KTH ROYAL INSTITUTE
OF TECHNOLOGY

Degree Project in Technology

Second cycle, 30 credits

Evaluating retrieval and summarisation performance of AI-Assistants built with Large Language Models and RAG-techniques (Retrieval Augmented Generation) in the domain of a LMS (Learning Management System)

A subtitle in the language of the thesis

LUDWIG KRISTOFFERSSON

Stockholm, Sweden, 2024

Evaluating retrieval and summarisation performance of AI-Assistants built with Large Language Models and RAG-techniques (Retrieval Augmented Generation) in the domain of a LMS (Learning Management System)

A subtitle in the language of the thesis

LUDWIG KRISTOFFERSSON

Master's Programme, Computer Science, 120 credits
Date: May 9, 2024

Supervisors: Michael Welle, Fredrik Enoksson
Examiner: Danica Jensfelt
School of Electrical Engineering and Computer Science
Host company: KTH IT
Swedish title: Detta är den svenska översättningen av titeln
Swedish subtitle: Detta är den svenska översättningen av undertiteln

Abstract

Foobar

Keywords

Canvas Learning Management System, Docker containers, Performance tuning

Sammanfattning

Foobar

Nyckelord

Canvas Lärplattform, Dockerbehållare, Prestandajustering

Acknowledgments

I would like to thank FEN for having yyyy. Or in the case of two authors:
We would like to thank xxxx for having yyyy.

Stockholm, May 2024
Ludwig Kristoffersson

Contents

List of Figures

List of Tables

List of acronyms and abbreviations

BERT	Bidirectional Encoder Representations from Transformers
CBOW	Continuous Bag-of-Words
CNN	Convolutional Neural Networks
ECM	Expectation-Confirmation Model
GAN	Generative Adversarial Network
GPT	Generative Pre-trained Transformers
GQA	Grouped-Query Attention
GRU	Gated Recurrent Units
IR	Information Retrieval
LLM	Large Language Models
LMS	Learning Manegement System
LSTM	Long Short-Term Memory
MMLU	Massive Multitask Language Understanding
MTEB	Massive Text Embedding Benchmark
NLP	Natural Language Processing
RAG	Retrieval Augmented Generation
RLHF	Reinforcement learning from human feedback
RNN	Recurrent Neural Network
seq2seq	Sequence-to-sequence
SMoE	Sparse Mixture of Experts
SWA	Sliding window attention
TAM	Technology Acceptance Model
TF-IDF	Term Frequency-Inverse Document Frequency

Chapter 1

Introduction

1.1 Background

This degree project will investigate Large Language Models (Large Language Models (LLM)) and Retrieval Augmented Generation (Retrieval Augmented Generation (RAG)) systems in the form of deploying an AI-Assistant in canvas course rooms. The degree project will investigate how to evaluate these systems in very specialised domains and benchmark various models, approaches and techniques.

The reason this research is important is that LLMs have gained widespread attention and we are likely to see large-scale adoption of these models into various applications. Understanding how to benchmark and evaluate these systems in specialised domains will be crucial to understand how to build these systems, which techniques to use, and which models work well.

Many organisations need to, due to commercial and regulatory compliance, host all AI-models themselves. This aspect is also interesting to evaluate, i.e. how well open source and commercially licensed models compare against the closed source models, such as GPT-4 by OpenAI.

The research will be carried out within the e-learning management object at KTH, who are responsible for the digital learning environment at KTH. The object consists of two teams at the KTH IT department and one team at the digital learning unit at the ITM-school. The university hosts thousands of courses with domain specific information, such as assignments, lectures and schedules, that aren't part of the public domain and therefore not part of the training set of LLMs.

All the work done by KTH IT aims to improve the operations at the university. Among this is reducing the administrative burden undertaken by teachers and teaching assistants (TAs). KTH IT wants to investigate if AI-assistants can be deployed into the canvas course rooms to reduce the workload of teachers and TAs which would help them focus on teaching, helping students and improve the quality of the education. KTH IT wants to see if it's feasible to deploy an AI assistant into the canvas course rooms.

1.2 Problem

LLMs have gained widespread use since its popularisation by ChatGPT. Their abilities to summarise large bodies of text and follow user instructions have proven very useful in many contexts. However, considering their limited context window (and drawbacks of models with larger context window [1]) deploying useful applications with a chat based interface still rely upon integrating a RAG system, introduced by Lewis et al. [2]. These can retrieve relevant information needed to answer a user's query from outside data sources and inject them into the conversation.

Some announced but currently unreleased models, such as the gemini family of models [**?**], have been reported to show great recall performance and reasoning abilities over millions of tokens. This could significantly reduce the importance of RAG systems in applications which utilise LLMs and external datasets to create intelligent systems with domain specific knowledge. However, even though no exact figures are presented by the Gemini time, inference speed (the time taken to produce a response to a prompt) seems to be significantly slower than shorter contexts. This would again highlight the importance of efficient RAG systems. Still, other approaches than traditional GPUs have been shown recently [3] by the Groq team to greatly increase inference speed.

Evaluating large language models is notoriously difficult. There are objective and automated metrics that can be used for tasks such as evaluating a model's summarisation capabilities, as shown by Basyal and Sanghvi [4]. However, for more complicated evaluations it gets trickier. In their seminal instructGPT paper Ouyang et al. at OpenAI try to evaluate *"how well a model can follow instructions"* [5] which is a very subjective question. They essentially relied upon human labellers to judge the overall quality of each response generated by the model.

In their Gemini-paper the Gemini team discuss the benchmarks used for their largest model. The team states that benchmarks are often designed to test shorter prompts whereas their longer prompts challenge tests used in traditional evaluation methods that rely heavily on manual evaluation. This highlights the relevance of good evaluation metrics. Regardless of context size or inference speed, evaluation of models tends to be very general. Which makes sense, when considering their general application.

When releasing their Mixtral model [6] the Mistral AI team used a range of benchmark tests, such as MMLU, PIQA, GSM8K etc. Massive Multitask Language Understanding (MMLU) [7] benchmarks a LLMs proficiency in understanding and reasoning across various subjects such as humanities, STEM, and professional and everyday knowledge, by evaluating its performance on 57 tasks, to test its ability to generalise and apply knowledge. PIQA (*Physical Interaction: Question Answering*) [8], evaluates a language models understanding of physical commonsense by asking them to predict the outcome of physical interactions in various scenarios through multiple-choice questions. GSM8K (*The Grade School Math*) [9] tests the ability to solve elementary-level mathematics word problems.

Evaluation of how well LLMs perform is an open research question. As shown above LLM developers often utilise multiple testsuites. These are oftentimes, as shown above, very general tests. When implementing LLMs in practical applications good performance often relies upon very good raw summarisation performance and reasoning abilities. Since the domain specific knowledge is provided to the model, raw built-in knowledge isn't crucial. It is more important for the model to learn the task at hand using very few examples and within the given domain understand the question being asked by a user. Further, as argued by by Siriwardhana et al., the training data of LLMs include the knowledge of datasets such as Wikipedia [10] which means that evaluation methods in very specialised domains hold higher value than generalised domains. These brand new domains, that with certainty haven't been seen during training, tests the models zero-shot, and depending on the implementation, few-shot learning abilities.

The research question for this project is *Which language model and which retrieval techniques do students prefer using?* and *Is it possible to deploy an AI-assistant using a completely open source toolchain?*.

I believe the answer to the first question is that the closed source alternatives will be preferred by the students, however, I think the results will show it is possible to deploy an open source based AI assistant too.

1.2.1 Original problem and definition

The core challenge addressed in this thesis is the effective deployment and evaluation of AI-assistants powered by LLM and RAG techniques in a specialised domain, specifically within the Learning Manegement System (LMS) of Canvas course rooms at KTH. This involves assessing the practicality and efficiency of integrating AI-Assistants built upon LLMs and RAG techniques into the educational settings to aid in reducing administrative burdens on educators and enhancing student interaction with course materials.

The original problem stems from the need to understand whether AI-assistants can effectively handle the domain-specific data intrinsic to educational platforms that are not included in their initial training datasets. Furthermore, the project aims to compare the efficacy and acceptability of open-source versus proprietary AI models in real-world educational applications.

1.3 Purpose

The purpose of this thesis is two-fold: firstly, to innovate within the educational technology space by integrating AI-assistants to potentially reduce workload and improve informational access within Canvas course rooms. Secondly, the thesis aims to contribute to academic knowledge by providing empirical data on the performance of these AI systems in a controlled educational setting. The dual purpose of this thesis ensures it not only investigates the immediate needs of KTH's digital learning environment but also enriches the scientific community's understanding of applied AI within a specialised domain, such as education.

This research is intended to benefit educational institutions by potentially offering a tool that improves operational efficiency and students by providing an alternative, possibly more effective way of interacting with course content. In addition the research will bring benefits for researchers within AI and education. Ethically, the study focuses on the sustainable development of AI technologies by emphasising open-source solutions, aiming to democratise advanced technological developments and reduce reliance on proprietary models.

