

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/381898598>

GPT-4o: The Cutting-Edge Advancement in Multimodal LLM

Preprint · July 2024

DOI: 10.36227/techriv.171986596.65533294/v1

CITATIONS

7

READS

2,396

2 authors:



Raisa Islam

New Mexico Institute of Mining and Technology

7 PUBLICATIONS 48 CITATIONS

[SEE PROFILE](#)



Owana Marzia Moushi

University of Nebraska-Lincoln

8 PUBLICATIONS 16 CITATIONS

[SEE PROFILE](#)

GPT-4o: The Cutting-Edge Advancement in Multimodal LLM

Raisa Islam

Computer Science

New Mexico Institute of Mining and Technology
Socorro, NM, USA
raisila.islam@student.nmt.edu

Owana Marzia Moushi

Electrical and Computer Engineering
University of Nebraska - Lincoln
Omaha, NE, USA
omoushi2@huskers.unl.edu

Abstract—GPT-4o marks a significant advancement in AI technology, enhancing multimodal capabilities. OpenAI has launched several GPT models over the years, with GPT-4o being the latest. This paper provides a concise overview of these models, focusing on their key features and technological advancements. The main objective is to present a brief overview of GPT-4o, including its technological innovations. GPT-4o offers substantial improvements over its predecessors by introducing multimodal capabilities, larger context windows, efficient tokenization, and faster processing speeds, achieving state-of-the-art performance in text, audio, video, and image generation and understanding. We have compared GPT-4o with ten top LLMs using metrics such as throughput, response time, and latency, where GPT-4o demonstrated clear superiority. Additionally, this paper explores various application domains, highlighting GPT-4o's versatility and potential to modernize multiple aspects of human life.

Index Terms—GPT-4o, LLM, OpenAI, AI, ChatGPT, multimodal

I. INTRODUCTION

Artificial Intelligence (AI) is one of the most popular cutting-edge technology in this era which is gaining attention because of its revolutionizing aspects in our life [1]. Natural Language Processing (NLP), a subset of AI, focuses on the interaction between computers and human languages, aiming to enable machines to understand, interpret, and generate human language meaningfully. The advent of Large Language Models (LLMs) has initiated a paradigm shift and revolutionized the field of NLP by leveraging deep learning techniques to process and generate natural language with unprecedented accuracy and fluency [2]. These models are trained on vast amounts of textual data, allowing them to capture intricate linguistic patterns and contextual nuances. As a result, LLMs have significantly enhanced applications in various domains, including chatbots, language translation, sentiment analysis, and content creation, driving forward the capabilities of AI in understanding and utilizing human language [3] [4].

OpenAI's Generative Pre-trained Transformer (GPT) models have fundamentally transformed the fields of AI and NLP over the past ten years. GPT models are built upon the transformer architecture, which has demonstrated exceptional effectiveness, particularly in developing LLMs. The first version, GPT-1, was launched in 2017. Subsequent versions, including GPT-2 and GPT-3, culminated in the groundbreaking ChatGPT.

Since its release in November 2022, ChatGPT has gained immense popularity, reaching 1 million users within a week and 100 million users within two months [5]. ChatGPT aims to simplify various tasks, such as coding, math solving, and problem-solving, by using a combination of pre-trained unsupervised models and fine-tuned supervised models to provide human-like responses [3]. However, the last version, GPT-4, had limitations in video chatting, image processing, and audio processing. To address these shortcomings and advance toward a more digital world, OpenAI has released a new version of ChatGPT, called GPT-4o.

The recent version of ChatGPT is based on GPT-4o architecture which is gaining success on the previous chatbots. It was released on May 13, 2024, called GPT-4omni (or GPT-4o) the latest multimodal LLM from OpenAI [6]. The term "omni," derived from the Latin word "omnis," meaning "all" or "every," highlights the model's omni-modal capabilities [7]. GPT-4o can process and understand multimodal inputs, including text, images, audio, and video, making it a significant advancement in AI technology. It is the first LLM capable of interpreting emotions from videos, enhancing user experience by analyzing various data types. Although the model is already available for use, OpenAI continues to improve its capabilities, with updates expected soon [8]. The model's efficiency in handling different types of data results in high success rates and reduced processing times and costs [9]. For users seeking unlimited access and enhanced features, OpenAI offers a paid version, such as ChatGPT Plus, which includes a higher message limit and upcoming macOS desktop app support [10] [11]. GPT-4o is twice as fast as GPT-4 Turbo (released after GPT-4) and represents a milestone in the digital interaction between machines and humans, promising to significantly impact the future of digital communication [7] [12] [13].

Tech giant Apple and OpenAI are collaborating to integrate ChatGPT into Apple's ecosystem, focusing on the upcoming iOS 18 operating system [14]. This partnership aims to enhance the user experience by embedding ChatGPT's advanced language capabilities into Apple services like Siri, making them more intuitive and responsive. Features include real-time conversation synchronization, and voice input through Whisper. These enhancements will allow Siri to process

	GPT-4o	GPT-4	GPT-3.5	GPT-3
Initial Release Date	May 13, 2024	March 14, 2023	March 15, 2022	June 11, 2020
Modality	Audio, video, text, images	Text and images	Text	Text
Context window [2] [15]	128000	8192	4096	2048
Parameters [2] [16] [17]	Yet-to-Disclose (YTD)	1.76 trillion	175 billion	175 billion
Cost per 1M token [18] [19]	Input: \$5 Output: \$15	Input: \$30 Output: \$60	Input: \$1.5 Output: \$2	\$0.4 to \$20
Decoder layers [16] [17] [20]	YTD	120	96	96
Prompting Method [17] [21]	YTD	Chain-of-Thought, n -shot ¹	n -shot	n -shot
Training Data [17] [22]	YTD	~ 13T tokens which includes both text data and code data	Over 570GB of text data	About 45TB text data
Response Time [23] [24]	30% faster than GPT-4	94ms per token	35ms per token	Unknown
Performance	State-of-the-art performance	Advanced in handling complex tasks, slower response time than GPT-4o	Improved version of GPT-3, lacks depth of understanding	Poor on complex tasks

TABLE I: Summary of latest GPT models

complex queries more effectively and provide functionalities such as instant answers, tailored advice, creative inspiration, professional input, and educational support. This integration aligns with Apple's broader strategy to incorporate AI-driven features while ensuring privacy by powering most features on-device.

A. Evolution of GPT

Table I provides an overview of the latest GPT models to understand the improvement OpenAI is making for the updated GPT models. GPT-4o significantly enhances the capabilities of its predecessors by incorporating multimodal functionalities, including text, audio, image, and video processing. It offers faster response times and reduced costs compared to GPT-4, while maintaining high accuracy and performance, surpassing the improvements seen in GPT-3.5 and GPT-3.

Contribution: The main contributions of this paper are as follows:

- We have provided an in-depth overview of GPT-4o, detailing the technologies that contribute to its advancement over previous models. This study also summarizes the key features and advantages of GPT-4o, highlighting the significant improvements it offers compared to its predecessors.
- We have analyzed the text and vision performance comparisons provided by the OpenAI team. Additionally, a comparison of GPT-4o with other leading LLMs using metrics such as throughput, response time, and latency has been discussed. The results demonstrate the superior performance of GPT-4o.
- Furthermore, we have presented the various fields where GPT-4o can be applied, showcasing its versatility and potential impact.

The structure of this paper is as follows: Section II provides an overview of the newly launched GPT-4o, including its technologies, features, advantages, and challenges. Section III analyzes the performance of GPT-4o by comparing it with popular LLM models, and Section IV explores the application domains where GPT-4o can be utilized. Finally, Section V concludes the paper.

¹the value of n starts from 0

II. GPT-4O OVERVIEW

This section provides an overview of GPT-4o, detailing the technologies employed in its development, its features, and its advantages. While GPT-4o incorporates numerous updated technologies, it also faces some challenges that need to be addressed. These aspects, including both the strengths and areas for improvement, are comprehensively discussed here.

A. Technology

GPT-4o builds upon the architecture of its predecessors, integrating enhancements in context window size, tokenization efficiency, and multimodal capabilities, which include processing text, audio, video, and images. This single model is trained end-to-end across text, vision, and audio, ensuring all inputs and outputs are managed by the same neural network [8]. It employs refined reinforcement learning with human feedback (RLHF) [25], significantly improving its alignment with human values and ethical standards. GPT-4o also features an advanced transformer architecture with enhanced self-attention mechanisms, allowing for better comprehension and generation of nuanced, contextually relevant responses. Utilizing Nvidia's most advanced GPUs [26], known for their parallel processing capabilities, GPT-4o efficiently handles the massive computations required by its sophisticated architecture. This model gives better performance for scalability since it uses larger training sessions [25]. GPT-4o's rate limits are 5x higher than GPT-4 Turbo—up to 10 million tokens per minute. These advancements make GPT-4o a versatile tool for applications ranging from conversational AI to content generation and data analysis. The following subsection discusses the technologies that are being used in GPT-4o.

1) o200k_base Tokenizer: GPT-4o introduced a new *o200k_base* tokenizer algorithm, marking a shift from the *cl100k_base* tokenizer used by GPT-4, GPT-4 Turbo, and GPT-3.5 Turbo². Tokenization, which breaks down text into smaller units called tokens, is critical in NLP. The *o200k_base* tokenizer improves upon previous methods by being faster and more efficient, allowing GPT-4o to process and generate language at unprecedented speeds. It enhances semantic coherence in generated text and improves the handling of multiple languages [27], expanding GPT-4o's applicability across various linguistic contexts.

²<https://github.com/openai/tiktoken>

2) *RAG-GPT*: RAG-GPT³ is an advanced implementation of Retrieval-Augmented Generation (RAG) technology, designed to enhance the capabilities of LLMs by integrating them with efficient document retrieval systems. This integration allows RAG-GPT to provide more accurate and contextually relevant responses by fetching and incorporating information from extensive knowledge bases during query processing. Integration of RAG-GPT in GPT-4o enables delivering precise and grounded answers, making it particularly useful for applications requiring detailed and up-to-date information. This approach not only improves the quality of generated content but also enables the handling of specialized and complex queries with greater reliability.

3) *Context Window*: GPT-4o features an impressive context window size of 128k tokens [6]. The larger context window allows the model to maintain and process a much larger amount of information within a single interaction, enhancing its ability to understand and generate responses that are contextually relevant over extended conversations. This large context window is particularly beneficial for complex tasks requiring extensive context retention, tracking multiple threads of a conversation, and integration of multiple pieces of information.

4) *Cloud Infrastructure and API Access*: GPT-4o is designed to be deployed on scalable cloud infrastructures (Microsoft Azure) [28] [29], offering flexible API access for diverse applications. This cloud-based deployment model allows for seamless scaling to meet varying user demands and workloads.

B. Features

OpenAI is improving its models to make the world digital by adding some additional features which are discussed below:

1) *Multimodality*: GPT-4o is the newest multimodal LLM. It can understand and generate spoken languages, recognizing speech to transcribe text and using text-to-speech to generate speech [8]. Its vision capabilities allow it to interpret and generate visual content, such as recognizing images, generating new images, and solving problems by analyzing uploaded images. In text interaction, GPT-4o excels in NLP, enabling it to write essays, answer questions, provide summaries, and create stories or poetry based on user input.

2) *Enhanced Interaction*: Users can interact with ChatGPT more dynamically, interrupting and receiving responses in real-time. The model can detect nuances in users' emotions and respond in various emotive tones, making conversations more natural and engaging [30]. Additionally, it can handle interactive Q&A sessions which require extensive knowledge [6].

3) *Data Analysis*: Users can create interactive tables and charts from uploaded data in a variety of file formats. ChatGPT automatically generates an interactive table view, enabling users to scroll through all rows and columns. Users can create interactive charts by specifying chart type or automated selection, customize the graphics of these charts, and generate summaries to explain their findings [31].

³<https://github.com/gpt-open/rag-gpt>

4) *Multilingual*: GPT-4o is capable of responding to 50 different non-English languages [6] [12]. This proficiency ensures effective communication and content generation for a global audience, making it a valuable tool for diverse linguistic applications.

5) *Memory*: ChatGPT introduced memory capabilities to save users from repeating information across conversations and to enhance future interactions. Users have control over the memory; they can instruct it to remember or forget specific details, inquire about what it remembers, and manage these settings or disable memory entirely [32].

C. Advantages

The following section explains the important advantages that GPT-4o has made over the previous chatbots.

1) *Faster Response*: GPT-4o can respond to audio inputs in as little as 232 milliseconds, with an average of 320 milliseconds, which is similar to human response time in a conversation [8].

2) *Cost-effective*: This GPT-4o model is cost-effective since we are using one model for various data types of input. This model will help analyze the video, security purposes, sports, and also for content analysis [13]. Most importantly, this chatbot has around 50% reduced processing cost than the previous model GPT-4 Turbo [27].

3) *Safe and Reliable*: GPT-4o is safe and reliable compared to the previous model. This model is based on the feedback of humans with a backbone of reinforcement learning which makes it more reliable. They also reduce the amount of misleading content generation by using the feedback [25]

D. Challenges

Despite the numerous advancements of GPT-4o, it still has noteworthy limitations. The outage on June 4th, 2024, caused by a major system issue, affected all users of ChatGPT-related services and lasted several hours⁶. The exact technical causes were not detailed, but such incidents typically involve a mix of software bugs, infrastructure issues, or configuration errors. Furthermore, GPT-4o's audio models are limited to preset voices, and the model's pronunciation or explanations can sometimes be incorrect [8]. Additionally, data breaches remain a significant concern in the digital world, emphasizing the need to protect user data and comply with data protection regulations to maintain trust and legal compliance. Ensuring responsible AI use, avoiding biases, and adhering to ethical practices are crucial considerations [29].

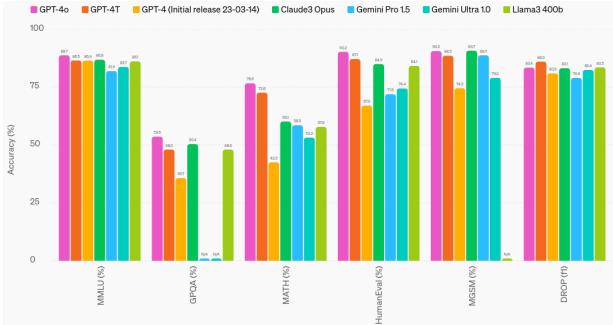
III. PERFORMANCE EVALUATION

OpenAI [8] provided a performance comparison between its models and their counterparts. Figure 1a illustrates the text evaluation results, showing that GPT-4o achieved state-of-the-art (SOTA) performance across the MMLU (88.7%),

⁵All vision evals are 0-shot, with MMMU, MathVista, and ChartQA as 0-shot CoT

⁶recommended to check original images for better view

⁶<https://status.openai.com/>



(a) Text Evaluation

Eval Sets	GPT-4o	GPT-4T 2024-04-09	Gemini 1.0 Ultra	Gemini 1.5 Pro	Claude Opus
MMMU (%) (val)	69.1	63.1	59.4	58.5	59.4
MathVista (%) (testmin)	63.8	58.1	53.0	52.1	50.5
AI2D (%) (test)	94.2	89.4	79.5	80.3	88.1
ChartQA (%) (test)	85.7	78.1	80.8	81.3	80.8
DocVQA (%) (test)	92.8	87.2	90.9	86.5	89.3
ActivityNet (%) (test)	61.9	59.5	52.2	56.7	
EgoSchema (%) (test)	72.2	63.9	61.5	63.2	

(b) Vision Understanding Evals⁴Fig. 1: Performance comparison provided by OpenAI [8]⁵

GPQA (53.6%), Math (76.6%), and HumanEval (90.2%) benchmarks. For the MGSM benchmark, Claude 3 Opus performs best (90.7%), followed by GPT-4o (90.5%). For the DROP benchmark, GPT-4T performed the best (86.0), followed by Gemini Ultra 1.0 (83.5) and GPT-4o (83.4). Figure 1b presents a performance comparison across various vision evaluation sets. Similar to the text evaluation set, GPT-4o consistently outperforms other models in most metrics, achieving SOTA in MMLU (69.1%), MathVista (63.8%), AI2D (94.2%), ChartQA (85.7%), DocVQA (92.8%), ActivityNet (61.9%), and EgoSchema (72.2%). GPT-4T follows closely but slightly lags behind GPT-4o. For the selected evaluation sets, it is clear from Figure 1 that GPT-4o demonstrates strong capabilities in these tasks, while Gemini 1.0 Ultra, Gemini 1.5 Pro, and Claude Opus show competitive but generally lower performance, with some exceptions in specific tasks.

OpenAI offers an evaluation framework called Evals⁷ on GitHub. This framework provides tools for assessing LLMs and includes an open-source registry of benchmarks. It enables users to create and run evaluations using datasets to generate prompts, assess the quality of model outputs, and compare performance across various datasets and models.

GPT-4o	GPT-4T
Claude 3 Opus	Claude 3 Haiku
Gemini 1.5 Pro	Gemini 1.5 Flash
Mixtral 8x22B	Mixtral 8x7B
DBRX	Command-R+

TABLE II: LLMs for performance comparison

Artificial Analysis [33] provides tools to compare selected LLMs by calling APIs and generating comparison graphs. To further analyze the performance of the GPT-4o model, we selected 10 popular LLMs (listed in Table II) using these tools. Figure 2 illustrates a relative performance analysis of the listed LLMs based on throughput, response time, and latency for a single query and a 10k token prompt length. These graphs collectively highlight the strengths and weaknesses of each model.

⁷<https://github.com/openai/evals>⁸Generated from <https://artificialanalysis.ai/models/gpt-4o/prompt-options/single/long#performance>

Figure 2a shows the latency comparison, defined as the time to the first token received (in seconds) after an API request is sent. Mixtral 8x7B and Mixtral 8x22B exhibit the lowest latencies, at 0.59 and 0.81 seconds, respectively. GPT-4o has a lower-than-average latency, receiving the first token in 1.04 seconds.

Figure 2b depicts throughput, measured in tokens per second received while the model generates tokens (i.e. after the first chunk has been received from the API). Gemini 1.5 Flash and Claude 3 Haiku have the highest throughput, scoring 89 each, followed by Mixtral 8x7B with a score of 73. GPT-4o ranks fourth, with a throughput of 64 tokens per second, outperforming the average.

Finally, Figure 2c evaluates the total response time, defined as the time to receive a 100-token response, estimated based on latency and throughput. Mixtral 8x7B and Claude 3 Haiku have the fastest response times, at 2.2 and 2.3 seconds, respectively. GPT-4o has a response time of 2.7 seconds.

Overall, the analysis in Figure 2 indicates that GPT-4o performs better than most models but lags behind Mixtral 8x7B and Claude 3 Haiku.

IV. APPLICATION DOMAIN

GPT-4o has the potential to significantly modernize and digitize the world by simplifying various aspects of human life. Its versatile applications span numerous fields, as detailed below, showcasing its substantial impact on a wide range of sectors.

a) Education: GPT-4o can guide students step-by-step to solve any math problems omitting the necessity of extra tutor [8] [30]. Additionally, it can be utilized in academic research by generating summaries of research papers, suggesting research topics, and providing insights from large datasets. These capabilities accelerate the research process and contribute to academic advancements by enabling researchers to quickly understand and explore extensive amounts of information.

b) Medical: GPT-4o can analyze medical images and patient data, aiding doctors in diagnosing diseases more accurately and quickly [29]. It enhances patient interaction by handling inquiries, providing information on medical conditions,

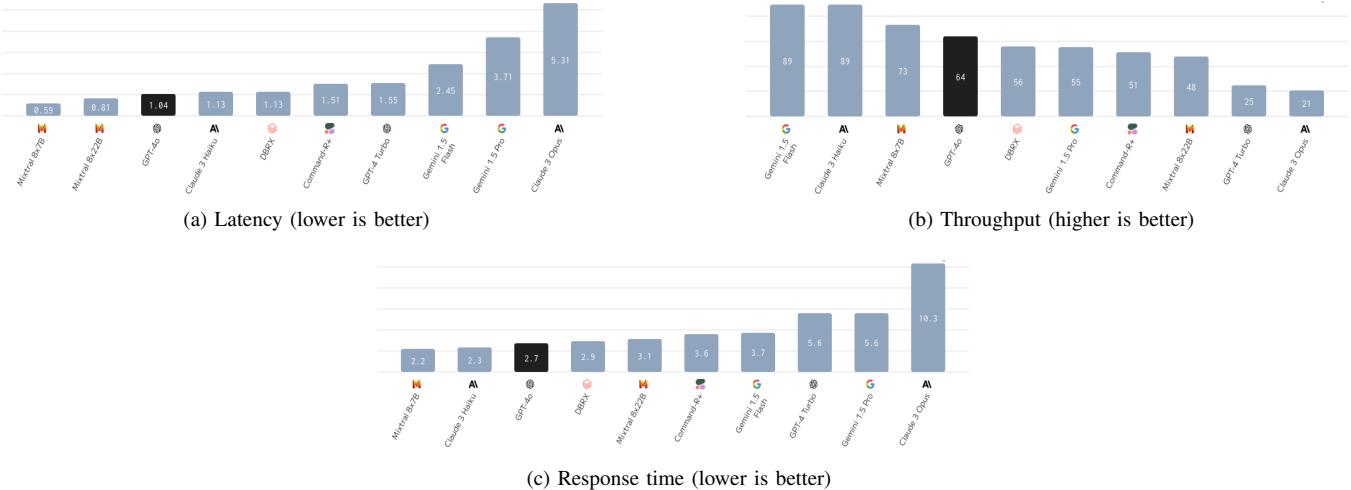


Fig. 2: Performance comparison⁸

and managing appointment scheduling through chatbots. It can also support learners with visual impairments by offering both speech-to-text and text-to-speech solutions [34].

c) *Customer Support*: GPT-4o can be tailored to specific business needs, enabling AI-powered chatbots to handle banking queries, transaction details, and account management 24/7, reducing the need for human intervention [29]. It supports advanced virtual assistants capable of effective responses, including visual data [25]. With real-time emotion analysis and multilingual capabilities, GPT-4o enhances global reach and customer satisfaction.

d) *Finance*: GPT-4o can analyze financial data and forecast market trends, enabling institutions to manage risks and make informed investment decisions [29].

e) *Content Generation*: GPT-4o can be used for creative and analytical purposes, i.e., making posters, creating visual representations from text input, designing characters, and producing outputs in different styles. It can also design logos, create artwork of people or objects, print text in different fonts, and generate 3D images.

f) *Code Review*: GPT-4o, can also review code effectively. It can identify the appropriate notes and comments while analyzing the code [6] [35]. Moreover, if provided with a picture of a desktop displaying the code, GPT-4o can detect issues or problems within the code [12].

V. CONCLUSION

GPT-4o represents a significant advancement in the field of AI, offering substantial improvements over its predecessors in terms of multimodal capabilities, context window size, tokenization efficiency, and processing speed. By integrating advanced technologies like refined RLHF and utilizing powerful hardware such as Nvidia's GPUs, GPT-4o achieves remarkable performance in understanding and generating text, audio, video, and images. Its applications span various domains, including education, medicine, customer support, finance, and

content generation, showcasing its versatility and potential to modernize and digitize numerous aspects of human life.

However, despite these advancements, GPT-4o still faces challenges such as ensuring ethical AI use, protecting user data, and overcoming technical issues like system outages. Addressing these challenges will be crucial for maximizing the model's benefits and maintaining user trust.

REFERENCES

- [1] T. Wu, S. He, J. Liu, *et al.*, “A brief overview of chatgpt: The history, status quo and potential future development,” *IEEE/CAA Journal of Automatica Sinica*, vol. 10, no. 5, pp. 1122–1136, 2023. DOI: 10.1109/JAS.2023.123618.
- [2] Y. Xu, L. Hu, J. Zhao, Z. Qiu, Y. Ye, and H. Gu, *A survey on multilingual large language models: Corpora, alignment, and bias*, 2024. DOI: 10.48550/arXiv.2404.00929.
- [3] Y. K. Dwivedi, N. Kshetri, L. Hughes, *et al.*, ““So what if ChatGPT wrote it?” Multidisciplinary perspectives on opportunities, challenges and implications of generative conversational AI for research, practice and policy,” *International Journal of Information Management*, vol. 71, p. 102 642, 2023.
- [4] P. P. Ray, “ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope,” *Internet of Things and Cyber-Physical Systems*, 2023.
- [5] H.-W. Cheng, “Challenges and limitations of ChatGPT and artificial intelligence for scientific research: a perspective from organic materials,” *AI*, vol. 4, no. 2, pp. 401–405, 2023.
- [6] S. M. Kerner, *GPT-4o explained: Everything you need to know*, <https://www.techtarget.com/whatis/feature/GPT-4o-explained-Everything-you-need-to-know>, [Accessed 26-05-2024], May 2024.

- [7] GeeksforGeeks, *OpenAI Launches GPT-4o Omni*, <https://www.geeksforgeeks.org/openai-announces-gpt-4o-omni/>, [Accessed 25-05-2024], May 2024.
- [8] OpenAI, *Hello GPT-4o*, <https://openai.com/index/hello-gpt-4o/>, [Accessed 10-06-2024], May 2024.
- [9] K. Doyle, 'The "o" is for omni' and other things you should know about GPT-4o, <https://www.jasper.ai/blog/what-is-gpt-4o>, [Accessed 26-05-2024], May 2024.
- [10] R. Montti, *OpenAI Announces GPT-4o Omni*, <https://www.searchenginejournal.com/openai-announces-chatgpt-4o-omni/516189/>, [Accessed 25-05-2024], May 2024.
- [11] M. Zeff and Gizmodo, *OpenAI's new ChatGPT sounds more human than ever*, <https://qz.com/openai-new-chatgpt-gpt4-omni-voice-human-ai-1851475246>, [Accessed 26-05-2024], May 2024.
- [12] K. Wiggers, *OpenAI debuts GPT-4o 'omni' model now powering ChatGPT*, <https://techcrunch.com/2024/05/13/openais-newest-model-is-gpt-4o/>, [Accessed 26-05-2024], May 2024.
- [13] K. Gomez, *Harnessing the Power of GPT-4 Omni for Multimodal Processing: A Comprehensive Guide*, <https://medium.com/@kyeg/harnessing-the-power-of-gpt-4-omni-for-multimodal-processing-a-comprehensive-guide-9301ae001576>, [Accessed 24-05-2024], May 2024.
- [14] *Introducing Apple Intelligence, the personal intelligence system that puts powerful generative models at the core of iPhone, iPad, and Mac*, <https://www.apple.com/newsroom/2024/06/introducing-apple-intelligence-for-iphone-ipad-and-mac>, [Accessed 11-06-2024], Jun. 2024.
- [15] *Introducing GPT-4o: our fastest and most affordable flagship model*, <https://platform.openai.com/docs/models/gpt-4o>, [Accessed 06-06-2024], 2024.
- [16] M. Schreiner, *GPT-4 architecture, datasets, costs and more leaked*, <https://the-decoder.com/gpt-4-architecture-datasets-costs-and-more-leaked>, [Accessed 07-06-2024], Jul. 2023.
- [17] T. B. Brown, B. Mann, N. Ryder, et al., *Language models are few-shot learners*, 2020. arXiv: 2005.14165 [cs.CL].
- [18] OpenAI, *Pricing*, <https://openai.com/api/pricing>, [Accessed 06-06-2024], 2024.
- [19] D. Sheremetov and A. Bitkina, *OpenAI API Pricing 2024: Understanding GPT-3 Pricing In-Depth*, <https://onix-systems.com/blog/how-much-does-it-cost-to-use-gpt-models>, [Accessed 07-06-2024], Jun. 2023.
- [20] C. Lang, *ChatGPT's Architecture - Decoder Only? Or Encoder-Decoder?* <https://datascience.stackexchange.com/questions/118260/chatgpts-architecture-decoder-only-or-encoder-decoder>, [Accessed 07-06-2024], Dec. 2023.
- [21] OpenAI and J. A. et al., *GPT-4 Technical Report*, 2024. arXiv: 2303.08774 [cs.CL].
- [22] N. Bijani, *What are the differences between GPT, GPT3, GPT 3.5, GPT turbo GPT 4?* <https://www.codiste.com/what-difference-between-gpt-gpt3-gpt-3-5-gptturbo-gpt-4>, [Accessed 08-06-2024], Oct. 2023.
- [23] T. Pungas, *GPT-3.5 and GPT-4 response times*, https://www.taivo.ai/_gpt-3-5-and-gpt-4-response-times, [Accessed 09-06-2024], May 2023.
- [24] E. Eckert, *GPT-4 vs GPT-4o: The Ultimate AI Smackdown!* <https://supernormal.com/blog/gpt-4-vs-gpt-4o>, [Accessed 09-06-2024], Jun. 2024.
- [25] C. D. Under, *OpenAI GPT-4o: The Next Generation of Omni-Multimodal AI*, <https://medium.com/@cognidownunder/openai-gpt-4o-the-next-generation-of-omni-multimodal-ai-e8d64d211a2c>, [Accessed 26-05-2024], May 2024.
- [26] R. LeFebvre and J. Ledford, *OpenAI Reveals New GPT 4o: AI For Everyone*, <https://www.lifewire.com/openai-reveals-gpt-4o-8647637>, [Accessed 06-06-2024], May 2024.
- [27] M. Rajguru, *Exploring the New Frontier of AI: OpenAI's GPT-4o For Indic Languages*, <https://techcommunity.microsoft.com/t5/ai-azure-ai-services-blog/exploring-the-new-frontier-of-ai-openai-s-gpt-4-o-for-indic/ba-p/4142383>, [Accessed 25-05-2024], May 2024.
- [28] *Introducing GPT-4o: OpenAI's new flagship multimodal model now in preview on Azure*, <https://azure.microsoft.com/en-us/updates/new-openai-model-on-azure/>, [Accessed 06-06-2024], May 2024.
- [29] V. K. Upadhyay, *OpenAI : GPT-4o (Comprehensive Guide)*, <https://vivekupadhyay1.medium.com/openai-gpt-4o-comprehensive-guide-ccf15fd93870>, [Accessed 06-06-2024], May 2024.
- [30] A. Toolz, *15 Abilities of GPT4o that you Won't Believe*, <https://aitoolzai.medium.com/15-abilities-of-gpt4o-that-you-wont-believe-8cba07c1cf1f>, [Accessed 31-05-2024], May 2024.
- [31] OpenAI, *Data analysis with ChatGPT*, <https://help.openai.com/en/articles/8437071-data-analysis-with-chatgpt>, [Accessed 04-06-2024], Jun. 2024.
- [32] OpenAI, *Memory and new controls for ChatGPT*, <https://openai.com/index/memory-and-new-controls-for-chatgpt>, [Accessed 04-06-2024], Feb. 2024.
- [33] Artificial Analysis, <https://artificialanalysis.ai>, [Accessed 05-06-2024], 2024.
- [34] A. Kuzdeuov, O. Mukayev, S. Nurgaliyev, A. Kubolsyn, and H. A. Varol, "Chatgpt for visually impaired and blind," in *2024 International Conference on Artificial Intelligence in Information and Communication (ICAIIIC)*, 2024, pp. 722–727. DOI: 10.1109/ICAIIIC60209.2024.10463430.
- [35] D. Eastman, *Reviewing Code With GPT-4o, OpenAI's New 'Omni' LLM*, <https://thenewstack.io/reviewing-code-with-gpt-4o-openais-new-omni-lm/>, [Accessed 26-05-2024], May 2024.

Context, Language Modeling, and Multimodal Data in Finance

Sanjiv Das, Connor Goggins, John He, George Karypis, Sandeep Krishnamurthy, Mitali Mahajan, Nagpurnanand Prabhala, Dylan Slack, Rob van Dusen, Shenghua Yue, Sheng Zha, and Shuai Zheng

Sanjiv Das

is a professor of finance at Santa Clara University and an Amazon scholar at Amazon Web Services in Santa Clara, CA and Palo Alto, CA.

srdas@scu.edu

Connor Goggins

is a software development engineer at Amazon Web Services in Palo Alto, CA.

cggoggin@amazon.com

John He

is a software development engineer at Amazon Web Services in Palo Alto, CA.

hezhijia@amazon.com

George Karypis

is a professor of computer science at the University of Minnesota and senior principal scientist at Amazon Web Services in Palo Alto, CA.

gkarypis@amazon.com

Sandeep Krishnamurthy

is a software development manager at Amazon Web Services in Palo Alto, CA.

krsandee@amazon.com

Mitali Mahajan

is a graduate student at Santa Clara University in Santa Clara, CA.

mmahajan@scu.edu

Nagpurnanand Prabhala

is a professor of finance at Johns Hopkins University in Baltimore, MD.

prabhala@jhu.edu

Dylan Slack

is a graduate student at University of California in Irvine, CA.

dsslack@uci.edu

Rob van Dusen

is a graduate student at the University of Chicago in Chicago, IL.

rvanduse@chicagobooth.edu

Shenghua Yue

is a software development engineer at Amazon Web Services in Palo Alto, CA.

yuesheng@amazon.com

Sheng Zha

is a senior applied scientist at Amazon Web Services in New York, NY.

zhasheng@amazon.com

Shuai Zheng

is an applied scientist at Amazon Web Services in Palo Alto, CA.

shzheng@amazon.com

*All articles are now categorized by topics and subtopics. [View at PM-Research.com](#).

KEY FINDINGS

- Machine learning based on multimodal data provides meaningful improvement over models based on numerical data alone.
- Context-rich models perform better than context-free models.
- Pretrained language models that mix common text and financial text do better than those pretrained on financial text alone.

ABSTRACT

The authors enhance pretrained language models with Securities and Exchange Commission filings data to create better language representations for features used in a predictive model. Specifically, they train RoBERTa class models with additional financial regulatory text, which they denote as a class of RoBERTa-Fin models. Using different datasets, the authors assess whether there is material improvement over models that use only text-based numerical features (e.g., sentiment, readability, polarity), which is the traditional approach adopted in academia and practice. The RoBERTa-Fin models also outperform generic bidirectional encoder representations from transformers (BERT) class models that are not trained with financial text. The improvement in classification accuracy is material, suggesting that full text and context are important in classifying financial documents and that the benefits from the use of mixed data, (i.e., enhancing numerical tabular data with text) are feasible and fruitful in machine learning models in finance.

TOPICS

[**Quantitative methods, big data/machine learning, legal/regulatory/public policy, information providers/credit ratings***](#)

In recent papers, Araci (2019), Desola, Hanna, and Nonis (2019), and Yang, Uy, and Huang (2020) showed that improvements in language models using additional training with financial text resulted in better prediction of sentiment of news headlines (i.e., financial phrases). In this article, we assess whether language models improve on the simple use of text-based numerical features (e.g., sentiment, readability, positivity, negativity, riskiness, litigiousness; see Loughran and McDonald 2020). The existing practice in much of the finance literature is to create word-based features by applying

finance-specific dictionaries such as the Loughran–McDonald (LM) word lists.¹ Such word scoring leads to numerical features that are widely used in regression analysis of regulatory filings, tweets, news, and so on. These features have proven to be quite successful, and a vast literature has exploited them. This article shows that the recent developments in language modeling, such as bidirectional encoder representations from transformers (BERT), can extend these successes.

Text is more versatile, extensive, and multifaceted than mere tabular (numerical) data (Gentzkow, Kelly, and Taddy 2019). Furthermore, numerical time-series data may be less useful given the huge structural shifts in economies from recent major events such as trade wars and pandemics. Therefore, finance companies are exploring the advantages of using text, which is plentiful in finance and forward-looking. Examples of use cases are as follows:

- Credit analysis of firms using text in their Securities and Exchange Commission (SEC) filings and news. Additional text improves models based purely on tabular data. This is useful to financial institutions with a lending business, rating agencies, and managers of corporate bond funds who would like to use text to enhance their credit quality models for asset allocation (see Bodnaruk, Loughran, and McDonald 2015; Bonsall et al. 2017; and Ertugrul et al. 2017).
- Asset managers using text features for prediction and portfolio construction. Using text features enables companies to discover other similar companies (e.g., semantically similar firms). In the opposite case, semantic distance may be used for diversification of portfolios. Machine learning (ML) using big data was used by Routledge (2019).
- Classification models based on text are being used to predict corporate performance. Various textual features, such as sentiment, readability, tone, size, risk, and uncertainty, have been established as useful in rank ordering the future performance of firms, and these are table-stakes features that companies employ in text processing to make portfolio decisions.² Comparison of text across quarters and years offers useful predictive information, as shown by Cohen, Malloy, and Nguyen (2020).

These use cases are examples that have been shown to provide better performance than pure tabular data in several academic studies and industry white papers.³ Several papers also provide comprehensive surveys of textual analysis in finance, and an excellent recent survey is given by Loughran and McDonald (2020).⁴

Against this backdrop of the text mining literature in finance and accounting, we next discuss the research questions in this article.

RESEARCH QUESTIONS

An advantage of word-based scoring to create numerical variables is that the dimension of the feature set remains small and supports simple, explainable methods such as regression analysis. This is why it has been popular and practical among the financial community, both in academia and in practice. The addition of text poses

¹ See sraf.nd.edu/textual-analysis/resources/.

² See Antweiler and Frank (2004), Das and Chen (2007), Hoberg and Phillips (2016), and Bach et al. (2019).

³ For more examples, see Hafez et al. (2020a, 2020b), Cong, Liang, and Zhang (2019), Cong et al. (2019), and Chebonenko, Gu, and Muravjev (2018).

⁴ Other surveys are provided by Li (2010b), Das (2014), Kearney and Liu (2014), Loughran and McDonald (2016), and Gentzkow, Kelly, and Taddy (2019).

interesting technical problems, the two most salient being (1) how to combine text and numerical tabular data in one predictive model; and (2) language models such as BERT being easiest to apply to short text (i.e., a few sentences) and, in fact, working best at the single sentence level.

Adapting classifiers to handle long text is an important problem and has only recently been getting attention. Long documents are ubiquitous in finance, and regulatory filings are just one example—there are financial news reports, analyst reports, and so on, all of which run into hundreds of words. Therefore, using language models such as BERT as they are is unlikely to be effective given that input is restricted to maximum sequence lengths that mostly vary from 128 to 1,024 tokens and more often are on the lower side. This is good for news headlines and tweets but not good for reports and long documents, which have been the subject of suggested improved models (e.g., Liu et al. 2018 and Lee and Hsiang 2019 for patents, Wan et al. 2019 for legal documents, and Pappagari et al. 2019 for transcripts of earnings calls, all of which are widely used in natural language processing [NLP] applications).

Most recently, generating long sequences with sparse attention transformers also has been gaining traction (Child et al. 2019; Zaheer et al. 2020). Sparse transformers typically deal with documents with sequence lengths of several thousands. SEC filings, however, generally have more than 50,000 words, and this can be doubled after byte-pair encoding (BPE) tokenization. Some of the experiments here will deal with long documents, and we will try to address these issues using the simplest approaches that give good results. Given this, we consider the following research experiments in this article:

1. Using a benchmark dataset of financial news headlines (composed of one or two sentences), from the work of Malo et al. (2014) and Araci (2019), a comparison of word-based scoring and full language modeling determines how important context is in the classification of new sentiment. This is redone on another dataset as an additional experiment, which examines whether the full text in SEC 10-K/Q and 8K filings contains predictive information beyond that from word-based features. These experiments examine whether adding text to tabular data generally improves classification in the finance domain.
2. We conduct a comparison of general-purpose language models of the BERT class (BERT, RoBERTa, DistilBERT) versus a language model specifically trained on financial regulatory filings (which we denote RoBERTa-Fin) to assess whether pretrained financial language models add value to general purpose financial models. This has already been explored in work by Araci (2019), who undertook further pretraining of BERT using a subset of the Reuters TRC2 news articles dataset. Here, the article experiments with a model that is pretrained only on SEC 10-K/Q filings and another that is trained on SEC filings and Wiki text, both of which contain formal financial text. In this experiment, a RoBERTa model with dynamic masking and no next sentence prediction (NSP; used in BERT) is used in the pretraining scheme.
3. We assess how these models perform when predicting filing week returns, a classification exercise that is hard in efficient markets. Markets are by and large efficient, and time-series approaches to predicting stock movements have been rather unsuccessful, even at high frequency (Malkiel 2003). However, a growing literature has found that quarterly earnings and stock performance rankings are weakly predictable.⁵

⁵ See Antweiler and Frank (2004); Bodnaruk, Loughran, and McDonald (2015); Bonsall et al. (2017); Brown and Tucker (2011); Bushee, Gow, and Taylor (2018); Das and Chen (2007); Ertugrul et al. (2017); Hoberg and Phillips (2016); Jegadeesh and Wu (2013); Li (2010a); Li and Zhao (2014); Loughran, McDonald, and Yun (2009); Loughran and McDonald (2011, 2013, 2014, 2015); Price et al. (2012); and Tetlock, Saar-Tsechansky, and Macskassy (2008).

4. We conduct experiments to assess whether long documents can be classified accurately using document summaries instead. This is especially important because the signal in numerical variables may be swamped by noisy text in long documents. Using a summary may reduce the noise, but it also loses context, which can degrade classification. This interesting trade-off is explored briefly.

DATA

Standard Features

The SEC 10-K and 10-Q filings for all tickers in the S&P 500 index every quarter for the past 10 years (2010–2019) were downloaded. These data come from SEC’s open data platform, EDGAR, which has been shown to improve the informativeness of markets; see Gao and Huang (2020). This delivered over 20,000 documents. These documents were used to pretrain a RoBERTa-Fin language model.

As is standard practice in finance, one may wish to focus on a limited portion of the SEC 10-K/Q text (i.e., the management discussion and analysis (MD&A) section) because this section is where management provides a forward-looking analysis of the company’s financials. Therefore, this section was parsed out as well.