1.4 Goals

1. **Technological Efficacy:** To evaluate the accuracy, speed, and reliability of responses by AI-assistants utilising both proprietary and open-source models in handling domain-specific content, such as the course rooms in canvas.

2. **User Preference:** To understand the preferences of students regarding the usability, information quality, and overall experience of interacting with an AI-assistant built upon different models and retrieval techniques.

3. **Operational Feasibility:** To assess the feasibility of integrating an AI-assistant built on fully open-source technologies within an academic setting, considering logistical, technical, and regulatory constraints.

4. **Educational Impact:** To explore the potential of AI-assistants to reduce administrative burdens on educators and improve information accessibility for students.

5. **Comparative Analysis:** To perform a comparative study between various LLM models and RAG-techniques.

1.5 Research Methodology

This project primarily employs an empirical study based on data collection and qualitative insights that will be used to evaluate the implementation of AI-assistants in the educational domain.

1.5.1 System Design and Implementation

Model selection Different models, including proprietary and open-source, with different sizes (number of active parameters), will be tested. The relevant models will be included in the study for testing.

RAG technique selection Various configurations of RAG systems will be tested to identify the most effective method for enhancing the AI's responses with respect to the layout of the data in Canvas course rooms. The relevant techniques will be included in the study for testing.

Implement AI Assistant Design and implement the system that will be used in the study.

Construct study questions Craft the questions that will be asked to students and implement them in the AI assistant.

Find courses to test with Find willing course administrators that want to participate in the study with their students.

1.5.2 Evaluation Design

Study Participants The study will involve students using the AI-assistant and providing feedback on their experiences.

Experimental Setup Controlled experiments will be conducted where participants use different configurations of the AI-assistant for typical student questions.

Data Collection Methods Data will be collected through integrated survey questions within the chat interface, capturing real-time feedback on the AI-assistant's performance and student satisfaction.

1.5.3 Analysis Techniques

Quantitative Analysis Statistical methods will analyse usage data and response accuracy to quantitatively assess the AI-assistant's performance.

Qualitative Analysis Feedback and open-ended responses will be analysed textually to understand user perceptions and contextual effectiveness of the AI-assistant.

This methodology was chosen for its ability to provide a comprehensive evaluation of both the technical capabilities and the practical usability of AI-assistants, offering insights into their potential benefits and limitations in the specific context for this study.

1.6 Delimitations

This project has several delimitations that define the scope and boundaries of the research to ensure a focused and manageable study. The key delimitations are;

- **Model Scope:** The project will not involve the development of new models or the fine-tuning of existing models. This includes LLM and embedding functions. The study will utilise pre-trained models offered by bigger vendors or the open source community.
- **Data Limitations:** Only existing courses within KTH's Canvas LMS will be utilised for the study. No new course content will be created, and no modifications will be made to existing course materials beyond what is necessary for the integration and testing of the AI-assistants.

- **Course Data Access:** The project will not use Canvas APIs for data integration. All interactions with the Canvas platform will be through existing interfaces, or data will be scraped and used from the Canvas web interface.

- **Geographic and Cultural Constraints:** The study is limited to the KTH environment, which may not represent other educational settings in different cultural or geographic contexts or languages. The findings might not be directly transferable to other institutions or countries without additional localisation and adaptation.

1.7 Structure of the thesis

Chapter 2 presents relevant background information about xxx. Chapter 3 presents the methodology and method used to solve the problem. …

Chapter 2

Background

This chapter provides the necessary background for understanding the research conducted within this thesis. This chapter also showcases the related work for this thesis and how the research relates to it.

2.1 Neural Networks

Neural network models are a type of models within the broader field of machine learning whose design have been inspired by human brains. These models allow computers to recognise patterns and solve complex problems. The backpropagation algorithm was popularised by Rumelhart, Hinton, and Williams [11]. This algorithm efficiently computes the gradient of the loss function with respect to the weights of the network by propagating the error back from the output layer to the input layer. This method is critical to understand all machine learning pipelines because it enables the network to adjust its weights in a way that minimises the error, thereby improving the model's predictions over time.

Building on backpropagation, Yann LeCun et al. [12] introduced the Convolutional Neural Networks (CNN) architecture in 1998. These are a specialised kind of neural network for processing data, such as images, which can be converted to a matrix. CNNs utilise layers with convolving filters that apply the learned weights across subsections of the input data. This reduces the amount of parameters in the network and improves its efficiency.

These are two steps in the evolution of neural network models, particularly the developments in CNNs and other deep learning technologies, are central for setting the stage for even more complex architectures aimed at processing not just visual data, but sequential data such as text. This will eventually lead to Large Language Models (LLM), which leverage deep learning techniques to understand and *generate* human language. LLMs are built upon the principles of neural networks. Understanding the models we commonly refer to as LLMs involves understanding models such as Transformer models, Bidirectional Encoder Representations from Transformers (BERT), and other encoder-decoder networks.

2.1.1 Recurrent Neural Networks (RNNs)

A Recurrent Neural Network (Recurrent Neural Network (RNN)) is a type of neural network that is good for modelling sequential data. They are significantly different from other neural networks in their ability to maintain memory of previous inputs using an internal state. This state which is maintained inside the network while it's running, will influence the network's output. RNNs proved to be fundamental in tasks where context was crucial, such as language modelling and generation of text.

In an RNN, each neuron, its most basic building block, processes a part of the sequence, receiving both the current input x_t and the output from the previous step h_{t-1}, this is known as the "hidden state". The core of an RNN operation involves updating this hidden state using:

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t + b)$$

where W_{hh} and W_{xh} are the weights for the hidden state and input, respectively, and b is a bias. The updated state h_t is used in the next step to generate the output y_t via:

$$y_t = W_{hy}h_t + b_y$$

However, RNNs often struggle with maintaining a longer context due to problems like vanishing and exploding gradients, as written by Hochreiter and Schmidhuber [13]. This was a problem other RNN models tried to mitigate as it significantly reduce their usefulness in various tasks. The vanishing gradient problem makes it difficult for the RNN to learn connections between events that occur at longer distances in the input sequence because the gradient of the loss function decays exponentially with the length of the input sequence.

This led to the development of more sophisticated variants like Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRU)s were developed. LSTMs [13], use input, output, and "forget gates" to manage information flow, which allows them to maintain stable gradients. GRUs, which was proposed by Cho et al. [14], simplifies this by merging the gates and states, reducing complexity while preserving performance across various tasks.

2.1.2 Sequence-to-Sequence Models

Sequence-to-sequence (seq2seq) models are designed to process sequences of data, such as text or speech, and generate corresponding output sequences. Sutskever et al. [15] were the first to introduce these models which typically consist of two main components: an encoder and a decoder. The encoder will process the input and convert it into a dense vector. This vector encodes the entire input sequence which is then passed to the decoder, which generates the output. This architecture proved very useful in certain tasks such as translating text between languages. Bahdanau, Cho, and Bengio built upon this concept with attention mechanisms [16] which would allow the decoder to focus on a specific piece of the input for small parts of the output, which improved the models ability to focus on longer sequences.

2.1.3 Transformer Models

The Transformer model, introduced by Vaswani et al. [17], was a new approach for seq2seq networks, with a self-attention mechanism which was different from the recurrent design of RNNs. The new transformer architecture introduced by Vaswani et al. allowed the network to weigh the importance of different tokens in the input data irrespective of their sequential position. Where a token is a sequence of characters that can be treated as a single logical entity in the input and output sequence.

The key innovation of the Transformer is its ability to handle dependencies between single tokens or sequences of tokens at long distances from each other. This makes the transformer architecture especially good at understanding context in text data.

The introduction of the transformer model was foundational in the field, and today most models use this architecture, see section 2.3 and 2.3.2.

2.1.4 BERT and Advances in Encoder-Decoder Models

BERT was introduced by Devlin et al. [18] in 2018 and was a major improvement within natural language processing. The BERT model optimised token representations bidirectionally which means that it was refining the understanding of each token by looking at the tokens before and after each token. BERT was built on the transformer model's encoder which allowed for pre-training on large text corpora, followed by fine-tuning for various tasks such as sentiment analysis and question answering.

Encoder-decoder models are important in machine learning for tasks that involve converting one sequence into another, such as machine translation or speech-to-text. In this type of model the encoder processes the input sequence and compresses information into what's known as a context vector, this is a condensed representation of the input data. The decoder takes this context vector and generates an output sequence token by token. Each of these two components may be built using recurrent networks, convolutional networks, or more commonly nowadays, transformer architectures.

In contrast to traditional encoder-decoder models, encoder-only models, such as BERT, focus on generating an output based on an input without the need for a decoder. These models are typically used for tasks that require deep understanding of language context like sentence classification.

Decoder-only models, like the Generative Pre-trained Transformers (GPT) (see section 2.3), focus on generating sequences from a given context or starting point. These models are very good in situations where the model needs to exhibit "creative" properties, such as when generating text completions.

Parallel to BERT, other encoder-decoder models like the Transformer [17] and seq2seq networks with attention mechanisms [16] have shown great results when translating sequences in tasks like machine translation, exemplified by Google's Neural Machine Translation system [19], and speech recognition, as seen in Apple's Siri voice assistant [20].

2.2 Generative AI

Generative AI is a term used to describe a subset of artificial intelligence technologies that are designed to create new content. This can be images such as with DALL-E [21], text with models like GPT-3 [22] or movies [23]. These models are capable of generating realistic and arguably novel outputs by understanding and simulating the underlying structure of the training data. One of the most popular frameworks in Generative AI includes Generative Adversarial Network (GAN)s, introduced by Goodfellow et al. [24], which consist of two neural networks, the generator and the discriminator. These two networks will compete against each other. The generator creates items that are as realistic as possible, and the discriminator evaluates them. This process runs until the discriminator can no longer accurately separate generated items from the training data.

2.3 State-of-the-Art Large Language Models

LLM represent a significant breakthrough in Natural Language Processing (NLP). They are capable of understanding and generating text similar to that written by humans. In recent years, several cutting-edge LLMs have been developed by prominent companies and research institutions that have gained wide-spread use. This section gives an overview of some notable examples of these advanced LLMs.

2.3.1 OpenAI's GPT Series

OpenAI's GPT series of language models have over the past few years featured some of the most widely used language models. GPT-1 was first released in 2017 followed by GPT-2, GPT-3, and GPT-4 (with various variants of these models). GPT-3, in particular, with its 175 billion parameters, has demonstrated strong capabilities in tasks such as text completion, question answering, and even code generation [22]. These models are some of the most widely used models, primarily due to their popularisation by the product from the same company, ChatGPT *.

2.3.2 Mistral

Mistral is a french firm that has released a few models that has gained widespread adoption in the open source community. As of writing, *Mistral-7B-Instruct-v0.2* had 2,297,845 million downloads on huggingface last month †, and Mixtral-8x7B-Instruct-v0.1 had 628,927 ‡.

Mistral 7B v0.1 [25] was their first major model to get widespread notoriety. The model is a 7-billion-parameter language model which was small enough to run on consumer-grade GPUs. The model utilised Grouped-Query Attention (GQA)[26] and Sliding window attention (SWA) [27] techniques to achieve impressive results across various benchmarks, including reasoning, mathematics, and code generation tasks. *Mistral 7B v0.1 instruct* is a related fine-tuned model.

The "instruct" version of generative AI models, such as the Mistral 7B, has been fine-tuned to follow prompted instructions. In contrast, the base model simply generates output based on the provided prompt. This process was first

*chat.openai.com

†The huggingface page for Mistral-7B-Instruct-v0.2

‡The huggingface page for Mixtral-8x7B-Instruct-v0.1

published by the team at OpenAI [5], however it's also employed by mistral and other model vendors. This approach is commonly used for models deployed in AI assistants or chat applications.

The *Mixtral of Experts* model [6], is a variant of the Mistral model that introduces a Sparse Mixture of Experts (SMoE) architecture, as described by Jiang et al. *Mixtral-8x7B-Instruct-v0.1* employs 8 feedforward blocks (experts) in each layer, with a router network selecting two experts for processing and combining their outputs at each timestep. The model has access to 47 billion parameters, but effectively only utilise 13 billion parameters during inference, which makes the model easier to deploy on GPUs with less amounts of memory.

2.3.3 Google's Language Models

Google has two major families of model, the first being the Gemini family, as introduced in a series of papers by Google's team [28], consists of models like Gemini Ultra, Pro, and Nano, each of these models are designed for specific applications and more importantly size of GPU. Where the larger models require enterprise-grade GPUs that are expensive to operate. Gemini 1.5 extended on these models with an even larger context window by effectively processing and recalling information across millions of tokens in a multi-modal context (tokens include both text, audio and image tokens) [29]. This is the first model to demonstrate resilience to the problem first described by Nelson et al. where the model would be biassed towards instructions or data in the beginning and end of larger prompts [1].

Goggles Gemma family of models [30] represents Google's effort to provide state-of-the-art, lightweight models to the open source community. These models, available in sizes of 2 billion and 7 billion parameters. The models demonstrate worse performance against their Gemini class of models across all tasks such language understanding and reasoning. However, the Gemma models' size make them easier to deploy on smaller consumer-grade GPUs.

2.3.4 The LLama family of models

In February 2023, Meta AI released LLaMA [31] in four distinct sizes: 7, 13, 33, and 65 billion parameters. The model utilised features such as SwiGLU activation functions, rotary positional embeddings, and root-mean-squared layer-normalisation to achieve comparable results to OpenAIs GPT-

3 model. Despite being initially released under a noncommercial licence, the weights of LLaMA were leaked, prompting widespread unauthorised use. This accelerated its adoption across various applications.

Later in July of 2023, Meta released LLaMA-2 [32] which was built upon the foundational models of its predecessor with enhanced data sets of 2 trillion tokens, fine-tuning capabilities, and improved dialogue system performance through specialised LLaMA-2 Chat models, these are similar to the instruct models mentioned in section 2.3.2. LLaMA-2 had a 40% larger training corpus and extended the context length to 4,000 tokens. The release included model sizes from 7 to 70 billion parameters. These models were released under a similar licence to the first LLaMA models.

Recently, in April 2024, Meta AI released LLaMA-3, this time with two models, one 8 billion parameter model and one 70 billion parameter model. These were open source and available online * from day one under a commercial licence. The model was pre-trained on approximately 15 trillion tokens. Meta announced an, as of writing, future release of a 400 billion parameter model.

2.3.5 Notable other vendors

Besides the major players such as OpenAI, Google, and Meta, there exists a vast array of players, of varying size, that also develops language models. These include, but are not limited to, Anthropic, IBM and DeepMind (which is also a part of Google).

2.4 Prompt engineering

Prompt engineering is the name given to the technique that evolved from the use of language models. This is the task of optimising the performance of a LLM such as GPT-4, LLaMA, and others. This involves crafting the input text, or *"prompt"* to these models in a way that guides them to produce desired outputs [33, 34].

Prompt engineering is defined as the practice of designing input prompts that maximise the efficacy and accuracy of LLM outputs. It is a key factor in the success of deploying LLM-based applications. The process of prompt engineering involves several key techniques. A prompt should, according to Chen et al. include clear instructions and enough contextual details to guide

*The GitHub repository for LLaMA-3

the model towards providing the expected answer in the expected format. There are numerous advanced techniques such as "role-prompting", zero-shot, one-shot, and few-shot prompting that can improve the performance of LLM.

For instance, Kathiriya et al. [33] demonstrates that role-prompting produces responses with heightened professional relevance. Similarly, Chen et al. highlight how few-shot prompting can refine the model's ability to perform complex analytical tasks by providing some targeted examples. Both of these studies show how prompt engineering techniques can improve performance.

Figure 2.1, taken from the paper published by Chen et al. [34] illustrates an example of role-prompting. In this example the LLM is instructed to assume the role of an expert in artificial intelligence, which aligns its responses with specific professional knowledge.

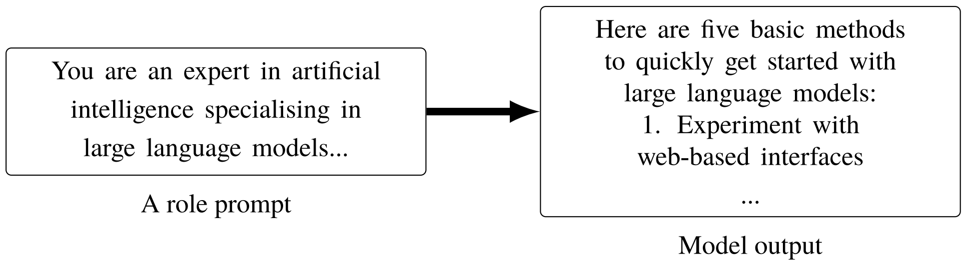

Figure 2.1: Role prompting example.

Another technique known as few-shot prompting, is shown in figure 2.2, taken from the paper written by Kathiriya et al. [33]. With this technique the model is provided with multiple examples to better understand the task. If only one example is given, this is referred to as "one-shot" prompting. Similarly, if no example is given, then the prompt is referred to as a "zero-shot" prompt.