We also downloaded the financial phrase bank data prepared by Malo et al. (2014) to undertake experiments on financial news headlines to assess the value of context. This dataset contains three sentiment classes—negative, neutral, and positive—and thus offers a good test bed for comparing word-based NLP classification and language model–based classification in the finance domain.

Finally, to create another dataset, we downloaded some 10-K/Q and 8K forms for 645 tickers of companies that took Paycheck Protection Program (PPP) loans; see Balyuk, Prabhala, and Puri (2020). This resulted in 5,810 filings for 2020 Q1 and Q2, for which we examine filing week return performance.

Pretraining a Language Model

What are language models? Language models are collections of conditional and unconditional probabilities of words in sequences of words. Therefore, a language model is a probability distribution over sequences of words. More accurately, given a sequence of words, we can compute the probability of all those words occurring in that sequence. The key word here is *sequence*—this provides context that we assume to be important.

We use the 10-K/Q filings data for the past decade of S&P 500 tickers to pretrain a RoBERTa class model (Liu et al. 2019). For pretraining, the entire text of the 10-K/Q filing was used, not just the MD&A section, to ensure a broader context of financial language was captured. For classification, passing the original documents through a language model generates new word embeddings that reflect context-transformed data, which may then be passed into another neural network for the final classification task.

Pretrained BERT (Devlin et al. 2019) models and variants such as RoBERTa (Liu et al. 2019) have been proven to achieve state-of-the-art results on a wide variety of NLP tasks, including question answering (SQuAD v1.1), natural language inference, and others. The pretrained BERT models generate bidirectional representations for sentences; that is, they assign probabilities to forward and backward sequences of words. The representations can be used as additional features or can be finetuned in downstream NLP tasks.

The BERT model is pretrained with the book corpus and English Wikipedia. Araci (2019; FinBERT), Beltagy, Lo, and Cohan (2019; SciBERT), Lee et al. (2019; BioBERT), and Huang, Altosaar, and Ranganath (2019; ClinicalBERT) showed that using a BERT model pretrained with a domain-specific corpus may improve the accuracy of domain-specific NLP tasks. For this reason, we have trained two models, RoBERTa-Fin1 and RoBERTa-Fin2, the former using only the last decade's S&P 500 10-K/10-Q reports and the latter using both, as well as the last decade's filings and Wikipedia text. The original RoBERTa model is pretrained on Wikipedia, book corpus, openweb text, and stories, but for RoBERTa-Fin2 we only used Wiki text because we do not want the finance text to be overwhelmed by other data sources. In particular, we used dynamic masking and removed NSP loss in the pretraining scheme of RoBERTa. The methodology we describe here may be used to pretrain other language models in the finance domain using proprietary text as well.

To train these models with more than 100 million parameters, a number of graphics processing units (GPUs) were required. We used eight Amazon Web Services EC2 p3dn.24xlarge instances. Each instance has 92 vCPUs and eight Nvidia V100 Tensor core GPUs. As a result, we needed a distributed deep learning training framework to run the pretraining job using all GPUs on the eight instances. Pretraining data were distributed into 256 shards. Using dynamic masking, the training instances with random masked tokens were generated on the fly during pretraining. We selected Uber's Horovod framework⁶ with openMPI⁷ on top of Nvidia Collective Communications Library.⁸ A representation learning script from the Amazon Web Services GluonNLP⁹ library was used for pretraining. See Exhibit 1 for a schematic of the pretraining model. We trained two versions, base and large; the former takes between two and seven days of pretraining, and the total number of training iterations is 250,000 with a minibatch size of 2,048 and sequence length of 512.¹⁰

Summarization

One approach to handling long documents is to summarize the document first and then submit it to a classifier. This has potential, especially if the summary keeps signal and drops noise in the document. However, it may also lose context. Furthermore, the type of the summary may matter, and we will discuss these issues in light of the following approach to summarization, called a *Jaccard summarizer*. This summarizer is based on breaking the original large document into separate sentences. Once this is done, the Jaccard similarity¹¹ is computed for each pair of sentences. If there are n sentences, then the number of similarity computations is $n(n - 1)/2$. We sum the Jaccard values to get n total similarity scores. The summary is then the top sentences from a ranked list based on similarity scores. This summary distills the document to sentences that are most similar to all of the other sentences. We may further reweight the similarity scores based on special lexicons (word lists) and then

⁶ <https://github.com/horovod/horovod>.

⁷ <https://www.open-mpi.org/>.

⁸ <https://developer.nvidia.com/ncc>.

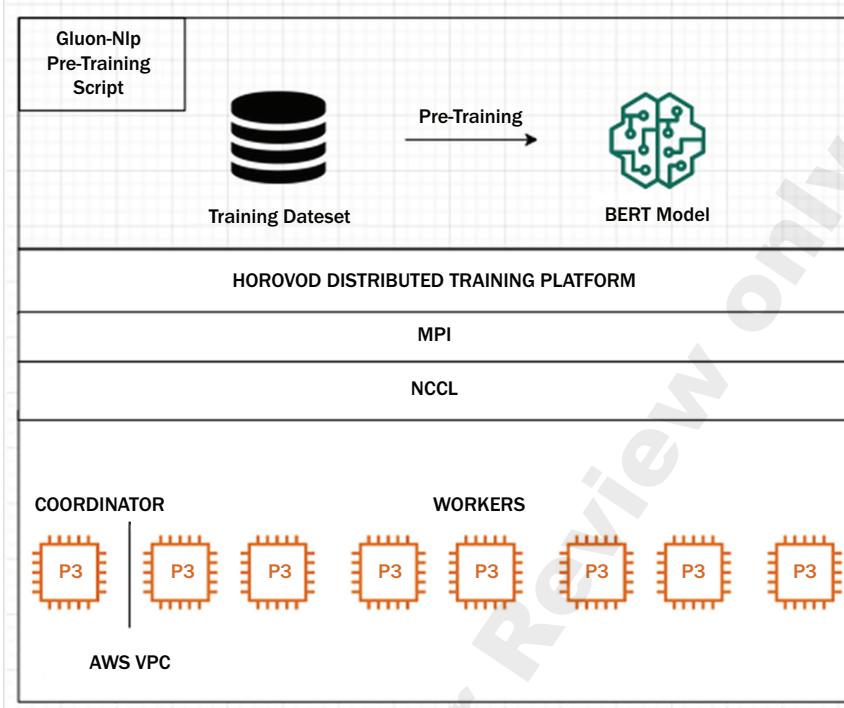
⁹ https://gluon-nlp.mxnet.io/examples/word_embedding/index.html.

¹⁰ The very first step in applying BPE encoding is to split the word or input into characters. We then use the merge operations learned during BPE training to combine pairs recursively. If there is no match in the merge operations, the input will remain in its split form, as a sequence of characters. For example, word embeddings may be split into subword tokens [em, ##bed, ##ding, ##s]. This is also one reason why it is hard to model the SEC filings; the number of tokenized tokens may be more than 100,000 (i.e., very large).

¹¹ The size of the intersection set of word tokens from each sentence divided by the size of the union set.

EXHIBIT 1

Schema for Pretraining RoBERTa-Fin Models



NOTES: The data type used was float16, batch size was 2,048, learning rate was 0.000625, number of training steps was 250,000, optimizer was LANS (Zheng et al. 2020), warm-up ratio was 125%, maximum sequence length was 512, and number of Horovod slots per instance was 8. We used a vocabulary size of 32,000. We trained two RoBERTa-Fin models, with embedding sizes of 768 and 1,024.

select top sentences based on weighted similarity scores. For example, we may weight sentences using financial word lists, specifically the litigious words. Or we may use a lexicon of commonly used words in equity markets to get a summary that is geared toward the equity analyst, or fixed-income lexicons for the bond analyst.

RESULTS AND FINDINGS

In this section, we present the various experiments that were undertaken and the results and implications therefrom.

How Much Does Language Matter?

This section evaluates the sentiment classification of financial news sentences using different approaches for comparison. There is a vast literature using the LM word lists to score text to convert it into numerical variables for regression analysis. Loughran and McDonald (2020) specifically presented a review of several papers that use text features such as readability, sentiment, polarity, positivity, negativity, risk/uncertainty, and litigiousness, which are converted into numerical scores. These effectively eliminate language from the model specification. To assess how much language matters, we use the financial phrase bank dataset from Malo et al.

(2014). quantify news headlines into numerical scores, and build an ML model to classify sentiment using only the numerical scores. We then enhance the model with text to see how much language matters. The dataset is especially appropriate for this experiment because the text in each headline is short and thus has minimal context, even though it exceeds that of the numerical scores.

We used 2,264 documents. The dataset has three categories of hand-tagged sentiment labels. These are imbalanced: 303 negative, 1,391 neutral, and 570 positive. We scored each news article for readability (Gunning-Fog index). Using the LM word lists, we also scored the percentage of litigious, risk/uncertainty, negative, and positive words. Finally, we scored polarity, which is the difference in positive and negative words divided by the sum of both. As is typical in ML, we created normalized features from these variables using min-max scaling. This created six features in the dataset, and we dropped the column of text containing the news headlines. Using word vectors from FastText,¹² we also created an alternative scoring mechanism with a different construction of word lists, comprising the following word roots: positive, negative, certainty, uncertainty, litigious, risk, safe, and fraud. These are also used separate from the LM word lists and result in slightly higher accuracy on the same classification task because of the additional features beyond those of LM.

To avoid making and justifying choices in the ML model, we decided to implement an open-source AutoML tool out of the box. This has the benefit of eliminating modeler bias, in which the choice of model may bias the outcome of the experiment. Another useful aspect of AutoML modeling is that the model usually searches for the best ensemble of individual ML models that give good results; thus, we try to find the best model given the numerical features first, so as to set a difficult baseline before we add full text to the analysis. We chose to use AutoGluon Tabular,¹³ an open-source package. This is easy to implement and replicate. It runs an ensemble of models, such as random forest, Gini-based and entropy-based extra trees classifiers, K-nearest neighbors, CatBoost, NN, and light GBM. It is also fast and has high performance (see Erickson et al. 2020). AG-Tabular took less than 20 seconds to fit the best model it could find. The accuracy on the test dataset with the LM word scores is 66% with a Matthews (1975) correlation coefficient (MCC) of 0.18 (this lies in the range of -1 to +1, where 0 means no classification ability and -1 means perfect error always). Detailed results are reported in Exhibit 2. In addition, when we used our new features based on fastText, we had a small improvement in accuracy to 71%, and each other metric showed similar improvement. We note that, although AutoGluon allows additional options, such as hyperparameter tuning and autostacking of models, we did not select these to prevent these decisions being interpreted as modeler bias.

With purely numerical features, the model achieved a 2/3 prediction accuracy on the test dataset.

We now include an additional column with the text containing the original financial sentences from the news. Once again, AG-Tabular was applied and took a runtime of under 30 seconds. AG-Tabular converts text columns into TFIDF embeddings that are combined with the numerical columns and then submitted to the classifier. It applied the same models to the data and achieved an 86% accuracy level on the test dataset with an MCC of 0.72, compared with an accuracy of 66% and MCC of 0.18 when only numerical features were used. The other statistics are reported in Exhibit 2.

It is clear that the metrics are good across all classes. Although we get over 2/3 accuracy with just numerical text attributes, we can see that using all of the text (full

¹² fasttext.cc/from Facebook.

¹³ autogluon.mxnet.io/.

EXHIBIT 2

Comparison of Classification Performance on Tabular Data and TabText for the Sentiment Classification Task on the Financial Phrase Bank

Label/Metric	Tabular Only		
	(LM)	(New)	TabText
Accuracy	0.658	0.711	0.861
MCC	0.186	0.447	0.723
Negative			
Precision	0.625	0.625	0.973
Recall	0.089	0.149	0.643
F1	0.156	0.241	0.774
Neutral			
Precision	0.670	0.754	0.854
Recall	0.966	0.918	0.989
F1	0.791	0.828	0.917
Positive			
Precision	0.481	0.598	0.838
Recall	0.121	0.563	0.626
F1	0.194	0.580	0.717

NOTES: We report accuracy, precision, recall, F1 scores, and MCC on the test (holdout) dataset. When we add text to the dataset (denoted as TabText), we see an appreciable improvement in the classification. The LM column uses numerical scores based on LM word lists, and the column New uses word lists based on our word vector model.

language) matters in classifying financial news in this dataset, and we do better than when only using the numerical text scores.

Classification Using SEC Forms 10-K, 10-Q, and 8-K

The experiments in the previous section have shown that using all of the text improves on using numerical NLP scores from text (e.g., readability, risk) in a classification setting. We move on to classification experiments on large amounts of text: SEC filings. We extracted all SEC 10-K, 10-Q, and 8-K forms for the first two quarters of 2020 for a list of 645 tickers. These tickers were based on those used by Balyuk, Prabhala, and Puri (2020) and are related to firms that borrowed money provided by the federal government in the CARES Act of 2020 under the provisions of the PPP. This is not entirely necessary for our analysis, and we may just as well have chosen any random set of tickers; however, this is an interesting set because these firms may have something in common given that they all partook in the PPP program. Our SEC form extractor took 45 minutes to download and create a dataframe of 5,801 forms, of which 530 are 10-Ks, 704 are 10-Qs, and 4,567 are 8-Ks.

We also downloaded the stock prices for all tickers from December 2019 through July 2020, so as to span 2020 Q1 and Q2 with a month on each side.

After converting these to returns, we lined up the returns for each filing from five days before the filing date to five days after the filing date (11 days). We did the same for the S&P 500 return, and we also computed the excess return of the ticker over the S&P 500 return for all 11 days. We created a new variable: the sum of daily excess returns from day 0 through day 5 (i.e., from the day of the filing and the five following days, roughly a week of post-filing performance). Using this new variable, we created a binary label, taking a value of 1 if the variable was positive and 0 otherwise, for each row of the dataset; the ratio of variables is 0.45:0.55, respectively—fairly well balanced.

First, we submitted the combination of the long text of the SEC filings and the day 5 through day 1 returns for the ticker and the S&P 500 to AutoGluon for classification of the binary label. We implemented an 80/20 train/test split for the dataset. We achieved an accuracy of 58.5% in predicting five-day cumulative returns using the filing, with an MCC of 0.14, which is not high, although it is reasonable given that predicting stock movements is difficult in efficient markets.

Second, using a subset of 440 tickers, some of these firms first took the PPP loans and then returned the money. The reasons for returning the money and the cost–benefit trade-off of keeping and returning the money were analyzed in detail by Balyuk, Prabhala, and Puri (2020). Reasons for returning the money may include avoiding regulatory oversight, preventing markets from penalizing stock prices because taking a PPP loan sends a negative signal, or simply realizing *ex post* that the firm did not need the funding and returning the money would send a positive signal to the markets with a concomitant boost to the stock price. Our initial extraction of all filings for the 440 tickers in 2020 Q1 and Q2 resulted in 4110 filings.

EXHIBIT 3

Predicting Filings by Companies That Returned PPP Money

Label/Metric	Tabular Only	TabText
Accuracy	0.83	0.89
MCC	0.25	0.57
Label 0		
Precision	0.83	0.88
Recall	0.99	0.99
F1	0.91	0.93
Label 1		
Precision	0.79	0.93
Recall	0.10	0.40
F1	0.19	0.56

NOTES: There were 4,100 original filings. Dropping post-return filings yielded 3,191 with label 0 and 720 with label 1, totaling 3,911. This means that a model that always returns label 0 will have an accuracy of 0.816. The tabular data only comprise cumulative returns up to the filing date, and the TabText data contain tabular data and the full text of the filing.

returned the money and those that did not, because the two groups appear to have different average returns before and after the filing dates in the cross section of firms. Given that past returns are helpful, we further explore whether the text of the filing improves the ability to classify the text of firms that returned PPP money and those that did not.

The prediction performance is reported in Exhibit 3. The baseline accuracy is 81.6%, the return-only model has an accuracy of 83%, and the returns plus full text model achieves 89% accuracy, suggesting that adding text to tabular data improves classifier performance. The TFIDF approach handled long text well for these filings and improved classification accuracy for mixed numerical tabular and text models.

Adding Context with Language Models?

We now explore whether language models that have context can do even better than models that do not. A classifier with n -gram features that are not cognizant of context offers a baseline. The fact that we achieved 2/3 accuracy with just six numerical features (Exhibit 2), however, suggests that the large literature on textual scoring using keywords and lexicons is well placed. However, we saw that exploiting all of the text using TFIDF makes a huge difference in classification accuracy out of sample (increasing it from 66% to 86%), so we know that adding text n -grams does not result in massive overfitting and improves model performance over mere NLP scoring. The question is, can we do even better with the addition of context from transformer-based (Vaswani et al. 2017; Devlin et al. 2019; Liu et al. 2019; Sanh et al. 2020), language models?

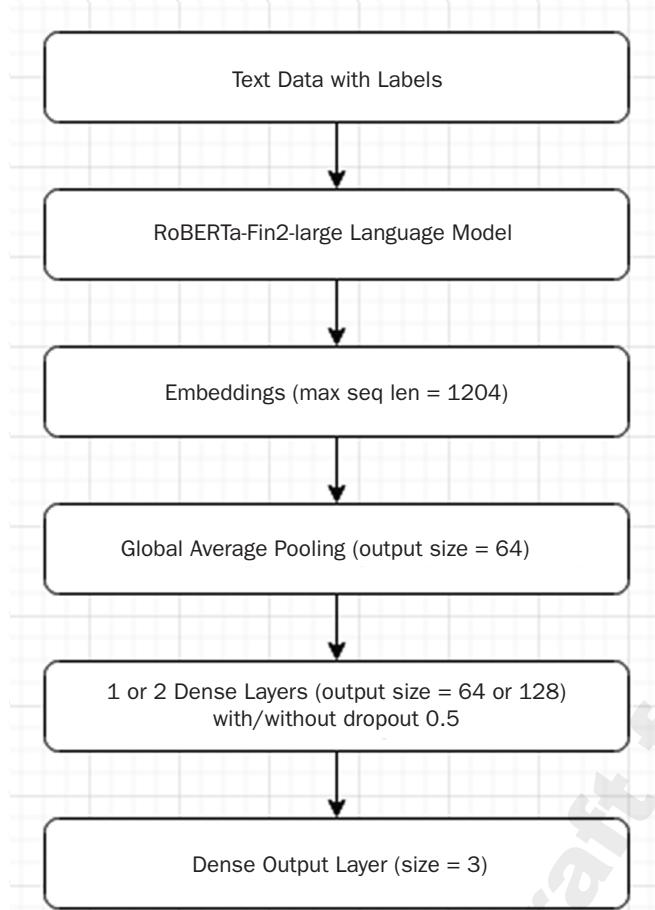
We assess different language models here, from the original BERT model to variants such as DistilBERT and RoBERTa. We also assess pretrained models, RoBERTa-Fin1, pretrained completely on the SEC 10-K/Q filings for S&P 500 companies for all quarters in the 2010–2019 period. We also pretrained a language model on Wikipedia text and all the text used in RoBERTa-Fin1; we denote this model as RoBERTa-Fin2. We will assess both base- and large-model variants.

For the subsample of 440 tickers of firms that returned the money, we applied a label of 1 for filings by firms that returned the money and 0 for those firms that did not. We also dropped all filings of companies that returned the money, starting from the date of the filing that stated the money was being returned, so as to not keep any information on or after the date on which the money was returned.

Using this dataset, we then conducted experiments to predict label 1 (companies that returned the PPP money) or 0 (those that did not) using the following two feature sets: (1) only cumulative returns on days –5 to 0 or (2) cumulative returns on days –5 to 0, plus the full text of the filing. We also computed the cumulative daily return from day 0 to day +5. We found that for firms that did not return the money (label 0), the mean day –5 to day 0 return was 3.5%, but it was only 0.2% for filings of firms that returned the money. This relationship reverses post-filing: Day 0 to day +5 return was 0.01% for firms that did not return the money but 0.28% for firms that returned the money. This suggests that, as a first cut, stock returns before filing dates may distinguish between firms that

EXHIBIT 4

Schematic of the Models Used after Applying BERT Models to the Text and Passing the Embeddings through an Average Pooling Layer



NOTE: We then use one or more dense layers of size 64 or 128 with and without dropout (0.5).

that uses context-free textual features. This is the case even given that NLP scores for readability, sentiment, and so on were used in the classifier and not used in the language model-based classifiers. Therefore, for financial text, context certainly matters, consistent with similar results in the legal (Lee and Hsiang 2019; Wan et al. 2019), biological (Lee et al. 2019), and science (Beltagy, Lo, and Cohan 2019) domains.

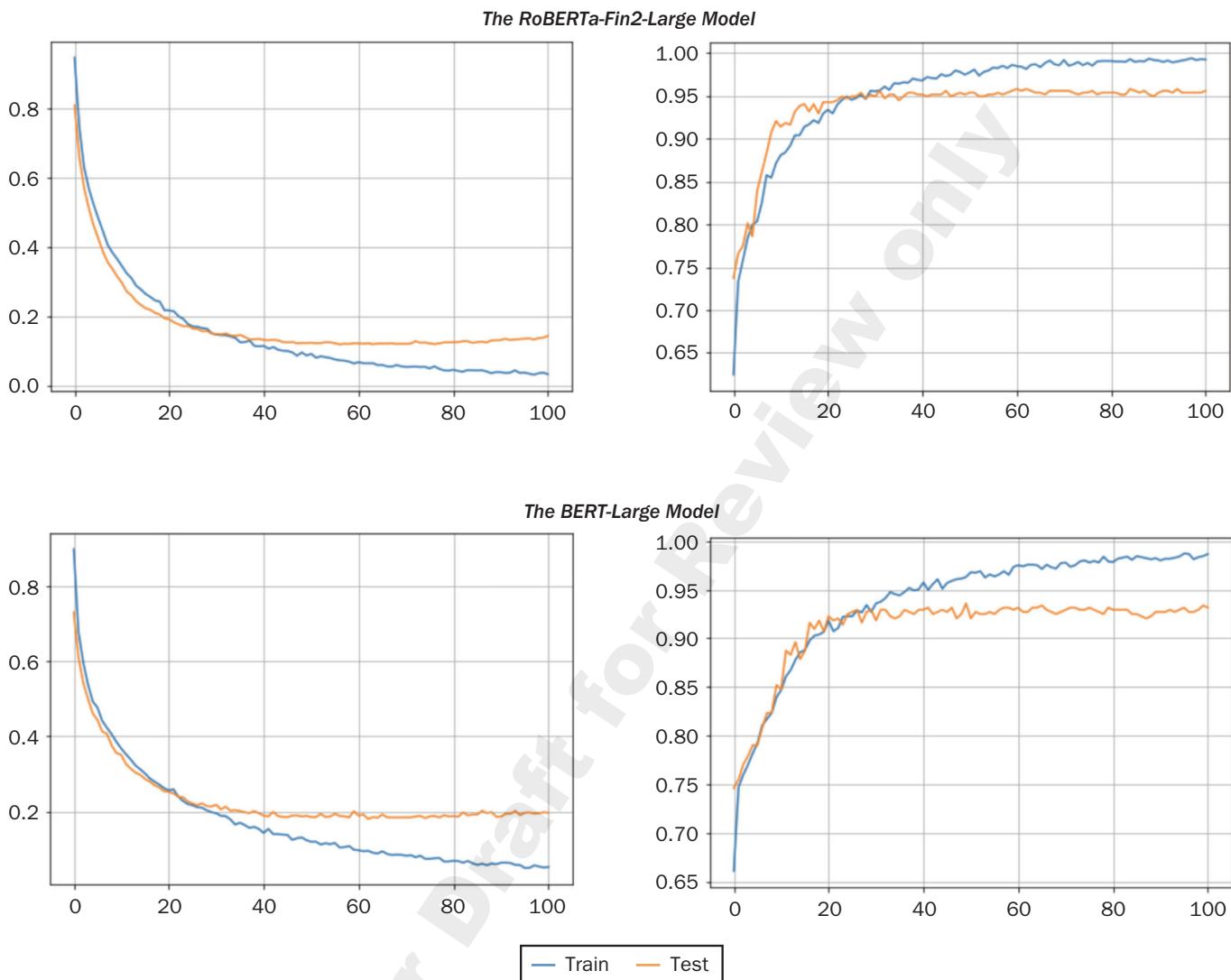
The pretrained RoBERTa class models, tuned for financial text, are of two types: (1) pretrained using only SEC filings data for the past decade (RoBERTa-Fin1) and (ii) pretrained using both the SEC data and Wiki text (RoBERTa-Fin2). A comparison of these models enables us to assess whether making the language model have only finance context and no other will improve its performance in classifying financial text. As we can see from a comparison of our new language models' performance (RoBERTa-Fin1 versus RoBERTa-Fin2) including both financial and nonfinancial context does improve the model over one that only has financial context—RoBERTa-Fin2 (both base and large) outperforms RoBERTa-Fin1. In fact, in comparison with RoBERTa-Fin1, traditional BERT models do better, which suggests that financial context alone may

The vast array of language models is used to exploit *transfer learning*. That is, we use the pretrained language model to modify the text input and create embeddings that are then passed into one or two additional layers in a neural network for training the specific classification task—in this case, scoring sentiment into three categories. We consider five different architectures for comparison across all the language models and feed the embeddings for text from the language model into standard feed-forward neural net layers. Rather than feeding the embeddings for only the CLS token, as is done with standard BERT models, we applied average pooling to the embeddings for all tokens and submitted that to the following architectures:

- Configuration 1: one dense layer (64), baseline model
- Configuration 2: one dense layer (128)
- Configuration 3: two dense layers (64, 64)
- Configuration 4: one dense layer (64), one dropout layer (0.5)
- Configuration 5: one dense layer (128), one dropout layer (0.5)

See Exhibit 4. All configurations have a final dense output layer of three neurons for the final classification. The dropout layers help in managing overfitting. We trained each model for 100 epochs and report the accuracy level on the test dataset for 30, 50, and 100 epochs. Exhibit 5 shows how the loss function and accuracy track with epoch for the training and validation datasets for the top two models. The RoBERTa-Fin2-large model has slightly higher accuracy and overfits marginally less than BERT-large.

Results for the financial news sentiment classification are presented in Exhibit 6. The language models clearly outperform TFIDF, which is an ensemble model

EXHIBIT 5**Progress of Training and Validation Loss and Accuracy for the RoBERTa-Fin2-Large Model and the BERT Large Model for Comparison**

NOTES: The RoBERTa-Fin2-large model pertains to the top two plots, and the BERT-large model is shown in the bottom two plots. The left-side plots display the loss function, and the right-side plots display model accuracy. We see that the accuracy levels out at around 30 epochs. At 100 epochs, it is 95.6% for the RoBERTa-Fin2-large model and 93.2% for standard BERT-large.

be too narrow for good classification and the best route is to further pretrain general language models with additional financial text. Finally, we also see that RoBERTa-Fin2 outperforms language models such as BERT that do not have extended pretraining with financial text, but the gains here are smaller. In sum, context matters most, and financial context matters incrementally.

Finally, we examine use of summarized text. It is likely that we may be able to improve classifier performance by using a summary of the text instead of all the text. This has the potential to remove some of the noise in the text and to avoid truncation of the text when it gets too long. We redid the experiment by replacing the full text of the SEC filings with the summary of the text. We did not find that summarization helps improve metrics (precision, recall, F1). Therefore, we conclude that the use

EXHIBIT 6

Comparison of Performance on the Sentiment Classification Task across Various Language Models

Model	Accuracy/Configuration No.		
	30 Epochs	50 Epochs	100 Epochs
AutoGluon		86.1%	
Distilbert hf	92.9%/4	92.9%/4	93.2%/4
Distilbert gluon	92.3%/1	92.3%/2	92.9%/4
BERT Base	92.1%/2,5	92.3%/4	93.4%/4
BERT Large	93.2%/4	92.3%/3,4	92.3%/3
RoBERTa Base	79.2%/1,2	81.2%/5	80.6%/5
RoBERTa Large	85.7%/3	86.1%/3	86.1%/2,5
RoBERTaFin1 Base	90.3%/4	90.3%/1,5	90.5%/2
RoBERTaFin1 Large	92.9%/2	92.9%/5	92.7%/1
RoBERTaFin2 Base	93.6%/2	94.3%/3	94.7%/2,3,5
RoBERTaFin2 Large	95.6%/3	95.6%/2,5	95.6%/1

NOTE: We report the results using various models on the same dataset and for the five neural net configurations mentioned in the text.

models adds context and provides an additional 7% improvement in classification accuracy. This suggests that context matters in classifying financial text. We also pretrained a RoBERTa-Fin2 large model on 10 years of SEC filings and showed that it provides a better model than standard transformer-based BERT, DistilBERT, and RoBERTa models. Therefore, pretraining on finance-specific text is also valuable.

We also attempted to handle long documents via summarization, which yields no material improvement in accuracy. More work needs to be done to determine the best methods for long documents, which are ubiquitous in finance. Further work will examine models (e.g., Zaheer et al. 2020) that tackle the long-document problem. The experiments here are extensible to a fuller examination using long-document methods for classification of regulatory filings, earnings call transcripts, analyst reports, and so on.

REFERENCES

- Antweiler, W., and, M. Z. Frank. 2004. “Is All That Talk Just Noise? The Information Content of Internet Stock Message Boards.” *The Journal of Finance* 59 (3): 1259–1294.
- Araci, D. “FinBERT: Financial Sentiment Analysis with Pre-Trained Language Models.” *arXiv* 1908.10063, 2019.
- Bach, M. P., Z. Krstic, S. Seljan, and L. Turulja. 2019. “Text Mining for Big Data Analysis in Financial Sector: A Literature Review.” *Sustainability* 11 (5): 1–27.
- Balyuk, T., N. Prabhala, and M. Puri. “On the Costs of Being Public and Government Aid: Evidence from the Paycheck Protection Program.” Working paper, Johns Hopkins University, 2020.
- Beltagy, I., K. Lo, and A. Cohan. “SciBERT: A Pretrained Language Model for Scientific Text.” *arXiv* 1903.10676, 2019.
- Bodnaruk, A., T. Loughran, and B. McDonald. 2015. “Using 10-K Text to Gauge Financial Constraints.” *Journal of Financial and Quantitative Analysis* 50, no. 4 (August): 623–646.

of text summaries for classification gives ambiguous results, and more testing is needed to assess the efficacy of summaries.

CONCLUDING DISCUSSION

In this article, using multiple datasets, we examined how much we can improve classification accuracy by extending the explanatory variables beyond the standard scoring for sentiment, readability, positivity, negativity, risk, litigiousness, and polarity used in the finance literature. We verify that these methods deliver a high baseline level of accuracy. By using full text, we are able to inject non-numerical, forward-looking information, and the improvement in classification performance is material. For the financial news dataset, there is a 20% improvement in accuracy. For a different experiment on SEC 10-K/Q and 8-K filings, the improvement over just using return series is 5%. In general, adding text to tabular data improves classification accuracy on news and return data.

Other experiments show that the use of language

- Bonsall, S. B., A. J. Leone, B. P. Miller, and K. Rennekamp. 2017. "A Plain English Measure of Financial Reporting Readability." *Journal of Accounting and Economics* 63, no. 2 (April): 329–357.
- Brown, S., and J. W. Tucker. 2011. "Large-Sample Evidence on Firms' Year-over-Year MD&A Modifications." *Journal of Accounting Research* 49 (2): 309–346.
- Bushee, B. J., I. D. Gow, and D. J. Taylor. 2018. "Linguistic Complexity in Firm Disclosures: Obfuscation or Information?" *Journal of Accounting Research* 56 (1): 85–121.
- Chebonenko, T., L. Gu, and D. Muravyev. "Text Sentiment's Ability to Capture Information: Evidence from Earnings Calls." Paper 2352524, SSRN, March 2018.
- Child, R., S. Gray, A. Radford, and I. Sutskever. "Generating Long Sequences with Sparse Transformers." *arXiv* 1904.10509, 2019.
- Cohen, L., C. Malloy, and Q. Nguyen. 2020. "Lazy Prices." *The Journal of Finance* 75 (3): 1371–1415.
- Cong, L. W., T. Liang, B. Yang, and X. Zhang. "Analyzing Textual Information at Scale." Paper 3449822, SSRN, November 2019.
- Cong, L. W., T. Liang, and X. Zhang. "Textual Factors: A Scalable, Interpretable, and Data-Driven Approach to Analyzing Unstructured Information." Paper 3307057, SSRN, September 2019.
- Das, S. R. 2014. "Text and Context: Language Analytics in Finance." *Foundations and Trends in Finance* 8 (3): 145–261.
- Das, S. R., and M. Y. Chen. 2007. "Yahoo! For Amazon: Sentiment Extraction from Small Talk on the Web." *Management Science* 53 (9): 1375–1388.
- Desola, V., K. Hanna, and P. Nonis. "FinBERT: Pre-Trained Model on SEC Filings for Financial Natural Language Tasks." 2019, UC Berkeley, Working Paper 266, DOI: [10.13140/RG.2.2.19153.89442](https://doi.org/10.13140/RG.2.2.19153.89442).
- Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova. "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding." version 2, *arXiv* 1810.04805, 2019.
- Erickson, N., J. Mueller, A. Shirkov, H. Zhang, P. Larroy, M. Li, and A. Smola. "AutoGluon-Tabular: Robust and Accurate AutoML for Structured Data." *arXiv* 2003.06505, 2020.
- Ertugrul, M., J. Lei, J. Qiu, and C. Wan. 2017. "Annual Report Readability, Tone Ambiguity, and the Cost of Borrowing." *Journal of Financial and Quantitative Analysis* 52, no. 2 (April): 811–836.
- Gao, M., and J. Huang. 2020. "Informing the Market: The Effect of Modern Information Technologies on Information Production." *The Review of Financial Studies* 33, no. 4 (April): 1367–1411.
- Gentzkow, M., B. Kelly, and M. Taddy. 2019. "Text as Data." *Journal of Economic Literature* 57, no. 3 (September): 535–574.
- Hafez, P., M. Kangrga, J. Guerrero-Colon, F. Gomez, and R. Matas. "Capturing Alpha from Your Own Digital Content." 2020a, www.ravenpack.com.
- Hafez, P., R. Matas, F. Gomez, M. Kangrga, B. Skorodumov, and A. Liu. "RavenPack News Sentiment Data Outperforms During Coronavirus Crisis." 2020b, www.ravenpack.com.
- Hoberg, G., and G. Phillips. 2016. "Text-Based Network Industries and Endogenous Product Differentiation." *Journal of Political Economy* 124, no. 5 (October): 1423–1465.
- Huang, K., J. Altosaar, and R. Ranganath. "ClinicalBERT: Modeling Clinical Notes and Predicting Hospital Readmission." *arXiv* 1904.05342, 2019.
- Jegadeesh, N., and D. Wu. 2013. "Word Power: A New Approach for Content Analysis." *Journal of Financial Economics* 110, no. 3 (December): 712–729.
- Kearney, C., and S. Liu. 2014. "Textual Sentiment in Finance: A Survey of Methods and Models." *International Review of Financial Analysis* 33 (May): 171–185.

- Lee, J., W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang. 2019. "BioBERT: A Pre-Trained Biomedical Language Representation Model for Biomedical Text Mining." *Bioinformatics* 36 (4): 1234–1240.
- Lee, J.-S., and J. Hsiang. "PatentBERT: Patent Classification with Fine-Tuning a Pre-Trained BERT Model." *arXiv* 1906.02124, 2019.
- Li, F. 2010a. "The Information Content of Forward-Looking Statements in Corporate Filings—A Naïve Bayesian Machine Learning Approach." *Journal of Accounting Research* 48 (5): 1049–1102.
- . 2010b. "Textual Analysis of Corporate Disclosures: A Survey of the Literature." *Journal of Accounting Literature* 29: 143–165.
- Li, J., and X. Zhao. "Complexity and Information Content of Financial Disclosures: Evidence from Evolution of Uncertainty Following 10-K Filings." 2014, <https://ssrn.com/abstract=2516622>.
- Liu, L., K. Liu, Z. Cong, J. Zhao, Y. Ji, and J. He. 2018. "Long Length Document Classification by Local Convolutional Feature Aggregation." *Algorithms* 11, no. 8 (August): 109.
- Liu, Y., M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. "RoBERTa: A Robustly Optimized BERT Pretraining Approach." *arXiv* 1907.11692, 2019.
- Loughran, T., and B. McDonald. 2011. "When Is a Liability Not a Liability? Textual Analysis, Dictionaries, and 10-Ks." *The Journal of Finance* 66 (1): 35–65.
- . 2013. "IPO First-Day Returns, Offer Price Revisions, Volatility, and Form S-1 Language." *Journal of Financial Economics* 109 (2): 307–326.
- . 2014. "Measuring Readability in Financial Disclosures." *The Journal of Finance* 69 (4): 1643–1671.
- . 2015. "The Use of Word Lists in Textual Analysis." *Journal of Behavioral Finance* 16, no. 1 (January): 1–11.
- . 2016. "Textual Analysis in Accounting and Finance: A Survey." *Journal of Accounting Research* 54 (4). 1187–1230.
- . 2020. "Textual Analysis in Finance." SSRN 3470272, June 2020.
- Loughran, T., B. McDonald, and H. Yun. 2009. "A Wolf in Sheep's Clothing: The Use of Ethics-Related Terms in 10-K Reports." *Journal of Business Ethics* 89, no. 1 (May): 39–49.
- Malkiel, B. G. *A Random Walk down Wall Street: The Time-Tested Strategy for Successful Investing*. New York: W.W. Norton, 2003.
- Malo, P., A. Sinha, P. Korhonen, J. Wallenius, and P. Takala. 2014. "Good Debt or Bad Debt: Detecting Semantic Orientations in Economic Texts." *Journal of the Association for Information Science and Technology* 65, no. 4 (April): 782–796.
- Matthews, B. W. 1975. "Comparison of the Predicted and Observed Secondary Structure of T4 Phage Lysozyme." *Biochimica et Biophysica Acta (BBA)—Protein Structure* 405, no. 2 (October): 442–451.
- Pappagari, R., P. Źelasko, J. Villalba, Y. Carmiel, and N. Dehak. "Hierarchical Transformers for Long Document Classification." *arXiv* 1910.10781, 2019.
- Price, S. M., J. S. Doran, D. R. Peterson, and B. A. Bliss. 2012. "Earnings Conference Calls and Stock Returns: The Incremental Informativeness of Textual Tone." *Journal of Banking & Finance* 36, no. 4 (April): 992–1011.
- Routledge, B. R. 2019. "Machine Learning and Asset Allocation." *Financial Management* 48 (4): 1069–1094.

- Sanh, V., L. Debut, J. Chaumond, and T. Wolf. "DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter." *arXiv* 1910.01108, 2020.
- Tetlock, P. C., M. Saar-Tsechansky, and S. Macskassy. 2008. "More Than Words: Quantifying Language to Measure Firms' Fundamentals." *The Journal of Finance* 63 (3): 1437–1467.
- Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. "Attention Is All You Need." *arXiv* 1706.03762, 2017.
- Wan, L., G. Papageorgiou, M. Seddon, and M. Bernardoni. "Long-Length Legal Document Classification." *arXiv* 1912.06905, 2019.
- Yang, Y., M. C. S. Uy, and A. Huang. "FinBERT: A Pretrained Language Model for Financial Communications." *arXiv* 2006.08097, 2020.
- Zaheer, M., G. Guruganesh, A. Dubey, J. Ainslie, C. Alberti, S. Ontanon, P. Pham, A. Ravula, Q. Wang, L. Yang, and A. Ahmed. "Big Bird: Transformers for Longer Sequences." *arXiv* 2007.14062, 2020.
- Zheng, S., H. Lin, S. Zha, and M. Li. "Accelerated Large Batch Optimization of BERT Pretraining in 54 Minutes." *arXiv* 2006.13484, 2020.

To order reprints of this article, please contact David Rowe at d.rowe@pageantmedia.com or 646-891-2157.



CSR-Bench: Benchmarking LLM Agents in Deployment of Computer Science Research Repositories

Yijia Xiao^{†*} Runhui Wang^{‡*} Luyang Kong^{‡*} Davor Golac[‡] Wei Wang[†]

[†]University of California, Los Angeles

[‡] Amazon Web Services

* Equal Contribution

{yijia.xiao,weiwang}@cs.ucla.edu

{runhuiw,luyankon,dgolac}@amazon.com

Abstract

The increasing complexity of computer science research projects demands more effective tools for deploying code repositories. Large Language Models (LLMs), such as Anthropic Claude and Meta Llama, have demonstrated significant advancements across various fields of computer science research, including the automation of diverse software engineering tasks. To evaluate the effectiveness of LLMs in handling complex code development tasks of research projects, particularly for NLP/CV/AI/ML/DM topics, we introduce CSR-Bench, a benchmark for Computer Science Research projects. This benchmark assesses LLMs from various aspects including accuracy, efficiency, and deployment script quality, aiming to explore their potential in conducting computer science research autonomously. We also introduce a novel framework, CSR-Agents, that utilizes multiple LLM agents to automate the deployment of GitHub code repositories of computer science research projects. Specifically, by checking instructions from markdown files and interpreting repository structures, the model generates and iteratively improves bash commands that set up the experimental environments and deploy the code to conduct research tasks. Preliminary results from CSR-Bench indicate that LLM agents can significantly enhance the workflow of repository deployment, thereby boosting developer productivity and improving the management of developmental workflows.