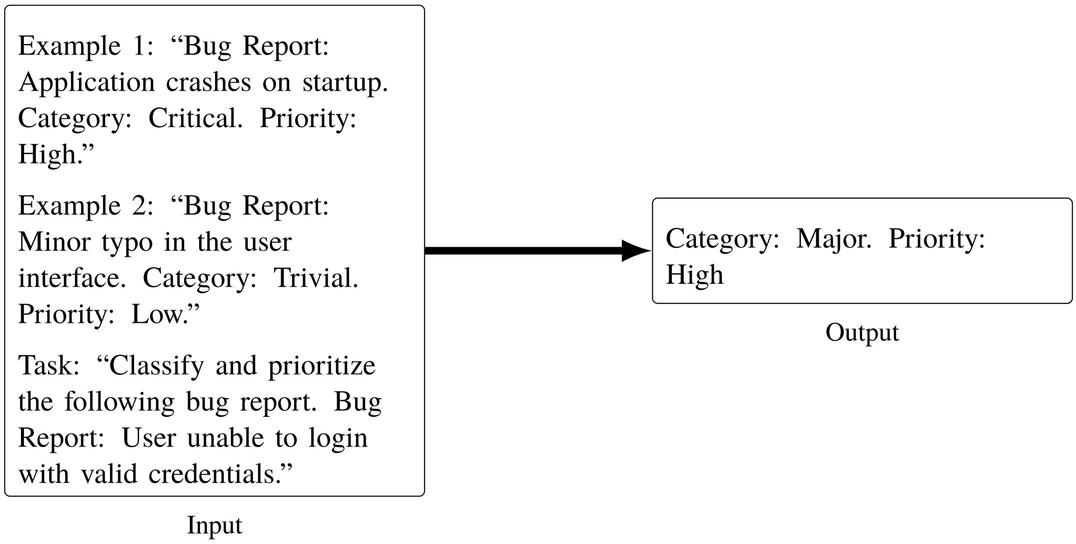

Figure 2.2: Few-shot prompting example.

2.5 Evaluating LLM performance

When an LLM vendor, such as Mistral, releases a new model its performance is evaluated using a series of well-established benchmarks. These benchmarks are essential for understanding the model’s capabilities in various cognitive tasks. This includes tasks such as mathematical reasoning, language understanding and program synthesis. This practice helps to quantify the models’ performance and provides a method to compare its performance against previously released models [9, 35, 36].

When Mistral released its first major model they used table 2.1 to compare its results against the LLaMA family of models. There are numerous benchmarks included in the table, that test various abilities of the model. For example, the *GSM8K* benchmark includes thousands of grade-school level maths problems that are designed to test mathematical reasoning [9]. Benchmarks like MBPP assess a model’s ability to understand and generate programming code from natural language descriptions [35]. A test like *MMLU* measures general world knowledge and problem solving ability [7]. Lastly, the *PIQA* benchmark, challenges a model with physical common sense questions [37]

Model	MMLU	HellAs	PIQA	Arc-e	Arc-c	HumanE	MBPP	Math	GSMBK
LLaMA 2 7B	44.4%	71.1%	77.9%	68.7%	43.2%	11.6%	26.1%	3.9%	16.0%
LLaMA 2 13B	55.6%	70.7%	80.8%	75.2%	48.8%	18.9%	35.4%	6.0%	34.3%
LLaMA 1 33B	56.8%	83.7%	82.2%	79.6%	54.4%	25.0%	40.9%	8.4%	44.1%
LLaMA 2 70B	69.9%	85.4%	82.6%	79.9%	56.5%	29.3%	49.8%	13.8%	69.6%
Mistral 7B	62.5%	81.0%	82.2%	80.5%	54.9%	26.2%	50.2%	12.7%	50.0%
Mixtral 8x7B	**70.6%**	**84.4%**	**83.6%**	**83.1%**	**59.7%**	**40.2%**	**60.7%**	**28.4%**	**74.4%**

Table 2.1: Table used by Mistral to compare the performance of *Mixtral 8x7B* to *Mistral 7B* and the Llama family of models.

All of these benchmarks are essential for developers and researchers to understand the limitations and capabilities of AI models. They ensure continuous improvements and innovations in the field. Each benchmark is sourced and created differently. Some models are tested against just a single benchmark such as OpenAI's codex model (now deprecated) [36]. However, most general purpose language models such as those released by Mistral, OpenAI, Meta, Google and more use a common set of benchmarks such as those in table 2.1 [6, 38, 31, 29].

2.6 Web crawling

Web crawling is a technique to systematically browse the World Wide Web to index the content of websites for search engines and other applications using automated programs known as web crawlers [39, 40]. This is a process that's crucial for the operation of search engines.

A web crawler starts with a list of URLs to visit. As the crawler visits these URLs, it identifies all the hyperlinks on the page and adds them to a database of known URLs to visit. After visiting a URL the crawler employs a method of selecting the next url to visit, which may be one of the hyperlinks it just found on the current page, or any url it might have found before. This method may vary depending on the implementation of the crawler. This process continues until a defined stop condition.

While the primary application of web crawling is in web search engines, it can also be used within various other domains. Web crawlers can be used for everything from monitoring changes in web pages to gather data from specific intranets or corporate knowledge bases.

Implementing an efficient web crawler involves addressing multiple technical challenges. These are primarily constructing an efficient crawler that can visit and process urls at scale. Additionally, the crawler must be able to index the content found on those websites, which may include various media types such as plaintext, images or document formats such as PDF.

2.7 Information Retrieval

Information Retrieval (IR) refers to the process of returning relevant information from a corpus of documents. The field primarily focuses on the retrieval of text data and is a core part of many applications such as search engines or AI agents.

The objective of information retrieval is to find material within an unstructured database [41]. This usually involves resolving the relevant documents in response to a user query. Information retrieval systems are usually measured against precision and recall metrics. These show how relevant the documents returned were, and how many of the relevant documents were returned.

$$\text{Precision} = \frac{\text{Number of Relevant Documents Retrieved}}{\text{Total Number of Documents Retrieved}} \tag{2.1}$$

$$\text{Recall} = \frac{\text{Number of Relevant Documents Retrieved}}{\text{Total Number of Relevant Documents in the Corpus}} \tag{2.2}$$

The core of IR is indexing and search algorithms. To index a corpus means processing all the documents in the corpus into a data structure that can later be used for retrieving docs. Search algorithms utilise this index to find documents that match the user's query [41].

The second problem of IR is to rank the returned documents. This is problem with many possible solutions.

2.7.1 Term frequency inverse document frequency

Term Frequency-Inverse Document Frequency (TF-IDF) is a measurement used to evaluate how important a word is to a document in a collection or corpus. This means it is a relative metric that is unique to the corpus being indexed. TF-IDF is calculated by multiplying two values

1. How many times a term appears in a document

2. The inverse document frequency of the term across a set of documents

Term frequency is calculated using the following formula

$$\text{TF}(t, d) = \frac{\text{Number of times term } t \text{ appears in document } d}{\text{Total number of terms in document } d} \tag{2.3}$$

The inverse document frequency is calculated using this formula

$$\text{IDF}(t, D) = \log\left(\frac{\text{Total number of documents in the corpus } D}{\text{Number of documents containing term } t}\right) \tag{2.4}$$

The complete formula for TF-IDF is the following

$$\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \times \text{IDF}(t, D) \tag{2.5}$$

This formula means the relevance for a token increases with the number of times that term appears in the document, but is offset by the frequency of the term in the entire corpus. This is a good way of adjusting for the fact that some words are generally more common than others, such as "a", "the", etc. [41].

2.7.2 Embedding Functions

Vector embeddings are a way of representing text or other media content, such as images, as a numerical vector that encapsulates their features. Figure 2.3 illustrates how a token, in this case *cat* and *dog*, is encoded into a vector.

$$\texttt{cat} \rightarrow \{0.042, 0.112, 0.236, 0.368, 0.491, 0.623, 0.784, 0.895, \ldots, 0.931\}$$

$$\texttt{dog} \rightarrow \{0.157, 0.209, 0.330, 0.501, 0.579, 0.619, 0.755, 0.832, \ldots, 0.874\}$$

Figure 2.3: Example embeddings for "cat" and "dog" strings.

For text content such a feature could be something abstract about a word that's even true in several languages. In text processing, one typically leverages the neural network of a language model to understand the contexts and co-occurrences of tokens. These networks have usually been trained on very large corpora of text and are thereby very good at placing semantically similar tokens close to each other in a vector space. For example, *football* and *soccer* may appear in similar contexts, leading the network to locate them near each other in a "meaning space", as can be seen in figure 2.4. Measured with something like levenshtein distance, the words are very far from each other, even though we know they are synonymous in many contexts. Processing text with a neural network and representing it with a vector can allow computers to perform complex tasks like text prediction with an understanding akin to human cognitive judgments [42].