1 Introduction

With the rapid evolution of Large Language Models (LLMs), it has been demonstrated that LLMs have increased reasoning ability over the last few years, making intelligent agents based on LLM possible. Current agent-related applications in computer science include code writing (Codex (Chen et al., 2021), Deepseek Code (Guo et al., 2024), CodeLLaMA (Rozière et al., 2023), etc.), code

base generation (MetaGPT (Hong et al., 2023), Agentless (Xia et al., 2024a) and CodeStar(Li et al., 2023)), code correction (SWE Bench (Jimenez et al., 2024)), and more.

In computer science research projects, the associated codebases grow very rapidly, and a self-consistent codebase typically has several parts, including the instruction file (e.g., the README file), associated code packages, and related data. The instruction file usually contains an overview of the project, including environment setup, data preparation (e.g., data download and pre-processing, model weights download and preparation), model training process, performance evaluation, and the setup of a demo project (e.g., a chatbot project that interacts with users). In computer science, prestigious conferences, including NAACL, ACL, ICLR, NeurIPS, CVPR, KDD, etc., encourage researchers to release source code for reproducibility of their accepted papers, and GitHub is the top choice for maintaining codebases for most researchers. A major step of computer science research is reproducing existing work, which is essential to gain insights and propose novel methodologies. However, even for well-documented and self-consistent projects, the setup process requires manual efforts and cannot be fully automated; many steps of setting up a code repository are rather mechanical, such as installing/updating dependency packages to configure the environment, downloading data, updating the relevant script/data directories, etc., which is tedious and often time-consuming.

To tackle such challenges, we propose to use LLM agents for automating the deployment of code repositories of research projects, and build a benchmark, **Computer Science Research Benchmark** (CSR-Bench) for evaluating LLM Agents on code repository deployment tasks. We also propose a multi-agent collaborative framework, CSR-Agents, to automate the deployment of code repositories by coordinating multiple LLM agents with different

expertise and iterative improvement with provided tools. From more than ten major NLP/CV/AI/ML/DM conferences, we collected the 100 highly rated repositories¹, which are carefully selected from an initial candidate set that has more than 1500 top-star repositories. Our selection criteria include topic diversity and self-containment² so that CSR-Bench can provide a comprehensive evaluation of LLM agents on code deployment tasks including instruction generation, command execution, and self-improvement with tools.

To the best of our knowledge, CSR-Bench is the first benchmark for the deployment of computer science research projects, providing a reference for evaluating LLM agents. We note that the success of code deployment depends not only on code generation, but also on many non-coding factors including tasks like experimental environment setup, data/model preparation, correcting bash commands, searching for solutions, and etc. Our proposed multi-agent framework, CSR-Agents, aims to achieve automation of code deployment, which can accelerate the progress for computer science research projects.

Our contributions are as follows:

- We introduce CSR-Bench to assess LLM’s ability to understand instruction manuals and complex project structures, generate executable commands for code deployment, and solve errors during deployment.
- We propose the CSR-Agents framework, which leverages multi-agent cooperation with specialized capabilities including instruction comprehension, command execution, error log analysis, and error correction with searching and retrieval tools.
- We design a standardized testing system for reproducibility in CSR-Bench, which can make CSR-Agents a CI/CD³ standard system of computer science code repository deployment, ensuring ease of use, reusability, and improving communication and collaboration efficiency in computer science research projects.
- We evaluated a wide range of foundation models for CSR-Agents on CSR-Bench. Results indicate that LLM agents can potentially

accelerate the process of repository deployment, thereby boosting researcher productivity. However, it is still challenging to achieve full automation.

2 Related Work

Coding LLMs. Large Language Models (LLMs) have become the go-to solution for a wide array of coding tasks due to their exceptional performance in both code generation and comprehension (Chen et al., 2021). These models have been successfully applied to various software engineering activities, including program synthesis (Patton et al., 2024; Chen et al., 2021; Li et al., 2022a; Iyer et al., 2018), code translation (Pan et al., 2024; Roziere et al., 2020, 2021), program repair (Xia et al., 2023; Xia and Zhang, 2023; Monperrus, 2018; Bouzenia et al., 2024), and test generation (Deng et al., 2023; Xia et al., 2024b; Deng et al., 2024; Lemieux et al., 2023; Kang et al., 2023). Beyond general-purpose LLMs, specialized models have been developed by further training on extensive datasets of open-source code snippets. Notable examples of these code-specific LLMs include CODEX (Chen et al., 2021), CodeLlama (Rozière et al., 2023), StarCoder (Li et al., 2023; Lozhkov et al., 2024), and DeepSeek-Coder (Guo et al., 2024). Additionally, instruction-following code models have emerged, refined through instruction-tuning techniques. These include models such as CodeLlama-Inst (Rozière et al., 2023), DeepSeek-Coder-Inst (Guo et al., 2024), WizardCoder (Luo et al., 2023), Magicoder (Wei et al., 2023), and OpenCodeInterpreter (Zheng et al., 2024).

Benchmarking LLM-based coding tasks. To assess the capabilities of LLMs in coding, a variety of benchmarks have been proposed. Among the most widely utilized are HUMAN-EVAL (Chen et al., 2021) and MBPP (Austin et al., 2021), which are handcrafted benchmarks for code generation that include test cases to validate the correctness of LLM outputs. Other benchmarks have been developed to offer more rigorous tests (Liu et al., 2023a), cover additional programming languages (Zheng et al., 2023; Cassano et al., 2023), and address different programming domains (Jain et al., 2024; Hendrycks et al., 2021; Li et al., 2022b; Lai et al., 2023; Yin et al., 2022).

More recently, research has shifted towards evaluating LLMs on real-world software engineering challenges by operating on entire code reposi-

¹With appropriately permissive licenses.

²The information within the repository is mostly sufficient for a successful deployment.

³Continuous Integration and Continuous Delivery.

ries rather than isolated coding problems (Jimenez et al., 2024; Zhang et al., 2023; Liu et al., 2023b). A notable benchmark in this area is SWE-bench (Jimenez et al., 2024), which includes tasks requiring repository modifications to resolve actual GitHub issues. The authors of SWE-bench have also released a more focused subset, SWE-bench Lite (swe, 2024), which contains 300 problems centered on bug fixing that only involves single-file modifications in the ground truth patches. ML-Bench (Liu et al., 2023c) is a benchmark for evaluating large language models and agents for Machine Learning tasks on repository-level code. It involves 18 repositories and focuses on code generation and interactions with Jupyter Notebooks.

Repository-level coding. The rise of agent-based frameworks (Xi et al., 2023) has spurred the development of agent-based approaches to software engineering tasks. Devin (dev, 2024) (and its open-source counterpart OpenDevin (ope, 2024b)) is among the first comprehensive LLM agent-based frameworks. Devin employs agents to first perform task planning based on user requirements, then allows them to use tools like file editors, terminals, and web search engines to iteratively execute the tasks. SWE-agent (Yang et al., 2024) introduces a custom agent-computer interface (ACI), enabling the LLM agent to interact with the repository environment through actions like reading and editing files or running bash commands. Another agent-based approach, AutoCodeRover (Zhang et al., 2024), equips the LLM agent with specific APIs (e.g., searching for methods within certain classes) to effectively identify the necessary modifications for issue resolution. Beside these examples, a variety of other agent-based approaches have been developed in both open-source (Gauthier, 2024) and commercial products (Bouzenia et al., 2024; Chen et al., 2024; Ma et al., 2024; lin, 2024; fac, 2024; ibm, 2024; ope, 2024a; mar, 2024; ama, 2024).

Unlike existing benchmarks and agent-based frameworks, which focus on the code generation/completion tasks, our proposed CSR-Bench and CSR-Agents focus on the code deployment task, which is under-studied in the field.

3 CSR-Bench

In this section, we provide the problem statement for code deployment in CSR-Bench, introduce the code repository collection process of computer science research projects, and show their statistics.

3.1 Problem Statement

The CSR-Bench consists of a collection of computer science research repositories from GitHub and these repositories are used for evaluating the capabilities of LLMs in code deployment tasks. For each repository, the deployment tasks typically include: (1) setting up the environment; (2) preparing necessary data and model files; (3) conducting model training; (4) demonstration of inference; (5) performance evaluation. To complete these tasks, we prompt LLMs to generate executable bash commands by using the README file as the primary source of information and other repository contents (source code, bash scripts, directory structure, and etc.) as supplementary information.

Metric. During the evaluation in CSR-Bench, the large language model will be prompted to generate executable commands for the corresponding sections for each repository. we use the completion rate as a key metric, defined as the ratio between number of successfully executed commands and the total number of commands executed.

3.2 Repository Collection

In CSR-Bench, we aim to collect a diverse and comprehensive collection of code repositories of computer science-related research projects. GitHub is a good data source for this purpose and it provides tags for identifying most relevant repositories. Some example tags are “nlp”, “naacl”, and “emnlp2024”. Since CSR-Bench focuses on computer science-related repositories, we filter the repositories by tags of various conference names and categories to ensure they include diverse topics.

For repository selection, we use GitHub tags to obtain an initial set of over 1500 repositories that are relevant to computer science research topics and categorizing them into five areas: Natural Language Processing, Computer Vision, Large Language Models, Machine Learning, and Interdisciplinary topics. Notably, we collect repositories related to large language models because nowadays LLM-related research projects are increasingly popular due to its foundational impact in various areas of computer science.

We obtain 100 high-quality code repositories for CSR-Bench in the following steps. First, we categorize this initial set and sort them by the number of GitHub stars. Next, we manually check the content of each repository starting from the top of the sorted list. In this step, we only keep reposi-

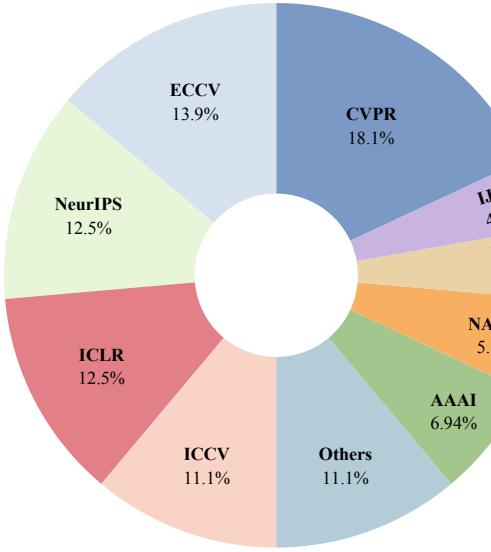


Figure 1: Conf Distribution of CSR-Bench

tories that contain sufficient information in their README files. We also skip the repositories that do not contain deployable code. Finally, we check the licenses of the repositories and make sure they are permissive.

3.3 Statistics of CSR-Bench

This section provides an in-depth analysis of the traits of repositories in CSR-Bench. We examine the diversity and breadth of topics covered, as well as detailed statistics about the documentation and structure of these repositories.

In CSR-Bench, the README files and directory structures provide critical insights into the usability and organization of repositories. We use the following figures to analyze the lengths of README file and number of files, and offer a quantitative view of content complexity and organizational depth. The length of README file is an important metric because the most LLMs have limits on the input token length. The number of files indicate the complexity of the code repository.

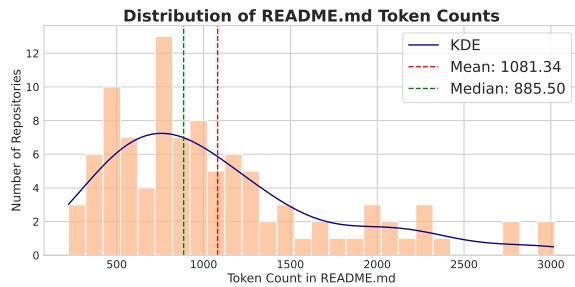


Figure 3: Number of Tokens per README

Figure 3 shows the distribution of token counts

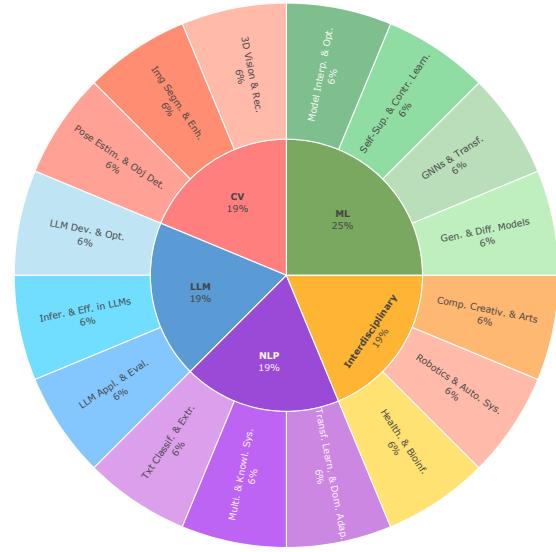


Figure 2: Topic Distribution of CSR-Bench

in README files, highlighting the extent of documentation, which is essential for user understanding and repository usability. Since the mean token counts is just over 1000 and the maximum counts is around 3000, most LLMs can take the full README files as input.

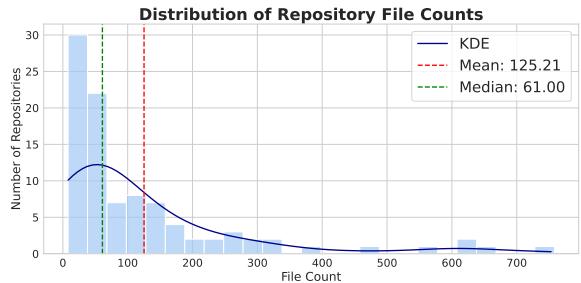


Figure 4: Number of Files per Repository

Figure 4 depicts the distribution of file counts in repositories, reflecting their complexity and scale based on the number of files. Because the number of files in most repositories are in the lower hundreds, it is feasible to leverage directory structure for code deployment with LLMs.

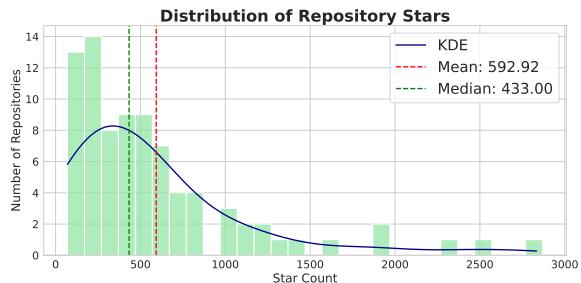


Figure 5: Stargazer Distribution

Figure 5 shows the distribution of star counts in selected repositories. The average count is over 590, which means that these repositories receives significant attention and indicates they are generally well maintained.

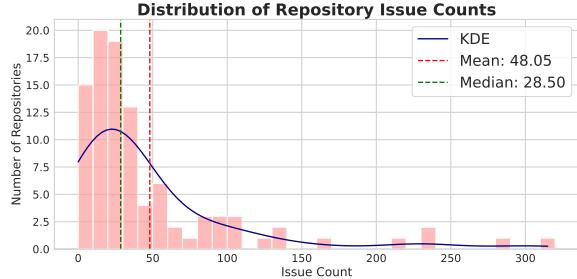


Figure 6: Size of Issue Database

Figure 6 shows the distribution of issues counts in selected repositories. On average, each repository contains over 48 issues, indicating these repositories have a good amount of engagements with the open-source community and sufficient support from the authors. Therefore, the information in the issues are valuable for the deployment tasks.

4 CSR-Agents

In this section, we propose CSR-Agents, a multi-agent framework that leverages LLMs for different tasks and achieves effective cooperation among agents for code deployment tasks. We introduce our standardized environment for code deployment, functionalities of different agents, and their cooperation workflow.

4.1 Code Deployment with LLM

For each repository, we use the README file under the root folder as the major source of information and prompt LLMs to generate executable bash commands for different steps of deployment including environment setup, data preparation, training, inference and evaluation.

To achieve reproducibility and safety, we use the Docker container to isolate the code deployment environment. We build a standard Docker image that is equipped with essential tools like bash, Conda, GCC, Make, Python, etc. for evaluations across all repositories and various LLMs. In the container, we use a counter to count the total number of scripts that needed to be executed and the number of scripts that were executed successfully. In this way, we safeguard the entire computing system, especially from bash command executions

that involve system-level permissions and could potentially break the whole system.

We note that the use of Docker is not part of the original repositories. The introduction of Docker into our evaluation system can ensure that each time we conduct the evaluation, we start from the same environment for fair comparisons across different LLMs, and the evaluation of one repository does not affect any other repositories. Another benefit of using Docker is that we can speed up the evaluation by running different evaluations in separate docker instances at the same time.

The LLM generated bash commands may not work well for successful deployment due to various reasons. First, in some README files, some basic directives (such as conda and pip) were missing, so the LLMs could not generate these commands at the first attempt. Secondly, the installation of packages may not be completed in a single attempt and usually needs several iterations of trial-and-error. Thirdly, some steps of the deployment require checking additional information like GitHub issues of the corresponding repositories and from the internet. To handle these issues, we design a group of LLM agents that cooperate effectively.

4.2 CSR-Agent: LLM Agent Design

In CSR-Agents, we adopt an iterative trial-and-error process for successful code deployment. Specifically, LLM takes README, directory structure, and error logs from failed command execution as input and generates bash commands to complete the deployment tasks. The system comprises five agents: Command Drafter, Script Executor, Log Analyzer, Issue Retriever, and Web Searcher. These agents collectively facilitate the deployment of a code repository. The complete workflow is shown in Figure 7.

Command Drafter: This agent reads the README files and directory structure and generates a draft script containing bash commands for deployment. Then, it divides the entire script into five sections, each corresponding to a step in the code repository deployment. This sectional division also serves as an evaluation standard later on.

Script Executor: This agent receives the draft commands from the Command Drafter and execute the commands in our standardized Docker environment. After execution, it collects logs from bash, including standard output and errors. Note that during bash script execution, no explicit return code

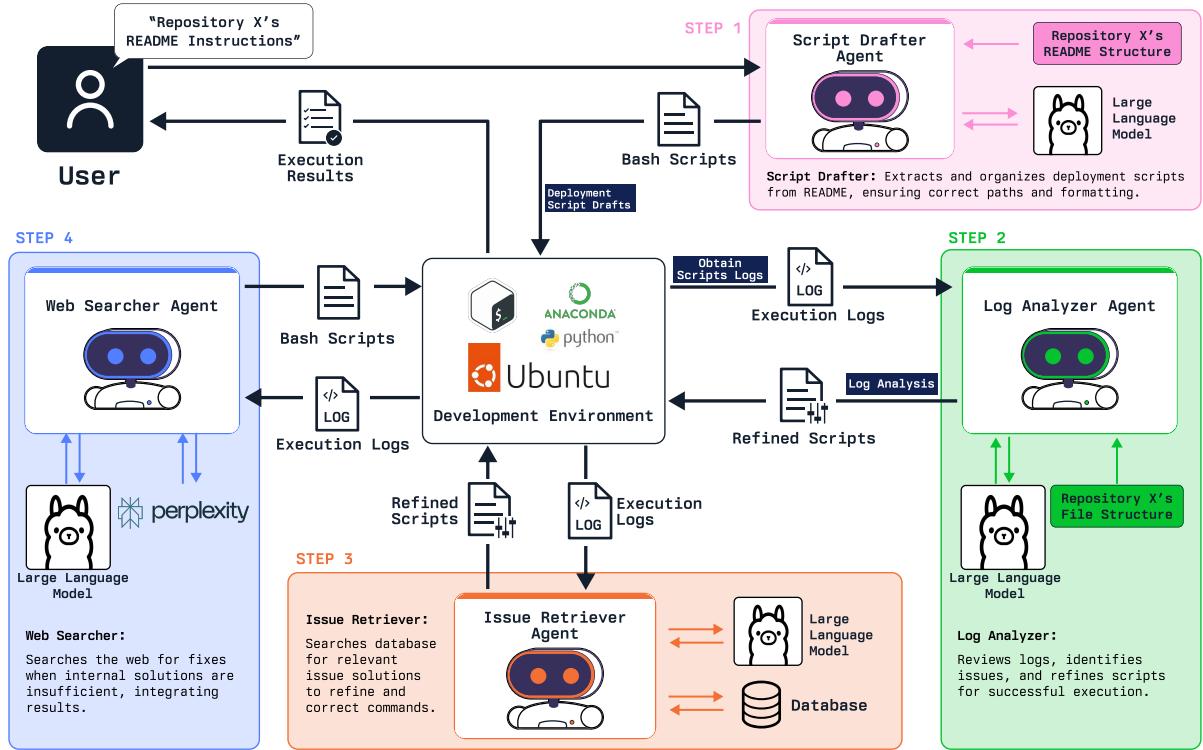


Figure 7: Workflow of CSR-Bench

is provided by bash. We experimented with setting predefined special prompt to bash and parse the return code from the returned output and error message. However, we find that a good quantity of commands do not have return code, making the feedback from bash not informative to other agents. To address this challenge, we leverage the LLM in the executor, instructing it to provide feedback based on the standard output and error messages. We then parse this feedback to generate a return code: if the return code is zero, then the command is executed successfully; otherwise, the command, output, and error messages are logged and sent to the Log Analyzer.

Log Analyzer: This agent reads the logs and the associated bash command, and checks for updates, missing prerequisites, or script paths that need updating. It also identifies any other missing components and returns a curated command for a successful execution. We note that this agent only reply on the internal knowledge of the LLM and may not be able to correctly solve the errors.

Issue Retriever: This agent takes in the command, standard output and error message and search them against the issue database we collected from the repository. It leverages RAG and its pipeline requires a search algorithm to query the in-

put against the database. In our design, the queries are the combination of the commands executed, standard output and error messages. The database contains GitHub issues and their communication records for the repository. We experimented with BM25 and Contriever as the retrieval algorithm and decided to use BM25 for (1) BM25's higher search speed and (2) the fact that error logs and issues generally share keywords, so sentimental search do not possess much advantage over lexical search.

Web Searcher: This agent utilizes the Perplexity API to obtain solutions for failed execution. If the pipeline reaches this stage, it indicates that the Log Analyzer and Issue Retriever failed to solve a failed command. The standard output, standard error, and the failed command are fed to Perplexity to search the web for solutions. The Web Searcher then analyzes the solutions from Perplexity and generates new bash commands to resolve the issue.

4.3 LLM Corporation Framework

The workflow operates as follows: Deployment commands are drafted from repository documentation, executed in a bash environment, and adjusted based on log analysis if errors arise. Additional information is retrieved from an issue database and web search if needed. The process concludes with

a summary that outlines successes and identifies steps needing further attention.

Drafting the Initial Commands The instructions from each repository’s README are fed into the Command Drafter LLM Agent to draft the necessary commands for deploying the repository. These commands are organized into five stages: prerequisite installation, data and model checkpoints downloading, training, inference, and evaluation. It is important to note that not every repository contains all these sections, and some sections may be empty.

Execution of Commands After the draft stage, the script drafted will be sent to the Script Executor. To provide LLM Agents with the Python interface, we implement a BashExecutor that encapsulates the bash binary executable file in it. If the commands execute successfully, the process is deemed successful and the deployment pipeline will return True. Though looks simple, the bash simulation was not straightforward during our experiments, since we need to consider the environment variables, stdout, stderr, etc. Initially, we used a subprocess to handle it, but the success rate was extremely low. Upon analyzing the logs, we discovered that no commands related to environment variable changes were reflected in subsequent instructions. For instance, if the Command changes the environment to a Python virtual environment or another conda environment, this change only applies to that specific command. Afterward, the Python interpreter and package manager revert to their default settings. Besides, not all command executions have a valid return code, therefore, we utilize an LLM to parse the standard output and standard error (if any) to obtain the return state of the execution.

Analyze the Execution Log If the execution is not successful and error occurs, the standard output, standard error, command, and return code are logged and analyzed. The Log Analyzer examines the error messages and attempts to refine the execution commands or adjust prerequisites to ensure the environment is prepared. We have a `max_attempt` argument that limits the number of retries for the log analyzer. If the issue persists after certain attempts, the workflow utilizes the web search tool to request external information.

Retrieve Augmented Generation from Issue Database After analyzing the execution log and making initial adjustments, the next step is to retrieve more insights using a Retrieve Augmented

Generation (RAG) approach. The Issue Retriever agent uses the logged command, output, and error messages to query the issue database for similar past problems or discussions. Leveraging the BM25 retrieval algorithm, it matches keywords from the logs to relevant entries. If a match is found, the agent extracts solutions or troubleshooting steps, feeding them back into the workflow to refine commands. If no relevant match is found, the process escalates to the Web Searcher for external information.

Search the Internet for External Information Using tools such as Perplexity, the WebSearcher agent integrates the command error logs and standard output with external information. This process refines the command and retries execution in the bash with a limited number of attempts. The agent records the information and transfers it to the deployment summarizer if the issue remains unresolved.

5 Evaluations

In this section, we evaluate CSR-Agents in CSR-Bench with a wide range of popular foundation LLM families, including Claude⁴, GPT-4 (Achiam et al., 2023), Llama-3 (Dubey et al., 2024), and Mistral⁵. For each LLM family, we experimented with different model sizes for thorough comparisons.

We show the completion rate in Table 1, Table 2, Table 3, and Table 4. In these tables, we use **S** to stand for the Setup stage, **D** to stand for the download stage, **T** to stand for the Training stage, **E** to stand for the Evaluation stage, and **I** for the inference stage.

5.1 Initial Drafter

As shown in Table 1, all models perform well on *Setup* and *Download* tasks (success rates around 0.23 to 0.28) but struggle with *Training*, *Evaluation*, and *Inference*, where success rates are close to zero. This indicates that the drafter agent handles basic installation effectively but has difficulty with complex tasks requiring updates to file paths and environment variables.

We note that this is also similar to the deployment process of a real researcher, where their first execution is more likely to fail and they need to analyze the errors and leverage tools like GitHub issues or search engines to solve the problem.

⁴<https://www.anthropic.com/news/claude-3-family>

⁵<https://mistral.ai/technology/#models>

Table 1: Drafter Success Metrics of Different Models.

Model Type	Name	S	D	T	E	I
Claude	Instant	0.232	0.189	0.007	0.000	0.000
	3-Haiku	0.253	0.239	0.046	0.005	0.052
	3-Sonnet	0.284	0.283	0.045	0.024	0.031
GPT	4o-Mini	0.242	0.229	0.008	0.016	0.029
	4o	0.261	0.238	0.039	0.022	0.031
	4-Turbo	0.271	0.252	0.028	0.039	0.032
Llama	3-70B	0.239	0.306	0.019	0.040	0.032
	3.1-8B	0.243	0.200	0.051	0.037	0.007
	3.1-70B	0.260	0.280	0.032	0.022	0.019
Mistral	Large	0.243	0.266	0.047	0.031	0.026
	Large-2	0.251	0.279	0.039	0.025	0.024

5.2 Log Analyzer

Table 2 shows noticeable improvements in all tasks compared to the drafter stage. Success rates for *Setup* and *Download* increase to around 0.34 to 0.40, while complex tasks see gains up to 0.18.

Table 2: Analyzer Success Metrics of Different Models.

Model Type	Name	S	D	T	E	I
Claude	Instant	0.342	0.353	0.104	0.109	0.151
	3-Haiku	0.350	0.301	0.132	0.037	0.130
	3-Sonnet	0.388	0.400	0.168	0.116	0.129
GPT	4o-Mini	0.347	0.317	0.118	0.078	0.131
	4o	0.362	0.353	0.148	0.115	0.145
	4-Turbo	0.353	0.322	0.161	0.094	0.148
Llama	3-70B	0.361	0.382	0.111	0.185	0.176
	3.1-8B	0.304	0.386	0.183	0.114	0.123
	3.1-70B	0.313	0.335	0.141	0.182	0.151
Mistral	Large	0.324	0.349	0.121	0.143	0.144
	Large-2	0.340	0.357	0.152	0.199	0.163

Analyzers leverage dynamic feedback from the execution of commands to refine scripts and try to correct errors from executed commands.

5.3 Issue Retriever

In the Issue Retriever stage (Table 3), success rates continue to improve, especially for complex tasks like **Training**, **Evaluation**, and **Inference**, reaching up to 0.25.

The results show that access to a knowledge base with informative discussions on issues of the code repository allows LLMs to retrieve solutions to execution errors from earlier stages, enhancing performance in complex operations.

5.4 Web Searcher

The Searcher Success Metrics in Table 4 exhibits the highest performance. Success rates for *Setup* and *Download* reach up to 0.46, and complex tasks improve to between 0.15 and 0.29.

Table 3: Issue Retriever Success Metrics of Different Models.

Model Type	Name	S	D	T	E	I
Claude	Instant	0.365	0.369	0.129	0.130	0.169
	3-Haiku	0.374	0.329	0.139	0.061	0.143
	3-Sonnet	0.442	0.436	0.254	0.183	0.163
GPT	4o-Mini	0.375	0.367	0.171	0.122	0.159
	4o	0.379	0.375	0.169	0.128	0.160
	4-Turbo	0.377	0.381	0.178	0.126	0.164
Llama	3-70B	0.364	0.399	0.113	0.159	0.154
	3.1-8B	0.305	0.389	0.182	0.115	0.122
	3.1-70B	0.312	0.334	0.143	0.179	0.153
Mistral	Large	0.357	0.358	0.174	0.155	0.153
	Large-2	0.359	0.380	0.181	0.155	0.152

Table 4: Searcher Success Metrics of Different Models.

Model Type	Name	S	D	T	E	I
Claude	Instant	0.388	0.406	0.151	0.131	0.190
	3-Haiku	0.385	0.338	0.155	0.070	0.163
	3-Sonnet	0.467	0.467	0.291	0.194	0.189
GPT	4o-Mini	0.412	0.405	0.201	0.131	0.179
	4o	0.415	0.407	0.198	0.130	0.183
	4-Turbo	0.416	0.406	0.200	0.133	0.182
Llama	3-70B	0.380	0.442	0.200	0.173	0.170
	3.1-8B	0.318	0.447	0.201	0.157	0.165
	3.1-70B	0.344	0.463	0.183	0.196	0.184
Mistral	Mistral-Large	0.375	0.450	0.199	0.174	0.168
	Mistral-Large-2	0.373	0.452	0.201	0.172	0.169

Web search enables models to find up-to-date solutions, resolving issues that previous agents could not, leading to substantial improvements.

5.5 Aggregated Results

We show the aggregated results of a single LLM on different tasks with different level of engagements of multi-agents in Figure 8, Figure 9, Figure 10, and Figure 11. In short, with more agents contributing to solving the tasks, the success rate increases across all tasks for all LLMs, which demonstrates the effectiveness of our proposed CSR-Agents.

5.6 Results Interpretation

The evaluation of the CSR-Bench involved assessing the performance of various large language models (LLMs) across key tasks: Setup, Download, Training, Inference, and Evaluation. These tasks are essential for deploying repositories within the CSR-Bench.

5.6.1 Task-Specific Outcomes

Setup and Download: Most models consistently performed well, reflecting their capability to initiate and manage basic deployment processes.

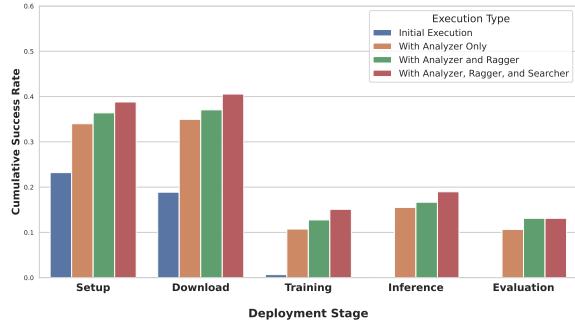


Figure 8: Performance of Claude 3 Sonnet

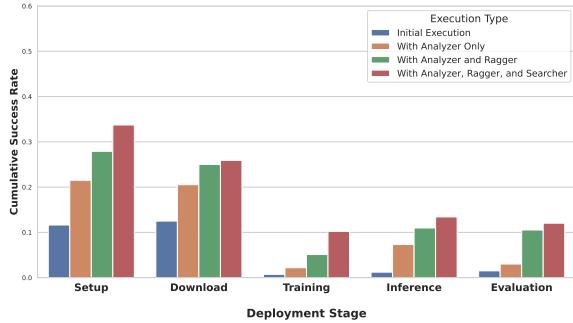


Figure 9: Performance of GPT 40

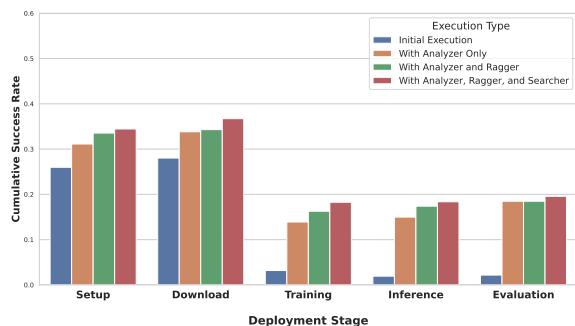


Figure 10: Performance of Llama 3.1 70B Instruct

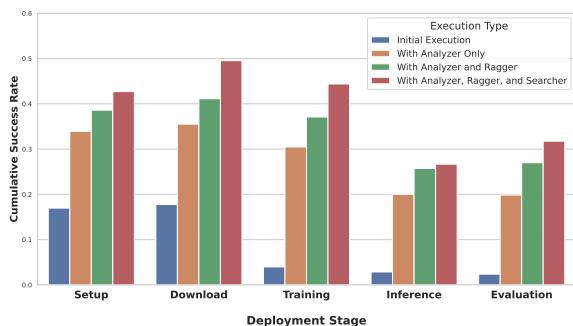


Figure 11: Performance of Mistral Large 2

Inference and Evaluation: Performance was less consistent, with some models demonstrating moderate success, but generally struggling with the complexity of these tasks.

Training: Training tasks are particularly challenging, with lower success rates across the board, indicating that current LLMs require further refinement to handle training processes effectively.

5.6.2 Overall Performance

The success metrics across different tasks and models indicate a wide variability in performance. Generally, models showed higher success rates in Setup and Download tasks, with performance tapering off in more complex tasks such as Inference, Evaluation, and Training. This pattern highlights the challenges LLMs face in handling the full deployment process autonomously.

The results demonstrate that while LLMs have made significant strides in automating repository deployment, their ability to manage complex tasks remains limited. Improvements are needed, particularly in the areas of Inference and Training, to achieve fully autonomous and reliable deployment of science repositories.

However, there is still a large gap between LLMs and real scientists even if the advanced tools are provided to the LLMs. To explain, it is not trivial to

handle the nuances in the experiment environment setup for the science repositories. For example, the hardware and software compatibility issues are very common in code deployment and often causes confusions even for domain experts.

6 Conclusion

The work introduces CSR-Bench, a benchmark designed to evaluate the capabilities of LLM agents in automating the deployment of GitHub repositories for scientific research. Our study highlights that while LLMs show potential in handling tasks like environment setup and data preparation, they face challenges in complex tasks such as training and inference, where success rates are notably lower.

Our multi-agent framework, CSR-Agents, exemplifies how LLMs can collaborate to tackle deployment challenges, offering a promising approach to improving automation in software engineering. However, the results indicate that further advancements are needed to fully realize autonomous and reliable deployment processes.

Overall, CSR-Bench serves as a crucial tool for assessing and improving LLM-driven deployment workflows in scientific research, paving the way for more efficient and automated computer science projects exploration.

Limitations

Although our benchmark framework supports several tools to facilitate large language model agents in the code deployment task, it does not actually improve the original reasoning capabilities of the large language models that are used in the agents. To improve LLMs' reasoning capabilities for this specific task, the community may resort to techniques like RLHF, which is orthogonal to this work. Our benchmark only focuses on code repositories that related to computer science research topics, and does not involve other types of repositories. Although this framework can be reused for other types of the repositories, we do not explore that direction in this work, and leave it to future works.

References

2024. Agent-101: A software engineering agent for code assistance developed by ibm research. https://github.com/swe-bench/experiments/blob/main/evaluation/lite/20240612_IBM_Research_Agent101/README.md/.
2024. Amazon q developer the most capable generative ai-powered assistant for software development. <https://aws.amazon.com/q/developer/>.
2024. Devin, ai software engineer. <https://www.cognition.ai/introducing-devin>.
2024. Factory bringing autonomy to software engineering. <https://www.factory.ai/>.
2024. Lingma agent. https://github.com/swe-bench/experiments/tree/main/evaluation/lite/20240622_Lingma_Agent.
2024. Marscode code and innovate faster with ai. <https://www.marscode.com/>.
- 2024a. Opencsg starship. <https://opencsg.com/product?class=StarShip/>.
- 2024b. Opendedvin: Code less, make more. <https://github.com/OpenDevin/OpenDevin/>.
2024. Swe-bench lite. <https://www.swebench.com/lite.html>.
- Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.
- Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. 2021. Program synthesis with large language models. *Preprint, arXiv:2108.07732*.
- Islem Bouzenia, Premkumar Devanbu, and Michael Pradel. 2024. Repairagent: An autonomous, llm-based agent for program repair. *arXiv preprint arXiv:2403.17134*.
- Federico Cassano, John Gouwar, Daniel Nguyen, Sydney Nguyen, Luna Phipps-Costin, Donald Pinckney, Ming-Ho Yee, Yangtian Zi, Carolyn Jane Anderson, Molly Q Feldman, et al. 2023. Multipl-e: A scalable and polyglot approach to benchmarking neural code generation. *IEEE Transactions on Software Engineering*.
- Dong Chen, Shaolin Lin, Muhan Zeng, Daoguang Zan, Jian-Gang Wang, Anton Cheshkov, Jun Sun, Hao Yu, Guoliang Dong, Artem Aliev, et al. 2024. Coder: Issue resolving with multi-agent and task graphs. *arXiv preprint arXiv:2406.01304*.

- Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Koplán, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.
- Yinlin Deng, Chunqiu Steven Xia, Haoran Peng, Chenyuan Yang, and Lingming Zhang. 2023. Large language models are zero-shot fuzzers: Fuzzing deep-learning libraries via large language models. In *32nd International Symposium on Software Testing and Analysis (ISSTA)*.
- Yinlin Deng, Chunqiu Steven Xia, Chenyuan Yang, Shizhuo Dylan Zhang, Shujing Yang, and Lingming Zhang. 2024. Large language models are edge-case fuzzers: Testing deep learning libraries via fuzzgpt. In *46th International Conference on Software Engineering (ICSE)*.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.
- Paul Gauthier. 2024. Aider is ai pair programming in your terminal. <https://aider.chat/>.
- Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Y Wu, YK Li, et al. 2024. Deepseek-coder: When the large language model meets programming—the rise of code intelligence. *arXiv preprint arXiv:2401.14196*.
- Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, and Jacob Steinhardt. 2021. Measuring coding challenge competence with apps. *NeurIPS*.
- Sirui Hong, Xiawu Zheng, Jonathan Chen, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, et al. 2023. Metagpt: Meta programming for multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352*.
- Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, and Luke Zettlemoyer. 2018. Mapping language to code in programmatic context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1643–1652.
- Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2024. Live-codebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint*.
- Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R Narasimhan. 2024. *SWE-bench: Can language models resolve real-world github issues?* In *The Twelfth International Conference on Learning Representations*.
- Sungmin Kang, Juyeon Yoon, and Shin Yoo. 2023. Large language models are few-shot testers: Exploring llm-based general bug reproduction. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, pages 2312–2323. IEEE.
- Yuhang Lai, Chengxi Li, Yiming Wang, Tianyi Zhang, Ruiqi Zhong, Luke Zettlemoyer, Wen-tau Yih, Daniel Fried, Sida Wang, and Tao Yu. 2023. Ds-1000: A natural and reliable benchmark for data science code generation. In *International Conference on Machine Learning*, pages 18319–18345. PMLR.
- Caroline Lemieux, Jeevana Priya Inala, Shuvendu K Lahiri, and Siddhartha Sen. 2023. Codamosa: Escaping coverage plateaus in test generation with pre-trained large language models. In *45th International Conference on Software Engineering (ICSE)*.
- Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, et al. 2023. Starcoder: may the source be with you!
- Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittweiser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, et al. 2022a. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097.
- Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittweiser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, Thomas Hubert, Peter Choy, Cyprien de Masson d’Autume, Igor Babuschkin, Xinyun Chen, Po-Sen Huang, Johannes Welbl, Sven Gowal, Alexey Cherepanov, James Molloy, Daniel J. Mankowitz, Esme Sutherland Robson, Pushmeet Kohli, Nando de Freitas, Koray Kavukcuoglu, and Oriol Vinyals. 2022b. *Competition-level code generation with alphacode*. *Science*.
- Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. 2023a. *Is your code generated by chatGPT really correct? rigorous evaluation of large language models for code generation*. In *Thirty-seventh Conference on Neural Information Processing Systems*.
- Tianyang Liu, Canwen Xu, and Julian McAuley. 2023b. Repobench: Benchmarking repository-level code auto-completion systems. *arXiv preprint arXiv:2306.03091*.
- Yuliang Liu, Xiangru Tang, Zefan Cai, Junjie Lu, Yichi Zhang, Yanjun Shao, Zexuan Deng, Helan Hu, Zengxian Yang, Kaikai An, et al. 2023c. Mil-bench: Large language models leverage open-source libraries for machine learning tasks. *arXiv e-prints*, pages arXiv–2311.
- Anton Lozhkov, Raymond Li, Loubna Ben Allal, Federico Cassano, Joel Lamy-Poirier, Nouamane Tazi, Ao Tang, Dmytro Pykhtar, Jiawei Liu, Yuxiang Wei, Tianyang Liu, Max Tian, Denis Kocetkov, Arthur Zucker, Younes Belkada, Zijian Wang, Qian Liu,

- Dmitry Abulkhanov, Indraneil Paul, Zhuang Li, Wen-Ding Li, Megan Risdal, Jia Li, Jian Zhu, Terry Yue Zhuo, Evgenii Zheltonozhskii, Nii Osae Osae Dade, Wenhao Yu, Lucas Krauß, Naman Jain, Yixuan Su, Xuanli He, Manan Dey, Edoardo Abati, Yekun Chai, Niklas Muennighoff, Xiangru Tang, Muhtasham Oblokulov, Christopher Akiki, Marc Marone, Cheng-hao Mou, Mayank Mishra, Alex Gu, Binyuan Hui, Tri Dao, Armel Zebaze, Olivier Dehaene, Nicolas Patry, Canwen Xu, Julian McAuley, Han Hu, Torsten Scholak, Sebastien Paquet, Jennifer Robinson, Carolyn Jane Anderson, Nicolas Chapados, Mostofa Patwary, Nima Tajbakhsh, Yacine Jernite, Carlos Muñoz Ferrandis, Lingming Zhang, Sean Hughes, Thomas Wolf, Arjun Guha, Leandro von Werra, and Harm de Vries. 2024. [Starcoder 2 and the stack v2: The next generation](#). Preprint, arXiv:2402.19173.
- Ziyang Luo, Can Xu, Pu Zhao, Qingfeng Sun, Xubo Geng, Wenxiang Hu, Chongyang Tao, Jing Ma, Qingwei Lin, and Dixin Jiang. 2023. Wizardcoder: Empowering code large language models with evol-instruct. arXiv preprint arXiv:2306.08568.
- Yingwei Ma, Qingping Yang, Rongyu Cao, Binhua Li, Fei Huang, and Yongbin Li. 2024. How to understand whole software repository? arXiv preprint arXiv:2406.01422.
- Martin Monperrus. 2018. *The living review on automated program repair*. Ph.D. thesis, HAL Archives Ouvertes.
- Rangeet Pan, Ali Reza Ibrahimzada, Rahul Krishna, Divya Sankar, Lambert Pouguem Wassi, Michele Merler, Boris Sobolev, Raju Pavuluri, Saurabh Sinha, and Reyhaneh Jabbarvand. 2024. Lost in translation: A study of bugs introduced by large language models while translating code. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, pages 1–13.
- Noah Patton, Kia Rahmani, Meghana Missula, Joydeep Biswas, and Işıl Dillig. 2024. Programming-by-demonstration for long-horizon robot tasks. *Proceedings of the ACM on Programming Languages*, 8(POPL):512–545.
- Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémie Rapin, et al. 2023. Code llama: Open foundation models for code. arXiv preprint arXiv:2308.12950.
- Baptiste Rozière, Marie-Anne Lachaux, Lowik Chanusot, and Guillaume Lample. 2020. Unsupervised translation of programming languages. *Advances in neural information processing systems*, 33:20601–20611.
- Baptiste Rozière, Jie M Zhang, Francois Charton, Mark Harman, Gabriel Synnaeve, and Guillaume Lample. 2021. Leveraging automated unit tests for unsupervised code translation. arXiv preprint arXiv:2110.06773.
- Yuxiang Wei, Zhe Wang, Jiawei Liu, Yifeng Ding, and Lingming Zhang. 2023. Magicoder: Source code is all you need. arXiv preprint arXiv:2312.02120.
- Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. 2023. The rise and potential of large language model based agents: A survey. arXiv preprint arXiv:2309.07864.
- Chunqiu Steven Xia, Yinlin Deng, Soren Dunn, and Lingming Zhang. 2024a. Agentless: Demystifying llm-based software engineering agents. arXiv preprint arXiv:2407.01489.
- Chunqiu Steven Xia, Matteo Paltenghi, Jia Le Tian, Michael Pradel, and Lingming Zhang. 2024b. Universal fuzzing via large language models. In *46th International Conference on Software Engineering (ICSE)*.
- Chunqiu Steven Xia, Yuxiang Wei, and Lingming Zhang. 2023. Automated program repair in the era of large pre-trained language models. In *Proceedings of the ACM/IEEE 45th International Conference on Software Engineering, ICSE ’23*.
- Chunqiu Steven Xia and Lingming Zhang. 2023. Keep the conversation going: Fixing 162 out of 337 bugs for \$0.42 each using chatgpt. arXiv preprint arXiv:2304.00385.
- John Yang, Carlos E Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. 2024. Swe-agent: Agent-computer interfaces enable automated software engineering. arXiv preprint arXiv:2405.15793.
- Pengcheng Yin, Wen-Ding Li, Kefan Xiao, Abhishek Rao, Yeming Wen, Kensen Shi, Joshua Howland, Paige Bailey, Michele Catasta, Henryk Michalewski, Alex Polozov, and Charles Sutton. 2022. Natural language to code generation in interactive data science notebooks.
- Fengji Zhang, Bei Chen, Yue Zhang, Jacky Keung, Jin Liu, Daoguang Zan, Yi Mao, Jian-Guang Lou, and Weizhu Chen. 2023. Repocoder: Repository-level code completion through iterative retrieval and generation. arXiv preprint arXiv:2303.12570.
- Yuntong Zhang, Haifeng Ruan, Zhiyu Fan, and Abhik Roychoudhury. 2024. Autocoderover: Autonomous program improvement. Preprint, arXiv:2404.05427.
- Qinkai Zheng, Xiao Xia, Xu Zou, Yuxiao Dong, Shan Wang, Yufei Xue, Zihan Wang, Lei Shen, Andi Wang, Yang Li, et al. 2023. Codegeex: A pre-trained model for code generation with multilingual evaluations on humaneval-x. arXiv preprint arXiv:2303.17568.
- Tianyu Zheng, Ge Zhang, Tianhao Shen, Xueling Liu, Bill Yuchen Lin, Jie Fu, Wenhui Chen, and Xiang Yue. 2024. Opencodeinterpreter: Integrating code generation with execution and refinement. arXiv preprint arXiv:2402.14658.

A Appendix

A.1 Categories and Subcategories

Computer Vision and Graphics

• 3D Vision and Reconstruction

- Structured3D: Structured 3D modeling dataset for computer vision and graphics (ECCV 2020)
- RayDF: Light ray-based 3D rendering and learning (NeurIPS 2023)
- SSDNeRF: Diffusion-based NeRF for 3D reconstruction (ICCV 2023)
- NeuralPull: Learning distance functions for point clouds (ICML 2021)
- DAD-3DHeads: Dataset for 3D head alignment and reconstruction (CVPR 2022)
- LeGO-LOAM: Lidar-based odometry and mapping for robotics

• Image Segmentation, Processing, and Enhancement

- Unsupervised-Semantic-Segmentation: Unsupervised methods for semantic segmentation (ICCV 2021)
- Seg-Uncertainty: Scene adaptation for self-driving cars (IJCAI 2020, IJCV 2021)
- RGBD_Semantic_Segmentation_PyTorch: RGBD scene recognition and segmentation (ECCV 2020)
- BCNet: Occlusion-aware segmentation (CVPR 2021)
- Deep_Metric: Metric learning for image retrieval
- Multiview2Novelview: Novel view synthesis from multiple views (ECCV 2018)
- NeRCo: Low-light image enhancement using neural representations (ICCV 2023)
- Transfiner: High-quality instance segmentation with mask transformers (CVPR 2022)

• Pose Estimation and Object Detection

- HybrIK: Hybrid kinematics for 3D human pose (CVPR 2021)
- SmoothNet: Human pose refinement in videos (ECCV 2022)

- FSA-Net: Head pose estimation from images (CVPR 2019)
- VisualDet3D: Monocular 3D detection for autonomous driving
- DEVIANT: Monocular 3D object detection for autonomous driving (ECCV 2022)
- OWOD: Open World Object Detection with incremental learning (CVPR 2021)

Machine Learning Models and Techniques

• Generative and Diffusion Models

- DDM2: Diffusion models for MRI denoising (ICLR 2023)
- DDNM: Diffusion models for zero-shot image restoration (ICLR 2023)
- diffusion-point-cloud: Generating 3D point clouds with diffusion models (CVPR 2021)
- GeoDiff: Diffusion models for molecular structures (ICLR 2022)
- probabilistic_unet: Conditional segmentation with probabilistic UNet (NeurIPS 2018)
- LTSF-Linear: Linear models for time series forecasting (AAAI 2023)
- unconditional-time-series-diffusion: Diffusion models for time series prediction (NeurIPS 2023)
- dyffusion: Spatiotemporal forecasting using diffusion models (NeurIPS 2023)

• Graph Neural Networks and Transformers

- GraphSAINT: Scalable graph neural networks with sampling (ICLR 2020, IPDPS 2019)
- Graph-Transformer: Applying transformers to graph data (WWW 2022)
- graphtransformer: Generalized transformers for graphs (DLG-AAAI 2021)
- GPT-GNN: Pre-training techniques for graph neural networks (KDD 2020)
- GraphMAE: Masked autoencoders for graph learning (KDD 2022)

• Self-Supervised and Contrastive Learning

- skip-connections-matter: Impact of skip connections on adversarial example transferability (ICLR 2020)

- SparK: BERT-style pretraining on convolutional networks (ICLR 2023)
- imbalanced-semi-self: Semi-supervised learning for class-imbalanced datasets (NeurIPS 2020)
- CURL: Image enhancement via neural curve layers (ICPR 2020)

- **Model Interpretability and Optimization**

- rrl: Rule-based learning for interpretable classification (NeurIPS 2021)
- gradient-descent-the-ultimate-optimizer: Advanced optimization techniques (NeurIPS 2022)

Large Language Models (LLMs)

- **LLM Development and Optimization**

- llama-recipes: Fine-tuning scripts for LLaMA models
- llama3: Official GitHub repository for LLaMA 3
- alpaca-lora: Instruct-tuning LLaMA on consumer hardware
- MetaGPT: Multi-agent AI development framework using LLMs
- Stanford_Alpaca: Training and dataset generation for Alpaca models

- **Inference and Efficiency in LLMs**

- mistral-inference: Efficient inference for Mistral models
- PowerInfer: Fast LLM serving on consumer-grade GPUs
- direct-preference-optimization: Implementation of direct preference optimization (DPO)
- GPTCache: Efficient caching for LLMs integrated with LangChain and llama_index

- **LLM Applications and Evaluation**

- alpaca_eval: Automatic evaluation for instruction-following LLMs
- DART: Enhancing few-shot learning with differentiable prompts (ICLR 2022)
- LRV-Instruction: Reducing hallucination in multi-modal models (ICLR 2024)
- alpaca_farm: Simulated framework for reinforcement learning from human feedback (RLHF)

- FastChat: Open-source platform for training and evaluating large language models

Natural Language Processing (NLP)

- **Text Classification and Information Extraction**

- HVPNet: Multimodal entity and relation extraction (NAACL 2022)
- PURE: Simple approach to relation extraction (NAACL 2021)
- SimCSE: Contrastive sentence embeddings (EMNLP 2021)
- Text-GCN: Text classification using graph convolutional networks (AAAI 2019)

- **Multimodal and Knowledge-Based Systems**

- OntoProtein: Protein function prediction using knowledge graphs (ICLR 2022)
- tree-of-thought-llm: Structured problem-solving with LLMs (NeurIPS 2023)
- storm: Automated report generation using large language models
- SWE-agent: AI-based code bug fixing using LLMs

- **Transfer Learning and Domain Adaptation**

- SubpopBench: Benchmark for subpopulation shifts in domain generalization (ICML 2023)
- naacl_transfer_learning_tutorial: Transfer learning techniques in NLP (NAACL 2019)
- private-transformers: Training transformers with differential privacy
- imbalanced-regression: Handling data imbalance in regression tasks (ICML 2021)

Domain-Specific Applications

- **Healthcare and Bioinformatics**

- imbalanced-regression: Addressing imbalances in healthcare data (ICML 2021)
- hyena-dna: Long-range genomic modeling with Hyena
- OntoProtein: Protein function prediction with pretraining (ICLR 2022)
- GeoDiff: Molecular structure generation using diffusion models (ICLR 2022)

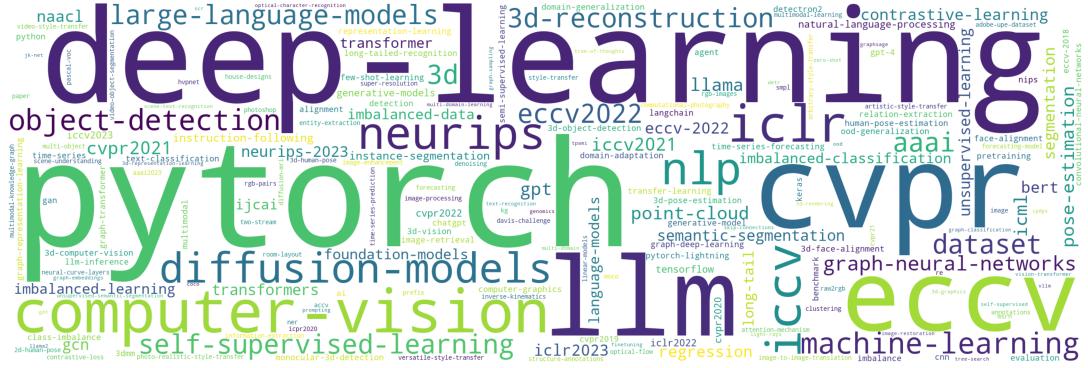


Figure 12: Distribution of Topics in GSRBench100

Table 5: Categories, Subcategories, and Abbreviations

Category	Subcategory	Subcategory Abbreviation
CV	3D Vision & Reconstruction	3D Vision & Rec.
	Image Segmentation & Enhancement	Img Segm. & Enh.
	Pose Estimation & Object Detection	Pose Estim. & Obj Det.
ML	Generative & Diffusion Models	Gen. & Diff. Models
	Graph Neural Networks & Transformers	GNNs & Transf.
	Self-Supervised & Contrastive Learning	Self-Sup. & Contr. Learn.
	Model Interpretability & Optimization	Model Interp. & Opt.
NLP	Text Classification & Extraction	Txt Classif. & Extr.
	Multimodal & Knowledge Systems	Multi. & Knowl. Sys.
	Transfer Learning & Domain Adaptation	Transf. Learn. & Dom. Adap.
LLM	LLM Development & Optimization	LLM Dev. & Opt.
	Inference & Efficiency in LLMs	Infer. & Eff. in LLMs
	LLM Applications & Evaluation	LLM Appl. & Eval.
Interdisciplinary	Healthcare & Bioinformatics	Health. & Bioinf.
	Robotics & Autonomous Systems	Robotics & Auto. Sys.
	Computational Creativity & Arts	Comp. Creativ. & Arts

- tape: Benchmark tasks for protein sequence modeling
 - **Robotics and Autonomous Systems**
 - LeGO-LOAM: Lidar-based odometry and mapping for robotics
 - SuperGlobal: Image retrieval using global features (ICCV 2023)
 - Deep_Metric: Embedding learning for metric-based image retrieval
 - FEARTracker: Robust visual tracking (ECCV 2022)
 - **Computational Creativity and Visual Arts**
 - CCPL: Artistic style transfer for images and videos (ECCV 2022)
 - GANgealing: GAN-based visual alignment (CVPR 2022)
 - NeRCo: Low-light image enhancement using neural representations (ICCV 2022)

2023)

A.2 Word Cloud of Topics

Figure 12 presents a word cloud that visualizes the frequency of topics across the repositories, highlighting the primary focus areas within GSR-Bench100. It provides a quick overview of the thematic concentration of the dataset.

B Repository list and Commit ID

We specify the commit ID we used in our benchmark dataset for reproducibility. Please see table 6 and table 7 for details.

C Categories, Subcategories, and Abbreviations

C.0.1 Example Text from README

Below are case studies of how models (GPT-4o, Claude 3, LLaMA 3.1, and Mistral) extract the commands from a Repository.

Table 6: GitHub Repository Details - Part 1

Repository URL	Commit ID	Branch
https://github.com/cleardusk/3DDFA_V2	1b6c67601abffc1e9f248b291708aef0e43b55ae	master
https://github.com/lkeab/BCNet	d6580e8a2a0b5e71c0ae6913ed0340c101d35723	main
https://github.com/JarrentWu1031/CCPL	d0b6b326d7d624b5e8d1543a3a84a745a08fd883	main
https://github.com/sjmoran/CURL	4be9753a8063f9833423e4aa5947ae7a64b114f8	master
https://github.com/tiangexiang/CurveNet	c2e7cf642b7e08d9aec5b70263f6989a85c9e191	main
https://github.com/PinataFarms/DAD-3DHeads	3acc5c2a1177d354a1247c49e44a83ad682ea6a1	main
https://github.com/zjunlp/DART	d418ded8bd548ef25f2d030990e707a497e93483	main
https://github.com/StanfordMIMI/DDM2	d07be20ad36446f8e35621d4b0d92c7cf54c169e	main
https://github.com/wyhuai/DDNM	00b58eac7843a4c99114fd8fa42da7aa2b6808af	main
https://github.com/abhi1kumar/DEVIANT	009955f3bbb21c38a687eaae59dfcb82eca93e7	main
https://github.com/shenweichen/DSIN	e8ba406eeda0916214897d44866bffc419c3edb0	master
https://github.com/bnu-wangxun/Deep_Metric	04ca51093db13135a04c3c94401bc898c6af0c40	master
https://github.com/tjiiv-cprg/EPro-PnP	42412220b641ae9e8943ceba516b3175631d370	main
https://github.com/PinataFarms/FEARTracker	0a3bd039918909c79c1b7e55a4fb7807520abde	main
https://github.com/shamangary/FSA-Net	4361d0e48103bb215d15734220c9d17e6812bb48	master
https://github.com/lm-sys/FastChat	92a6d1fc69a88ea169c0b01065ce44f1e690a2c	main
https://github.com/acbull/GPT-GNN	f26e13c69ddc8a3f2580cb16d0b9a1c73d89f4bc	master
https://github.com/zilliztech/GPTCache	75ab7ec7b871c8399a95d5bf528441f2856250dd	main
https://github.com/MinkaiXu/GeoDiff	ea0ca48045a2f7abfccd7f0df449e45eb6eae638	main
https://github.com/daiquocnguyen/Graph-Transformer	99c88a116148fdaa8d3071fcc548e5c471ae607f	master
https://github.com/THUDM/GraphMAE	b14f080c919257b495e3cb6474286d5252d6a635	main
https://github.com/GraphSAINT/GraphSAINT	c9b1e340d7b951465ac4a9251ee93832e68b003	master
https://github.com/zjunlp/HVPNeT	52c77f7835a295d9c8534997b1316c42b2662972	main
https://github.com/IDEA-Research/HumanSD	c5db29dd66a3e40afa8b4bed630f0aa7ea001880	main
https://github.com/Jeff-sjtu/HybrIK	9b8681dcf3c902dd5dacc01520ba04982990e1e2	main
https://github.com/princeton-nlp/LM-BFF	c282f5210019c299d29ecc7b459266f2b14fbaf	main
https://github.com/FuxiaoLiu/LRV-Instruction	0a5ab538ed96c3e0c9835b5fe02cc8f7fa0bf8fa	main
https://github.com/cure-lab/LTSF-Linear	0c113668a3b88c4c4ee586b8c5ec3e539c4de5a6	main
https://github.com/RobustFieldAutonomyLab/LeGO-LOAM	896a7a95a8bc510b76819d4cc48707e344bad621	master
https://github.com/tfzhou/MATNet	c8b95e527c486c304f711cc7dfb060f31abe19f	master
https://github.com/MIVRC/MSRN-PyTorch	a0e038de7eb42e21d2e88c38e6490b61a02c566e	master
https://github.com/princeton-nlp/MeZo	552cb1b710767f9a6e1dc8f9645d7640376f9941	main
https://github.com/geekan/MetaGPT	5446c7e490e7203c61b2ff31181551b2c0f4a86b	main
https://github.com/shaohua0116/Multiview2Novelview	a5e236f3c564bf287c8a09d855fd2134ba86b299	master
https://github.com/Ysz2022/NeRCo	6b0e1112231d0902976ad76357044de582a307f3	main
https://github.com/mabaorui/NeuralPull	c093a52308a9b74446d24cc6c1b0fee5ee5bb7bb	master
https://github.com/JosephKJ/OWOD	23890f188cd1a6801c6ac0e3dacd78b8572b8c29	master
https://github.com/zjunlp/OntoProtein	6360f458e11670ecfaf853ee68f2087b31439dc0	main
https://github.com/princeton-nlp/PURE	b1e9cad39bec10eb3c355dc5a8e4e75dd0afebf5	main
https://github.com/SJTU-IPADS/PowerInfer	61cac9bf25e60336bbad27ada9dbb809204473ac	main
https://github.com/charlesCXX/RGBD_Semantic_Segmentation_PyTorch	32b3f86822d278103a13ea6f93f9668d3b631398	master
https://github.com/vLAR-group/RayDF	ca6c663523b777732788a5d8100d36251a482b31	master
https://github.com/Paranioar/SGRAF	50d0c6f9caf759099b28371046f780342357c405	main
https://github.com/Lakonik/SSDNeRF	b9d195db76b715c475b24287362d9627d77d3bb	main
https://github.com/shamangary/SSR-Net	f98b6cbe1c9c8c78649e5a331f94113564521525	master
https://github.com/princeton-nlp/SWE-agent	36e430d27ffd11269738d92d6c521cab2207dc8	main
https://github.com/daveredrum/ScanRefer	9d7483053e8d29acf4d4b4e1bc28f1564f5dddb	master
https://github.com/layumi/Seg-Uncertainty	6fce9eae141c2c0592b3e7c1b3e5f8ee7b1ce9a6	master
https://github.com/princeton-nlp/SimCSE	7edb07e05cec0d5293fc1696b578d8056dba76ef	main
https://github.com/cure-lab/SmoothNet	c03e93e8a14f55b9aa087dc2751a7a5e2d50b0	main

Table 7: GitHub Repository Details - Part 2

Repository URL	Commit ID	Branch
https://github.com/keyu-tian/Spark	a63e386f8e5186bc07ad7fce86e06b08f48a61ea	main
https://github.com/bertjiazheng/Structured3D	d06d1b241b290ae7ed6b311d710ffafffc567de	master
https://github.com/YyzHarry/SubpopBench	4d3dbbe21029666ef19d040e110ec22908640c5b	main
https://github.com/ShihaoShao-GH/SuperGlobal	86946964b907e6f28a7264add2c15640fae30009	main
https://github.com/codeKgu/Text-GCN	70b970a52efcb80235cf0ae3e578eaf80278d5f3	master
https://github.com/wvngansbeke/Unsupervised-Semantic-Segmentation	dfdf5fa0a1542fb2b26824b4059bd2bb1240c5c94b	main
https://github.com/tloen/alpaca-lora	8bb8579e403dc78e37fe81ffb253c413007323f	main
https://github.com/tatsu-lab/alpaca_eval	32c8c0d068205c38b02003b67e0becc82a8f1ac2	main
https://github.com/tatsu-lab/alpaca_farm	30717ddae735365de756ee2085191b491a71788d	main
https://github.com/adobe/antialiased-cnns	b27a34a26f3ab039113d44d83c54d0428598ac9c	master
https://github.com/richardaecn/class-balanced-loss	1d7857208a2abc03d84e35a9d5383af8225d4b4d	master
https://github.com/luost26/diffusion-point-cloud	1e30d48d018820fb7c67c8b3190215bd41878e4	main
https://github.com/eric-mitchell/direct-preference-optimization	f8b8c0f49dc92a430bae41585f9d467d3618fe2f	main
https://github.com/Rose-STL-Lab/dyffusion	832574f6f788a0cd4a4d75e8f59b3c07c7e8446b	main
https://github.com/wpeebles/gangealing	ffa6387c7ffd3f7de76bdc693dc2272e274e9bfd	main
https://github.com/openai/gpt-3	d7a9bb505df6f630f9bab3b30c889e52f22eb9ea	master
https://github.com/kach/gradient-descent-the-ultimate-optimizer	b3b047e02ca6d32e0e61e34a0ca6e0bc57e55bdf	main
https://github.com/microsoft/graphrag	61b5eea34783c58074b3c53f1689ad8a5ba6b6ee	main
https://github.com/graphdeeplearning/graphtransformer	c9cd49368eed4507f9ae92a137d90a7a9d7efc3a	main
https://github.com/HazyResearch/hyena-dna	d553021b483b82980aa4b868b37ec2d4332e198a	main
https://github.com/YyzHarry/imbalanced-regression	a6fdc45d45c04e6f5c40f43925bc66e580911084	main
https://github.com/YyzHarry/imbalanced-semi-self	b91ad29fd8805ddf0a146f735905b0c869e68ae4	master
https://github.com/meta-llama/llama	8fac8befd776bc03242fe7bc2236cdb41b6c609c	main
https://github.com/meta-llama/llama-recipes	8c1418e93b817cb6734a9cf095b270f5a0f48f5	main
https://github.com/meta-llama/llama3	18f515a3c3c5f02cf45c6ac56cc5d039488e867a	main
https://github.com/state-spaces/mamba	a71bb5a83bfa289b5807aefc1767232dee77b35e	main
https://github.com/kwotsin/mimicry	a7fd06c4aff1e6af8dc4c4a35ed6636e434c766	master
https://github.com/mistralai/mistral-finetune	5b8adb54a1263664d52dab6f94581bf24d7b59e3	main
https://github.com/mistralai/mistral-inference	1f583071dc7aad2ca35cb9896140316ffce5b65	main
https://github.com/YyzHarry/multi-domain-imbalance	2efbfef34542e365293f798d79f70cee5e54303	main
https://github.com/JiangWenPL/multiperson	e8ae029cc691f3f9c3958a23f762f3d72cf65c54	master
https://github.com/zju3dv/mvpose	38b958f423f2de2bf7562f5a386c27440eab8c53	master
https://github.com/huggingface/naacl_transfer_learning_tutorial	dc976775bb11ede24a77e2ce161450089c5e169	master
https://github.com/baudm/parseq	1902db043c029a7e03a3818c616c06600af574be	main
https://github.com/lxuechen/private-transformers	18ccc4eab7355e4ac96051a82434796f6aa4624b	main
https://github.com/SimonKohl/probabilistic_unet	7a2e79d549184d0f3a47d0deaa054a70b0f54a3f	master
https://github.com/Jeff-sjtu/res-loglikelihood-regression	203dc3195ee5a1led6f47c066fdb83247511359	master
https://github.com/12wang3/rrl	f8d0886b23c4e15f63c62c248b97d4eb73386ad1	main
https://github.com/csdongxian/skip-connections-matter	9b2e5cca9b673efcac253e16b2f55f6cd41a8692	master
https://github.com/xuchen-ethz/snarf	ae0c893cc049f0f8270eaa401e138dff5d4637b9	main
https://github.com/tatsu-lab/stanford_alpaca	761dc5bfbdeeffa89b8bff5d038781a4055f796a	main
https://github.com/stanford-oval/storm	263d894e34e2a399a18f80fbf3c9028dc06e1a1d	main
https://github.com/songlab-cal/tape	6d345c2b2bbf52cd32cf179325c222af92aec7e	master
https://github.com/SysCV/transfiner	5b61fb53d8df5484f44c8b7d8415f398fd283ddc	main
https://github.com/yzhq97/transmomo.pytorch	0e4d2f0e7e8af159e018c8a85f9d789de963a83a	master
https://github.com/princeton-nlp/tree-of-thought-llm	ab400345c5ea39d28ea6d73be0e417b11113c87	master
https://github.com/amazon-science/unconditional-time-series-diffusion	3eafeffdffef4d2f5344ac63e5627a746c331a3	main
https://github.com/elliottwu/unsup3d	dc961410d61684561f19525c2f7e9ee6f4dacb91	master
https://github.com/dddzg/up-detr	44c0c6eb4bddf409a41cbaae31b7360062495199	master
https://github.com/Owen-Liuyuxuan/visualDet3D	ad229f2e491cba2c48f9bb2c211fe672294e2398	master

REPOSITORY ID: 3DDFA_V2**README****Requirements**

See requirements.txt , tested on macOS and Linux platforms . The Windows users may refer to FQA for building issues . Note that this repo uses Python3 . The major dependencies are PyTorch , numpy , opencv-python , and onnxruntime , etc . If you run the demos with --onnx flag to do acceleration , you may need to install libomp first , i.e., brew install libomp on macOS .

Usage

```
# 1. Clone this repo  
git clone https://github.com/cleardusk/3DDFA_V2.git  
cd 3DDFA_V2  
  
# 2. Build the cython version  
sh ./build.sh  
  
# 3. Run demos  
# Running on a still image  
python3 demo.py -f examples/inputs/emma.jpg --onnx  
  
# Running on videos  
python3 demo_video.py -f examples/inputs/videos/214.avi --onnx  
  
# Running on webcam  
python3 demo_webcam_smooth.py --onnx
```

The implementation of tracking is simply by alignment . If the head pose > 90 degrees or the motion is too fast , the alignment may fail . A threshold is used to trickly check the tracking state , but it is unstable .

GPT-4o:

Extracted commands for Environment Setup / Requirement / Installation:

```
git clone https://github.com/cleardusk/3DDFA_V2.git; cd 3DDFA_V2; sh ./build.sh
```

Error message from bash shell:

```
"stderr": "Traceback (most recent call last):\n  File \"/workspace/3DDFA_V2/FaceBoxes/utils/build.py\", line 15, in <module>\n    from Cython.Distutils import build_ext\nModuleNotFoundError: No module named 'Cython'\nTraceback (most recent call last):\n  File \"/workspace/3DDFA_V2/Sim3DR/setup.py\", line 7, in <module>\n    from Cython.\nBuild import cythonize\nModuleNotFoundError: No module named 'Cython'",
```

Analyzer suggested solution:

```
"command": "git clone https://github.com/cleardusk/3DDFA_V2.git; cd 3DDFA_V2;\npip install cython; sh ./build.sh"
```

Table 8: Commands and errors for setting up 3DDFA_V2 repository using GPT-4o.

LLaMA 3.1:

Extracted commands for Environment Setup / Requirement / Installation:

```
pip install -r requirements.txt  
brew install libomp
```

Claude 3:

Extracted commands for Environment Setup / Requirement / Installation:

```
git clone https://github.com/cleardusk/3DDFA_V2.git  
cd 3DDFA_V2  
sh ./build.sh
```

Error message from bash shell:

```
"stderr": "Traceback (most recent call last):\n  File \"/workspace/3DDFA_V2/FaceBoxes/\nutils/build.py\", line 15, in <module>\n    from Cython.Distutils import build_ext\nModuleNotFoundError: No module named 'Cython'\nTraceback (most recent call last):\n  File \"/workspace/3DDFA_V2/Sim3DR/setup.py\", line 7, in <module>\n    from Cython.\nBuild import cythonize\nModuleNotFoundError: No module named 'Cython'",
```

Analyzer suggested solution:

```
"command": "pip install Cython && sh ./build.sh",
```

Mistral:

Extracted commands for Environment Setup / Requirement / Installation:

```
git clone https://github.com/cleardusk/3DDFA_V2.git && cd 3DDFA_V2 && sh ./build.sh &&  
brew install libomp
```

Error message from bash shell:

```
"stderr": "bash: brew: command not found"
```

Analyzer suggested solution:

```
"command": "sudo apt-get install homebrew && git clone https://github.com/cleardusk/\n3DDFA_V2.git && cd 3DDFA_V2 && sh ./build.sh && brew install libomp",
```

Table 9: Commands and errors for setting up 3DDFA_V2 repository using LLaMA 3.1, Claude 3 and Mistral.



GPT-4o: The Cutting-Edge Advancement in Multimodal LLM

Raisa Islam and Owana Marzia Moushi

EasyChair preprints are intended for rapid dissemination of research results and are integrated with the rest of EasyChair.

July 2, 2024

GPT-4o: The Cutting-Edge Advancement in Multimodal LLM

Raisa Islam

Computer Science

New Mexico Institute of Mining and Technology
Socorro, NM, USA
raisila.islam@student.nmt.edu

Owana Marzia Moushi

Electrical and Computer Engineering
University of Nebraska - Lincoln
Omaha, NE, USA
omoushi2@huskers.unl.edu

Abstract—GPT-4o marks a significant advancement in AI technology, enhancing multimodal capabilities. OpenAI has launched several GPT models over the years, with GPT-4o being the latest. This paper provides a concise overview of these models, focusing on their key features and technological advancements. The main objective is to present a brief overview of GPT-4o, including its technological innovations. GPT-4o offers substantial improvements over its predecessors by introducing multimodal capabilities, larger context windows, efficient tokenization, and faster processing speeds, achieving state-of-the-art performance in text, audio, video, and image generation and understanding. We have compared GPT-4o with ten top LLMs using metrics such as throughput, response time, and latency, where GPT-4o demonstrated clear superiority. Additionally, this paper explores various application domains, highlighting GPT-4o's versatility and potential to modernize multiple aspects of human life.

Index Terms—GPT-4o, LLM, OpenAI, AI, ChatGPT, multimodal

I. INTRODUCTION

Artificial Intelligence (AI) is one of the most popular cutting-edge technology in this era which is gaining attention because of its revolutionizing aspects in our life [1]. Natural Language Processing (NLP), a subset of AI, focuses on the interaction between computers and human languages, aiming to enable machines to understand, interpret, and generate human language meaningfully. The advent of Large Language Models (LLMs) has initiated a paradigm shift and revolutionized the field of NLP by leveraging deep learning techniques to process and generate natural language with unprecedented accuracy and fluency [2]. These models are trained on vast amounts of textual data, allowing them to capture intricate linguistic patterns and contextual nuances. As a result, LLMs have significantly enhanced applications in various domains, including chatbots, language translation, sentiment analysis, and content creation, driving forward the capabilities of AI in understanding and utilizing human language [3] [4].

OpenAI's Generative Pre-trained Transformer (GPT) models have fundamentally transformed the fields of AI and NLP over the past ten years. GPT models are built upon the transformer architecture, which has demonstrated exceptional effectiveness, particularly in developing LLMs. The first version, GPT-1, was launched in 2017. Subsequent versions, including GPT-2 and GPT-3, culminated in the groundbreaking ChatGPT.

Since its release in November 2022, ChatGPT has gained immense popularity, reaching 1 million users within a week and 100 million users within two months [5]. ChatGPT aims to simplify various tasks, such as coding, math solving, and problem-solving, by using a combination of pre-trained unsupervised models and fine-tuned supervised models to provide human-like responses [3]. However, the last version, GPT-4, had limitations in video chatting, image processing, and audio processing. To address these shortcomings and advance toward a more digital world, OpenAI has released a new version of ChatGPT, called GPT-4o.

The recent version of ChatGPT is based on GPT-4o architecture which is gaining success on the previous chatbots. It was released on May 13, 2024, called GPT-4omni (or GPT-4o) the latest multimodal LLM from OpenAI [6]. The term "omni," derived from the Latin word "omnis," meaning "all" or "every," highlights the model's omni-modal capabilities [7]. GPT-4o can process and understand multimodal inputs, including text, images, audio, and video, making it a significant advancement in AI technology. It is the first LLM capable of interpreting emotions from videos, enhancing user experience by analyzing various data types. Although the model is already available for use, OpenAI continues to improve its capabilities, with updates expected soon [8]. The model's efficiency in handling different types of data results in high success rates and reduced processing times and costs [9]. For users seeking unlimited access and enhanced features, OpenAI offers a paid version, such as ChatGPT Plus, which includes a higher message limit and upcoming macOS desktop app support [10] [11]. GPT-4o is twice as fast as GPT-4 Turbo (released after GPT-4) and represents a milestone in the digital interaction between machines and humans, promising to significantly impact the future of digital communication [7] [12] [13].

Tech giant Apple and OpenAI are collaborating to integrate ChatGPT into Apple's ecosystem, focusing on the upcoming iOS 18 operating system [14]. This partnership aims to enhance the user experience by embedding ChatGPT's advanced language capabilities into Apple services like Siri, making them more intuitive and responsive. Features include real-time conversation synchronization, and voice input through Whisper. These enhancements will allow Siri to process

	GPT-4o	GPT-4	GPT-3.5	GPT-3
Initial Release Date	May 13, 2024	March 14, 2023	March 15, 2022	June 11, 2020
Modality	Audio, video, text, images	Text and images	Text	Text
Context window [2] [15]	128000	8192	4096	2048
Parameters [2] [16] [17]	Yet-to-Disclose (YTD)	1.76 trillion	175 billion	175 billion
Cost per 1M token [18] [19]	Input: \$5 Output: \$15	Input: \$30 Output: \$60	Input: \$1.5 Output: \$2	\$0.4 to \$20
Decoder layers [16] [17] [20]	YTD	120	96	96
Prompting Method [17] [21]	YTD	Chain-of-Thought, n -shot ¹	n -shot	n -shot
Training Data [17] [22]	YTD	~ 13T tokens which includes both text data and code data	Over 570GB of text data	About 45TB text data
Response Time [23] [24]	30% faster than GPT-4	94ms per token	35ms per token	Unknown
Performance	State-of-the-art performance	Advanced in handling complex tasks, slower response time than GPT-4o	Improved version of GPT-3, lacks depth of understanding	Poor on complex tasks

TABLE I: Summary of latest GPT models

complex queries more effectively and provide functionalities such as instant answers, tailored advice, creative inspiration, professional input, and educational support. This integration aligns with Apple's broader strategy to incorporate AI-driven features while ensuring privacy by powering most features on-device.

A. Evolution of GPT

Table I provides an overview of the latest GPT models to understand the improvement OpenAI is making for the updated GPT models. GPT-4o significantly enhances the capabilities of its predecessors by incorporating multimodal functionalities, including text, audio, image, and video processing. It offers faster response times and reduced costs compared to GPT-4, while maintaining high accuracy and performance, surpassing the improvements seen in GPT-3.5 and GPT-3.

Contribution: The main contributions of this paper are as follows:

- We have provided an in-depth overview of GPT-4o, detailing the technologies that contribute to its advancement over previous models. This study also summarizes the key features and advantages of GPT-4o, highlighting the significant improvements it offers compared to its predecessors.
- We have analyzed the text and vision performance comparisons provided by the OpenAI team. Additionally, a comparison of GPT-4o with other leading LLMs using metrics such as throughput, response time, and latency has been discussed. The results demonstrate the superior performance of GPT-4o.
- Furthermore, we have presented the various fields where GPT-4o can be applied, showcasing its versatility and potential impact.

The structure of this paper is as follows: Section II provides an overview of the newly launched GPT-4o, including its technologies, features, advantages, and challenges. Section III analyzes the performance of GPT-4o by comparing it with popular LLM models, and Section IV explores the application domains where GPT-4o can be utilized. Finally, Section V concludes the paper.

¹the value of n starts from 0

II. GPT-4O OVERVIEW

This section provides an overview of GPT-4o, detailing the technologies employed in its development, its features, and its advantages. While GPT-4o incorporates numerous updated technologies, it also faces some challenges that need to be addressed. These aspects, including both the strengths and areas for improvement, are comprehensively discussed here.

A. Technology

GPT-4o builds upon the architecture of its predecessors, integrating enhancements in context window size, tokenization efficiency, and multimodal capabilities, which include processing text, audio, video, and images. This single model is trained end-to-end across text, vision, and audio, ensuring all inputs and outputs are managed by the same neural network [8]. It employs refined reinforcement learning with human feedback (RLHF) [25], significantly improving its alignment with human values and ethical standards. GPT-4o also features an advanced transformer architecture with enhanced self-attention mechanisms, allowing for better comprehension and generation of nuanced, contextually relevant responses. Utilizing Nvidia's most advanced GPUs [26], known for their parallel processing capabilities, GPT-4o efficiently handles the massive computations required by its sophisticated architecture. This model gives better performance for scalability since it uses larger training sessions [25]. GPT-4o's rate limits are 5x higher than GPT-4 Turbo—up to 10 million tokens per minute. These advancements make GPT-4o a versatile tool for applications ranging from conversational AI to content generation and data analysis. The following subsection discusses the technologies that are being used in GPT-4o.

1) o200k_base Tokenizer: GPT-4o introduced a new *o200k_base* tokenizer algorithm, marking a shift from the *cl100k_base* tokenizer used by GPT-4, GPT-4 Turbo, and GPT-3.5 Turbo². Tokenization, which breaks down text into smaller units called tokens, is critical in NLP. The *o200k_base* tokenizer improves upon previous methods by being faster and more efficient, allowing GPT-4o to process and generate language at unprecedented speeds. It enhances semantic coherence in generated text and improves the handling of multiple languages [27], expanding GPT-4o's applicability across various linguistic contexts.

²<https://github.com/openai/tiktoken>

2) *RAG-GPT*: RAG-GPT³ is an advanced implementation of Retrieval-Augmented Generation (RAG) technology, designed to enhance the capabilities of LLMs by integrating them with efficient document retrieval systems. This integration allows RAG-GPT to provide more accurate and contextually relevant responses by fetching and incorporating information from extensive knowledge bases during query processing. Integration of RAG-GPT in GPT-4o enables delivering precise and grounded answers, making it particularly useful for applications requiring detailed and up-to-date information. This approach not only improves the quality of generated content but also enables the handling of specialized and complex queries with greater reliability.

3) *Context Window*: GPT-4o features an impressive context window size of 128k tokens [6]. The larger context window allows the model to maintain and process a much larger amount of information within a single interaction, enhancing its ability to understand and generate responses that are contextually relevant over extended conversations. This large context window is particularly beneficial for complex tasks requiring extensive context retention, tracking multiple threads of a conversation, and integration of multiple pieces of information.

4) *Cloud Infrastructure and API Access*: GPT-4o is designed to be deployed on scalable cloud infrastructures (Microsoft Azure) [28] [29], offering flexible API access for diverse applications. This cloud-based deployment model allows for seamless scaling to meet varying user demands and workloads.

B. Features

OpenAI is improving its models to make the world digital by adding some additional features which are discussed below:

1) *Multimodality*: GPT-4o is the newest multimodal LLM. It can understand and generate spoken languages, recognizing speech to transcribe text and using text-to-speech to generate speech [8]. Its vision capabilities allow it to interpret and generate visual content, such as recognizing images, generating new images, and solving problems by analyzing uploaded images. In text interaction, GPT-4o excels in NLP, enabling it to write essays, answer questions, provide summaries, and create stories or poetry based on user input.

2) *Enhanced Interaction*: Users can interact with ChatGPT more dynamically, interrupting and receiving responses in real-time. The model can detect nuances in users' emotions and respond in various emotive tones, making conversations more natural and engaging [30]. Additionally, it can handle interactive Q&A sessions which require extensive knowledge [6].

3) *Data Analysis*: Users can create interactive tables and charts from uploaded data in a variety of file formats. ChatGPT automatically generates an interactive table view, enabling users to scroll through all rows and columns. Users can create interactive charts by specifying chart type or automated selection, customize the graphics of these charts, and generate summaries to explain their findings [31].

³<https://github.com/gpt-open/rag-gpt>

4) *Multilingual*: GPT-4o is capable of responding to 50 different non-English languages [6] [12]. This proficiency ensures effective communication and content generation for a global audience, making it a valuable tool for diverse linguistic applications.

5) *Memory*: ChatGPT introduced memory capabilities to save users from repeating information across conversations and to enhance future interactions. Users have control over the memory; they can instruct it to remember or forget specific details, inquire about what it remembers, and manage these settings or disable memory entirely [32].

C. Advantages

The following section explains the important advantages that GPT-4o has made over the previous chatbots.

1) *Faster Response*: GPT-4o can respond to audio inputs in as little as 232 milliseconds, with an average of 320 milliseconds, which is similar to human response time in a conversation [8].

2) *Cost-effective*: This GPT-4o model is cost-effective since we are using one model for various data types of input. This model will help analyze the video, security purposes, sports, and also for content analysis [13]. Most importantly, this chatbot has around 50% reduced processing cost than the previous model GPT-4 Turbo [27].

3) *Safe and Reliable*: GPT-4o is safe and reliable compared to the previous model. This model is based on the feedback of humans with a backbone of reinforcement learning which makes it more reliable. They also reduce the amount of misleading content generation by using the feedback [25]

D. Challenges

Despite the numerous advancements of GPT-4o, it still has noteworthy limitations. The outage on June 4th, 2024, caused by a major system issue, affected all users of ChatGPT-related services and lasted several hours⁶. The exact technical causes were not detailed, but such incidents typically involve a mix of software bugs, infrastructure issues, or configuration errors. Furthermore, GPT-4o's audio models are limited to preset voices, and the model's pronunciation or explanations can sometimes be incorrect [8]. Additionally, data breaches remain a significant concern in the digital world, emphasizing the need to protect user data and comply with data protection regulations to maintain trust and legal compliance. Ensuring responsible AI use, avoiding biases, and adhering to ethical practices are crucial considerations [29].

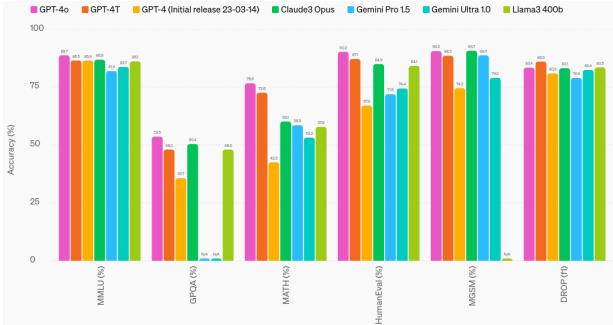
III. PERFORMANCE EVALUATION

OpenAI [8] provided a performance comparison between its models and their counterparts. Figure 1a illustrates the text evaluation results, showing that GPT-4o achieved state-of-the-art (SOTA) performance across the MMLU (88.7%),

⁵All vision evals are 0-shot, with MMMU, MathVista, and ChartQA as 0-shot CoT

⁶recommended to check original images for better view

⁶<https://status.openai.com/>



(a) Text Evaluation

Eval Sets	GPT-4o	GPT-4T 2024-04-09	Gemini 1.0 Ultra	Gemini 1.5 Pro	Claude Opus
MMMU (%) (val)	69.1	63.1	59.4	58.5	59.4
MathVista (%) (testmin)	63.8	58.1	53.0	52.1	50.5
AI2D (%) (test)	94.2	89.4	79.5	80.3	88.1
ChartQA (%) (test)	85.7	78.1	80.8	81.3	80.8
DocVQA (%) (test)	92.8	87.2	90.9	86.5	89.3
ActivityNet (%) (test)	61.9	59.5	52.2	56.7	
EgoSchema (%) (test)	72.2	63.9	61.5	63.2	

(b) Vision Understanding Evals⁴Fig. 1: Performance comparison provided by OpenAI [8]⁵

GPQA (53.6%), Math (76.6%), and HumanEval (90.2%) benchmarks. For the MGSM benchmark, Claude 3 Opus performs best (90.7%), followed by GPT-4o (90.5%). For the DROP benchmark, GPT-4T performed the best (86.0), followed by Gemini Ultra 1.0 (83.5) and GPT-4o (83.4). Figure 1b presents a performance comparison across various vision evaluation sets. Similar to the text evaluation set, GPT-4o consistently outperforms other models in most metrics, achieving SOTA in MMLU (69.1%), MathVista (63.8%), AI2D (94.2%), ChartQA (85.7%), DocVQA (92.8%), ActivityNet (61.9%), and EgoSchema (72.2%). GPT-4T follows closely but slightly lags behind GPT-4o. For the selected evaluation sets, it is clear from Figure 1 that GPT-4o demonstrates strong capabilities in these tasks, while Gemini 1.0 Ultra, Gemini 1.5 Pro, and Claude Opus show competitive but generally lower performance, with some exceptions in specific tasks.

OpenAI offers an evaluation framework called Evals⁷ on GitHub. This framework provides tools for assessing LLMs and includes an open-source registry of benchmarks. It enables users to create and run evaluations using datasets to generate prompts, assess the quality of model outputs, and compare performance across various datasets and models.

GPT-4o	GPT-4T
Claude 3 Opus	Claude 3 Haiku
Gemini 1.5 Pro	Gemini 1.5 Flash
Mixtral 8x22B	Mixtral 8x7B
DBRX	Command-R+

TABLE II: LLMs for performance comparison

Artificial Analysis [33] provides tools to compare selected LLMs by calling APIs and generating comparison graphs. To further analyze the performance of the GPT-4o model, we selected 10 popular LLMs (listed in Table II) using these tools. Figure 2 illustrates a relative performance analysis of the listed LLMs based on throughput, response time, and latency for a single query and a 10k token prompt length. These graphs collectively highlight the strengths and weaknesses of each model.

⁷<https://github.com/openai/evals>⁸Generated from <https://artificialanalysis.ai/models/gpt-4o/prompt-options/single/long#performance>

Figure 2a shows the latency comparison, defined as the time to the first token received (in seconds) after an API request is sent. Mixtral 8x7B and Mixtral 8x22B exhibit the lowest latencies, at 0.59 and 0.81 seconds, respectively. GPT-4o has a lower-than-average latency, receiving the first token in 1.04 seconds.

Figure 2b depicts throughput, measured in tokens per second received while the model generates tokens (i.e. after the first chunk has been received from the API). Gemini 1.5 Flash and Claude 3 Haiku have the highest throughput, scoring 89 each, followed by Mixtral 8x7B with a score of 73. GPT-4o ranks fourth, with a throughput of 64 tokens per second, outperforming the average.

Finally, Figure 2c evaluates the total response time, defined as the time to receive a 100-token response, estimated based on latency and throughput. Mixtral 8x7B and Claude 3 Haiku have the fastest response times, at 2.2 and 2.3 seconds, respectively. GPT-4o has a response time of 2.7 seconds.

Overall, the analysis in Figure 2 indicates that GPT-4o performs better than most models but lags behind Mixtral 8x7B and Claude 3 Haiku.

IV. APPLICATION DOMAIN

GPT-4o has the potential to significantly modernize and digitize the world by simplifying various aspects of human life. Its versatile applications span numerous fields, as detailed below, showcasing its substantial impact on a wide range of sectors.

a) Education: GPT-4o can guide students step-by-step to solve any math problems omitting the necessity of extra tutor [8] [30]. Additionally, it can be utilized in academic research by generating summaries of research papers, suggesting research topics, and providing insights from large datasets. These capabilities accelerate the research process and contribute to academic advancements by enabling researchers to quickly understand and explore extensive amounts of information.

b) Medical: GPT-4o can analyze medical images and patient data, aiding doctors in diagnosing diseases more accurately and quickly [29]. It enhances patient interaction by handling inquiries, providing information on medical conditions,

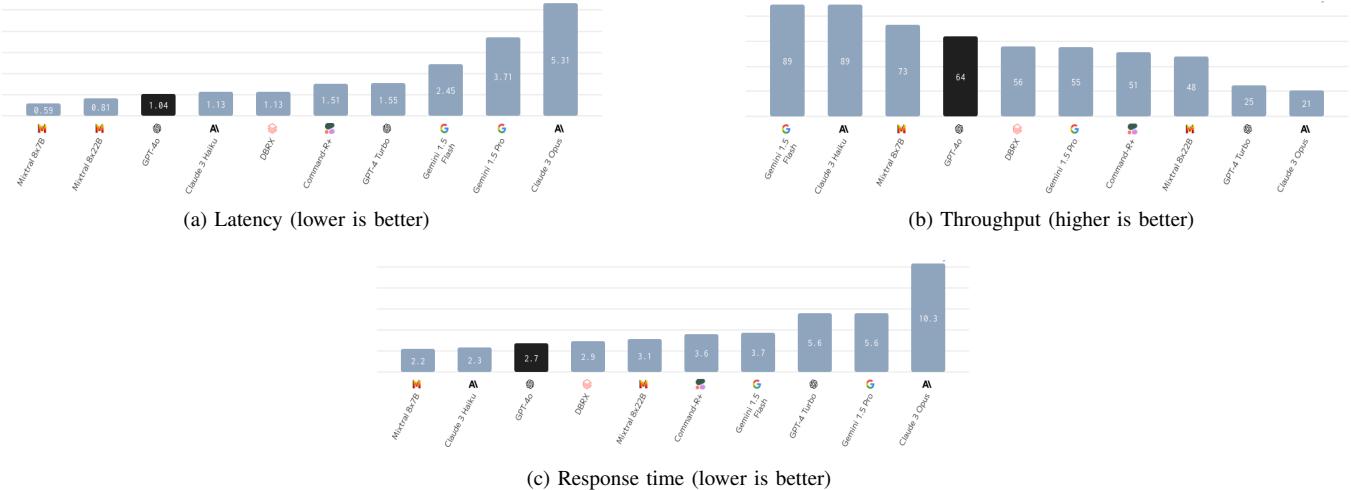


Fig. 2: Performance comparison⁸

and managing appointment scheduling through chatbots. It can also support learners with visual impairments by offering both speech-to-text and text-to-speech solutions [34].

c) *Customer Support*: GPT-4o can be tailored to specific business needs, enabling AI-powered chatbots to handle banking queries, transaction details, and account management 24/7, reducing the need for human intervention [29]. It supports advanced virtual assistants capable of effective responses, including visual data [25]. With real-time emotion analysis and multilingual capabilities, GPT-4o enhances global reach and customer satisfaction.

d) *Finance*: GPT-4o can analyze financial data and forecast market trends, enabling institutions to manage risks and make informed investment decisions [29].

e) *Content Generation*: GPT-4o can be used for creative and analytical purposes, i.e., making posters, creating visual representations from text input, designing characters, and producing outputs in different styles. It can also design logos, create artwork of people or objects, print text in different fonts, and generate 3D images.

f) *Code Review*: GPT-4o, can also review code effectively. It can identify the appropriate notes and comments while analyzing the code [6] [35]. Moreover, if provided with a picture of a desktop displaying the code, GPT-4o can detect issues or problems within the code [12].

V. CONCLUSION

GPT-4o represents a significant advancement in the field of AI, offering substantial improvements over its predecessors in terms of multimodal capabilities, context window size, tokenization efficiency, and processing speed. By integrating advanced technologies like refined RLHF and utilizing powerful hardware such as Nvidia's GPUs, GPT-4o achieves remarkable performance in understanding and generating text, audio, video, and images. Its applications span various domains, including education, medicine, customer support, finance, and

content generation, showcasing its versatility and potential to modernize and digitize numerous aspects of human life.

However, despite these advancements, GPT-4o still faces challenges such as ensuring ethical AI use, protecting user data, and overcoming technical issues like system outages. Addressing these challenges will be crucial for maximizing the model's benefits and maintaining user trust.

REFERENCES

- [1] T. Wu, S. He, J. Liu, *et al.*, “A brief overview of chatgpt: The history, status quo and potential future development,” *IEEE/CAA Journal of Automatica Sinica*, vol. 10, no. 5, pp. 1122–1136, 2023. DOI: 10.1109/JAS.2023.123618.
- [2] Y. Xu, L. Hu, J. Zhao, Z. Qiu, Y. Ye, and H. Gu, *A survey on multilingual large language models: Corpora, alignment, and bias*, 2024. DOI: 10.48550/arXiv.2404.00929.
- [3] Y. K. Dwivedi, N. Kshetri, L. Hughes, *et al.*, ““So what if ChatGPT wrote it?” Multidisciplinary perspectives on opportunities, challenges and implications of generative conversational AI for research, practice and policy,” *International Journal of Information Management*, vol. 71, p. 102 642, 2023.
- [4] P. P. Ray, “ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope,” *Internet of Things and Cyber-Physical Systems*, 2023.
- [5] H.-W. Cheng, “Challenges and limitations of ChatGPT and artificial intelligence for scientific research: a perspective from organic materials,” *AI*, vol. 4, no. 2, pp. 401–405, 2023.
- [6] S. M. Kerner, *GPT-4o explained: Everything you need to know*, <https://www.techtarget.com/whatis/feature/GPT-4o-explained-Everything-you-need-to-know>, [Accessed 26-05-2024], May 2024.

- [7] GeeksforGeeks, *OpenAI Launches GPT-4o Omni*, <https://www.geeksforgeeks.org/openai-announces-gpt-4o-omni/>, [Accessed 25-05-2024], May 2024.
- [8] OpenAI, *Hello GPT-4o*, <https://openai.com/index/hello-gpt-4o/>, [Accessed 10-06-2024], May 2024.
- [9] K. Doyle, 'The "o" is for omni' and other things you should know about GPT-4o, <https://www.jasper.ai/blog/what-is-gpt-4o>, [Accessed 26-05-2024], May 2024.
- [10] R. Montti, *OpenAI Announces GPT-4o Omni*, <https://www.searchenginejournal.com/openai-announces-chatgpt-4o-omni/516189/>, [Accessed 25-05-2024], May 2024.
- [11] M. Zeff and Gizmodo, *OpenAI's new ChatGPT sounds more human than ever*, <https://qz.com/openai-new-chatgpt-gpt4-omni-voice-human-ai-1851475246>, [Accessed 26-05-2024], May 2024.
- [12] K. Wiggers, *OpenAI debuts GPT-4o 'omni' model now powering ChatGPT*, <https://techcrunch.com/2024/05/13/openais-newest-model-is-gpt-4o/>, [Accessed 26-05-2024], May 2024.
- [13] K. Gomez, *Harnessing the Power of GPT-4 Omni for Multimodal Processing: A Comprehensive Guide*, <https://medium.com/@kyeg/harnessing-the-power-of-gpt-4-omni-for-multimodal-processing-a-comprehensive-guide-9301ae001576>, [Accessed 24-05-2024], May 2024.
- [14] *Introducing Apple Intelligence, the personal intelligence system that puts powerful generative models at the core of iPhone, iPad, and Mac*, <https://www.apple.com/newsroom/2024/06/introducing-apple-intelligence-for-iphone-ipad-and-mac>, [Accessed 11-06-2024], Jun. 2024.
- [15] *Introducing GPT-4o: our fastest and most affordable flagship model*, <https://platform.openai.com/docs/models/gpt-4o>, [Accessed 06-06-2024], 2024.
- [16] M. Schreiner, *GPT-4 architecture, datasets, costs and more leaked*, <https://the-decoder.com/gpt-4-architecture-datasets-costs-and-more-leaked>, [Accessed 07-06-2024], Jul. 2023.
- [17] T. B. Brown, B. Mann, N. Ryder, et al., *Language models are few-shot learners*, 2020. arXiv: 2005.14165 [cs.CL].
- [18] OpenAI, *Pricing*, <https://openai.com/api/pricing>, [Accessed 06-06-2024], 2024.
- [19] D. Sheremetov and A. Bitkina, *OpenAI API Pricing 2024: Understanding GPT-3 Pricing In-Depth*, <https://onix-systems.com/blog/how-much-does-it-cost-to-use-gpt-models>, [Accessed 07-06-2024], Jun. 2023.
- [20] C. Lang, *ChatGPT's Architecture - Decoder Only? Or Encoder-Decoder?* <https://datascience.stackexchange.com/questions/118260/chatgpts-architecture-decoder-only-or-encoder-decoder>, [Accessed 07-06-2024], Dec. 2023.
- [21] OpenAI and J. A. et al., *GPT-4 Technical Report*, 2024. arXiv: 2303.08774 [cs.CL].
- [22] N. Bijani, *What are the differences between GPT, GPT3, GPT 3.5, GPT turbo GPT 4?* <https://www.codiste.com/what-difference-between-gpt-gpt3-gpt-3-5-gptturbo-gpt-4>, [Accessed 08-06-2024], Oct. 2023.
- [23] T. Pungas, *GPT-3.5 and GPT-4 response times*, https://www.taivo.ai/_gpt-3-5-and-gpt-4-response-times, [Accessed 09-06-2024], May 2023.
- [24] E. Eckert, *GPT-4 vs GPT-4o: The Ultimate AI Smackdown!* <https://supernormal.com/blog/gpt-4-vs-gpt-4o>, [Accessed 09-06-2024], Jun. 2024.
- [25] C. D. Under, *OpenAI GPT-4o: The Next Generation of Omni-Multimodal AI*, <https://medium.com/@cognidownunder/openai-gpt-4o-the-next-generation-of-omni-multimodal-ai-e8d64d211a2c>, [Accessed 26-05-2024], May 2024.
- [26] R. LeFebvre and J. Ledford, *OpenAI Reveals New GPT 4o: AI For Everyone*, <https://www.lifewire.com/openai-reveals-gpt-4o-8647637>, [Accessed 06-06-2024], May 2024.
- [27] M. Rajguru, *Exploring the New Frontier of AI: OpenAI's GPT-4o For Indic Languages*, <https://techcommunity.microsoft.com/t5/ai-azure-ai-services-blog/exploring-the-new-frontier-of-ai-openai-s-gpt-4-o-for-indic/ba-p/4142383>, [Accessed 25-05-2024], May 2024.
- [28] *Introducing GPT-4o: OpenAI's new flagship multimodal model now in preview on Azure*, <https://azure.microsoft.com/en-us/updates/new-openai-model-on-azure/>, [Accessed 06-06-2024], May 2024.
- [29] V. K. Upadhyay, *OpenAI : GPT-4o (Comprehensive Guide)*, <https://vivekupadhyay1.medium.com/openai-gpt-4o-comprehensive-guide-ccf15fd93870>, [Accessed 06-06-2024], May 2024.
- [30] A. Toolz, *15 Abilities of GPT4o that you Won't Believe*, <https://aitoolzai.medium.com/15-abilities-of-gpt4o-that-you-wont-believe-8cba07c1cf1f>, [Accessed 31-05-2024], May 2024.
- [31] OpenAI, *Data analysis with ChatGPT*, <https://help.openai.com/en/articles/8437071-data-analysis-with-chatgpt>, [Accessed 04-06-2024], Jun. 2024.
- [32] OpenAI, *Memory and new controls for ChatGPT*, <https://openai.com/index/memory-and-new-controls-for-chatgpt>, [Accessed 04-06-2024], Feb. 2024.
- [33] Artificial Analysis, <https://artificialanalysis.ai>, [Accessed 05-06-2024], 2024.
- [34] A. Kuzdeuov, O. Mukayev, S. Nurgaliyev, A. Kunbolsyn, and H. A. Varol, "Chatgpt for visually impaired and blind," in *2024 International Conference on Artificial Intelligence in Information and Communication (ICAIIIC)*, 2024, pp. 722–727. DOI: 10.1109/ICAIIIC60209.2024.10463430.
- [35] D. Eastman, *Reviewing Code With GPT-4o, OpenAI's New 'Omni' LLM*, <https://thenewstack.io/reviewing-code-with-gpt-4o-openais-new-omni-lm/>, [Accessed 26-05-2024], May 2024.

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/381898598>

GPT-4o: The Cutting-Edge Advancement in Multimodal LLM

Preprint · July 2024

DOI: 10.36227/techriv.171986596.65533294/v1

CITATIONS

8

READS

2,522

2 authors:



Raisa Islam

New Mexico Institute of Mining and Technology

7 PUBLICATIONS 49 CITATIONS

[SEE PROFILE](#)



Owana Marzia Moushi

University of Nebraska-Lincoln

8 PUBLICATIONS 16 CITATIONS

[SEE PROFILE](#)

GPT-4o: The Cutting-Edge Advancement in Multimodal LLM

Raisa Islam

Computer Science

New Mexico Institute of Mining and Technology
Socorro, NM, USA
raisila.islam@student.nmt.edu

Owana Marzia Moushi

Electrical and Computer Engineering
University of Nebraska - Lincoln
Omaha, NE, USA
omoushi2@huskers.unl.edu

Abstract—GPT-4o marks a significant advancement in AI technology, enhancing multimodal capabilities. OpenAI has launched several GPT models over the years, with GPT-4o being the latest. This paper provides a concise overview of these models, focusing on their key features and technological advancements. The main objective is to present a brief overview of GPT-4o, including its technological innovations. GPT-4o offers substantial improvements over its predecessors by introducing multimodal capabilities, larger context windows, efficient tokenization, and faster processing speeds, achieving state-of-the-art performance in text, audio, video, and image generation and understanding. We have compared GPT-4o with ten top LLMs using metrics such as throughput, response time, and latency, where GPT-4o demonstrated clear superiority. Additionally, this paper explores various application domains, highlighting GPT-4o's versatility and potential to modernize multiple aspects of human life.

Index Terms—GPT-4o, LLM, OpenAI, AI, ChatGPT, multimodal

I. INTRODUCTION

Artificial Intelligence (AI) is one of the most popular cutting-edge technology in this era which is gaining attention because of its revolutionizing aspects in our life [1]. Natural Language Processing (NLP), a subset of AI, focuses on the interaction between computers and human languages, aiming to enable machines to understand, interpret, and generate human language meaningfully. The advent of Large Language Models (LLMs) has initiated a paradigm shift and revolutionized the field of NLP by leveraging deep learning techniques to process and generate natural language with unprecedented accuracy and fluency [2]. These models are trained on vast amounts of textual data, allowing them to capture intricate linguistic patterns and contextual nuances. As a result, LLMs have significantly enhanced applications in various domains, including chatbots, language translation, sentiment analysis, and content creation, driving forward the capabilities of AI in understanding and utilizing human language [3] [4].

OpenAI's Generative Pre-trained Transformer (GPT) models have fundamentally transformed the fields of AI and NLP over the past ten years. GPT models are built upon the transformer architecture, which has demonstrated exceptional effectiveness, particularly in developing LLMs. The first version, GPT-1, was launched in 2017. Subsequent versions, including GPT-2 and GPT-3, culminated in the groundbreaking ChatGPT.

Since its release in November 2022, ChatGPT has gained immense popularity, reaching 1 million users within a week and 100 million users within two months [5]. ChatGPT aims to simplify various tasks, such as coding, math solving, and problem-solving, by using a combination of pre-trained unsupervised models and fine-tuned supervised models to provide human-like responses [3]. However, the last version, GPT-4, had limitations in video chatting, image processing, and audio processing. To address these shortcomings and advance toward a more digital world, OpenAI has released a new version of ChatGPT, called GPT-4o.

The recent version of ChatGPT is based on GPT-4o architecture which is gaining success on the previous chatbots. It was released on May 13, 2024, called GPT-4omni (or GPT-4o) the latest multimodal LLM from OpenAI [6]. The term "omni," derived from the Latin word "omnis," meaning "all" or "every," highlights the model's omni-modal capabilities [7]. GPT-4o can process and understand multimodal inputs, including text, images, audio, and video, making it a significant advancement in AI technology. It is the first LLM capable of interpreting emotions from videos, enhancing user experience by analyzing various data types. Although the model is already available for use, OpenAI continues to improve its capabilities, with updates expected soon [8]. The model's efficiency in handling different types of data results in high success rates and reduced processing times and costs [9]. For users seeking unlimited access and enhanced features, OpenAI offers a paid version, such as ChatGPT Plus, which includes a higher message limit and upcoming macOS desktop app support [10] [11]. GPT-4o is twice as fast as GPT-4 Turbo (released after GPT-4) and represents a milestone in the digital interaction between machines and humans, promising to significantly impact the future of digital communication [7] [12] [13].

Tech giant Apple and OpenAI are collaborating to integrate ChatGPT into Apple's ecosystem, focusing on the upcoming iOS 18 operating system [14]. This partnership aims to enhance the user experience by embedding ChatGPT's advanced language capabilities into Apple services like Siri, making them more intuitive and responsive. Features include real-time conversation synchronization, and voice input through Whisper. These enhancements will allow Siri to process

	GPT-4o	GPT-4	GPT-3.5	GPT-3
Initial Release Date	May 13, 2024	March 14, 2023	March 15, 2022	June 11, 2020
Modality	Audio, video, text, images	Text and images	Text	Text
Context window [2] [15]	128000	8192	4096	2048
Parameters [2] [16] [17]	Yet-to-Disclose (YTD)	1.76 trillion	175 billion	175 billion
Cost per 1M token [18] [19]	Input: \$5 Output: \$15	Input: \$30 Output: \$60	Input: \$1.5 Output: \$2	\$0.4 to \$20
Decoder layers [16] [17] [20]	YTD	120	96	96
Prompting Method [17] [21]	YTD	Chain-of-Thought, n -shot ¹	n -shot	n -shot
Training Data [17] [22]	YTD	~ 13T tokens which includes both text data and code data	Over 570GB of text data	About 45TB text data
Response Time [23] [24]	30% faster than GPT-4	94ms per token	35ms per token	Unknown
Performance	State-of-the-art performance	Advanced in handling complex tasks, slower response time than GPT-4o	Improved version of GPT-3, lacks depth of understanding	Poor on complex tasks

TABLE I: Summary of latest GPT models

complex queries more effectively and provide functionalities such as instant answers, tailored advice, creative inspiration, professional input, and educational support. This integration aligns with Apple's broader strategy to incorporate AI-driven features while ensuring privacy by powering most features on-device.

A. Evolution of GPT

Table I provides an overview of the latest GPT models to understand the improvement OpenAI is making for the updated GPT models. GPT-4o significantly enhances the capabilities of its predecessors by incorporating multimodal functionalities, including text, audio, image, and video processing. It offers faster response times and reduced costs compared to GPT-4, while maintaining high accuracy and performance, surpassing the improvements seen in GPT-3.5 and GPT-3.

Contribution: The main contributions of this paper are as follows:

- We have provided an in-depth overview of GPT-4o, detailing the technologies that contribute to its advancement over previous models. This study also summarizes the key features and advantages of GPT-4o, highlighting the significant improvements it offers compared to its predecessors.
- We have analyzed the text and vision performance comparisons provided by the OpenAI team. Additionally, a comparison of GPT-4o with other leading LLMs using metrics such as throughput, response time, and latency has been discussed. The results demonstrate the superior performance of GPT-4o.
- Furthermore, we have presented the various fields where GPT-4o can be applied, showcasing its versatility and potential impact.

The structure of this paper is as follows: Section II provides an overview of the newly launched GPT-4o, including its technologies, features, advantages, and challenges. Section III analyzes the performance of GPT-4o by comparing it with popular LLM models, and Section IV explores the application domains where GPT-4o can be utilized. Finally, Section V concludes the paper.

¹the value of n starts from 0

II. GPT-4O OVERVIEW

This section provides an overview of GPT-4o, detailing the technologies employed in its development, its features, and its advantages. While GPT-4o incorporates numerous updated technologies, it also faces some challenges that need to be addressed. These aspects, including both the strengths and areas for improvement, are comprehensively discussed here.

A. Technology

GPT-4o builds upon the architecture of its predecessors, integrating enhancements in context window size, tokenization efficiency, and multimodal capabilities, which include processing text, audio, video, and images. This single model is trained end-to-end across text, vision, and audio, ensuring all inputs and outputs are managed by the same neural network [8]. It employs refined reinforcement learning with human feedback (RLHF) [25], significantly improving its alignment with human values and ethical standards. GPT-4o also features an advanced transformer architecture with enhanced self-attention mechanisms, allowing for better comprehension and generation of nuanced, contextually relevant responses. Utilizing Nvidia's most advanced GPUs [26], known for their parallel processing capabilities, GPT-4o efficiently handles the massive computations required by its sophisticated architecture. This model gives better performance for scalability since it uses larger training sessions [25]. GPT-4o's rate limits are 5x higher than GPT-4 Turbo—up to 10 million tokens per minute. These advancements make GPT-4o a versatile tool for applications ranging from conversational AI to content generation and data analysis. The following subsection discusses the technologies that are being used in GPT-4o.

1) o200k_base Tokenizer: GPT-4o introduced a new *o200k_base* tokenizer algorithm, marking a shift from the *cl100k_base* tokenizer used by GPT-4, GPT-4 Turbo, and GPT-3.5 Turbo². Tokenization, which breaks down text into smaller units called tokens, is critical in NLP. The *o200k_base* tokenizer improves upon previous methods by being faster and more efficient, allowing GPT-4o to process and generate language at unprecedented speeds. It enhances semantic coherence in generated text and improves the handling of multiple languages [27], expanding GPT-4o's applicability across various linguistic contexts.

²<https://github.com/openai/tiktoken>

2) *RAG-GPT*: RAG-GPT³ is an advanced implementation of Retrieval-Augmented Generation (RAG) technology, designed to enhance the capabilities of LLMs by integrating them with efficient document retrieval systems. This integration allows RAG-GPT to provide more accurate and contextually relevant responses by fetching and incorporating information from extensive knowledge bases during query processing. Integration of RAG-GPT in GPT-4o enables delivering precise and grounded answers, making it particularly useful for applications requiring detailed and up-to-date information. This approach not only improves the quality of generated content but also enables the handling of specialized and complex queries with greater reliability.

3) *Context Window*: GPT-4o features an impressive context window size of 128k tokens [6]. The larger context window allows the model to maintain and process a much larger amount of information within a single interaction, enhancing its ability to understand and generate responses that are contextually relevant over extended conversations. This large context window is particularly beneficial for complex tasks requiring extensive context retention, tracking multiple threads of a conversation, and integration of multiple pieces of information.

4) *Cloud Infrastructure and API Access*: GPT-4o is designed to be deployed on scalable cloud infrastructures (Microsoft Azure) [28] [29], offering flexible API access for diverse applications. This cloud-based deployment model allows for seamless scaling to meet varying user demands and workloads.

B. Features

OpenAI is improving its models to make the world digital by adding some additional features which are discussed below:

1) *Multimodality*: GPT-4o is the newest multimodal LLM. It can understand and generate spoken languages, recognizing speech to transcribe text and using text-to-speech to generate speech [8]. Its vision capabilities allow it to interpret and generate visual content, such as recognizing images, generating new images, and solving problems by analyzing uploaded images. In text interaction, GPT-4o excels in NLP, enabling it to write essays, answer questions, provide summaries, and create stories or poetry based on user input.

2) *Enhanced Interaction*: Users can interact with ChatGPT more dynamically, interrupting and receiving responses in real-time. The model can detect nuances in users' emotions and respond in various emotive tones, making conversations more natural and engaging [30]. Additionally, it can handle interactive Q&A sessions which require extensive knowledge [6].

3) *Data Analysis*: Users can create interactive tables and charts from uploaded data in a variety of file formats. ChatGPT automatically generates an interactive table view, enabling users to scroll through all rows and columns. Users can create interactive charts by specifying chart type or automated selection, customize the graphics of these charts, and generate summaries to explain their findings [31].

³<https://github.com/gpt-open/rag-gpt>

4) *Multilingual*: GPT-4o is capable of responding to 50 different non-English languages [6] [12]. This proficiency ensures effective communication and content generation for a global audience, making it a valuable tool for diverse linguistic applications.

5) *Memory*: ChatGPT introduced memory capabilities to save users from repeating information across conversations and to enhance future interactions. Users have control over the memory; they can instruct it to remember or forget specific details, inquire about what it remembers, and manage these settings or disable memory entirely [32].

C. Advantages

The following section explains the important advantages that GPT-4o has made over the previous chatbots.

1) *Faster Response*: GPT-4o can respond to audio inputs in as little as 232 milliseconds, with an average of 320 milliseconds, which is similar to human response time in a conversation [8].

2) *Cost-effective*: This GPT-4o model is cost-effective since we are using one model for various data types of input. This model will help analyze the video, security purposes, sports, and also for content analysis [13]. Most importantly, this chatbot has around 50% reduced processing cost than the previous model GPT-4 Turbo [27].

3) *Safe and Reliable*: GPT-4o is safe and reliable compared to the previous model. This model is based on the feedback of humans with a backbone of reinforcement learning which makes it more reliable. They also reduce the amount of misleading content generation by using the feedback [25]

D. Challenges

Despite the numerous advancements of GPT-4o, it still has noteworthy limitations. The outage on June 4th, 2024, caused by a major system issue, affected all users of ChatGPT-related services and lasted several hours⁶. The exact technical causes were not detailed, but such incidents typically involve a mix of software bugs, infrastructure issues, or configuration errors. Furthermore, GPT-4o's audio models are limited to preset voices, and the model's pronunciation or explanations can sometimes be incorrect [8]. Additionally, data breaches remain a significant concern in the digital world, emphasizing the need to protect user data and comply with data protection regulations to maintain trust and legal compliance. Ensuring responsible AI use, avoiding biases, and adhering to ethical practices are crucial considerations [29].

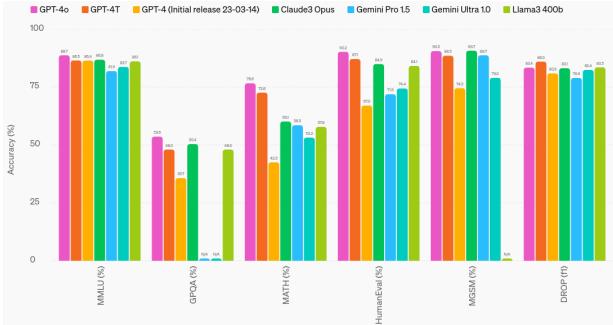
III. PERFORMANCE EVALUATION

OpenAI [8] provided a performance comparison between its models and their counterparts. Figure 1a illustrates the text evaluation results, showing that GPT-4o achieved state-of-the-art (SOTA) performance across the MMLU (88.7%),

⁵All vision evals are 0-shot, with MMMU, MathVista, and ChartQA as 0-shot CoT

⁶recommended to check original images for better view

⁶<https://status.openai.com/>



(a) Text Evaluation

Eval Sets	GPT-4o	GPT-4T 2024-04-09	Gemini 1.0 Ultra	Gemini 1.5 Pro	Claude Opus
MMMU (%) (val)	69.1	63.1	59.4	58.5	59.4
MathVista (%) (testmin)	63.8	58.1	53.0	52.1	50.5
AI2D (%) (test)	94.2	89.4	79.5	80.3	88.1
ChartQA (%) (test)	85.7	78.1	80.8	81.3	80.8
DocVQA (%) (test)	92.8	87.2	90.9	86.5	89.3
ActivityNet (%) (test)	61.9	59.5	52.2	56.7	
EgoSchema (%) (test)	72.2	63.9	61.5	63.2	

(b) Vision Understanding Evals⁴Fig. 1: Performance comparison provided by OpenAI [8]⁵

GPQA (53.6%), Math (76.6%), and HumanEval (90.2%) benchmarks. For the MGSM benchmark, Claude 3 Opus performs best (90.7%), followed by GPT-4o (90.5%). For the DROP benchmark, GPT-4T performed the best (86.0), followed by Gemini Ultra 1.0 (83.5) and GPT-4o (83.4). Figure 1b presents a performance comparison across various vision evaluation sets. Similar to the text evaluation set, GPT-4o consistently outperforms other models in most metrics, achieving SOTA in MMLU (69.1%), MathVista (63.8%), AI2D (94.2%), ChartQA (85.7%), DocVQA (92.8%), ActivityNet (61.9%), and EgoSchema (72.2%). GPT-4T follows closely but slightly lags behind GPT-4o. For the selected evaluation sets, it is clear from Figure 1 that GPT-4o demonstrates strong capabilities in these tasks, while Gemini 1.0 Ultra, Gemini 1.5 Pro, and Claude Opus show competitive but generally lower performance, with some exceptions in specific tasks.

OpenAI offers an evaluation framework called Evals⁷ on GitHub. This framework provides tools for assessing LLMs and includes an open-source registry of benchmarks. It enables users to create and run evaluations using datasets to generate prompts, assess the quality of model outputs, and compare performance across various datasets and models.

GPT-4o	GPT-4T
Claude 3 Opus	Claude 3 Haiku
Gemini 1.5 Pro	Gemini 1.5 Flash
Mixtral 8x22B	Mixtral 8x7B
DBRX	Command-R+

TABLE II: LLMs for performance comparison

Artificial Analysis [33] provides tools to compare selected LLMs by calling APIs and generating comparison graphs. To further analyze the performance of the GPT-4o model, we selected 10 popular LLMs (listed in Table II) using these tools. Figure 2 illustrates a relative performance analysis of the listed LLMs based on throughput, response time, and latency for a single query and a 10k token prompt length. These graphs collectively highlight the strengths and weaknesses of each model.

⁷<https://github.com/openai/evals>⁸Generated from <https://artificialanalysis.ai/models/gpt-4o/prompt-options/single/long#performance>

Figure 2a shows the latency comparison, defined as the time to the first token received (in seconds) after an API request is sent. Mixtral 8x7B and Mixtral 8x22B exhibit the lowest latencies, at 0.59 and 0.81 seconds, respectively. GPT-4o has a lower-than-average latency, receiving the first token in 1.04 seconds.

Figure 2b depicts throughput, measured in tokens per second received while the model generates tokens (i.e. after the first chunk has been received from the API). Gemini 1.5 Flash and Claude 3 Haiku have the highest throughput, scoring 89 each, followed by Mixtral 8x7B with a score of 73. GPT-4o ranks fourth, with a throughput of 64 tokens per second, outperforming the average.

Finally, Figure 2c evaluates the total response time, defined as the time to receive a 100-token response, estimated based on latency and throughput. Mixtral 8x7B and Claude 3 Haiku have the fastest response times, at 2.2 and 2.3 seconds, respectively. GPT-4o has a response time of 2.7 seconds.

Overall, the analysis in Figure 2 indicates that GPT-4o performs better than most models but lags behind Mixtral 8x7B and Claude 3 Haiku.

IV. APPLICATION DOMAIN

GPT-4o has the potential to significantly modernize and digitize the world by simplifying various aspects of human life. Its versatile applications span numerous fields, as detailed below, showcasing its substantial impact on a wide range of sectors.

a) Education: GPT-4o can guide students step-by-step to solve any math problems omitting the necessity of extra tutor [8] [30]. Additionally, it can be utilized in academic research by generating summaries of research papers, suggesting research topics, and providing insights from large datasets. These capabilities accelerate the research process and contribute to academic advancements by enabling researchers to quickly understand and explore extensive amounts of information.

b) Medical: GPT-4o can analyze medical images and patient data, aiding doctors in diagnosing diseases more accurately and quickly [29]. It enhances patient interaction by handling inquiries, providing information on medical conditions,

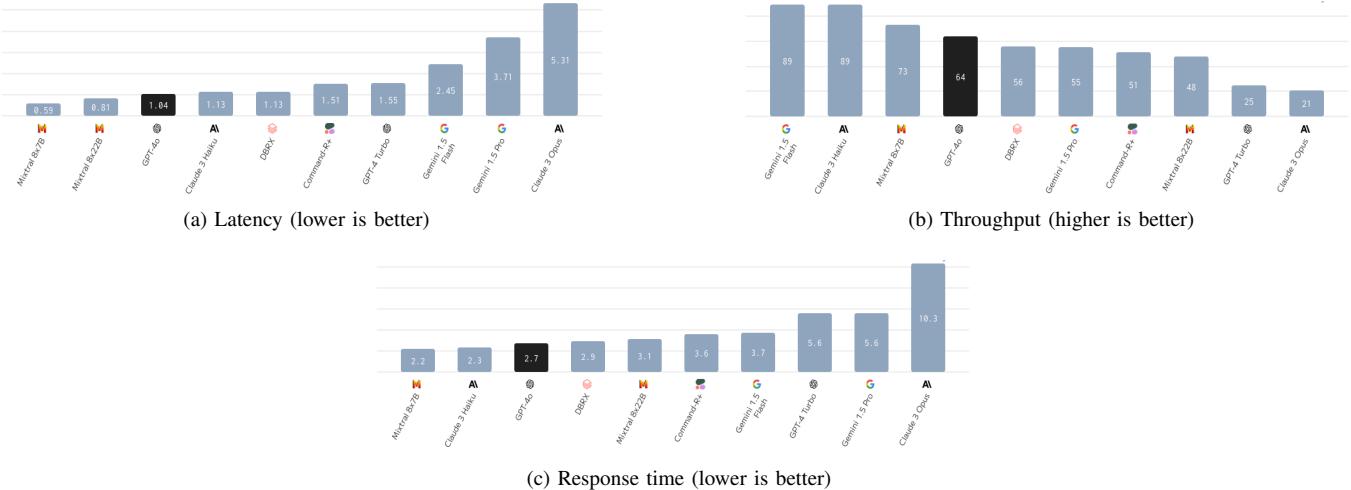


Fig. 2: Performance comparison⁸

and managing appointment scheduling through chatbots. It can also support learners with visual impairments by offering both speech-to-text and text-to-speech solutions [34].

c) *Customer Support*: GPT-4o can be tailored to specific business needs, enabling AI-powered chatbots to handle banking queries, transaction details, and account management 24/7, reducing the need for human intervention [29]. It supports advanced virtual assistants capable of effective responses, including visual data [25]. With real-time emotion analysis and multilingual capabilities, GPT-4o enhances global reach and customer satisfaction.

d) *Finance*: GPT-4o can analyze financial data and forecast market trends, enabling institutions to manage risks and make informed investment decisions [29].

e) *Content Generation*: GPT-4o can be used for creative and analytical purposes, i.e., making posters, creating visual representations from text input, designing characters, and producing outputs in different styles. It can also design logos, create artwork of people or objects, print text in different fonts, and generate 3D images.

f) *Code Review*: GPT-4o, can also review code effectively. It can identify the appropriate notes and comments while analyzing the code [6] [35]. Moreover, if provided with a picture of a desktop displaying the code, GPT-4o can detect issues or problems within the code [12].