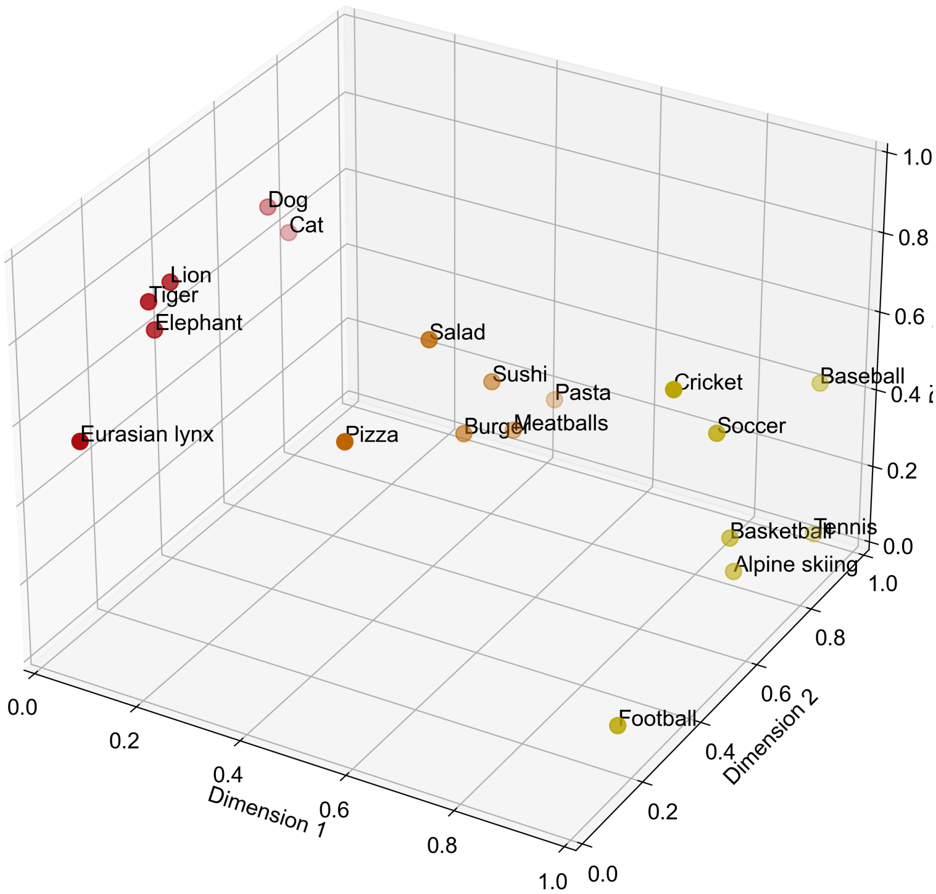

Figure 2.4: Simplified 3D space with simplified embeddings for various words

There are numerous types of models and techniques that have been developed to efficiently compute vector embeddings. One such is Continuous Bag-of-Words (CBOW) and Skip-gram models, introduced by Mikolov et al. [42]. These were early yet foundational methods for generating word embeddings. These models leverage large corpora to predict tokens from their context (CBOW), or with the context from the tokens (Skip-gram). Both of these techniques leverage the trained models ability to learn semantic and syntactic nuances of the tokens in the training corpus [42, 43].

Advancements in vector embedding technologies enhance NLP tasks such as text classification and sentiment analysis. Embedding models that can process language or images with nuance and precision are crucial for accurate real-world applications [44].

There are new Embedding functions released often, built-upon different language models and employing various different techniques. The evaluation of these embedding functions often remains constrained to a narrow set of tasks. Muennighoff et al. [45] tried to address this issue by introducing Massive Text Embedding Benchmark (MTEB), which spans 8 embedding tasks covering a total of 58 datasets and 112 languages. The leaderboard is currently actively maintained on hunggingface *.

Two models that rank highly on the leaderboard is Salesforce's open source *SFR-Embedding-Mistral* model which exemplifies advancements in embedding technology for text retrieval tasks [46]. Similarly, OpenAI has developed several closed source embedding models that also rank highly on the MTEB leaderboard [47, 48].

Embeddings are often used to compare documents against each other, or against a given user query. This is often done by computing similarity scores between words, phrases, or documents, which are represented as vectors in the embedding space. These scores quantify the closeness, or "similarity" between different texts.

The similarity between two vector representations is typically measured using the cosine similarity metric. This calculates the cosine of the angle between two vectors. This metric ranges from -1 (the exact opposite document) to 1 (the exact same document), with 0 indicating orthogonality (no similarity). The cosine similarity $\text{sim}(u, v)$ between two vectors u and v is defined as:

$$\text{sim}(u, v) = \frac{u \cdot v}{\|u\|\|v\|} \tag{2.6}$$

where $u \cdot v$ is the dot product of the vectors u and v, and $\|u\|$ and $\|v\|$ are the Euclidean norms of both vectors.

*Massive Text Embedding Benchmark (MTEB) Leaderboard on Huggingface

2.8 RAG

RAG is the process of integrating retrieval mechanisms into the generative models. This approach effectively combines the strengths of both retrieval and generative language modelling to enhance a model's ability to accurately recall factual information by utilising an external knowledge base during the generation process [2].

RAG was developed to address the limitations of large pre-trained language models that could compress a large training corpus into its weights, but could struggle with accessing and precisely manipulating this information when required. The term "hallucination" would come to describe the event where models would "recall" incorrect information, as shown in figure 2.5.

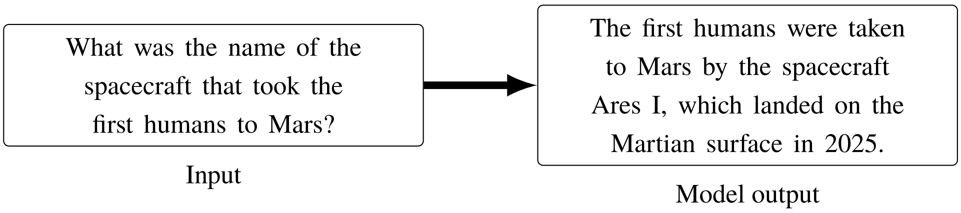

Figure 2.5: An example of how a model can hallucinate an answer to a question.

The fact that language models tend to have a propensity to hallucinate, and the simple fact that it is very time consuming to train language models, mean these models would often lag behind in knowledge-intensive applications where correctness is crucial. The integration of a non-parametric memory, or an external knowledge base, allows these models to retrieve relevant information during the generation process and thereby producing more accurate responses [2].

In a typical RAG setup, the systems architecture is split into two main components: the retriever and the generator. The retriever is a language model trained to search and fetch relevant documents. Nowadays, this process often utilises dense vector representations of documents (see section 2.7.2) which enables efficient and effective search [49].

The returned documents are then fed into the generator, this is a seq2seq model, which then synthesises the information into coherent text. The generator is often instructed to only use authoritative facts that are returned from the retriever and not rely on its internal knowledge for factual statements. This dual-component approach allows RAG to dynamically access a large corpus of knowledge while maintaining its ability to generate fluent and contextually appropriate language [2]. This method has shown significant improvement over a purely parametric-approach in various tasks such as question answering and fact verification [22, 50].

When google announced Gemini 1.5 [29] they claimed it could effectively recall knowledge over prompts as large as a million tokens. It remains to be seen if the hallucination, training-time and context length problems can be overcomed and remove the need for a RAG system when building knowledge-intensive applications on-top of a LLM.

2.9 AI Assistants

AI assistants is a type of AI-system that is designed to support human users by performing tasks that typically require human intelligence. Stuart Russell and Peter Norvig wrote in *Artificial Intelligence: A Modern Approach* that an assistant should interact with their environment to achieve specific goals rationally and effectively [51].

AI assistants must be good at NLP for effective communication with humans in addition to good knowledge representation such as with a RAG toolchain. The assistant must also possess good reasoning and decision-making abilities, as those exhibited by a modern LLM. An assistant should also utilise machine learning to improve from user interactions.

2.10 Measuring usability and acceptance of new technologies

To assess how effectively a user can interact with a technology, for instance, an AI assistant, Jakob Nielsen's "Usability Engineering" [52] is a seminal book that defines usability in terms of learnability, efficiency, memorability, safety, and satisfaction. All of these can be measured through specific metrics. The IBM Computer Usability Satisfaction Questionnaires, developed by Lewis [53], offer a tool that's been validated through the years to measure these dimensions.

Interactions with AI agents through conversation is very affected by the agent's ability to engage in social dialogue. Bickmore and Cassell [54] discuss the importance of dialogue in building engagement long-term between users and conversational agents. Their conversational agents communicated over the phone, but their framework for understanding the qualitative feedback from users about their experiences can also be applied with an AI assistant.

Technology Acceptance Model (TAM) was introduced by Davis [55] and is particularly relevant for examining the acceptance of new technologies such as AI assistants. TAM suggests that perceived usefulness and ease of use are key factors for whether a new technology is accepted and used. The model is useful for investigating users' attitudes towards the utility and usability of new technologies, not the least of which is AI and AI assistants.

Expectation-Confirmation Model (ECM) was introduced by Bhattacherjee [56] in 2001 and it extends the understanding of user satisfaction beyond initial acceptance which is outlined in TAM. ECM includes user expectations, perceived performance, and confirmation of expectations into the satisfaction assessment. This model is especially useful in assessing whether a technology meets or exceeds the users' expectations over time.