V. CONCLUSION

GPT-4o represents a significant advancement in the field of AI, offering substantial improvements over its predecessors in terms of multimodal capabilities, context window size, tokenization efficiency, and processing speed. By integrating advanced technologies like refined RLHF and utilizing powerful hardware such as Nvidia's GPUs, GPT-4o achieves remarkable performance in understanding and generating text, audio, video, and images. Its applications span various domains, including education, medicine, customer support, finance, and

content generation, showcasing its versatility and potential to modernize and digitize numerous aspects of human life.

However, despite these advancements, GPT-4o still faces challenges such as ensuring ethical AI use, protecting user data, and overcoming technical issues like system outages. Addressing these challenges will be crucial for maximizing the model's benefits and maintaining user trust.

REFERENCES

- [1] T. Wu, S. He, J. Liu, *et al.*, “A brief overview of chatgpt: The history, status quo and potential future development,” *IEEE/CAA Journal of Automatica Sinica*, vol. 10, no. 5, pp. 1122–1136, 2023. DOI: 10.1109/JAS.2023.123618.
- [2] Y. Xu, L. Hu, J. Zhao, Z. Qiu, Y. Ye, and H. Gu, *A survey on multilingual large language models: Corpora, alignment, and bias*, 2024. DOI: 10.48550/arXiv.2404.00929.
- [3] Y. K. Dwivedi, N. Kshetri, L. Hughes, *et al.*, ““So what if ChatGPT wrote it?” Multidisciplinary perspectives on opportunities, challenges and implications of generative conversational AI for research, practice and policy,” *International Journal of Information Management*, vol. 71, p. 102 642, 2023.
- [4] P. P. Ray, “ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope,” *Internet of Things and Cyber-Physical Systems*, 2023.
- [5] H.-W. Cheng, “Challenges and limitations of ChatGPT and artificial intelligence for scientific research: a perspective from organic materials,” *AI*, vol. 4, no. 2, pp. 401–405, 2023.
- [6] S. M. Kerner, *GPT-4o explained: Everything you need to know*, <https://www.techtarget.com/whatis/feature/GPT-4o-explained-Everything-you-need-to-know>, [Accessed 26-05-2024], May 2024.

- [7] GeeksforGeeks, *OpenAI Launches GPT-4o Omni*, <https://www.geeksforgeeks.org/openai-announces-gpt-4o-omni/>, [Accessed 25-05-2024], May 2024.
- [8] OpenAI, *Hello GPT-4o*, <https://openai.com/index/hello-gpt-4o/>, [Accessed 10-06-2024], May 2024.
- [9] K. Doyle, 'The "o" is for omni' and other things you should know about GPT-4o, <https://www.jasper.ai/blog/what-is-gpt-4o>, [Accessed 26-05-2024], May 2024.
- [10] R. Montti, *OpenAI Announces GPT-4o Omni*, <https://www.searchenginejournal.com/openai-announces-chatgpt-4o-omni/516189/>, [Accessed 25-05-2024], May 2024.
- [11] M. Zeff and Gizmodo, *OpenAI's new ChatGPT sounds more human than ever*, <https://qz.com/openai-new-chatgpt-gpt4-omni-voice-human-ai-1851475246>, [Accessed 26-05-2024], May 2024.
- [12] K. Wiggers, *OpenAI debuts GPT-4o 'omni' model now powering ChatGPT*, <https://techcrunch.com/2024/05/13/openais-newest-model-is-gpt-4o/>, [Accessed 26-05-2024], May 2024.
- [13] K. Gomez, *Harnessing the Power of GPT-4 Omni for Multimodal Processing: A Comprehensive Guide*, <https://medium.com/@kyeg/harnessing-the-power-of-gpt-4-omni-for-multimodal-processing-a-comprehensive-guide-9301ae001576>, [Accessed 24-05-2024], May 2024.
- [14] *Introducing Apple Intelligence, the personal intelligence system that puts powerful generative models at the core of iPhone, iPad, and Mac*, <https://www.apple.com/newsroom/2024/06/introducing-apple-intelligence-for-iphone-ipad-and-mac>, [Accessed 11-06-2024], Jun. 2024.
- [15] *Introducing GPT-4o: our fastest and most affordable flagship model*, <https://platform.openai.com/docs/models/gpt-4o>, [Accessed 06-06-2024], 2024.
- [16] M. Schreiner, *GPT-4 architecture, datasets, costs and more leaked*, <https://the-decoder.com/gpt-4-architecture-datasets-costs-and-more-leaked>, [Accessed 07-06-2024], Jul. 2023.
- [17] T. B. Brown, B. Mann, N. Ryder, et al., *Language models are few-shot learners*, 2020. arXiv: 2005.14165 [cs.CL].
- [18] OpenAI, *Pricing*, <https://openai.com/api/pricing>, [Accessed 06-06-2024], 2024.
- [19] D. Sheremetov and A. Bitkina, *OpenAI API Pricing 2024: Understanding GPT-3 Pricing In-Depth*, <https://onix-systems.com/blog/how-much-does-it-cost-to-use-gpt-models>, [Accessed 07-06-2024], Jun. 2023.
- [20] C. Lang, *ChatGPT's Architecture - Decoder Only? Or Encoder-Decoder?* <https://datascience.stackexchange.com/questions/118260/chatgpts-architecture-decoder-only-or-encoder-decoder>, [Accessed 07-06-2024], Dec. 2023.
- [21] OpenAI and J. A. et al., *GPT-4 Technical Report*, 2024. arXiv: 2303.08774 [cs.CL].
- [22] N. Bijani, *What are the differences between GPT, GPT3, GPT 3.5, GPT turbo GPT 4?* <https://www.codiste.com/what-difference-between-gpt-gpt3-gpt-3-5-gptturbo-gpt-4>, [Accessed 08-06-2024], Oct. 2023.
- [23] T. Pungas, *GPT-3.5 and GPT-4 response times*, https://www.taivo.ai/_gpt-3-5-and-gpt-4-response-times, [Accessed 09-06-2024], May 2023.
- [24] E. Eckert, *GPT-4 vs GPT-4o: The Ultimate AI Smackdown!* <https://supernormal.com/blog/gpt-4-vs-gpt-4o>, [Accessed 09-06-2024], Jun. 2024.
- [25] C. D. Under, *OpenAI GPT-4o: The Next Generation of Omni-Multimodal AI*, <https://medium.com/@cognidownunder/openai-gpt-4o-the-next-generation-of-omni-multimodal-ai-e8d64d211a2c>, [Accessed 26-05-2024], May 2024.
- [26] R. LeFebvre and J. Ledford, *OpenAI Reveals New GPT 4o: AI For Everyone*, <https://www.lifewire.com/openai-reveals-gpt-4o-8647637>, [Accessed 06-06-2024], May 2024.
- [27] M. Rajguru, *Exploring the New Frontier of AI: OpenAI's GPT-4o For Indic Languages*, <https://techcommunity.microsoft.com/t5/ai-azure-ai-services-blog/exploring-the-new-frontier-of-ai-openai-s-gpt-4-o-for-indic/ba-p/4142383>, [Accessed 25-05-2024], May 2024.
- [28] *Introducing GPT-4o: OpenAI's new flagship multimodal model now in preview on Azure*, <https://azure.microsoft.com/en-us/updates/new-openai-model-on-azure/>, [Accessed 06-06-2024], May 2024.
- [29] V. K. Upadhyay, *OpenAI : GPT-4o (Comprehensive Guide)*, <https://vivekupadhyay1.medium.com/openai-gpt-4o-comprehensive-guide-ccf15fd93870>, [Accessed 06-06-2024], May 2024.
- [30] A. Toolz, *15 Abilities of GPT4o that you Won't Believe*, <https://aitoolzai.medium.com/15-abilities-of-gpt4o-that-you-wont-believe-8cba07c1cf1f>, [Accessed 31-05-2024], May 2024.
- [31] OpenAI, *Data analysis with ChatGPT*, <https://help.openai.com/en/articles/8437071-data-analysis-with-chatgpt>, [Accessed 04-06-2024], Jun. 2024.
- [32] OpenAI, *Memory and new controls for ChatGPT*, <https://openai.com/index/memory-and-new-controls-for-chatgpt>, [Accessed 04-06-2024], Feb. 2024.
- [33] Artificial Analysis, <https://artificialanalysis.ai>, [Accessed 05-06-2024], 2024.
- [34] A. Kuzdeuov, O. Mukayev, S. Nurgaliyev, A. Kubolsyn, and H. A. Varol, "Chatgpt for visually impaired and blind," in *2024 International Conference on Artificial Intelligence in Information and Communication (ICAIIIC)*, 2024, pp. 722–727. DOI: 10.1109/ICAIIIC60209.2024.10463430.
- [35] D. Eastman, *Reviewing Code With GPT-4o, OpenAI's New 'Omni' LLM*, <https://thenewstack.io/reviewing-code-with-gpt-4o-openais-new-omni-lm/>, [Accessed 26-05-2024], May 2024.

Leveraging Large Language Models for Multimodal Search

Oriol Barbany^{1†} Michael Huang^{2*} Xinliang Zhu^{2*} Arnab Dhua²

¹Institut de Robòtica i Informàtica Industrial, CSIC-UPC ²Visual Search & AR, Amazon



Figure 1. **Overview:** This paper introduces a comprehensive pipeline for multimodal search, presenting a novel composed retrieval model that outperforms previous approaches significantly. Additionally, we propose a system that utilizes a Large Language Model (LLM) as an orchestrator to invoke both our proposed model and other off-the-shelf models. The resulting search interface offers a conversational search assistant experience, integrating information from previous queries and leveraging our novel model to enhance search capabilities.

Abstract

Multimodal search has become increasingly important in providing users with a natural and effective way to express their search intentions. Images offer fine-grained details of the desired products, while text allows for easily incorporating search modifications. However, some existing multimodal search systems are unreliable and fail to address simple queries. The problem becomes harder with the large variability of natural language text queries, which may contain ambiguous, implicit, and irrelevant information. Addressing these issues may require systems with enhanced matching capabilities, reasoning abilities, and context-aware query parsing and rewriting. This paper introduces a novel multimodal search model that achieves a new performance milestone on the Fashion200K dataset [19]. Additionally, we propose a novel search interface integrating Large Language Models (LLMs) to facilitate natural language interaction. This interface routes queries to search systems while conversationally engaging with users and considering previous searches. When coupled with our multimodal search model, it heralds a new era of shopping assistants capable of offering human-like interaction and enhancing the overall search experience.

1. Introduction

The Composed Image Retrieval (CIR) problem, also known as Text-Guided Image Retrieval (TGIR), involves finding images that closely match a reference image after applying text modifications. For instance, given a reference image of a blue dress and the instruction "replace blue with red", the retrieved images should depict red dresses resembling the reference.

It is natural for users to search for products using information from multiple modalities, such as images and text. Enabling visual search allows for finding visually similar correspondences and obtaining fine-grained results. Otherwise, text-only search tools would require extensive textual descriptions to reach the same level of detail. Thus, it is more natural and convenient for users to upload a picture of their desired product or a similar version rather than articulating their search entirely in words.

Traditional search engines often struggle to deliver precise results to users due to the challenges posed by overly specific, broad, or irrelevant queries. Moreover, these engines typically lack support for understanding natural language text and reasoning about search queries while conversationally engaging with the user.

In the context of the Fashion200K benchmark [19], several existing approaches fail to retrieve the correct query among the top matches. Concretely, most of the baselines considered in this work fail to retrieve the correct image

† Work performed during an internship at Amazon.

* Equal contribution.

among the top 10 matches in 60% of the cases, as shown in our results in Sec. 4.1.

In this paper, we propose to leverage pretrained large-scale models that can digest image and text inputs. We focus on improving the performance on the Fashion200K dataset [19] and achieve state-of-the-art results that improve upon previous work by a significant margin. However, all the queries in Fashion200K follow the simple formatting "replace {original_attribute} with {target_attribute}", which impedes generalizing to natural language text. For this reason, we develop a novel interactive multimodal search solution leveraging recent advances in LLMs and vision-language models that can understand complex text queries and route them to the correct search tool with the required formatting. Leveraging LLMs facilitates digesting natural language queries and allows taking contextual information into account. Moreover, the length of the context recent LLMs can consider allows for incorporating information from previous interactions. We include a high-level overview of our approach in Fig. 1. The main contributions of this work include:

- **Improved Multimodal Search:** We introduce a method that adapts foundational vision and language models for multimodal retrieval, which achieved state-of-the-art results on Fashion200k. We present the technical details in Sec. 3.1 and discuss the experimental results in Sec. 4.1.
- **Conversational Interface:** We propose an interface that harnesses state-of-the-art LLMs to interpret natural language inputs and route formatted queries to the available search tools. We describe the details of the backend in Sec. 3.2 and include examples in Sec. 4.2.

2. Related work

When tackling the CIR problem, the TIRG model [46] computes an image representation and modifies it with a text representation on the same space rather than fusing both modalities to create a new feature as in most of the other works. Crucially, this method is trained first on image retrieval and gradually incorporates text modifications.

The VAL framework [10] is based on computing image representations at various levels and using a transformer [45] conditioned on language semantics to extract features. Then, an objective function evaluates the feature similarities hierarchically.

The text and image encoders of a CLIP model [36] can be used for zero-shot retrieval with a simple Multi-Layer Perceptron (MLP) [40] and leveraging LLMs [5]. Another approach is to perform a late fusion of CLIP embeddings [4], which can be improved by fine-tuning the CLIP text encoder Baldrati et al. [3]. The hypothesis is that image and text embeddings obtained by CLIP are aligned, while the CIR problem requires a text representation that expresses differences w.r.t. the image representation.

CosMo [25] independently modulates the content and style of the reference image based on the modification text. This work assumes that style information is removed by simply performing instance normalization on the image features. With this assumption in mind, the normalized features are fed to the content modulator, which transforms them conditioned on text features. Then, the output of the content modulator is given to the style modulator, which along with the text features and the channel-wise statistics of the normalization, obtains the final representation.

FashionVLP [16] is based on extracting image features using a pretrained feature extractor, not only on the whole image but also on the cropped clothing, fashion landmarks, and regions of interest. The obtained image representations are concatenated with object tags extracted with an object detector, a class token, and the word tokens computed using BERT [13].

An alternative to tackle the problem of generic visual feature extractors not focusing on fashion-specific details without using the multiple inputs required in Goenka et al. [16], is proposed in FashionSAP [20]. FashionSAP leverages the FashionGen dataset [39] for fine-grained fashion vision-language pretraining. To do that, Han et al. [20] use a multi-task objective composed of retrieval and language modeling losses. CIR is then solved by fusing text and image features using multiple cross-attention layers, and the tasks included in the training objective are solved using different heads for each task.

CompoDiff [17] proposes to solve the CIR using a denoising transformer that provides the retrieval embedding conditioned on features of the reference image and the modifying text. Similarly to Rombach et al. [38], the diffusion process is performed in the latent space instead of the pixel space. Given that CompoDiff is a data-hungry method, Gu et al. [17] create a synthetic dataset of 18 million image triplets using StableDiffusion [38] for its training. CompoDiff performs better when using text features obtained with a T5-XL model [37] in addition to the text representations obtained with [36].

Koh et al. [23] uses a frozen LLM to process the input text and visual features that have been transformed with a learned linear mapping as in LLaVA [32]. To counteract the inferior expressiveness of causal attention over its bidirectional counterpart, Koh et al. [23] append a special [RET] token at the end of the outputs that allows the LLM to perform an extra attention step over all tokens. The hidden representations of [RET] are then mapped to an embedding space that is used for retrieval.

Couairon et al. [12] tackle a similar problem in which the transformation query is not a single word but a tuple of two words corresponding to the original and target attributes. As an example, for a reference image with caption "A cat is sitting on the grass", a source text "cat"

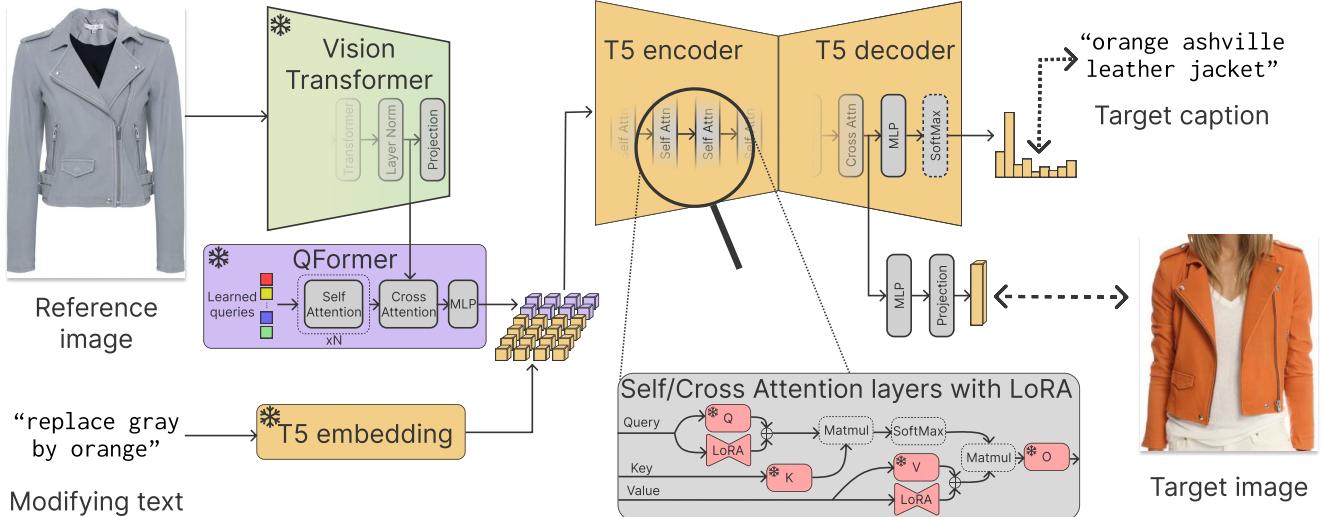


Figure 2. Proposed architecture: We extract visual features from the reference image \mathbf{x}_{ref} using a Vision Transformer [14], specifically, a pretrained CLIP [36] model with frozen weights. We extract features before the projection layer, which are then processed using a Querying transFormer (Q-Former), which performs cross-attention with a set of learned queries. The resulting output of the Q-Former is concatenated with the embeddings obtained from the modifying text (\mathbf{t}), which expresses a modification in the reference image. Subsequently, all this information is fed into a T5 model [37], an encoder-decoder LLM. We employ Low-Rank Adaptation (LoRA) [21] to learn low-rank updates for the query and value matrices in all attention layers, while keeping the rest of the parameters frozen. The output of the LLM yields a probability distribution from which a sentence is generated. To ensure alignment with the target caption (*i.e.*, the caption of the target image \mathbf{x}_{trg} , which corresponds to the caption of the reference image after incorporating the text modifications), a language modeling loss is used. The hidden states of the LLM are then projected into a space of embeddings used for retrieval. A retrieval loss term pushes together the embedding of the target image $\mathcal{G}(\mathbf{x}_{\text{trg}})$ and that obtained using the reference image and the modifying text $\mathcal{F}(\mathbf{x}_{\text{ref}}, \mathbf{t})$.

and a target text "dog", the model should be able to retrieve images of dogs sitting on the grass.

3. Method

In this section, we propose a model to perform an image search merging text and image inputs in Sec. 3.1. While this model outperforms alternative approaches by a large margin, it is trained on a dataset with specific formatting (see Sec. 4.1). Instead of artificially augmenting the vocabulary seen during training as in Gu et al. [17], we propose a conversational interface orchestrated by a LLM that can structure the queries to a format understandable to our multimodal search model.

Sec. 3.2 describes the principles of our approach. The proposed framework offers a modular architecture that allows interchanging search models with different formatting constraints while providing enhanced natural language understanding, a working memory, and a human-like shopping assistant experience.

3.1. Improved multimodal search

In the CIR problem, a dataset \mathcal{D} is composed of triplets with reference and target image as well as a modifying text, *i.e.*, $\mathcal{D} := \{(\mathbf{x}_{\text{ref}}^{(i)}, \mathbf{x}_{\text{trg}}^{(i)}, \mathbf{t}^{(i)})\}_{i \in [n]}$. The objective is to learn the

transformations

$$\mathcal{F} : \mathbf{x}_{\text{ref}} \times \mathbf{t} \mapsto \Psi \quad ; \quad \mathcal{G} : \mathbf{x}_{\text{trg}} \mapsto \Psi \quad (1)$$

along with a metric space (Ψ, d) with fixed $d : \Psi \times \Psi \rightarrow \mathbb{R}$ such that

$$d(\mathcal{F}(\mathbf{x}_{\text{ref}}, \mathbf{t}), \mathcal{G}(\mathbf{x}_{\text{trg}})) < d(\mathcal{F}(\mathbf{x}_{\text{ref}}, \mathbf{t}), \mathcal{G}(\mathbf{x}'_{\text{trg}})) \quad (2)$$

if \mathbf{x}_{ref} after applying the modifications described by \mathbf{t} is semantically more similar to \mathbf{x}_{trg} than it is to \mathbf{x}'_{trg} [6]. Commonly to other works [41, 43, 53], we normalize the space Ψ to the unit hypersphere for training stability, and choose d to be the cosine distance.

In this work, we use off-the-shelf foundational models for vision and language to compute the transformation \mathcal{F} . Concretely, we use an architecture similar to BLIP2 [29] and adapt it for the CIR problem. BLIP2 [29] uses a module referred to as the Q-Former, which allows ingesting image features obtained by a powerful feature extractor. These image features provide fine-grained descriptions of the input product and are transformed into the space of text embeddings of a LLM. Then, the LLM processes the fused text and image embeddings.

The Q-Former consists of two transformer submodules sharing the same self-attention layers to extract information from the input text and the image features. The image trans-

formers also contain a set of learnable query embeddings, which can be interpreted as a form of prefix tuning [30].

To generate image-only search embeddings using our model, one simply needs to input the images into the model and provide an empty string as the input text. Intuitively, this processes the images without any text modifications. In other words, we use

$$\mathcal{G}(\mathbf{x}) := \mathcal{F}(\mathbf{x}, "") \quad (3)$$

We illustrate the proposed architecture for \mathcal{F} in Fig. 2. We use the image part of the CLIP [36] model to obtain visual features and a T5 model [37] as LLM to process the modifying text and the visual features processed by the Q-Former.

We initialize the model using the BLIP2 weights with all the parameters frozen. The pretrained weights perform the task of image captioning, which is different from the task we are trying to solve. Instead, we define a new task that we refer to as *composed captioning*. The objective of this task is to generate the caption of the product that we would obtain by merging the information of the product in the input image and the text modifications.

We hypothesize that if the proposed model can solve the problem of *composed captioning*, the information captured by the LLM is enough to describe the target product. Intuitively, similarity search happens at a latent space close to the final text representations, making the CIR problem closer to the task of text-to-text retrieval. However, as the proposed model is able to capture fine-grained information by leveraging powerful visual representations, we are able to obtain an impressive retrieval performance. This is expected as the BLIP2 achieves state-of-the-art performance on Visual Question Answering (VQA) benchmarks, showing that image information can be effectively captured.

To adapt the LLM to this task while retaining its knowledge, we applied LoRA [21] to the query and value matrices of all the self-attention and cross-attention layers of the LLM. LoRA [21] learns a residual representation on top of some layers using matrices with low rank. Theoretically, this is supported by the fact that LLMs adapted to a specific task have low intrinsic dimension [1], and in practice it allows training with low computational resources and limited data. Moreover, only modifying a few parameters reduces the risk of catastrophic forgetting, observed in some studies where full fine-tuning of an LLM decreases the performance compared to using it frozen or fine-tuning it with parameter-efficient techniques [23, 31].

The hidden states of the T5 decoder are a sequence of tensors. Instead of using a class-like token as in Koh et al. [23] to summarize the information along the temporal dimension, we perform an average followed by layer normalization [2]. This technique was utilized in EVA [15], which improves over CLIP [36] in several downstream tasks. The

result is then projected to the embedding dimension using a ReLU-activated MLP and followed by normalization.

We train the model using a multi-task objective involving the InfoNCE loss [34], a lower bound on the mutual information [27], as retrieval term:

$$\begin{aligned} \mathcal{L}_{\text{InfoNCE}} &:= \mathbb{E}_i \left[\log \frac{\exp(S_{i,i} \cdot \tau)}{\sum_j \exp(S_{i,j} \cdot \tau)} \right] \\ S_{i,j} &:= \left\langle \mathcal{F}(\mathbf{x}_{\text{ref}}^{(i)}, \mathbf{t}^{(i)}), \mathcal{G}(\mathbf{x}_{\text{trg}}^{(j)}) \right\rangle, \end{aligned} \quad (4)$$

where τ is a learnable scaling parameter. Practically, given that our model has many parameters, the maximum batch sizes we can achieve have an order of magnitude of hundreds of samples. Given that this can affect the retrieval performance due to a lack of negative samples, we maintain a cross-batch memory as proposed in Wang et al. [47] and use it for the computation of Eq. (4).

On top of that, we add a standard maximum likelihood as a language modeling term \mathcal{L}_{LM} . We compute this objective using teacher forcing [50], based on providing the ground-truth outputs of previous tokens to estimate the next token, and cross-entropy loss. The final loss is

$$\mathcal{L} = \mathcal{L}_{\text{LM}} + \omega \mathcal{L}_{\text{InfoNCE}}, \quad (5)$$

where ω is a hyperparameter determining the relative importance of the retrieval task.

3.2. Conversational interface

Inspired by Visual ChatGPT [52], we connect a user chat to a prompt manager that acts as a middle-man to a LLM and provides it with access to tools. Differently from Wu et al. [52], these tools are not only to understand and modify images but also to perform searches with both unimodal and multimodal inputs.

From the user's perspective, the proposed framework allows implicitly using a search tool without requiring any input pattern. For example, interacting with a model like SIMAT [12] could be unintuitive as it requires two words with the original and target attributes. We trained our multimodal search model on Fashion200K [19], which only contains inputs of the form "replace {original_attribute} with {target_attribute}" (see Sec. 4.1). We could formulate this prompt using the same inputs that a model like SIMAT requires and thus modify them to match the training distribution of our model.

Since the LLMs can only ingest text information, we add image understanding tools to provide information about the images and their content, as well as search tools:

Image search: Image-only search based on CLIP [36] image embeddings. We use this tool internally when a user uploads an image to show an initial result to users, which

may inspire them to write the follow-up queries. The descriptions of the search results are provided to the LLM to enable Retrieval Augmented Generation (RAG) [26]

Multimodal search: The input of the multimodal search tool is an image and two text strings expressing the original and target attributes. We use our model and feed it the Fashion200K [19] prompt created from these attributes.

VQA model: We use the BLIP [28] pretrained base model¹ to facilitate image understanding to the LLM.

Our approach to providing image information to the LLM is similar to LENs [7], as it is a training-free method applicable to any off-the-shelf LLM.

3.2.1 Workflow

In this section, we describe the main events in the interface and the triggered actions.

Start: When a new user starts a new session, we create a unique identifier used to set a dedicated folder to store images and initialize the memory to store the context. The memory contains a conversation where the lines prefixed with "Human :" come from the user, and those starting with "AI :" are outputs of the LLM shown to the user.

Image input: When a user uploads an image, we store it in the session folder using file names with sequential numerical identifiers, *i.e.*, IMG_001.png, IMG_002.png, IMG_003.png, *etc.* Then we add a fake conversation to the memory:

```
Human: I provided a figure named {image_filename}. {
    description}
AI: Provide more details if you are not satisfied with
    the results.
```

where `description` is the text output of the search action.

Search: Every time a search tool is used, the results are shown to the user in a carousel of images. Additionally, we add the following information to the memory that will be provided to the LLM once invoked

```
Top-{len(image_descriptions)} results are: {
    image_descriptions}.
```

which contains the descriptions of the top retrieved images. These details help the LLM understand the fine-grained details (*e.g.*, brand, product type, technical specifications, color, *etc.*) and the multimodal search intention. We can interpret this as a form of RAG [26]. RAG is based on using an external knowledge base for retrieving facts to ground LLMs on the most accurate and up-to-date information.

Text input: Every time the user provides some text input, we invoke the LLM through the prompt manager. In this stage, the LLM can communicate directly to the user or use special formatting to call some tools. If the LLM wants to

¹<https://huggingface.co/Salesforce/blip-vqa-base>

perform a multimodal search, it can typically find the target attribute in the text input, which only needs to be formatted and simplified. However, in most cases, the original attribute is not included in the input text as it is implicit in the image. Generally, the descriptions contain enough information to perform the query. Otherwise, the LLM can use the VQA model to ask specific questions about the image.

3.2.2 Prompt manager

The prompt manager implements the workflow described in the previous section and empowers the LLM with access to different tools. The tool calls are coordinated by defining a syntax that processes the output of the LLM and parses the actions and the text visible to the user in the chat.

Every time the LLM is triggered, the prompt manager does so with a prompt that includes a description of the task, formatting instructions, previous interactions, and outputs of the tools.

We crafted a task description that specifies that the LLM can ask follow-up questions to the customers if the search intents are unclear or the query is too broad. In the prompt, we also include examples of use cases written in natural language. The formatting instructions describe when the LLM should use a tool, which are the inputs, how to obtain them, and what are the tool outputs.

For each tool, we have to define a name and a description that may include examples, input and output requirements, or cases where the tool should be used.

In this work, we test two prompt managers:

Langchain [9]: We take the Langchain prompts from Visual ChatGPT [52] and adapt them to our task. The syntax to use a tool is:

```
Thought: Do I need to use a tool? Yes
Action: Multimodal search
Action Input: IMG_001.png;natural;black
```

Our prompt manager: Inspired by the recent success of visual programming [18, 44], we propose to use a syntax similar to calling a function in programming languages:

```
SEARCH (IMG_001.png;natural;black)
```

In Fig. 1, we illustrate an example of a conversation and the actions that the prompt manager and the LLM trigger.

Visual programming typically performs a single call to a LLM, and the output is a single action or a series of actions whose inputs and outputs can be variables defined on the fly by other functions. While Langchain [9] allows performing multiple actions, it requires executing them one at a time. When the LLM expresses the intention to use a tool, Langchain calls the tool and prompts the LLM again with the output of such a tool. The visual programming approach only invokes the LLM once, saving latency and possible costs attributed to API calls. However, in visual programming, the LLM cannot process the output of tools

Table 1. **Quantitative results:** Recall@ k on the Fashion200K dataset [19]. Our method is able to successfully fuse image and text information and generate a representation that is useful to caption the resulting image and generate an embedding for retrieval purposes. Best results shown in **boldface**.

Method ↓	R@10	R@50	Average
RN [42]	40.5	62.4	51.4
MRN [22]	40.0	61.9	50.9
FiLM [35]	39.5	61.9	50.7
TIRG [46]	42.5	63.8	53.2
CosMo [25]	50.4	69.3	59.8
FashionVLP [16]	49.9	70.5	60.2
VAL [10]	53.8	73.3	63.6
Ours	71.4	91.6	81.5

but only use their outputs blindly. For the sake of simplicity, we restrict the custom prompt manager to handle single actions, but this could easily be extended following Gupta and Kembhavi [18], Surís et al. [44].

Additionally, we propose to include Chain-of-Thought (COT) [24, 49, 54, 55]. COT is a technique that enforces that the LLM reasons about the actions that should be taken. This simple technique has reportedly found numerous benefits. Following the example above, the complete output expected by the LLM would be as follows:

```
Thought: I can see that human uploaded an image of a
       deep v-neck tee. From the results, the color of the
       tee is natural. The user wants the color to be
       black instead. I have to call search.
Action: SEARCH(IMG_001.png;natural;black)
```

While Langchain and our prompt manager use the special prefix "Thought" to handle certain parts of the query, their purposes are distinct. In Langchain, the prefix is used to parse lines in the LLM output. If a line starts with this prefix, Langchain expects to find the question "Do I need to use a tool?" followed by "Yes" or "No", indicating whether a tool should be used. In contrast, our novel prompt manager does not impose any specific format on lines starting with the "Thought" prefix. Instead, these lines are solely dedicated to incorporating COT reasoning.

4. Experiments

4.1. Multimodal search on Fashion200K

Implementation details: We use the Flan T5 XL model [11], which is a 3 billion parameter LLM from the T5 family [37], finetuned using instruction tuning [48]. We obtain the visual features with CLIP-L model [36], a model with patch size 14 and 428 million parameter. In total, the model has around 3.5 million parameter, which requires splitting the model across different GPUs for training. Concretely, we use 8 NVIDIA V100 GPUs.

LoRA is performed with a rank of $r = 16$, scaling $\alpha = 32$ and dropout of 0.5 on the query and value matrices of the attention layers of the LLM. The hidden representation obtained from the LLM is transformed with a linear layer of size 1024, passed through a ReLU activation, and then transformed with another linear layer that yields an embedding of size 768. Such an embedding is normalized to have unit norm and used for retrieval.

We optimize the model with AdamW [33] with a learning rate of 10^{-5} and weight decay of 0.5 for a total of 300 epochs. The learning rate is linearly increased from 0 to the initial learning rate during the first 1000 steps. We set the weight of the language modeling loss as $\omega = 1$. The effective batch size considering all the GPUs is 4,096, and the total number of embeddings included in the cross-batch memory Wang et al. [47] is 65,536.

Dataset: The Fashion200K [19] is a large-scale fashion dataset crawled from online shopping websites. The dataset contains over 200,000 images with paired product descriptions and attributes. All descriptions are fashion-specific and have more than four words, e.g., "Beige v-neck bell-sleeve top". Similarly to Vo et al. [46], text queries for the CIR problem are generated by comparing the attributes of different images and finding pairs with one attribute difference. Then, a query is formed as "replace {original_attribute} with {target_attribute}".

When trained on Fashion-200K [19], our method achieves state-of-the-art results, improving the retrieval performance of competitive methods by 20% recall at positions 10 and 50. Tab. 1 includes the comparison with some of the CIR methods reviewed in Sec. 2 [10, 16, 25, 46], as well as the visual reasoning-based baselines RN [42], MRN [22], and FiLM [35].

One of the reasons is that the model can exploit the image and text understanding prior of a foundational model that can perform image captioning, and adapt it for the related task of *composed captioning*. The hidden representations of the model contain enough information to describe the target image and are effectively used for that purpose. Adapting to this new task becomes easier given the specific formatting of the modifying text, which facilitates extracting the important parts of the query.

The results show that it is possible to distill knowledge from a large vision and language model trained on large-scale datasets. While our model has billions of parameters, which is far more than the other models, we are able to learn a new task similar to the ones that the pretrained model could solve with only learning a few parameters consisting of a very small percentage of the total model size.

We include some qualitative examples in Fig. 3. These show that our model can successfully incorporate text information and modify the internal description formed about



(a) Successful examples

(b) Failure examples

Figure 3. **Qualitative results:** Examples of queries of the Fashion-200k dataset [19] and the 4 best matches. The correct matches are shown in **green** and incorrect ones in **red**. In the successful examples, we can see that our proposal is able to incorporate modifications to the input product involving changes to color and material among others. Despite not retrieving the correct products in the failure examples, almost all the retrieved images satisfy the search criteria.

the input image. The successful results in Fig. 3a show that the proposed model retrieves visually similar and can incorporate modifications of different attributes such as the color and the material.

The failures in Fig. 3b show that all the first retrieve results satisfy the search criteria, with some of them even belonging to the same product. This hints at our model having an even better performance in practice than what the benchmark reflects.

Overall, we can see from all the qualitative examples that all the top-ranked results are relevant. The only exception is the inclusion of the reference image, which is a common error in retrieval systems given that the search embedding is computed from such an image.

4.2. Search interface

One of the key drivers of performance is based on reformulating the examples. While the examples in Langchain are written using natural language, we advocate for using LLM model instructions. In this sense, the examples contain exactly the input that the LLM would receive including the product type, top- k product titles, and user input. Such examples also contain the expected model output including the COT reasoning and the action itself. This reinforces the format instructions and the benefits of RAG.

Note that the proposed reformulation introduces some redundancy w.r.t. the Langchain formatting instructions. Additionally, it requires to allocate much more space for examples. Despite these considerations, we find our approach beneficial. For a fair comparison, we also limit the full prompt to fit the context of the smallest LLM and empirically find that allocating more space to examples is ben-

eficial even if this is at the cost of removing the prefix.

We tested different LLMs for the search interface. Among all, GPT-3 [8], concretely the `text-davinci-003` model, was empirically found to be the best performing. Fig. 4 shows an example of our conversational interface displaying a real example in which composed retrieval is performed.

Besides GPT-3 [8], we compared different open-source models from the `transformers` library [51]. Surprisingly, these models performed poorly. Digging into the outputs of the LLMs we could see that one of the failure cases of Fastchat [56] had the following output:

```
Thought: Do I need to use a tool? Yes
Action: Multimodal altering search
Action Input: image_file: IMG_001.png, attribute value
of the product in the image: chair, desired
attribute value: sofa
```

While the former contains the correct action to take and the correct inputs, *i.e.* image file, negative text and positive text, it is not correctly formatted for Langchain. Instead, GPT-3 [8] is able to generate a correctly formatted output:

```
Thought: Do I need to use a tool? Yes
Action: Multimodal altering search
Action Input: IMG_001.png;chair;sofa
```

This example shows that FastChat [56] has the knowledge to perform a successful query but struggles to use the complicated formatting of Langchain. This example is the main motivation why we developed the novel prompt manager in Sec. 3.2.2.

5. Limitations

The model in Sec. 3.1 can achieve an impressive performance on Fashion200K. As discussed in Sec. 4.1, the char-

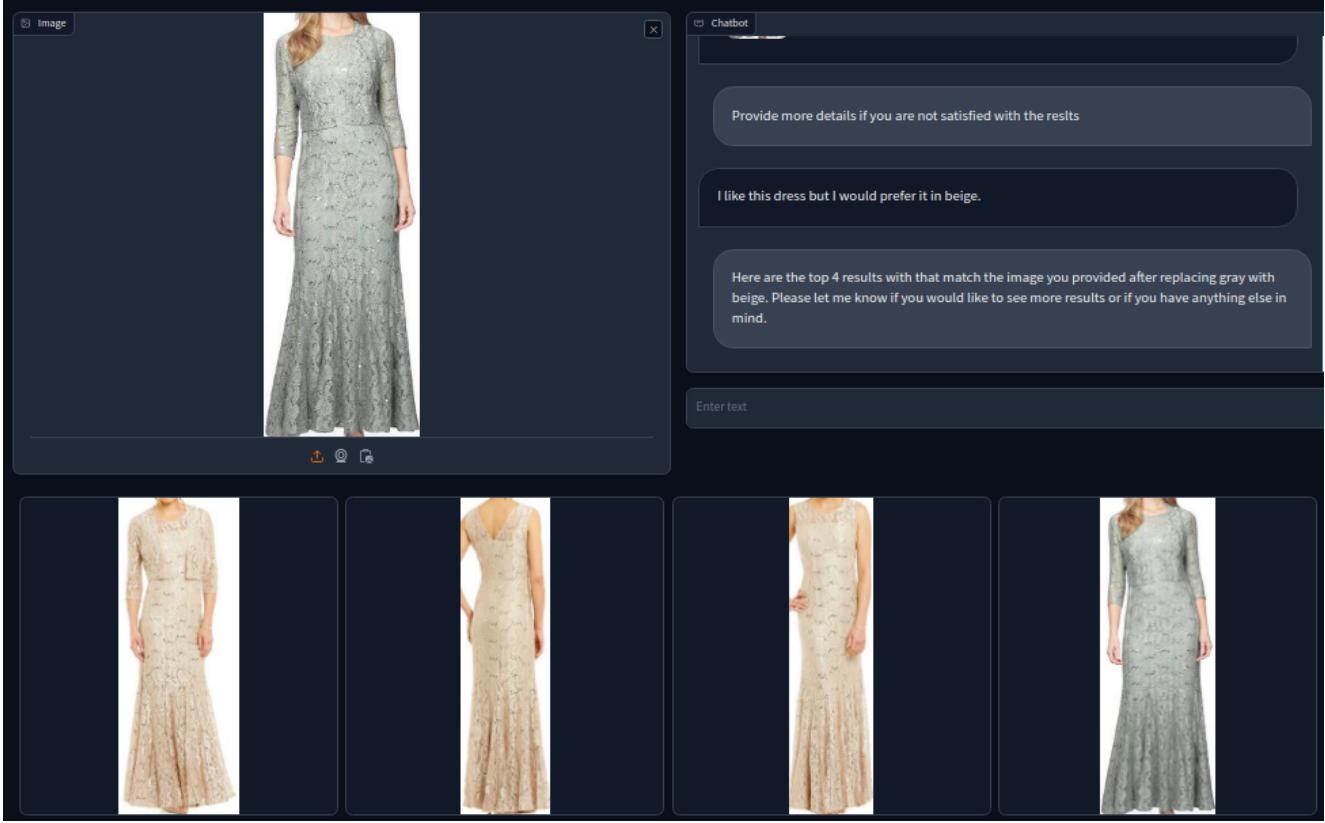


Figure 4. Proposed conversational multimodal search system: In this example, the user uploads an image from the Fashion200K dataset [19] and provides text input intending to search an a dress similar to the product in the image but in a different color. An LLM, specifically GPT-3 [8], processes the user’s prompt and invokes our novel multimodal search model with the uploaded image and a formatted text query. The desired attribute indicated by the user is “beige”, which can be inferred from the text input. The original attribute is required by the prompt used during the training of our model and is correctly identified by the LLM as “gray”. In this case, the LLM can obtain this information leveraging the RAG based on obtaining the product descriptions of the first matches using image search with the uploaded picture. The conversational nature of the interactions with the user offers an improved search experience.

acteristics of this dataset are ideal for our model to excel but may hinder generalizing to natural language queries. This is solved with our conversational interface, but the current setup is restricted to modifying a single attribute at a time.

Using hard prompts to encode the task description is simple and applicable to black-box models such as LLMs accessed through an API. However, it reduces the effective context length of LLMs and requires prompt engineering, which is a tedious process.

Although LLMs have a large context size, the prompt yields an effective input size that is relatively small, and the memory rapidly fills up. In practice, the memory gets truncated if conversations are too long, hence discarding the first interactions.

6. Conclusions

This paper presents a comprehensive pipeline to perform image retrieval with text modifications, addressing the CIR problem. Our novel composed retrieval model, built upon

the BLIP2 architecture [28] and leveraging LLMs, has demonstrated superior performance on the Fashion200K dataset [19] compared to previous models.

In this work, we also describe the integration of LLMs into a search interface, offering a conversational search assistant experience that enhances user interaction. We implement a prompt manager to enable using small LLMs and incorporate the COT [24, 49] and RAG [26] techniques to improve system performance.

Our experiments underscore the importance of addressing inherent challenges in multimodal search, including enhancing matching capabilities and handling ambiguous natural language queries.

Acknowledgments

The authors acknowledge René Vidal for constructive discussions. O.B. is part of project SGR 00514, supported by Departament de Recerca i Universitats de la Generalitat de Catalunya.

References

- [1] Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *International Joint Conference on Natural Language Processing*, 2021. 4
- [2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. arXiv:1607.06450, 2016. 4
- [3] Alberto Baldrati, Marco Bertini, Tiberio Uricchio, and Alberto Del Bimbo. Conditioned and composed image retrieval combining and partially fine-tuning CLIP-based features. In *CVPRW*, New Orleans, LA, USA, 2022. 2
- [4] Alberto Baldrati, Marco Bertini, Tiberio Uricchio, and Alberto Del Bimbo. Effective conditioned and composed image retrieval combining CLIP-based features. In *CVPR*, 2022. 2
- [5] Alberto Baldrati, Lorenzo Agnolucci, Marco Bertini, and Alberto Del Bimbo. Zero-Shot Composed Image Retrieval with Textual Inversion. In *ICCV*, 2023. 2
- [6] Aurélien Bellet, Amaury Habrard, and Marc Sebban. *Metric Learning*. Morgan & Claypool Publishers (USA), Synthesis Lectures on Artificial Intelligence and Machine Learning, pp 1-151, 2015. 3
- [7] William Berrios, Gautam Mittal, Tristan Thrush, Douwe Kiela, and Amanpreet Singh. Towards Language Models That Can See: Computer Vision Through the LENS of Natural Language. arXiv:2306.16410, 2023. 5
- [8] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners. In *NeurIPS*, 2020. 7, 8
- [9] Harrison Chase. LangChain. <https://github.com/langchain-ai/langchain>, 2022. 5
- [10] Yanbei Chen, Shaogang Gong, and Loris Bazzani. Image search with text feedback by visiolinguistic attention learning. In *CVPR*, 2020. 2, 6
- [11] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models. arXiv:2210.11416, 2022. 6
- [12] Guillaume Couairon, Matthijs Douze, Matthieu Cord, and Holger Schwenk. Embedding Arithmetic of Multimodal Queries for Image Retrieval. In *CVPRW*, 2022. 2, 4
- [13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 2
- [14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021. 3
- [15] Yuxin Fang, Wen Wang, Binhui Xie, Quan Sun, Ledell Wu, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. Eva: Exploring the limits of masked visual representation learning at scale. In *CVPR*, 2023. 4
- [16] Sonam Goenka, Zhaoheng Zheng, Ayush Jaiswal, Rakesh Chada, Yue Wu, Varsha Hedau, and Pradeep Natarajan. FashionVLP: Vision Language Transformer for Fashion Retrieval with Feedback. In *CVPR*, 2022. 2, 6
- [17] Geonmo Gu, Sanghyuk Chun, Wonjae Kim, HeeJae Jun, Yoohoon Kang, and Sangdoo Yun. CompoDiff: Versatile Composed Image Retrieval With Latent Diffusion. arXiv:2303.11916, 2023. 2, 3
- [18] Tanmay Gupta and Aniruddha Kembhavi. Visual Programming: Compositional Visual Reasoning Without Training. In *CVPR*, 2023. 5, 6
- [19] Xintong Han, Zuxuan Wu, Phoenix X. Huang, Xiao Zhang, Menglong Zhu, Yuan Li, Yang Zhao, and Larry S. Davis. Automatic spatially-aware fashion concept discovery. In *ICCV*, 2017. 1, 2, 4, 5, 6, 7, 8
- [20] Yunpeng Han, Lisai Zhang, Qingcai Chen, Zhijian Chen, Zhonghua Li, Jianxin Yang, and Zhao Cao. Fashion-sap: Symbols and attributes prompt for fine-grained fashion vision-language pre-training. In *CVPR*, 2023. 2
- [21] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models, 2021. arXiv:2106.09685 [cs]. 3, 4
- [22] Jin-Hwa Kim, Sang-Woo Lee, Donghyun Kwak, Min-Oh Heo, Jeonghee Kim, Jung-Woo Ha, and Byoung-Tak Zhang. Multimodal residual learning for visual qa. In *NeurIPS*, 2016. 6
- [23] Jing Yu Koh, Ruslan Salakhutdinov, and Daniel Fried. Grounding Language Models to Images for Multimodal Inputs and Outputs. In *ICML*, 2023. 2, 4
- [24] Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large Language Models are Zero-Shot Reasoners. In *NeurIPS*, 2022. 6, 8
- [25] Seungmin Lee, Dongwan Kim, and Bohyung Han. Cosmo: Content-style modulation for image retrieval with text feedback. In *CVPR*, 2021. 2, 6
- [26] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *NeurIPS*, 2020. 5, 8
- [27] Junnan Li, Ramprasaath R. Selvaraju, Akhilesh Deepak Gotmare, Shafiq Joty, Caiming Xiong, and Steven Hoi. Align before fuse: Vision and language representation learning with momentum distillation. In *NeurIPS*, 2021. 4
- [28] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified

- vision-language understanding and generation. In *ICML*, 2022. 5, 8
- [29] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models, 2023. 3
- [30] Xiang Lisa Li and Percy Liang. Prefix-Tuning: Optimizing Continuous Prompts for Generation, 2021. arXiv:2101.00190 [cs]. 4
- [31] Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In *NeurIPS*, 2022. 4
- [32] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual Instruction Tuning. arXiv:2304.08485, 2023. 2
- [33] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 6
- [34] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. arXiv:1807.03748, 2018. 4
- [35] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. Film: Visual reasoning with a general conditioning layer. In *AAAI*, 2018. 6
- [36] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. In *ICML*, 2021. 2, 3, 4, 6
- [37] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. 2, 3, 4, 6
- [38] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. arXiv:2112.10752, 2021. 2
- [39] Negar Rostamzadeh, Seyedarian Hosseini, Thomas Boquet, Wojciech Stokowiec, Ying Zhang, Christian Jauvin, and Chris Pal. Fashion-gen: The generative fashion dataset and challenge. arXiv:1806.08317, 2018. 2
- [40] Kuniaki Saito, Kihyuk Sohn, Xiang Zhang, Chun-Liang Li, Chen-Yu Lee, Kate Saenko, and Tomas Pfister. Pic2Word: Mapping Pictures to Words for Zero-shot Composed Image Retrieval. In *CVPR*, 2023. 2
- [41] Artsiom Sanakoyeu, Vadim Tschernezki, Uta Büchler, and Björn Ommer. Divide and conquer the embedding space for metric learning. In *CVPR*, 2019. 3
- [42] Adam Santoro, David Raposo, David G.T. Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. A simple neural network module for relational reasoning. In *NeurIPS*, 2017. 6
- [43] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015. 3
- [44] Dídac Surís, Sachit Menon, and Carl Vondrick. ViperGPT: Visual Inference via Python Execution for Reasoning. arXiv:2303.08128, 2023. 5, 6
- [45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2
- [46] Nam Vo, Lu Jiang, Chen Sun, Kevin Murphy, Li-Jia Li, Li Fei-Fei, and James Hays. Composing Text and Image for Image Retrieval - an Empirical Odyssey. In *CVPR*, 2019. 2, 6
- [47] Xun Wang, Haozhi Zhang, Weilin Huang, and Matthew R Scott. Cross-batch memory for embedding learning. In *CVPR*, 2020. 4, 6
- [48] Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*, 2022. 6
- [49] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. Chain of Thought Prompting Elicits Reasoning in Large Language Models. In *NeurIPS*, 2022. 6, 8
- [50] Ronald J. Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1989. 4
- [51] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierrick Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2020. 7
- [52] Chenfei Wu, Shengming Yin, Weizhen Qi, Xiaodong Wang, Zecheng Tang, and Nan Duan. Visual ChatGPT: Talking, Drawing and Editing with Visual Foundation Models. arXiv:2303.04671, 2023. 4, 5
- [53] Chao-Yuan Wu, R Manmatha, Alexander J Smola, and Philipp Krahenbuhl. Sampling matters in deep embedding learning. In *ICCV*, 2017. 3
- [54] Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. Automatic Chain of Thought Prompting in Large Language Models. In *ICLR*, 2023. 6
- [55] Zhuosheng Zhang, Aston Zhang, Mu Li, Hai Zhao, George Karypis, and Alex Smola. Multimodal Chain-of-Thought Reasoning in Language Models. arXiv:2302.00923, 2023. 6
- [56] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging Ilm-as-a-judge with mt-bench and chatbot arena. arXiv:2306.05685, 2023. 7

UTRECHT UNIVERSITY

MASTER ARTIFICIAL INTELLIGENCE

THESIS (44 ECTS)

**Enhancing Customer Support Chatbots
with LLMs: Comparative Analysis of
Few-Shot Learning, Fine-Tuning, and
RAG including the Proposal of
an Integrated Architecture**

Author

R.J. WOERING
6570941

Supervisors

Dr. Marijn SCHRAAGEN
Dr. Shihan WANG

External Supervisor

Wouter van OOSTEN MSc.
ing. Maarten KRUIT

December 4, 2024



Abstract

Recent years have seen a growing interest in advanced chatbots, mainly due to breakthroughs in Artificial Intelligence (AI). Particularly, the integration of Large Language Models (LLMs) has increased chatbot capabilities, making them increasingly viable in various domains, such as customer support. This thesis delves into the area of chatbots, enhanced by LLMs and different techniques to increase context-specific capabilities like fine-tuning, few-shot learning, and retrieval-augmented generation, within the context of customer support. This thesis consisted of two separate phases, the first compared the effectiveness of fine-tuning, few-shot learning, and RAG, against each other to identify the most effective method for enhancing chatbot responses. An evaluation framework was developed, combining automated metrics with human judgment to analyze chatbot performance across various customer service scenarios, based on multiple metrics. The results indicate that RAG outperforms other methods on all metrics, demonstrating superior response quality in customer interactions. In the second phase, a final chatbot architecture was proposed, based on these findings, leveraging the strengths of RAG while integrating a handover module and template enhancements to address its limitations. The proposed architecture was tested using human judgment and compared to the RAG method for the first phase. A second evaluation showed that the complete architecture showed overall improved performance and reliability, with considerably higher scores on most metrics. The study highlights the crucial role of RAG in advancing chatbot intelligence and offers insights for the development of more effective LLM-driven customer service support bots.

Keywords: Chatbots, Customer support, Few-shot learning, Fine-tuning, Large Language Models, Natural Language Processing, Retrieval-Augmented Generation

Contents

1	Introduction	1
1.1	Background	1
1.1.1	The need for customer support bots and LLMs	2
1.1.2	Some challenges of chatbots and LLMs	2
1.2	Research gap	3
1.3	Research question	4
1.4	Current study	5
1.4.1	Few-shot learning	5
1.4.2	Fine-tuning	5
1.4.3	Retrieval-Augmented Generation	6
1.4.4	Evaluation	6
2	Related work	8
2.1	Chatbots	8
2.1.1	History	9
2.1.2	Core methods and technologies	9
2.1.3	Architectures and frameworks	12
2.1.4	Evaluation methods	19
2.1.5	Challenges	23
2.2	Customer end	24
2.2.1	User experience	25
2.2.2	Customer perception	26
2.2.3	Customer preferences and expectations	27
3	Methods	29
3.1	Research overview	29
3.2	Data process	30
3.2.1	Data collection	31
3.2.2	Clearing up process	31
3.2.3	Knowledge Base	33
3.3	Phase 1 - Initial response generator	34
3.3.1	Retrieval database	34

3.3.2	Response Generator	35
3.3.3	Preparations for the first evaluation	40
3.3.4	Evaluation	42
3.4	Phase 2 - Complete chatbot architecture	44
3.4.1	Chatbot architecture	44
3.4.2	Response generator changes	45
3.4.3	Handover module	46
3.4.4	Preparations for the second evaluation	48
3.4.5	Evaluation	49
4	Results	50
4.1	Chatbot responses	50
4.1.1	Zero-shot responses	50
4.1.2	Few-shot learning responses	51
4.1.3	Fine-tuning responses	51
4.1.4	RAG responses	51
4.2	Phase 1: Evaluation	53
4.2.1	Survey overview	53
4.2.2	Relevance and Accuracy	54
4.2.3	Completeness and Conciseness	56
4.2.4	Empathy and Friendliness	57
4.2.5	Impact of experience	58
4.2.6	Initial questions	58
4.2.7	Human vs. Automatic metrics	59
4.3	Phase 2: Evaluation	62
4.3.1	Survey overview	62
4.3.2	RAG vs. changed architecture	63
4.3.3	Templates and the handover module performance	65
4.3.4	Initial scenarios	66
5	Discussion	68
5.1	Phase 1: Research	68
5.1.1	Main research question 1	69
5.1.2	Other discussion points	72
5.2	Phase 2: Research	76
5.2.1	Template enhancements and handover module decisions	76
5.2.2	Main research question 2	79
5.2.3	Other discussion points	81
5.3	Overall limitations	84
5.3.1	Data limitations	84
5.3.2	Limitations in response generation	85
5.3.3	Experimental limitations	86
5.4	Conclusion	86

5.4.1 Our recommendations	87
5.5 Future work	88
5.5.1 Architectural changes	88
5.5.2 Other LLMs	90
5.5.3 Improve retrieval process	90
5.5.4 New evaluation metrics	91
5.5.5 Business aspect of LLMs	92
Appendix	105
A Data types table	105
B Initial questions	107
B.1 Original questions	107
B.2 English translated questions	109
C Initial scenarios	111
C.1 Original scenarios	111
C.2 English translated scenarios	113
D Templates 1st phase	114
D.1 Chat history template	114
D.2 Few-shot conversion template	115
E Detailed statistical test results	116
F Best- and worst-performing questions	117
F.1 Relevance	117
F.2 Accuracy	118
F.3 Completeness	119
F.4 Conciseness	120
F.5 Empathy	121
F.6 Friendliness	122
G Initial question scores	123
G.1 Relevance	123
G.2 Accuracy	124
G.3 Completeness	125
G.4 Conciseness	126
G.5 Empathy	127
G.6 Friendliness	128

H Best- and worst-performing scenarios	129
H.1 Relevance	129
H.2 Accuracy	130
H.3 Completeness	131
H.4 Conciseness	132
H.5 Empathy	133
H.6 Friendliness	134
I BERTScore comparison	135
J Discrepancies	136
J.1 Question 2 discrepancy	136
J.2 Question 6 discrepancy	137
J.3 Question 22 discrepancy	138
K Incorrect combination of sources	139
L Enhanced templates	140
L.1 Response creation	140
L.2 Chat history	141
L.3 Relevance	142
L.4 Specificity	143
M Handover module check examples	144
M.1 First relevance check	144
M.2 Second relevance check	144
M.3 Specificity check	145
N Repetition example specificity check	146

Chapter 1

Introduction

Chatbots, also called dialog systems, have been around ever since the first chatbot ELIZA was introduced in 1966. Since then chatbots have become more and more complex ([Braggaar et al., 2023](#)). In the past few years, however, the evolution of chatbots has exploded, particularly with the advent of Large Language Models (LLMs). A well-known example of this development is ChatGPT, being one of the standout chatbots in revolutionizing how the general public perceives chatbots and interacts with them ([Topsakal and Akinci, 2023](#)). The surge in chatbot capabilities has opened many new possibilities, with chatbots now serving casual chit-chatting but also complex problem-solving ([Noever and McKee, 2023](#)), emotional support ([Liu et al., 2021](#)) and customer support ([Suthar, 2018](#)).

This research contains a case study conducted at a Dutch software company and aims to delve into the area of chatbots in combination with LLMs. We will focus on the strengths and limitations of LLMs. This study will also delve into different Natural Language Processing techniques that are used for chatbots, but also for LLMs in particular. Furthermore, we will focus on the role of these chatbots in addressing customer queries and resolving customer's problems. The significance of this focus cannot be overstated, as customer support plays a pivotal role in shaping the overall customer experience and influencing key metrics such as customer retention and satisfaction ([Misischia et al., 2022](#)). Customer support chatbots, combined with LLMs, have the potential to transform the way companies engage with their customers. By leveraging different machine learning methods in combination with LLMs, these chatbots can comprehend and respond to a wide range of customer queries, providing accurate and personalized assistance tailored to each individual's needs.

1.1 Background

We will cover some background information about chatbots, the perspective of the customer and companies, and Large Language Models, in this section.

1.1.1 The need for customer support bots and LLMs

Up until a couple of years ago, customer support was mostly done over the phone, by email, or using forums and message boards (Hardalov et al., 2018), however, chatbots have been introduced as another way of connecting with the customer (Nuruzzaman and Hussain, 2020). One of the primary motivations behind the adoption of customer support chatbots is to enhance service quality and improve customer satisfaction (Meyer-Waarden et al., 2020). By automating routine tasks and handling repetitive queries, chatbots enable customer service representatives to focus their efforts on more complex issues, thereby streamlining operations and maximizing efficiency (Cui et al., 2017; Hardalov et al., 2018). This not only reduces response times but also ensures consistency in service delivery, regardless of the amount of questions received. Chatbots can also operate effortlessly 24/7 so that customer support is available all the time (Hardalov et al., 2018). Moreover, the use of chatbots in customer support has been shown to significantly reduce operational costs for companies. By automating routine interactions and minimizing the need for human intervention, companies can achieve substantial savings in staffing and operational expenses, while simultaneously increasing the scalability of their customer support operations (Bavaresco et al., 2020). Furthermore, customer support chatbots can contribute to fostering customer loyalty and brand advocacy. By engaging in dialogues (Nuruzzaman and Hussain, 2020) and providing personalized assistance, chatbots create positive experiences for customers (Huseynov, 2023). Satisfied customers are more likely to recommend the brand to others and become repeat customers themselves (Ginting et al., 2023), thereby contributing to long-term business success.

Integrating LLMs with customer support chatbots can help carry out these advantages of chatbots. LLMs are trained on vast amounts of data and can offer some advantages compared to previous NLP methods. According to Yao et al. (2024), they possess deep language comprehension, human-like text generation capabilities, contextual awareness, and robust problem-solving skills. These skills can help improve the quality of the chatbot, however, one thing to keep in mind is that while LLMs have impressive general knowledge they lack context-specific knowledge. For example, an LLM cannot know a company-specific question like: *Does the software have ...?*, when the answer is not provided in the training data. It might be able to 'reason' to get to the answer in a zero-shot setting, but in this setting, the answer is not entirely reliable (Brown et al., 2020). To improve the context-specific knowledge of the LLM, different state-of-the-art NLP methods can be used in combination with the LLM, like few-shot learning (FSL), fine-tuning, and retrieval-augmented generation (RAG). These will be covered in detail in section 1.4.

1.1.2 Some challenges of chatbots and LLMs

While customer support chatbots enhanced by Large Language Models (LLMs) offer significant improvements in service efficiency and customer interaction, they also introduce specific challenges that can impact their effectiveness. These include issues with response accuracy, data security, and maintaining the quality of interactions, which are crucial for

customer satisfaction and trust (Ngai et al., 2021).

Even though LLMs show promising results they still come with some challenges that need to be taken into consideration. LLMs are successful on questions that probe knowledge which appears widely in their pre-training (Kandpal et al., 2023), however, questions that are sparse in the training data are less successful and can result in hallucinations (Wang et al., 2023a). Attacks like prompt injection (Liu et al., 2023) could possibly expose unwanted company data and thus should be addressed when using an LLM. The fairness and biases of an LLM resulting from the training data can also become a problem when these are encountered.

For customer support bots, it is important to recognize that the performance of these chatbots can significantly affect a brand's image negatively if not managed properly (Suthar, 2018). Challenges such as ensuring the reliability of chatbot responses (Ngai et al., 2021), particularly in complex or sensitive contexts (Bavaresco et al., 2020), having a good hand-over module (Braggaar et al., 2023), and maintaining the security of sensitive data are critical. Furthermore, we have to take the structure and quality of the Knowledge Base (KB) used in fine-tuning and training these chatbots into consideration (Ngai et al., 2021). The data for customer support is often organized in Q&A pairs or as detailed conversation logs and plays a crucial role in the effectiveness of these systems. In chapter 2 we will delve deeper into these challenges, examining both the potentials and limitations of current technologies and proposing directions for future research to enhance the capabilities of chatbots in customer support environments.

1.2 Research gap

Previous studies in the field of chatbots have focused on different architectures proposed by the authors, with multiple different purposes. Many of these architectures had a structure with something similar to an NLG (Natural Language Generation) and NLU (Natural Language Understanding) unit. Using LLMs as part of either of these units has not been widely investigated yet, especially considering recent advancements in LLM capabilities. Developing an LLM-based chatbot from scratch by training a Transformer model (Roller et al., 2020) is an area that has been studied before. Other areas focused more on specific state-of-the-art NLP methods in combination with pre-trained LLMs. Three of the most promising Natural Language Processing and (NLP) methods for enhancing the context-specific abilities of LLMs, as identified in recent studies, include few-shot learning (Bae et al., 2022), fine-tuning (Liu et al., 2021), and Retrieval-Augmented Generation (RAG) (Lewis et al., 2020). These methods will be discussed in detail in section 1.4. Each method can significantly enhance the context-specific capabilities of LLMs, making them highly suitable for application in customer support bots. Few-shot learning, fine-tuning, and RAG have made notable contributions to the field of chatbot systems. However, there remains a substantial research gap in a detailed comparison of these methods. Evaluating the performance of each method can help identify the strengths and weaknesses, which in turn can help determine the most suitable method for various scenarios. Addressing this

gap can help refine the context-specific performance of LLMs and could influence architectural designs regarding customer support bots. This can also be an important aspect to companies, for which customer support bots must perform as intended.

1.3 Research question

To address this gap the following research question with sub-questions has been formulated:

RQ 1: How do few-shot learning, RAG, and fine-tuning compare in enhancing the relevance, accuracy, completeness, conciseness, empathy, and friendliness of responses generated by Large Language Models in customer support bots?

- SQ 1.1: How do few-shot learning, RAG, and fine-tuning differ in their ability to generate responses that are relevant and accurate to customer questions in a support chatbot?
- SQ 1.2: Which method among few-shot learning, RAG, and fine-tuning most effectively balances completeness and conciseness in responses generated by a customer support chatbot?
- SQ 1.3: In what ways do few-shot learning, RAG, and fine-tuning impact the perceived empathy and friendliness of a customer support chatbot's responses?

Although the initial research question addresses the evaluation of multiple generation methods using LLMs on different metrics, a gap still exists in creating a comprehensive chatbot that builds on these findings. Therefore, we define a second main research question focused on establishing a complete architecture:

There remains a gap in creating a full-fledged chatbot that builds on the findings of the first research question. We, therefore, define a second main research question regarding a complete architecture:

RQ 2: How does the implementation of the best-performing method with template refinements, based on the findings from RQ 1, in combination with a handover module, affect the relevance, accuracy, completeness, conciseness, empathy, and friendliness of responses in customer support chatbots?

Assessing the effectiveness of these methods both independently and the best-performing method as part of an integrated architecture is crucial for developing more efficient and user-friendly chatbots.

1.4 Current study

To address the questions this study will primarily focus on implementing the three methods mentioned earlier. These methods are few-shot learning in combination with a pre-trained LLM (Brown et al., 2020), fine-tuning the LLM on the knowledge base (Bae et al., 2022) and using Retrieval-Augmented Generation (RAG) with the help of the Python framework LangChain and apply this to an LLM (Pandya and Holia, 2023). We also implement a zero-shot model that is just a standard LLM, to be used as the baseline during evaluation. Each of these methods will be evaluated using automatic and human evaluation metrics to find out what methods work best. More details about the evaluation can be found in subsection 1.4.4. The results can also suggest that one method might work well in one scenario, while another works better in the other. In this case, different methods can be combined into a fully functional architecture. Each of the methods explored in this study are described below.

1.4.1 Few-shot learning

Few-shot Learning (FSL) is a type of machine learning problem (specified by experience 'E', task 'T', and performance 'P'), where E contains only a limited number of examples with supervised information for the target class as well as prior knowledge about other classes (Wang et al., 2020). T is to distinguish the target class from the other classes using E. So FSL is designed to enable models to learn new tasks or recognize new objects with very little data. Traditional machine learning and deep learning models often require vast amounts of labeled data to achieve high performance. This requirement poses a challenge in situations where collecting or labeling large datasets is impractical, expensive, or time-consuming, as is the case for a customer support bot, where the amount of (labeled) data is limited. Few-shot learning seeks to overcome this limitation by using prior knowledge the model has gained during its initial training phase, allowing it to adapt to new tasks with only a limited number of examples (Brown et al., 2020). The idea of few-shot learning is the concept of "learning to learn", so a few-shot model is able to distinguish between inputs of different classes. If a dataset with pictures of animals, labeled with the animal name, contains five types of animals, then the amount of different classes of animals in the dataset would be five.

There are different types of few-shot learning namely: zero-shot learning, one-shot learning, and few-shot learning. The 'zero', 'one', or 'few' refers to the number of examples that are shown for each of the different classes in the dataset. In this study a pre-trained LLM will be used as the main model, then few-shot learning using a custom dataset will be applied to this model to let it function as a customer support bot.

1.4.2 Fine-tuning

Fine-tuning an LLM is the process of taking a pre-trained model and continuing the training phase on a smaller, task-specific dataset. This additional training phase allows the

model to adjust its weights and biases, which is the fine-tuning of its parameters (Brown et al., 2020), to improve upon the requirements of the specific task or domain, in this case, customer support. So the overall goal of fine-tuning is to maintain the model's generalized language abilities while optimizing its performance for particular requirements. The fine-tuning phase can take a significant amount of computing resources and training time, especially when using a larger LLM, because of the many parameters that have to be tuned. Using bigger LLMs can improve quality, because of the greater amount of parameters it has, however, this is not strictly the case (Roller et al., 2020). A bigger model can also make the fine-tuning process even more resource-intensive. This study will focus on a pre-trained LLM and tune it using the available dataset.

1.4.3 Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG), is a method of combining an LLM with contextual information to generate more accurate and reliable answers (Chen et al., 2024). The contextual information can be a dataset that contains the information that the LLM is supposed to know, as a task-specific knowledge base (KB). RAG can be used to facilitate the development and deployment of applications that leverage Large Language Models (LLMs). LangChain is a framework that can be used to implement RAG applications, it also makes the many complexities associated with integrating language models into applications a lot more accessible (Topsakal and Akinci, 2023). This framework makes use of chains so that multiple LLMs can be used in a chain to perform different tasks. This study will use LangChain along with a task-specific KB, to extract the relevant information from the KB based on a customer query. This context information can then be used in combination with an LLM to create an answer based on the context information. The advantage of using LangChain is that the KB can actively be updated and this new information can immediately be used again without re-training or re-learning anything.

1.4.4 Evaluation

Evaluating the performance of chatbot systems is an intricate process that involves several automated and human-based assessment methods. The objective is to ensure that the responses generated by the chatbot are not only contextually relevant, but also accurate, and satisfactory from the user's perspective. A multi-layered evaluation framework is used to achieve this, incorporating human judgment and different automatic evaluation metrics. The different evaluation methods are covered in more detail in subsection 2.1.4.

1.4.4.1 Human evaluation

Humans are used on two different occasions. On the first occasion, each of the three methods will generate a response to a pre-crafted question. These responses are then presented to human evaluators, who assess the responses based on different criteria such as accuracy,

relevance, completeness, conciseness, empathy, and friendliness on a scale of one to five. For this, a survey will be used that contains some questions about each of the criteria.

The second occasion is when we choose the best-performing method and use it in combination with a handover module and template enhancements. We will use the pre-crafted scenarios in this case, which were based on the questions from the first evaluation. It is also different compared to the first evaluation because the participants will now be able to actively chat with the chatbot itself. A survey will be used to capture the quality of the user experience and chatbot performance during the interaction. The questions will again be about the criteria accuracy, relevance, completeness, conciseness, empathy, and friendliness, however, on this occasion, participants evaluate the overall chatbot interactions instead of pre-made conversations, again on a scale of one to five.

1.4.4.2 Automatic evaluation methods

Automatic evaluation metrics such as BERTScore (Zhang et al., 2019), BLEU (Papineni et al., 2002), DeepEval (ConfidentAI, 2023), METEOR (Banerjee and Lavie, 2005), ROUGE-L (Lin, 2004), and response length are crucial in the initial stages of assessment. BERTScore, on the other hand, leverages contextual embeddings from BERT to assess the semantic similarity between the generated and reference texts. The BLEU score measures the precision of n-grams in a chatbot's response by comparing it to a set of reference responses. Both will be used as a potential relevance metric along with DeepEval. DeepEval is a more nuanced evaluation framework specifically designed for LLMs containing multiple evaluation metrics. This study will use these different metrics to track the performance of the generated responses. METEOR is like BLEU but accounts for synonyms, stemming, and paraphrasing, making it more effective in capturing key information. Similarly, ROUGE-L focuses on recall by measuring the overlap of phrases and sequences. Both METEOR and ROUGE-L are used for evaluating the completeness of generated responses. These metrics, by providing quantitative feedback, can help improve response generation algorithms. Response length measures the number of words in a response which could indicate the conciseness of a generated response.

While automated metrics can provide valuable evaluations, they might fall short in capturing the subtleties of human language, and the nuances of satisfactory user experiences. Liu et al. (2016) also showed that BLEU is not suitable for evaluating chatbots as it has low correlations with human judgment. Therefore, human evaluation still serves as an important piece of the overall evaluation process.

Chapter 2

Related work

Customer support and chatbots have received considerable attention in academic and industrial research. In this section, we will explore the evolution and impact of chatbots, in general, but also within the domain of customer service. In section 2.1, we provide a comprehensive overview of chatbot technology, covering key aspects in the realm of chatbots.

In section 2.2, we will delve deeper into the perspective of the customer, by reviewing studies that shed light on the effectiveness, challenges, user experiences, and user perceptions of chatbot implementations in customer support. These studies offer insights into what makes a chatbot successful from a user's standpoint, as well as methods to improve customer satisfaction and expectations.

2.1 Chatbots

Chatbots can be categorized into two types: task-oriented chatbots and chat-oriented chatbots. Task-oriented chatbots are designed to handle specific tasks, such as booking tickets, scheduling appointments, or customer support tasks (Li et al., 2019). Chat-oriented chatbots aim to simulate human-like interactions and are designed to engage users in conversation, while not being limited to specific tasks (Mitsuda et al., 2022). This is an important distinction to make as the architecture and evaluation methods can differ between the two. This section begins with a concise history of chatbots in subsection 2.1.1, looking at the way chatbots have been made over the years and what changed compared to the current-day chatbots. In subsection 2.1.2, we then proceed to discuss modern-day methods and technologies that are used in the domain. This will include an analysis of various strategies employed to make chatbots more effective, such as natural language processing, machine learning techniques, and user feedback loops. After this, we will take a look at architectures and frameworks that are proposed in related literature in subsection 2.1.3. Different evaluation methods from past literature are discussed in subsection 2.1.4, and we will finish by discussing some challenges involved with the creation of chatbots in subsection 2.1.5.

2.1.1 History

The first well-regarded chatbot, ELIZA ([Weizenbaum, 1966](#)), was developed to simulate a Rogerian psychotherapist using sets of rules and keywords. With these rules, it could either respond to the user with a hard-coded response or transform a user utterance into a viable response. Another early chatbot was called Parry ([Colby et al., 1971](#)), and behaved like a paranoid schizophrenic patient. These types of rule-based and pattern-matching chatbots have been the primary way of creating a chatbot for a long time, and are still used to this day ([Webb, 2000](#); [Banchs and Li, 2012](#); [Al-Ajmi and Al-Twairesh, 2021](#)). For these types of chatbots, it is generally the case that the more extensive the database of rules is, the more capable a chatbot responds to user utterances ([Adamopoulou and Moussiades, 2020](#)). Over the years, more and more complex rules and scripted responses ([Braggaar et al., 2023](#)) were used, to improve the performance of these bots. There are however some downsides to these rule-based chatbots. Because responses are automated, and repeated, they might not feel original or 'human' ([Adamopoulou and Moussiades, 2020](#)). Grammatically or syntactically complex queries also prove difficult for rule-based chatbots ([Adamopoulou and Moussiades, 2020](#)).

Following the era of rule-based systems, research in chatbot development has shifted more towards machine learning and neural network-based methodologies. This period saw the introduction of more sophisticated models capable of understanding and generating natural language more effectively. The development of IBM Watson in 2010 was a landmark in chatbot research ([Ferrucci et al., 2010](#)), showcasing the potential of Artificial Intelligence (AI) in processing natural language.

The introduction of Large Language Models (LLMs) like OpenAI's GPT series and Google's BERT and T5 in the late 2010s and early 2020s revolutionized chatbot technology ([Gokul, 2023](#); [Yao et al., 2024](#)). These models show unprecedented capabilities in many areas of Natural Language (NL). This resulted in a new wave of chatbots that are not only capable of engaging in more complex conversations but also able to adapt to a wide range of tasks without extensive reprogramming.

2.1.2 Core methods and technologies

This section will discuss the more recent approaches in the field of chatbots and how these work. Over the years, more advanced Natural Language Processing (NLP) techniques have been progressively incorporated into chatbots, resulting in enhanced capabilities to analyze user utterances and craft coherent, contextually relevant responses. Within chatbots, two components stand out for their roles in understanding and responding to user inputs: the Natural Language Understanding (NLU) unit and the Natural Language Generation (NLG) unit ([Bavaresco et al., 2020](#)).

The NLU unit plays an important role in identifying intent and extracting entities from the user utterance ([Weld et al., 2022](#)). Effectively identifying user intents and entities using NLU is important for understanding the user utterance and thus for creating a good chatbot ([Avishka et al., 2021](#)). Once the user's intent and relevant entities have been identified,

the NLG unit takes over the task of generating a fitting response. Using the processed intents and entities from the NLU unit a response needs to be constructed that is not only grammatically correct but also contextually appropriate, to ensure a smooth and natural interaction. The NLG unit also has a critical role in enhancing the quality of a chatbot (Perera and Nand, 2017), the study also showed the capabilities and ways of utilizing NLG. It is however important to note that in recent developments within chatbot technology, the traditional distinction between Natural Language Understanding (NLU) and Natural Language Generation (NLG) units is becoming less pronounced. This is especially evident in decoder-only models like GPT, where a single unified architecture handles both the understanding and generation of language.

These components are part of a larger architecture that analyzes, 'understands', and generates human language in a way that is both meaningful and useful given the context. The shift towards machine learning and neural network-based chatbots represents a significant advancement in making interactions more human-like and responsive to the nuanced needs of users. As indicated by Weld et al. (2022), the integration of NLU and NLG into chatbot architectures can help chatbots towards more intelligent and adaptive systems capable of engaging in complex dialogues.

In the section below we cover different NLP methods used for processing text information. We begin with earlier types of NLP models, including Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) networks, and Gated Recurrent Units (GRU), progressing towards the more recent advancements with Large Language Models (LLMs).

2.1.2.1 RNNs, LSTMs and GRUs

Recurrent Neural Networks (RNNs) are a class of artificial neural networks designed to recognize patterns in sequences of data such as text (Elman, 1990). Unlike traditional feed-forward neural networks (Rosenblatt, 1958), where the flow of information is unidirectional from input to output, RNNs have connections that form directed cycles. This architecture allows information to persist within the network, making RNNs well-suited for processing sequences of inputs. RNNs contain some memory of previous inputs in the sequence, because of the hidden state that is used. The hidden state 'captures' the information about what has been processed so far. This is done by updating the hidden state at every step of the sequence, using the previous hidden state along with the current input.

Long Short-Term Memory (LSTM) networks are an advanced type of RNN architecture designed to address the limitations of traditional RNNs (Hochreiter and Schmidhuber, 1997), particularly their difficulty in learning long-range dependencies within sequences. LSTMs are specifically engineered to avoid the vanishing gradient problem (Hochreiter, 1998), allowing them to remember information for long periods, making them highly effective for a wide range of sequential data processing tasks. The fundamental part of the LSTM is the memory cell which can store information over long periods. Each of these cells also contains mechanisms that allow it to retain or discard information, making LSTMs capable of learning what to keep and what to forget over long sequences. The

LSTM has three gates: the input, forget, and output gates. The input gate determines which values from the input should be used to update the memory state. The forget gate decides what information should be thrown away from the block's state. The output gate determines what part of the current state should be output to the next layer of the network or used to compute the prediction.

Gated Recurrent Units (GRUs) are a type of Recurrent Neural Network (RNN) architecture ([Cho et al., 2014](#)). Just like the LSTM the GRU is designed to solve the vanishing gradient problem associated with traditional RNNs. GRUs simplify the LSTM model while retaining its ability to capture long-term dependencies in sequence data. Instead of three gates like the LSTM, GRUs only have two gates: the update and reset gate. The update gate controls the extent to which the new state is a blend of the previous state and the candidate state. It decides how much of the past information needs to be passed along to future steps, this helps the model in learning long-term dependencies. The reset gate determines how much of the past information to forget. It allows the model to decide if and how to ignore the previous state's information when generating the new state. This simplification can reduce the computational complexity and the number of parameters that need to be learned, potentially speeding up training and inference.

[Mangal et al. \(2019\)](#) compared an RNN, LSTM and GRU for script generation, trained on the same data. In Table 2.1 we have shown the first five lines of an example, taking JON as input, provided by the authors.

Model	Generated text
RNN	JON: JO JO JO JO
LSTM	JON: Good. They will stand what was sweet desor I'm sorry. ARYA would also still have me fear the word flast? BRAN? Hodor hapes the moment then is in a larrer brothel. OLENNA: With a wrong little chareea.
GRU	JON: I didn't work to discuss down the Prince Dorne, she nods. MARWYN: The Lannisters have wed it in the skolloors. CUT TO: OLDTOWN - CITADEL TOILDER: You're a good man. YARA: I think we can win fighting for me. That's what I want. We are leaving your name.

Table 2.1: *The first five lines of the generated examples, taking JON as starting input for RNNs, LSTMs, and GRUs, from [Mangal et al. \(2019\)](#)*

2.1.2.2 Transformer-based LLMs

Large Language Models (LLMs), based mostly on the Transformer architecture ([Vaswani et al., 2017](#)), have revolutionized the field of NLP by demonstrating remarkable capabilities in a wide range of linguistic tasks, from text generation and translation to question answering and sentiment analysis. The Transformer architecture relies on self-attention mechanisms and positional encodings to process words in relation to all other words in a sentence. This allows LLMs to capture the context and process text more effectively than previous models, like RNNs or LSTMs, which process text sequentially. The attention mechanism in LLMs enables the model to weigh the importance of different words within a sentence or document for a given task. This is crucial for understanding the nuanced meanings of words in different contexts. An LLM is trained unsupervised on vast amounts of text data to gain a general understanding of language. This training allows the model to be fine-tuned for more specific tasks, such as question-answering. In addition to fine-tuning, LLMs can be utilized in zero-shot and few-shot settings, combined with in-context learning ([Brown et al., 2020](#)).