2.11 Related Work

2.11.1 The Open Source Models by Mistral and Meta

Mistral and Meta have both released a series of open-source models within their respective LLaMA and Mistral families. They've both made substantial contributions to the field of NLP. The LLaMA models offered very capable models that could fit within different computing envelopes, with various levels of compute capacity [31, 32]. Similarly, Mistral's models, including *Mistral-7B* and its Mixtral variants, have been made widely available and demonstrated robust capabilities. While the performance of these models is noteworthy, their most significant impact lies in their open-source licensing. This approach has democratised access to cutting-edge research and models, significantly accelerating the pace of innovation in the field compared to proprietary models from vendors such as OpenAI.

2.11.2 The Proprietary Models by OpenAI

The models released by OpenAI, which includes all the models from GPT-1 through GPT-4 have led the field across most benchmarks and text capabilities. Each new model has incorporated architectural improvements, larger datasets and new training methodologies. All of which have increased the models' ability to understand complex text structures and generate coherent text. OpenAI's contributions primarily lie in their scaled transformer architectures and fine-tuning techniques [22]. OpenAI have also done major advancements in LLM alignment research, for instance with their development of *instructGPT* [5]. InstructGPT was an effort to get models to follow human instructions. This was done through a combination of techniques such as Reinforcement learning from human feedback (RLHF) and supervised learning. These techniques have also shown to be able to increase truthfulness and reduce toxicity in the model's output, which may have been inherent in the models' training dataset.

2.11.3 Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

The introduction of RAG by Lewis et al. has been crucial for the development of NLP systems. The model they proposed has been shown to combine the generative power of a LLM with the factual accuracy of verified knowledge bases. From a research perspective, RAG is a method of addressing "hallucinations" in generative models [2]. "Hallucinations" refers to instances where a LLM asserts a fact that is provably untrue as though it were accurate, see figure 2.5. With the techniques proposed by Lewis et al. a system can be constructed around the model that ensures only verified facts are used to answer input prompts.

2.11.4 Gemini 1.5: Unlocking Multimodal Understanding Across Millions of Tokens of Context

There are two dominant theories for how AI systems will be constructed. The first theory advocates for a RAG-based approach. As discussed in sections 2.8 and 2.11.3, RAG effectively integrates an external knowledge base into an AI system. However, there is the alternative theory of longer context windows. The context window refers to the maximum amount of tokens the model can consider at one time when generating responses. The context window typically consists of the input prompt, whatever extra content has been added to the model (could be multi-modal content such as images) and the models' output. With a large enough context window, theoretically, an entire knowledge-base could be incorporated into a prompt. This would remove the need from intercepting the models generation process, which is the core component of RAG, since the entire knowledge base is part of the prompt.

Previously, Liu et al. has shown that models with larger prompts have a bias towards the beginning and the end of the prompt [1]. This effectively meant that a knowledge base with similar facts injected into the beginning or end of the prompt would have a higher likelihood of being included in the response, regardless of their correctness.

When Google announced their Gemini 1.5 model they released results that suggest that it could process and integrate information across vast context windows using an extended transformer architecture [29]. The model could do this without a bias in precision or recall depending on where the data existed in the prompt. Even though this model isn't widely available yet, Google's results suggest that it is possible to construct a model that can fit a large knowledge base within the context window of a LLM.

Larger context windows generally require larger models which are more expensive to run. Larger prompts also take longer compute responses for. It remains an open research question whether the necessity of RAG-systems will remain in the future.

2.11.5 Measuring Massive Multitask Language Understanding

The MMLU benchmark was introduced by Hendrycks et al. in 2020 to provide a framework for evaluating LLM across a broad range of tasks [7]. The test covers tasks in fields such as elementary mathematics, US history, computer science and law. It is designed to test a model's world knowledge and problem solving abilities. The benchmark is significant as it is the current most widely used general intelligence test on LLMs.

As highlighted by many, not the least Google in their Gemini 1.5 paper [29], there is a pressing need for new benchmarking approaches. Traditional benchmarks do not sufficiently challenge new models, especially in the multimodal domain where text, images, video, and audio are combined [29]. Current evaluation methods rely heavily on human labelling and annotation, they are increasingly seen as inadequate for more complex prompts and responses.

In addition most benchmarks are biassed towards the training dataset of the models. The model's general intelligence is measured by data that is available to the model during training. This includes historical events, facts and problems. For instance, included in the training set are programming problems very similar to the programming problems included in benchmarks such as *PIQA* [37]. To test the models true intelligence there is a need for benchmarks that measure the models abilities on data it hasn't seen before.

Chapter 3

Method or Methods

3.1 Research Process

3.2 Research Paradigm

3.3 Data Collection

3.3.1 Sampling

3.3.2 Sample Size

3.3.3 Target Population

3.4 Experimental design and Planned Measurements

3.4.1 Test environment/test bed/model

3.4.2 Hardware/Software to be used

3.5 Assessing reliability and validity of the data collected

3.5.1 Validity of method

3.5.2 Reliability of method

3.5.3 Data validity

3.5.4 Reliability of data

3.6 Planned Data Analysis

3.6.1 Data Analysis Technique

3.6.2 Software Tools

Chapter 4

What you did

4.1 Hardware/Software design . . . /Model/Simulation model & parameters/. . .

4.2 Implementation . . . /Modeling/Simulation/. . .

4.2.1 Some examples of coding

4.2.2 Some examples of figures in tikz

4.2.2.1 Azure's Form Recognizer

Chapter 5

Results and Analysis

In this chapter, we present the results and discuss them.

5.1 Major results

Some statistics of the delay measurements are shown in table... The delay has been computed from the time the GET request is received until the response is sent.

5.2 Reliability Analysis

5.3 Validity Analysis

Chapter 6

Discussion

diskussion här

Chapter 7

Conclusions and Future work

7.1 Conclusions

7.2 Limitations

7.3 Future work

Due to the breadth of the problem, only some of the initial goals have been met. In these section we will focus on some of the remaining issues that should be addressed in future work. ...

7.3.1 What has been left undone?

The prototype does not address the third requirment, *i.e.,* a yearly unavailability of less than 3 minutes; this remains an open problem. ...

7.3.1.1 Cost analysis

The current prototype works, but the performance from a cost perspective makes this an impractical solution. Future work must reduce the cost of this solution; to do so, a cost analysis needs to first be done. ...

7.3.1.2 Security

A future research effort is needed to address the security holes that results from using a self-signed certificate. Page filling text mass. Page filling text mass. ...

7.3.2 Next obvious things to be done

In particular, the author of this thesis wishes to point out xxxxxx remains as a problem to be solved. Solving this problem is the next thing that should be done. ...

7.4 Reflections

One of the most important results is the reduction in the amount of energy required to process each packet while at the same time reducing the time required to process each packet.

References

[1] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang, "Lost in the Middle: How Language Models Use Long Contexts," Nov. 2023, arXiv:2307.03172 [cs]. [Online]. Available: http://arxiv.org/abs/2307.03172 [Pages 2, 14, and 26.]

[2] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," Apr. 2021, arXiv:2005.11401 [cs]. [Online]. Available: http://arxiv.org/abs/2005.11401 [Pages 2, 23, 24, and 26.]

[3] D. Abts, G. Kimmell, A. Ling, J. Kim, M. Boyd, A. Bitar, S. Parmar, I. Ahmed, R. DiCecco, D. Han, J. Thompson, M. Bye, J. Hwang, J. Fowers, P. Lillian, A. Murthy, E. Mehtabuddin, C. Tekur, T. Sohmers, K. Kang, S. Maresh, and J. Ross, "A software-defined tensor streaming multiprocessor for large-scale machine learning," in *Proceedings of the 49th Annual International Symposium on Computer Architecture*. New York New York: ACM, Jun. 2022, pp. 567–580. [Online]. Available: https://dl.acm.org/doi/10.1145/3470496.3527405 [Page 2.]

[4] L. Basyal and M. Sanghvi, "Text Summarization Using Large Language Models: A Comparative Study of MPT-7b-instruct, Falcon-7b-instruct, and OpenAI Chat-GPT Models," Oct. 2023, arXiv:2310.10449 [cs]. [Online]. Available: http://arxiv.org/abs/2310.10449 [Page 2.]

[5] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. Christiano, J. Leike, and R. Lowe, "Training language models to follow instructions with human feedback," Mar. 2022, arXiv:2203.02155 [cs]. [Online]. Available: http://arxiv.org/abs/2203.02155 [Pages 2, 14, and 26.]

[6] A. Q. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford, D. S. Chaplot, D. d. l. Casas, E. B. Hanna, F. Bressand, G. Lengyel, G. Bour, G. Lample, L. R. Lavaud, L. Saulnier, M.-A. Lachaux, P. Stock, S. Subramanian, S. Yang, S. Antoniak, T. L. Scao, T. Gervet, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed, "Mixtral of Experts," Jan. 2024, arXiv:2401.04088 [cs]. [Online]. Available: http://arxiv.org/abs/2401.04088 [Pages 3, 14, and 18.]

[7] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt, "Measuring Massive Multitask Language Understanding," Jan. 2021, arXiv:2009.03300 [cs]. [Online]. Available: http://arxiv.org/abs/2009.03300 [Pages 3, 17, and 27.]

[8] Y. Bisk, R. Zellers, R. L. Bras, J. Gao, and Y. Choi, "PIQA: Reasoning about Physical Commonsense in Natural Language," Nov. 2019, arXiv:1911.11641 [cs]. [Online]. Available: http://arxiv.org/abs/1911.11641 [Page 3.]

[9] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman, "Training Verifiers to Solve Math Word Problems," Nov. 2021, arXiv:2110.14168 [cs]. [Online]. Available: http://arxiv.org/abs/2110.14168 [Pages 3 and 17.]

[10] S. Siriwardhana, R. Weerasekera, E. Wen, T. Kaluarachchi, R. Rana, and S. Nanayakkara, "Improving the Domain Adaptation of Retrieval Augmented Generation (RAG) Models for Open Domain Question Answering," *Transactions of the Association for Computational Linguistics*, vol. 11, pp. 1–17, Jan. 2023. [Online]. Available: https://doi.org/10.1162/tacl_a_00530 [Page 3.]

[11] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986, publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/323533a0 [Page 9.]

[12] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998, conference Name: Proceedings of the IEEE. [Online]. Available: https://ieeexplore.ieee.org/document/726791 [Page 9.]

[13] S. Hochreiter and J. Schmidhuber, "Long Short-term Memory," *Neural computation*, vol. 9, pp. 1735–80, Dec. 1997. [Page 10.]

[14] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, A. Moschitti, B. Pang, and W. Daelemans, Eds. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1724–1734. [Online]. Available: https://aclanthology.org/D14-1179 [Page 10.]

[15] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," Dec. 2014, arXiv:1409.3215 [cs]. [Online]. Available: http://arxiv.org/abs/1409.3215 [Page 11.]

[16] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," May 2016, arXiv:1409.0473 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1409.0473 [Pages 11 and 12.]

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention Is All You Need," Aug. 2023, arXiv:1706.03762 [cs]. [Online]. Available: http://arxiv.org/abs/1706.03762 [Pages 11 and 12.]

[18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," May 2019, arXiv:1810.04805 [cs]. [Online]. Available: http://arxiv.org/abs/1810.04805 [Page 11.]

[19] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and J. Dean, "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation," Oct. 2016, arXiv:1609.08144 [cs]. [Online]. Available: http://arxiv.org/abs/1609.08144 [Page 12.]

[20] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury,

"Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, Nov. 2012, conference Name: IEEE Signal Processing Magazine. [Online]. Available: https://ieeexplore.ieee.org/document/6296526 [Page 12.]

[21] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever, "Zero-Shot Text-to-Image Generation," Feb. 2021, arXiv:2102.12092 [cs]. [Online]. Available: http://arxiv.org/abs/2102.12092 [Page 12.]

[22] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language Models are Few-Shot Learners," Jul. 2020, arXiv:2005.14165 [cs]. [Online]. Available: http://arxiv.org/abs/2005.14165 [Pages 12, 13, 24, and 26.]

[23] OpenAI, "Video generation models as world simulators," Mar. 2024. [Online]. Available: https://openai.com/index/video-generation-models-as-world-simulators [Page 12.]

[24] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Networks," *Advances in Neural Information Processing Systems*, vol. 3, Jun. 2014. [Page 12.]

[25] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. l. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed, "Mistral 7B," Oct. 2023, arXiv:2310.06825 [cs]. [Online]. Available: http://arxiv.org/abs/2310.06825 [Page 13.]

[26] J. Ainslie, J. Lee-Thorp, M. de Jong, Y. Zemlyanskiy, F. Lebrón, and S. Sanghai, "GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints," Dec. 2023, arXiv:2305.13245 [cs]. [Online]. Available: http://arxiv.org/abs/2305.13245 [Page 13.]

[27] A. Roy, M. Saffar, A. Vaswani, and D. Grangier, "Efficient Content-Based Sparse Attention with Routing Transformers," Oct. 2020,

arXiv:2003.05997 [cs, eess, stat]. [Online]. Available: http://arxiv.org/abs/2003.05997 [Page 13.]

[28] G. Team, R. Anil, S. Borgeaud, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, and K. Millican, "Gemini: A Family of Highly Capable Multimodal Models," Apr. 2024, arXiv:2312.11805 [cs]. [Online]. Available: http://arxiv.org/abs/2312.11805 [Page 14.]

[29] G. Team, M. Reid, N. Savinov, D. Teplyashin, Dmitry, Lepikhin, T. Lillicrap, J.-b. Alayrac, R. Soricut, and A. Lazaridou, "Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context," Apr. 2024, arXiv:2403.05530 [cs]. [Online]. Available: http://arxiv.org/abs/2403.05530 [Pages 14, 18, 24, and 27.]

[30] G. Team, T. Mesnard, C. Hardin, R. Dadashi, S. Bhupatiraju, S. Pathak, L. Sifre, M. Rivière, M. S. Kale, and J. Love, "Gemma: Open Models Based on Gemini Research and Technology," Apr. 2024, arXiv:2403.08295 [cs]. [Online]. Available: http://arxiv.org/abs/2403.08295 [Page 14.]

[31] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample, "LLaMA: Open and Efficient Foundation Language Models," Feb. 2023, arXiv:2302.13971 [cs]. [Online]. Available: http://arxiv.org/abs/2302.13971 [Pages 14, 18, and 25.]

[32] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom, "Llama 2: Open Foundation and Fine-Tuned Chat Models," Jul. 2023, arXiv:2307.09288 [cs]. [Online]. Available: http://arxiv.org/abs/2307.09288 [Pages 15 and 25.]

[33] S. Kathiriya, M. Mullapudi, and A. Shende, “The Power of Prompt Engineering: Refining Human -AI Interaction with Large Language Models in The Field of Engineering,” *International Journal of Science and Research (IJSR)*, vol. 12, Nov. 2023. [Pages 15 and 16.]

[34] B. Chen, Z. Zhang, N. Langrené, and S. Zhu, “Unleashing the potential of prompt engineering in Large Language Models: a comprehensive review,” Oct. 2023, arXiv:2310.14735 [cs]. [Online]. Available: http://arxiv.org/abs/2310.14735 [Pages 15 and 16.]

[35] J. Austin, A. Odena, M. Nye, M. Bosma, H. Michalewski, D. Dohan, E. Jiang, C. Cai, M. Terry, Q. Le, and C. Sutton, “Program Synthesis with Large Language Models,” Aug. 2021, arXiv:2108.07732 [cs]. [Online]. Available: http://arxiv.org/abs/2108.07732 [Page 17.]

[36] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. d. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H. Guss, A. Nichol, A. Paino, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse, A. N. Carr, J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, and W. Zaremba, “Evaluating Large Language Models Trained on Code,” Jul. 2021, arXiv:2107.03374 [cs]. [Online]. Available: http://arxiv.org/abs/2107.03374 [Pages 17 and 18.]

[37] Y. Bisk, R. Zellers, R. L. Bras, J. Gao, and Y. Choi, “PIQA: Reasoning about Physical Commonsense in Natural Language,” Nov. 2019, arXiv:1911.11641 [cs]. [Online]. Available: http://arxiv.org/abs/1911.11641 [Pages 17 and 27.]

[38] M. AI, “Introducing Meta Llama 3: The most capable openly available LLM to date,” Apr. 2024. [Online]. Available: https://ai.meta.com/blog/meta-llama-3/ [Page 18.]

[39] G. Pant and P. Srinivasan, “Crawling the Web,” *Web Dynamics*, Jul. 2003. [Page 18.]

[40] M. Najork, "Web Crawler Architecture," L. Liu and M. T. Özsu, Eds. Boston, MA: Springer US, 2009, pp. 3462–3465, book Title: Encyclopedia of Database Systems. [Online]. Available: http://link.springer.com/10.1007/978-0-387-39940-9_457 [Page 18.]

[41] M. Christopher D., R. Prabhakar, and S. Hinrich, *Introduction to Information Retrieval*. Cambridge University Press, 2008. [Online]. Available: https://nlp.stanford.edu/IR-book/html/htmledition/irbook.html [Pages 19 and 20.]

[42] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," Sep. 2013, arXiv:1301.3781 [cs]. [Online]. Available: http://arxiv.org/abs/1301.3781 [Pages 21 and 22.]

[43] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," in *Advances in Neural Information Processing Systems*, vol. 26. Curran Associates, Inc., 2013. [Online]. Available: https://papers.nips.cc/paper_files/paper/2013/hash/9aa42b31882ec039965f3c4923ce901b-Abstract.html [Page 22.]

[44] J. Pennington, R. Socher, and C. Manning, "GloVe: Global Vectors for Word Representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, A. Moschitti, B. Pang, and W. Daelemans, Eds. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543. [Online]. Available: https://aclanthology.org/D14-1162 [Page 22.]

[45] N. Muennighoff, N. Tazi, L. Magne, and N. Reimers, "MTEB: Massive Text Embedding Benchmark," Mar. 2023, arXiv:2210.07316 [cs]. [Online]. Available: http://arxiv.org/abs/2210.07316 [Page 22.]

[46] R. Meng, Y. Liu, S. Rayhan Joty, C. Xiong, Y. Zhou, and S. Yavuz, "SFR-Embedding-Mistral: Enhance Text Retrieval with Transfer Learning," Feb. 2024. [Online]. Available: https://blog.salesforceairesearch.com/sfr-embedded-mistral/ [Page 22.]

[47] OpenAI, "New and improved embedding model," Dec. 2024. [Online]. Available: https://openai.com/index/new-and-improved-embedding-model [Page 22.]

[48] ——, "New embedding models and API updates," Jan. 2024. [Online]. Available: https://openai.com/index/new-embedding-models-and-api-updates [Page 22.]

[49] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M.-W. Chang, "REALM: Retrieval-Augmented Language Model Pre-Training," Feb. 2020, arXiv:2002.08909 [cs]. [Online]. Available: http://arxiv.org/abs/2002.08909 [Page 23.]

[50] D. S. Sachan, M. Patwary, M. Shoeybi, N. Kant, W. Ping, W. L. Hamilton, and B. Catanzaro, "End-to-End Training of Neural Retrievers for Open-Domain Question Answering," Jun. 2021, arXiv:2101.00408 [cs]. [Online]. Available: http://arxiv.org/abs/2101.00408 [Page 24.]

[51] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*, third edition, global edition ed., ser. Prentice Hall series in artificial intelligence. Boston Columbus Indianapolis New York San Francisco Upper Saddle River Amsterdam Cape Town Dubai London Madrid Milan Munich Paris Montreal Toronto Delhi Mexico City Sao Paulo Sydney Hong Kong Seoul Singapore Taipei Tokyo: Pearson, 2016. [Page 24.]

[52] J. Nielsen, "Chapter 2 - What Is Usability?" in *Usability Engineering*, J. Nielsen, Ed. San Diego: Morgan Kaufmann, Jan. 1993, pp. 23–48. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B978008052029250005X [Page 24.]

[53] J. Lewis and J. R., "IBM Computer Usability Satisfaction Questionnaires: Psychometric Evaluation and Instructions for Use," *International Journal of Human-Computer Interaction*, vol. 7, p. 57, Feb. 1995. [Page 24.]

[54] T. Bickmore and J. Cassell, "Social Dialogue with Embodied Conversational Agents," Jan. 2005. [Page 24.]

[55] F. D. Davis, "Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology," *MIS Quarterly*, vol. 13, no. 3, pp. 319–340, 1989, publisher: Management Information Systems Research Center, University of Minnesota. [Online]. Available: https://www.jstor.org/stable/249008 [Page 25.]

[56] A. Bhattacherjee, "Understanding Information Systems Continuance: An Expectation-Confirmation Model," *MIS Quarterly*, vol. 25, pp. 351–370, Sep. 2001. [Page 25.]

Appendix A

Supporting materials

Appendix B

Something Extra

TRITA-EECS-EX- 2023:0000

€€€€ For DIVA €€€€

{
"Author1": { "Last name": "Kristoffersson",
"First name": "Ludwig",
"Local User Id": "u100001",
"E-mail": "ludwigkr@kth.se",
"organisation": {"L1": "School of Electrical Engineering and Computer Science",
}
},
"Cycle": "2",
"Course code": "DA231X",
"Credits": "30.0",
"Degree1": {"Educational program": "Master's Programme, Computer Science, 120 credits"
,"programcode": "TCSCM"
,"Degree": "Masters degree"
,"subjectArea": "Technology"
},
"Title": {
"Main title": "Evaluating retrieval and summarisation performance of AI-Assistants built with Large Language Models and RAG-techniques (Retrieval Augmented Generation) in the domain of a LMS (Learning Management System)",
"Subtitle": "A subtitle in the language of the thesis",
"Language": "eng" },
"Alternative title": {
"Main title": "Detta är den svenska översättningen av titeln",
"Subtitle": "Detta är den svenska översättningen av undertiteln",
"Language": "swe"
},
"Supervisor1": { "Last name": "Welle",
"First name": "Michael",
"Local User Id": "u100003",
"E-mail": "mwelle@kth.se",
"organisation": {"L1": "School of Electrical Engineering and Computer Science",
"L2": "COLLABORATIVE AUTONOMOUS SYSTEMS DIVISION OF ROBOTICS, PERCEPTION AND LEARNING" }
},
"Supervisor2": { "Last name": "Enoksson",
"First name": "Fredrik",
"Local User Id": "u100003",
"E-mail": "fen@kth.se",
"organisation": {"L1": "",
"L2": "UNIT OF DIGITAL LEARNING" }
},
"Examiner1": { "Last name": "Jensfelt",
"First name": "Danica",
"Local User Id": "u1d13i2c",
"E-mail": "danik@kth.se",
"organisation": {"L1": "School of Electrical Engineering and Computer Science",
"L2": "COLLABORATIVE AUTONOMOUS SYSTEMS DIVISION OF ROBOTICS, PERCEPTION AND LEARNING" }
},
"Cooperation": { "Partner_name": "KTH IT"},
"National Subject Categories": "10201, 10206",
"Other information": {"Year": "2024", "Number of pages": "1,53"},
"Copyrightleft": "copyright",
"Series": { "Title of series": "TRITA-EECS-EX" , "No. in series": "2023:0000" },
"Opponents": { "Name": "A. B. Normal & A. X. E. Normalè"},
"Presentation": { "Date": "2022-03-15 13:00"
,"Language":"eng"
,"Room": "via Zoom https://kth-se.zoom.us/j/ddddddddddd"
,"Address": "Isafjordsgatan 22 (Kistagången 16)"
,"City": "Stockholm" },
"Number of lang instances": "2",
"Abstract[eng]": €€€€
€€€€,
"Keywords[eng]": €€€€
Canvas Learning Management System, Docker containers, Performance tuning
€€€€,
"Abstract[swe]": €€€€
€€€€,
"Keywords[swe]": €€€€
Canvas Lärplattform, Dockerbehållare, Prestandajustering €€€€,
}

acronyms.tex

```
%%% Local Variables:
%%% mode: latex
%%% TeX-master: t
%%% End:
% The following command is used with glossaries-extra
\setabbreviationstyle[acronym]{long-short}
% The form of the entries in this file is \newacronym{label}{acronym}{phrase}
%                                      or \newacronym[options]{label}{acronym}{phrase}
% see "User Manual for glossaries.sty" for the  details about the options, one example is shown below
% note the specification of the long form plural in the line below
\newacronym[longplural={Debugging Information Entities}]{DIE}{DIE}{Debugging Information Entity}
%
% The following example also uses options
\newacronym[shortplural={OSes}, firstplural={operating systems (OSes)}]{OS}{OS}{operating system}

% note the use of a non-breaking dash in long text for the following acronym

\newacronym{KTH}{KTH}{KTH Royal Institute of Technology}

\newacronym{LMS}{LMS}{Learning Manegement System}
\newacronym{RAG}{RAG}{Retrieval Augmented Generation}
\newacronym{LLM}{LLM}{Large Language Models}
\newacronym{RNN}{RNN}{Recurrent Neural Network}
\newacronym{CNN}{CNN}{Convolutional Neural Networks}
\newacronym{LSTM}{LSTM}{Long Short-Term Memory}
\newacronym{GRU}{GRU}{Gated Recurrent Units}
\newacronym{BERT}{BERT}{Bidirectional Encoder Representations from Transformers}
\newacronym{GAN}{GAN}{Generative Adversarial Network}
\newacronym{NLP}{NLP}{Natural Language Processing}
\newacronym{GPT}{GPT}{Generative Pre-trained Transformers}
\newacronym{GQA}{GQA}{Grouped-Query Attention}
\newacronym{SWA}{SWA}{Sliding window attention}
\newacronym{SMoE}{SMoE}{Sparse Mixture of Experts}
\newacronym{IR}{IR}{Information Retrieval}
\newacronym{TF-IDF}{TF-IDF}{Term Frequency-Inverse Document Frequency}
\newacronym{CBOW}{CBOW}{Continuous Bag-of-Words}
\newacronym{MTEB}{MTEB}{Massive Text Embedding Benchmark}
\newacronym{seq2seq}{seq2seq}{Sequence-to-sequence}
\newacronym{TAM}{TAM}{Technology Acceptance Model}
\newacronym{ECM}{ECM}{Expectation-Confirmation Model}
\newacronym{RLHF}{RLHF}{Reinforcement learning from human feedback}
\newacronym{MMLU}{MMLU}{Massive Multitask Language Understanding}
\newacronym{GUI}{GUI}{Graphical user interface}
```