Chatbots based on LLMs like ChatGPT resulted in a huge public appeal with regards to multi-purpose conversational agents ([Topsakal and Akinci, 2023](#)). ChatGPT based on GPT-3.5 has shown great performance in multiple domains as well as in higher-order and critical thinking ([Lo, 2023](#)). It also showed to be capable of complex problem solving by being able to play multiple roles in verbal games like twenty questions ([Noever and McKee, 2023](#)). These results highlight the promise of LLMs, especially given that this field of research is still quite new. Considering that newer models such as GPT-4 significantly outperform earlier versions of ChatGPT based on GPT-3.5, and even perform remarkably well in many different human exams ([OpenAI, 2023](#)), it is obvious that these models have great potential.

2.1.3 Architectures and frameworks

This section will cover different proposed architectures from literature, discussing the different methods, implementations, and structures used.

2.1.3.1 Retrieval- and rule-based architectures

A few studies still focus on retrieval and rule-based chatbots for their architecture or use some sort of hybrid system for this.

In [Khanna et al. \(2015\)](#) the authors describe an architecture for rule-based chatbots called SARANG Bot. It uses the KB as the brain of the chatbot, which consists of suitable replies, given possible questions. The KB also does the ambiguity handling, data handling and error handling of the system. The study from [Reshma and Balakrishnan \(2018\)](#) also proposed a framework for architectures that rely on rule-based systems. The framework is based on the ALICE chatbot framework and contains a KB engine and a chat engine. The KB consists of text files and Artificial Intelligence Markup Language (AIML) files. The

paper discusses how to integrate big data into a chatbot framework for the KB engine along with an AIML knowledge base for the chatbot engine. The proposed architecture solves the limitation of static queries, with the use of the KB engine that analyses the big dataset.

2.1.3.2 Hybrid architectures

[Al-Ajmi and Al-Twairesh \(2021\)](#) proposed a hybrid dialogue system that could be used for Arabic flight booking. It is a hybrid system that uses an NLU component, a dialogue manager (DM), and an NLG component. The Wit.ai framework was used as the NLU component, to create and train entities. Additionally, when a new message was received it was added as a new training example for the NLU component. For the DM, a telegram bot was used. The different information slots were filled if the info was available and requested for when the information wasn't given. The NLG component was a rule-based system and thus hard-coded using the slot information. The results show that the chatbot was effective and easy to use as participants gained experience over time. It also became smarter over time due to its self-feeding property.

Another hybrid system was proposed in [Ngai et al. \(2021\)](#). In this paper, the authors propose a new design for a knowledge-based conversational agent, specifically for customer support. It contains a knowledge module, a dialogue module, a handover module, and an adapter. The dialogue module is responsible for handling conversations with the customer, using an NLP engine. The NLP engine performs a variety of different tasks like named entity recognition, syntax parsing, and intent classification. The knowledge module contains the KB, where question and answer (Q&A) pairs are stored. An answer gets generated in a retrieval and template-based manner. New Q&A pairs are classified and organized, after this a human gets notified to review if the new pair should be added to the KB. The handover module is responsible for switching between human and chatbot customer service. So when the chatbot cannot answer the question the handover module will route the customer to customer service. The adapter lets the chatbot connect to many different platforms. The prototype showed satisfactory results, a sign that the system is effective.

The design of the empathic social chatbot XiaoIce is also a hybrid system described in [Zhou et al. \(2020\)](#). XiaoIce was developed on an empathic computing framework, which is a framework designed to recognize human feelings and states as well as understand user intents. It consists of a human experience layer, a conversation layer, and the KB. The user experience layer is for connecting to different chatting platforms and has some components for processing inputs (text, speech, images) and responses. The conversation layer has a dialogue manager that keeps track of the dialogue state and uses the dialogue policy to generate an answer. Three answers are generated in three different ways: a Retrieval-Based Generator using paired data, a Retrieval-Based Generator using unpaired data, and a neural response generator based on a GRU-RNN model. The three answers get ranked by a Response Candidate Ranker which is based on a boosted tree ranker from ([Wu et al.,](#)

(2010). The emphatic computing module is to identify the contents and empathic aspects of an input. The KB consists of human profiles as well as some databases containing text-to-text pairs or text-image pairs. The results show good performance on both Emotional Quotient (EQ) and Intelligence Quotient (IQ).

Nuruzzaman and Hussain (2020) propose the IntelliBot, which has a hybrid architecture and is a domain-specific bot for the insurance industry. The framework uses an NLU unit for understanding the input, it then uses a strategy selector to choose between one of four strategies: template-based, knowledge-based, internet-retrieved, and generative-based (RNN model). The response analyzer unit eliminates incoherent or questionable responses, or scores valid answers based on relevance. This is done by using the Jaccard similarity between question and response with a predefined threshold. The generative-based strategy was by far the most used strategy from the bot. Compared to other chatbots the results from IntelliBot are much better in all basic conversations as well as domain-specific questions.

As in Nuruzzaman and Hussain (2020), the study from Cui et al. (2017) also used a hybrid architecture with multiple strategies. The authors in this study proposed a chatbot called SuperAgent for e-commerce websites. It uses a web crawler to search the webpage for product information, Q&A pairs and customer reviews, which are sent to the KB. When a query is received there are four different engines that process the query in parallel. These are the FactQA which answers questions regarding facts about the product using the KB, the FAQ search for Q&A pairs which answers questions using the KB, the opinion-oriented text Q&A to answer opinion-oriented questions using the customer reviews in the KB, lastly there is the chit-chat engine designed to reply to greeting queries and queries that cannot be answered. If one of the answers from the first three models has high confidence then the chatbot will return that answer, otherwise it will default to the chit-chat engine. SuperAgent shows improved user experience for online shopping, mainly due to the convenience of information acquisition especially on pages with lots of user-generated content.

An architecture to propose a Dutch financial chatbot was done in the study by Wetstein (2017). It used a Q&A dataset as well as two models, a vector space (VS) and a machine learning (ML) model, for the chatbot. Both followed roughly the same steps, however, the ML model was trained on a question-to-question dataset. In the VS model, the final answer was chosen based on the highest similarity between user utterance and the dataset question. In the ML model, the answer with the highest confidence from the model was chosen as the final answer. The study concluded that the overall answer quality was not very high, likely due to the small dataset.

Another architecture in the realm of chatbots was proposed by Adamopoulou and Moussiades (2020). The goal of this architecture was to be general and to contain all crucial elements according to the authors. The architecture works as follows, the user message gets analyzed for cognitive services like sentiment analysis, translator, or spell checker as well as intent classification and entity recognition. The dialog manager handles ambiguity, makes a call to the KB for information, and does the error handling. All this information

gets fed to the generation component to form a response. This is a general architecture and each component can use different methods or models for it to work.

2.1.3.3 RNN-, LSTM-, and GRU-based architectures

The chatbot, as described by [Bansal et al. \(2021\)](#), was designed for programmers to ask questions about subroutines. The architecture consisted of the following components: a dialogue strategy management system, a KB, and an NLU/NLG component. The dialogue strategy management makes decisions on how to respond and how to extract information to form a response. The KB consisted of the source code of subroutines, which included over 2.1M methods. The NLU/NLG component used RNNs with word embedding vector spaces to implement the encoder and decoder, where the encoder implements the NLU and the decoder the NLG. The results showed that the chatbot was able to reliably answer questions about subroutines. However, the authors explicitly stated that the Q&A system was not intended to be used on its own, but would serve as a component of a larger hypothetical chatbot.

[Acharya et al. \(2020\)](#) uses an LSTM to generate response candidates. In the architecture, the LSTM is used for intent classification and entity recognition to generate response candidates, based on the responses from the KB. The response selector will then use the context and the candidates to select the most appropriate response. It also uses various APIs to communicate to avoid installing dedicated applications. In [Zhou et al. \(2018\)](#) the chatbot is based on a general seq-2-seq using GRUs. The model is trained on a large corpus of conversations that was first annotated with emotion using a bi-directional LSTM. The goal of the Emotional Chatting Machine (ECM) was to generate a response for an incoming query that is coherent with the emotion category, using the emotion embedding, the internal memory, and the external memory. The results show that the ECM was able to not only generate responses in content but also in the appropriate emotion.

2.1.3.4 Transformer-based architectures

[Vlasov et al. \(2019\)](#) propose a Transformer Embedding Dialogue (TED) architecture to use for developing architectures for dialogue tasks. The goal of the paper is to show that multi-turn conversations are better modeled by a transformer architecture. In each dialogue turn the input of the transformer consists of the user input, previous inputs, and system actions. The relevance of the previous dialogue turns is learned and calculated for each turn, using a loss function for the similarity. The TED policy was compared to a specialized Recurrent Embedding Dialogue Policy (REDP) proposed in [Vlasov et al. \(2018\)](#) and an LSTM policy. The REDP is an LSTM-based model designed to handle long-range history dependencies. The LSTM policy is identical to the TED policy, however, the transformer model was replaced by an LSTM. The results show that the TED policy performs on par with the REDP and and better than the LSTM policy while being faster, simpler, and more general. They also used a modular approach which received better F1 scores compared to the original end-to-end policy. This is likely due to the extra supervision it receives.

However, while the amount of mistakes decreased, the kind of mistakes it makes is still the same.

EVA was proposed by Zhou et al. (2021) and is a transformer-based Chinese dialogue model. They used a data sampling technique for efficient pre-training. The dataset used for the model consisted of 1.4B context-response pairs. The model proved more effective in sensibleness, specificity, and the sensibleness and specificity average (SSA) metric by Adiwardana et al. (2020), compared to previous models. PlugMed is another task-specific chatbot proposed by Dou et al. (2023) and used in context learning. It consists of the prompt generation module and the response ranking module. The LLM generates a few prompts based on similar full-context examples and recent utterance examples from the dialogue history. The ranking module uses a small language model (SLM) to evaluate the responses of the LLM. The best response gets picked as the ultimate output. The results show that the prompts result in greater inclusion of medical terms and yield more accurate dialogue intents.

Generative Multimodal Entity Linking (GEMEL) is a framework proposed by Shi et al. (2023), to directly generate target entity names. The framework receives a multi-modal input and then returns the entity as output with the model's and vision encoder's parameters frozen. The authors use in-context learning on the LLM to let it better comprehend the MEL task, by constructing a prompt template with some training examples. Such a template would contain image and textual information of the mention, a hand-crafted question containing the mention as well as the entity name as the answer. The framework outperforms state-of-the-art methods on two MEL datasets.

2.1.3.5 ParlAI framework

In Miller et al. (2017) the authors provided an open-source software platform for dialogue research called ParlAI, to share test, and train dialog models. Its integration with Amazon Mechanical Turk supports efficient data collection, human evaluation, and the application of online and reinforcement learning techniques. It could support over 20 tasks on initial release and provides researchers with a comprehensive toolkit for developing advanced dialog models, which can help with comparing and improving upon existing architectures. The goal of this platform is to enable the systematic development and evaluation of dialog agents to further push the field of dialogue systems.

This framework was used by Roller et al. (2020) to fine-tune three different-sized LLMs, which consisted of 90M, 2.7B, and 9.4B parameters. They used three types of architectures: retrieval, generative, and retrieve-and-refine models, which were all based on the Transformer model. First, they pre-trained their three models using the Fairseq (Ott et al., 2019) toolkit and the pushshift.io Reddit discussions data. For the fine-tuning process, the data consisted of multiple different datasets. They showed that their models had improved performance in engagingness and humanness by human evaluation, however, they still make mistakes sometimes. It was also likely that these issues became more apparent, the longer the conversations went on, showing that these transformer models still have a lim-

ited dialogue history. They also found that the 2.7B parameter model outperformed the 90M parameter model by a lot, however, the 9.4B parameter model didn't outperform the 2.7B model even though the model has a lower perplexity. This shows that there are more factors besides perplexity to take into consideration. ParlAI was also used for fine-tuning the mental health dialogue system called Serena, proposed by [Brocki et al. \(2023\)](#). The LLM used in this study was the 2.7B parameter model from [Roller et al. \(2020\)](#). The model was fine-tuned on counseling and psychotherapy sessions, which consisted of 14.3K Q&A pairs. Serena showed promising results for health counseling, even though there are still challenges with it. Serena often hallucinates knowledge about the user, pretending to have detailed knowledge of the user while this might not be the case. Another issue of Serena is that it will always respond with a question, feedback from users showed that this behavior can be seen as annoying or rude.

2.1.3.6 Pre-trained LLMs with fine-tuning

The fine-tuning of a pre-trained LLM is a prominent method in literature. [Kasahara et al. \(2022\)](#) proposed an architecture for persona-based dialogue systems that uses prompt tuning to optimize the desired LLM outputs. The architecture consists of an input with persona information tokens (tokens that embed persona information), utterance tokens, and response tokens. The persona information tokens are embedded while those of the pre-trained LLM are frozen, so only this information is tuned. The results show that, when using the same LLMs, prompt-tuning works better than fine-tuning in different metrics.

In [Liu et al. \(2021\)](#) the emotional support conversation (ESC) framework is proposed as well as the ESConv dataset containing richly annotated conversations. With the ESConv dataset, they want to find out how much state-of-the-art generative models improve using strategy annotation, as well as find out the effectiveness of these models for emotional support. BlenderBot and DiabloGPT were used as the backbone models and then fine-tuned using the ESConv dataset and some different strategies. The results show potential improvements for these models to provide emotional support. The study by [Zheng et al. \(2023\)](#) proceeded on this subject. In this study, the ExTES dataset was created, with the help of ChatGPT. This dataset was used to fine-tune the LLaMa model with different training techniques, to optimize the LLM for emotional support interactions. The Open-AI's API was used to create around 11K conversations in multiple domains of emotional support. To generate responses three methods were used: DiabloGPT with causal language modeling (CLM) training, LLaMa adapter-tuning [Zhang et al. \(2023\)](#), and LLaMa LoRa-tuning ([Hu et al., 2021](#)). CLM is the task of predicting the next token in a sequence of tokens, by only attending to tokens on the left. LLaMa adapter-tuning is a lightweight adoption method that only needs fine-tuning on 1.2M parameters upon the 7B frozen parameters. This method prepends a set of learnable adaption prompts to the transformer blocks at higher levels. LLaMa LoRa-tuning reduces the number of trainable parameters by 10K times. It achieves this by using trainable rank decomposition matrices in each layer of the Transformer architecture. All of the methods were parameter-efficient tuning

methods. They also trained these models on the ESConv dataset in [Liu et al. \(2021\)](#). The results showed that the models performed better while fine-tuned on the ExTES dataset compared to the ESConv dataset. And especially the two LLaMa methods trained on the ExTES dataset showed promising emotional support across many scenarios.

The Action-Based Conversations Dataset (ABCD) is introduced by [Chen et al. \(2021a\)](#), which is a fully labeled dataset containing over 10K conversations and 55 different user intents. They used an LLM and used their dataset and two new dialogue tasks to fine-tune the model. They used the transformer-based LLM BERT in this study, but they would like to experiment further on GPT models in the future. The results showed that the models performed decently in one of the dialogue tasks and quite poor in the other. The paper by [Brown et al. \(2020\)](#) presents such a GPT model. In this case, GPT-3, which is a generative language model with 175B parameters, is built on the Transformer architecture. This architecture enables GPT-3 to effectively process and generate human-like text by considering the relevance of different parts of the input text. GPT-3's extensive pre-training across diverse data sources allows it to perform a broad array of tasks without task-specific tuning, demonstrating remarkable few-shot, one-shot, and zero-shot learning capabilities across various domains.

2.1.3.7 RAG and the LangChain framework

The authors from [Lewis et al. \(2020\)](#) introduce a model that combines the generative capabilities of LLMs with a retrieval mechanism. This retrieval-augmented generation (RAG) approach enables the model to dynamically retrieve relevant information from a large corpus of texts during the generation process. This method not only enhances the model's ability to produce accurate and contextually relevant responses but also offers a way to provide provenance for the model's outputs. Additionally, the approach facilitates the updating of the model's world knowledge by incorporating new information into the retrievable corpus without the need to retrain the model's parameters. The RAG model demonstrates improved performance on a range of knowledge-intensive NLP tasks, showcasing its potential to use the rich and broad knowledge encapsulated in pre-trained LLMs, combined with precise tasks.

Sahaay was proposed by [Pandya and Holia \(2023\)](#) for automating customer support and is powered by the LangChain framework. They used web-scraping techniques to retrieve the data like FAQs, product manuals, forums, etc. This information is chunked into smaller pieces of information, embedded, and then stored in a vector store. Google's Flan-T5 was used as the LLM, this model then used some context from the vector store to generate an answer. They used the Gradio API framework to then display the chatbot for a user to interact with. The results show that the biggest model (XXL) performed better than the smaller model. It also concludes that LangChain can revolutionize the customer-company relationship and help facilitate the fast development of LLM applications. This was also confirmed by the study of [Topsakal and Akinci \(2023\)](#). In the paper by [Chen et al. \(2024\)](#), the authors examine the impact of RAG on various LLMs. It introduces a new benchmark,

the Retrieval-Augmented Generation Benchmark (RGB), to systematically evaluate LLMs across four abilities essential for RAG: noise robustness, negative rejection, information integration, and counterfactual robustness. The evaluation, conducted on six state-of-the-art LLMs, revealed challenges in negative rejection, information integration, and dealing with false information, despite some level of noise robustness being observed. The authors highlight the need for further improvements in LLMs' use of RAG to ensure reliable and accurate responses.

2.1.3.8 Pre-trained LLMs and FSL

The study of [Chada and Natarajan \(2021\)](#) proposed a simple few-shot fine-tuning framework leveraging pre-trained LLMs. For the pre-training, they used the BERT, BART, and T5 models, with all slightly different pre-training frameworks. They fine-tuned the models by concatenating the question, the answer as mask, and the context together, and filled in the real answer for the mask as the target. The results show that these few-shot models outperform the base models by a lot.

Few-shot learning was also applied in the study [Bae et al. \(2022\)](#). The authors propose a data collection system that uses in-context few-shot learning to create role-specific LLMs. The preferred role gets specified and a few dialog examples (few-shot learning) get fed into the 39B parameter-sized LLM from [Kim et al. \(2021\)](#), which generates entire dialogues and these get filtered by humans. Humans labeled the point where the first out-of-bounds (inappropriate answer) occurred in the conversation. The authors then used the utterances before the out-of-bounds as positive examples and the utterance after this point as a negative example. Utterances after the negative example were not taken into consideration anymore. The dialogue model is then trained with supervised learning on the created dataset. By chatting with the system the users can also give extra feedback to the system. To assess the response quality the SSA metric was used. The model shows competitiveness to humans in sensibleness.

The FlexKBQA framework by [Li et al. \(2024\)](#) introduces a novel approach for few-shot knowledge base question answering (KBQA). It utilizes LLMs for generating synthetic data, which trains a lightweight model for efficient KBQA. It employs automated algorithms for sampling diverse programs, such as SPARQL queries, from a knowledge base, converting them into natural language questions using LLMs. FlexKBQA also implements an execution-guided self-training method, enhancing the framework's adaptability. This approach shows substantial improvements in KBQA performance under few-shot and zero-shot settings.

2.1.4 Evaluation methods

Evaluating chatbots as well as Large Language Models (LLMs) is a critical step in understanding their capabilities, limitations, and the contexts in which they can be effectively deployed. The evaluation process typically involves a range of techniques and metrics designed to measure various aspects of model performance, including linguistic abilities,

Automatic Evaluation Metric	Studies
Distinct-n	(Zhou et al., 2020), (Bae et al., 2022), (Kasahara et al., 2022)
FLEX benchmark	(Bragg et al., 2021)
RAG benchmark	(Chen et al., 2024)
BLEU	(Cho et al., 2014), (Kim et al., 2021), (Zhou et al., 2021), (Hardalov et al., 2018), (Li et al., 2015), (Liu et al., 2021)
QuAD-tuned Q-BLEU-1	(Lewis et al., 2020)
Rouge-L	(Dou et al., 2023), (Zhou et al., 2020), (Zhou et al., 2021), (Hardalov et al., 2018), (Lewis et al., 2020), (Liu et al., 2021)
METEOR, CIDEr, SPICE	(Zhou et al., 2021)
Bert-Score	(Dou et al., 2023)
Embedding Average, Greedy Matching Vector Extrema	(Hardalov et al., 2018), (Liu et al., 2016) (Liu et al., 2021)
YNAT, KLUE-STS	(Kim et al., 2021)
Exact Match-score	(Li et al., 2024), (Miller et al., 2017)
GPTjudge	(Lin et al., 2022)
Perplexity	(Zhou et al., 2018), (Liu et al., 2021) (Adiwardana et al., 2020)
Unigram F1-score	(Zhou et al., 2020)
DeepEval	(ConfidentAI, 2023)

Table 2.2: Automatic evaluation methods used in different studies.

knowledge retention, and reasoning, but also human metrics like relevance, humanness, empathy, etc. This part provides a detailed overview of the methods and considerations involved in the evaluation of LLMs used in past studies for both automatic evaluation and human evaluation. Some of the different metrics will be discussed in short.

2.1.4.1 Automatic evaluation

Natural Language Understanding (NLU) and Generation (NLG) metrics evaluate the model's ability to parse user inputs accurately and generate appropriate responses. In Table 2.2, different evaluation metrics can be found, that were used in the papers described above. Distinct-n (Li et al., 2015) is calculated by dividing the distinct number of n-grams by the total number of words. It is a metric used to report the degree of diversity in generated responses. The FLEX benchmark (Bragg et al., 2021) and RAG benchmark (Chen et al., 2024) are specific evaluation methods for FSL and RAG respectively. BLEU-n (Papineni et al., 2002) looks at co-occurrences between n-grams of the golden response and the system's

response. However, (Liu et al., 2016) showed that it is not very suitable for evaluating chatbots, as these evaluations don't correlate well with human judgment. QuAD-tuned Q-BLEU-1 is a variant of BLEU with a higher weight for matching entities. According to Lewis et al. (2020), it has a higher correlation with human judgment for question generation compared to standard metrics. ROUGE-L (Lin, 2004) is an evaluation metric for summarization of texts. It is based on the Longest Common Subsequence (LCS) between the golden response and the system's response. For the LCS, the words occurring in both responses do not have to be adjacent, unlike the BLEU metric. METEOR (Banerjee and Lavie, 2005) was designed to address some of BLEU's weaknesses, by using alignment based on exact token-matching instead of n-grams, making it more effective for semantic relevance and coverage of key information.

BERTScore (Zhang et al., 2019) is an automatic evaluation method for text generation. It uses vector embeddings for both the golden response and the system's response, the cosine similarity is used to compare each word from both sentences to each other and then uses greedy matching to find the highest similarities between the responses. The authors argue that BERTScore works better compared to metrics like BLEU or METEOR because these metrics fail to match paraphrases and fail to capture long-range dependencies, both of which BERTScore tries to solve using contextualized token embeddings. Other embedding-based evaluation methods are embedding average, greedy matching (Rus and Lintean, 2012), and vector extrema (Forgues et al., 2014). GPT-judge, proposed by (Lin et al., 2022), is a GPT-3 6.7B model fine-tuned to classify the truthfulness of an answer with yes or no, another model was fine-tuned to evaluate informativeness. The perplexity metric evaluates how well sequence-2-sequence models predict the test set data, with lower perplexity generally resulting in better scores.

Lastly, we have the DeepEval framework for LLM evaluation (ConfidentAI, 2023). The framework uses LLMs to perform the evaluations and contains multiple evaluation metrics like relevance, bias, faithfulness, and toxicity, but you can also create your own custom metric to perform evaluations.

2.1.4.2 Human evaluation

Human evaluation is often also an important part of testing the output and responses of an LLM. You can use human evaluators to rate many aspects of the chatbot and its responses across a conversation. This involves assessing whether the chatbot's responses make sense within the broader context of the dialogue and remain consistent with previous statements. OpenAI, for example, uses crowd-sourced feedback in ChatGPT by providing the users with two answers and letting them pick which of the two is preferable. The human evaluation metrics are more subjective compared to the automatic metrics, depending on what the subject of the research was and what the authors focused on. Some measures also use different names while having related meanings. The table Table 2.3 shows the different human evaluation metrics. Generally, the accuracy, coherence, relevance, and fluency of responses were measures that studies looked at in some way. Studies regarding emotional

Human Evaluation Metric	Studies
Sensibleness, SSA	(Adiwardana et al., 2020), (Zhou et al., 2020)
Specificity	(Adiwardana et al., 2020), (Lewis et al., 2020), (Zhou et al., 2020)
Coherence, Persona, Style, Safety	(Bae et al., 2022)
Fluency	(Liu et al., 2021), (Bae et al., 2022), (Kasahara et al., 2022)
Accuracy, Completeness, Conciseness	(Bansal et al., 2021)
Role Consistency, Correctness, Necessity, Empathy, Richness	(Dou et al., 2023)
Actuality	(Lewis et al., 2020)
Truth Score	(Lin et al., 2022)
Identification, Comforting, Suggestion, Overall	(Liu et al., 2021)
Quantity, Relation, Manner	(Ngai et al., 2021)
Quality (Relevance)	(Nuruzzaman and Hussain, 2020), (Ngai et al., 2021) (Bansal et al., 2021), (Kasahara et al., 2022)
Performance	(Pandya and Holia, 2023)
ACUTE-Eval, Humanness	(Roller et al., 2020)
Engagingness	(Roller et al., 2020), (Kasahara et al., 2022)
Content, Emotion	(Zhou et al., 2018)

Table 2.3: *Human evaluation methods used in different studies.*

support (Liu et al., 2021) focused more on the helpfulness, empathy, and comforting aspects of the chatbot.

The Sensibleness and Specificity Average (SSA) metric was proposed by Adiwardana et al. (2020). It takes the average of sensibleness (does the response make sense, given the context) and specificity (is the response specific for the given context). The SSA was proposed as a proxy for human likeness, which also can penalize generic responses. They also showed that SSA correlates with the Perplexity metric, arguing that SSA is a good proxy for human judgment on fundamental attributes, such as sensibleness and specificity. How the chatbot acted was also measured in different ways, using for example humanness (Roller et al., 2020), empathy (Dou et al., 2023), personal manner (Ngai et al., 2021), and engagingness (Roller et al., 2020; Kasahara et al., 2022). Most studies with human evaluations used a survey with questions in the domain of their evaluation measures, using some sort of point scale.

2.1.5 Challenges

While leveraging customer support bots and Large Language Models (LLMs) promises significant enhancements in handling customer queries efficiently, it's essential to acknowledge the challenges that could occur when deploying the bot.

LLM-specific challenges address the need for continuous learning and adaptation of these systems to evolving user needs and expectations requires ongoing investment in terms of both time and resources. While the chatbot itself can have some challenges like technical limitations and ethical considerations. Additionally, data privacy and customer trust might become challenges as well. Together, these factors show the complexity of deploying customer support bots and LLMs effectively. These factors also need to be considered to maximize their potential while mitigating the inherent risks and challenges. We will first cover LLM-specific challenges and then address those of chatbots itself.

2.1.5.1 LLM-specific challenges

One of the first problems that comes to mind is NLU and how the chatbot handles a user utterance. Despite advancements in NLP and the development of LLMs, customer support bots still can struggle with the ambiguity and variability of human language. While most architectures described earlier showed promising results, none were perfect yet.

Maintaining a coherent conversation that spans multiple interactions while accurately retaining context from previous interactions is another significant challenge for transformer-based models (Roller et al., 2020). One expects a seamless dialogue where the bot can recall and use past interactions as well as understand the conversation's progression. This means there needs to be some sort of memory retention mechanism and context-management capabilities within the bot's architecture.

While LLMs can be very promising on data that is widely available in the training set, they can also have the problem of not handling long-tail knowledge, that is knowledge about sparse data in the training set, very well (Kandpal et al., 2023). These can also result in hallucinations of the LLM. Hallucinations are system-generated responses that defy universally accepted world knowledge (Wang et al., 2023a). Besides insufficiently broad training data, these hallucinations can also occur from the model not learning its training distribution well enough (Lin et al., 2022) and because of the randomness that comes with every decoding step of LLMs (Lee et al., 2022). The problem with hallucinations is the likelihood of generating inaccurate or false data (Wang et al., 2023a; Gill et al., 2024), resulting in misinforming the user.

Another possible challenge could be fairness. AI-driven customer support bots could be susceptible to biases present in their training data, but also in the specific knowledge base that is used. This, in turn, can lead to unfair or discriminatory responses towards users. Addressing these biases, by for example changing the KB, and ensuring that bots operate responsibly is important for maintaining trust and fairness.

2.1.5.2 Chatbot-specific challenges

The chatbot needs to respond to sensitive and private conversations with the customer accurately and reliably (Ngai et al., 2021), however, the system also needs to handle the user's data in a responsible manner (Bavaresco et al., 2020). From the company's perspective, the KB that is used for chatbots also needs to be well protected. Attacks like prompt injection (Liu et al., 2023) could possibly expose unwanted sensitive company data. These bots often handle sensitive personal information, thus it is necessary to put security measures in place to prevent data breaches and ensure compliance. Besides a secure KB, updating the KB is also important to provide up-to-date information. A well-structured KB can help with this process (Ngai et al., 2021).

Personalizing interactions to meet individual customer preferences is another important area for enhancing the user experience but remains a complex challenge. You could for example risk impersonal interactions with a one-size-fits-all approach which could result in customers feeling alienated or frustrated with the lack of empathy (Liu and Sundar, 2018; Daher et al., 2020), thus adding empathy and personality can be a good thing. However, it is important to note that when customers put too much trust in your system, this can result in eventually misplaced trust (Abercrombie et al., 2023). For example when a bot provides an obviously wrong answer (like with hallucinations) but the user believes it because of the large amount of trust they put in it. These user experiences will be discussed further in section 2.2.

Keeping in mind these challenges, corporations can use chatbots to enhance customer support. Prioritizing user privacy and security and continually improving the architecture can result in a good-performing chatbot. The following section will explore the user's experience with a customer support bot and their expectations in more detail.

2.2 Customer end

The customer experience is the most important aspect of a customer support bot. Customer support bots have been shown to improve user experience (Nguyen, 2019; Misischia et al., 2022) and enhance service quality (Meyer-Waarden et al., 2020). However, it is worth noting that a poorly performing bot can have a negative impact on user engagement and user retention, while also resulting in a negative brand image (Suthar, 2018). This shows that understanding the user is very important to a company when creating such a bot. As such, researching the different aspects of the customer end when it comes to chatbots is crucial in this regard. In this section, we will cover customer experience, customer perception, customer expectations, and customer preferences as the most prominent and important subjects of this. We will go into detail about these subjects, highlighting the different aspects they present found in previous literature.

2.2.1 User experience

The user experience (UX) of a chatbot significantly influences user perceptions of such systems. According to [Trivedi \(2019\)](#) and [Jenneboer et al. \(2022\)](#), the UX of a chatbot encompasses three key dimensions: service quality, system quality, and information quality. We will cover these topics in depth below.

2.2.1.1 Service quality

Past research underscores the importance of service quality in chatbot interactions, characterized by dimensions like assurance, responsiveness, and empathy. [Jenneboer et al. \(2022\)](#) detail these dimensions, noting that their presence can significantly enhance user satisfaction. Research also shows that extrinsic values (focusing on efficiency and economic values) can be enhanced by the usability of chatbots, which directly contributes to service quality ([Chen et al., 2021b](#)). They also concluded that by making customers feel valued, chatbots effectively enhance customer satisfaction, a direct outcome of improved service quality. Making interactions feel more personalized and attentive helps to build trust with the user ([Nicolescu and Tudorache, 2022](#)). [Haugeland et al. \(2022\)](#) also states that introducing humanlike characteristics and ensuring chatbots can entertain, exhibit novelty, provide inspiration, and support social interaction can enhance the overall service quality. However, these anthropomorphic features need to meet users' pre-interaction expectations, as failing to do so can lead to dissatisfaction, particularly in scenarios where users approach chatbots with specific needs, or under stress ([Nicolescu and Tudorache, 2022](#)). [Kvale et al. \(2020\)](#) delves into how effective problem resolution impacts customer satisfaction, which is a key element of service quality. The author showed that the ability of a chatbot to resolve issues efficiently directly influences how users perceive the quality of the service.

2.2.1.2 System quality

System quality is another important factor in the UX of chatbots, involving aspects such as response time, usability, reliability, availability, and adaptability. These elements are crucial for ensuring that chatbots are perceived as effective tools and in turn user satisfaction ([Trivedi, 2019](#); [Meyer-Waarden et al., 2020](#); [Jenneboer et al., 2022](#)). [El Bakkouri et al. \(2022\)](#) and [Haugeland et al. \(2022\)](#) also agreed that the success of chatbots hinges on these core attributes, especially in service-oriented chatbot applications. The functionality of chatbots significantly influences their adoption and usage, with instrumental qualities such as technology adoption and usage as highlighted by [Meyer-Waarden et al. \(2020\)](#) being crucial. [Nicolescu and Tudorache \(2022\)](#) identified the quality of the system, privacy and security considerations, and responsiveness as critical elements that contribute to the overall quality. Moreover, visual aesthetics also play a role ([Meyer-Waarden et al., 2020](#)), supporting the notion that a well-designed interface enhances the usefulness and ease of use, contributing to a positive user experience.

2.2.1.3 Information quality

The accuracy, relevance, and value of the information provided by chatbots are critical components of information quality that directly influence user trust and loyalty, as well as the perceived competence of the chatbot (Meyer-Waarden et al., 2020). In Jenneboer et al. (2022) the authors point out that high-quality information strengthens the user's trust in the chatbot and, by extension, the company, ultimately affecting the customer's loyalty. El Bakkouri et al. (2022) confirms this by stating that accurate, relevant, and valuable information is essential for fostering trust and loyalty among users. This is especially significant in digital customer service platforms where the exchange of information is a primary function of the chatbot.

2.2.2 Customer perception

Understanding how customers perceive chatbots is an important task as well, as these perceptions significantly impact the customer's satisfaction and ongoing engagement. Customer perception is shaped by individual interactions, influenced by prior expectations, word-of-mouth, and personal biases toward technology. Wang et al. (2023c) showed that chat-oriented (social-oriented) chatbots enhance customer satisfaction more than task-oriented chatbots, by increasing trust. This section will cover trust and reliability as well as human-like interactions with the bot.

2.2.2.1 Trust and reliability

Studies indicate that the reliability of chatbots and the trust customers place in them are foundational for their successful adoption. El Bakkouri et al. (2022) highlights how customer trust correlates with the overall system design and the brand's adept use of technology. Trust is not solely built on functionality, but also on how securely and respectfully customer data is handled, underscoring the importance of privacy and security measures, and is a determinant for user satisfaction according to Joshi (2021). Chen et al. (2021b) and Meyer-Waarden et al. (2020) both emphasize that trust is enhanced when chatbots deliver not only functional but also valuable interactions, especially for initial experiences with the bot (Joshi, 2021). Misischia et al. (2022) adds to this that personality traits of chatbots, such as trustworthiness and empathy, contribute significantly to building trust with users. However, in Meyer-Waarden et al. (2020) the authors point out that trust alone does not guarantee increased usage unless the chatbot is perceived as genuinely useful, reflecting the need for effectiveness in addition to reliability. Furthermore, Hjerpbaek et al. (2021) underscores a potential erosion of trust when customers receive misleading information from chatbots, suggesting that accuracy, transparency, and reliability are critical.

2.2.2.2 Human-like interactions

The degree to which chatbots exhibit human-like characteristics significantly influences customer perception. [Nicolescu and Tudorache \(2022\)](#) and [Chen et al. \(2021b\)](#) discuss how anthropomorphic features can enhance the user experience by making chatbots seem more relatable and engaging. These features, however, must be carefully balanced with performance expectations. If chatbots fail to meet high expectations, especially in complex interaction scenarios, dissatisfaction can ensue. Moreover, the findings from [Meyer-Waarden et al. \(2020\)](#) reveal that, while chatbots can mimic certain human interactions, such as empathy and social cues, these features do not always translate to perceived usefulness, particularly in transaction-focused contexts where straightforward functionality is more valued.

2.2.3 Customer preferences and expectations

How customers prefer to interact with a chatbot is a valuable area to look into. Things like personalized support can influence the customer's behavior towards a chatbot. Furthermore, the expectations customers hold for their interactions with chatbots significantly influence their future engagement with this technology. In [Wang et al. \(2023c\)](#) the authors stated that incorporating humor in chatbot responses, personalized responses, and expressing empathy can positively affect consumer perceptions of customer support bots. By understanding these expectations, developers can adjust chatbot design to better meet user needs and enhance overall satisfaction. By tackling personalization, expectations, and customer behavior when it comes to chatbots, we can get a better idea of what the customer wants.

2.2.3.1 Customization and personalization

The ability of chatbots to offer customized and personalized experiences is an important factor in meeting customer expectations. [Chen et al. \(2021b\)](#) demonstrates that personalized interactions significantly enhance satisfaction by meeting specific customer needs and preferences. Similarly, [Kvale et al. \(2020\)](#) suggests that adjusting chatbot responses to the nature of customer queries can improve satisfaction levels, indicating the importance of context-aware support strategies. [Jenneboer et al. \(2022\)](#) points out that the lack of personality and natural responses fall short of meeting user expectations for personalized, human-like interactions. [Nicolescu and Tudorache \(2022\)](#) also notes the importance of chatbots delivering personalized and engaging interactions, which can significantly enhance user satisfaction. Furthermore, [Chen et al. \(2021b\)](#) and [Haugeland et al. \(2022\)](#) discuss how customization not only makes chatbots more engaging but also more effective at meeting individual user needs. The design and deployment of chatbots, therefore, should incorporate elements that allow for extensive customization, to ensure that each interaction feels adapted to the specific user.

2.2.3.2 Expectations of a chatbot

Studies like those by [Nicolescu and Tudorache \(2022\)](#) and [Kvale et al. \(2020\)](#) demonstrate that customers value chatbots that can understand and respond appropriately within the context of their requests. This includes the ability to handle complex problems effectively, as highlighted by [Lei et al. \(2021\)](#), who notes that in settings like tourism, users prefer interactions that resemble human capabilities due to the urgent and complex nature of their questions. [Kuylenstierna and Trägårdh \(2021\)](#) and [El Bakkouri et al. \(2022\)](#) emphasize that younger generations and users involved in more complex service tasks expect chatbots to perform tasks efficiently and effectively, approaching the level of human agents. [Kuylenstierna and Trägårdh \(2021\)](#) further states that younger generations still would rather interact with human service compared to a chatbot. So the ability of chatbots to meet these high expectations directly impacts their adoption and user satisfaction. To achieve this, the user also expects chatbots to function on any basic smartphone with reliable internet connectivity according to [Joshi \(2021\)](#).

2.2.3.3 Impact on customer behavior

In [Jenneboer et al. \(2022\)](#) the authors concluded that adoption of chatbots is lower than expected for consumers, this is often due to the lack of personality and unnatural responses from the chatbot. Human-like chatbots can improve the satisfaction and adoption rates of a chatbot. In [Chen et al. \(2021b\)](#) the authors also note that customers who experience higher satisfaction, due to better service quality, are more likely to recommend the chatbot to others, this is also confirmed by [Haugeland et al. \(2022\)](#). Positive interactions with chatbots are likely to increase user engagement, improve conversion rates, and attract potential buyers by offering a seamless blend of traditional customer service with the convenience of online support ([Misischia et al., 2022](#)). This shows the impact of chatbot quality on customer actions such as advocacy and brand loyalty ([Trivedi, 2019; Jenneboer et al., 2022](#)). Chatbot identity disclosure also impacts customer behavior, affecting purchase decisions and engagement levels as shown by [El Bakkouri et al. \(2022\)](#). This suggests that transparency about chatbot automation can influence how users interact with and perceive the service.

Chapter 3

Methods

This research contains a case study conducted at a Dutch software company known for its specialized inspection software, for various markets including real estate, installation engineering, and roofing. The focus of this study is to evaluate different response generation methods for chatbots to enhance the efficiency of the company's customer support department. By implementing a chatbot as a first line of support, the customer service team can redirect their attention toward resolving more complex questions. Additionally, the chatbots can be used to assist new employees or employees without a customer support background, enabling them to provide support effectively with the help of the chatbot.

Now we will give a detailed outline of the current study. We will first look at the structure of the current research in section 3.1. Then we give an overview of the data available for this research and the methods used for its processing in section 3.2. Following this, the first phase of the thesis will be covered in section 3.3, including each component within it. Finally, we will describe the final chatbot architecture in section 3.4.

3.1 Research overview

The chatbot in this research is primarily task-oriented, aiming to effectively answer user queries. However, it needs to possess the ability to engage in human-like conversations when assisting customers, as highlighted in subsection 2.2.2, meaning it needs to be somewhat chat-oriented as well. This dual focus ensures that the chatbot not only provides accurate information but also maintains a pleasant chatting experience with users. The design of the structure and the evaluation methods in this thesis have been selected to achieve both objectives.

The proposed research, illustrated in Figure 3.1, consists of three parts, each containing a structured sequence of steps. The first part begins with the data collection, then the data processing, which includes data filtering and sanitizing, to ultimately create the Knowledge Base (KB). We then start with the first phase, where a response generator is developed, which consists of the query handler, a processing step, and the generator. Following the creation of the response generator, an initial evaluation is conducted using both

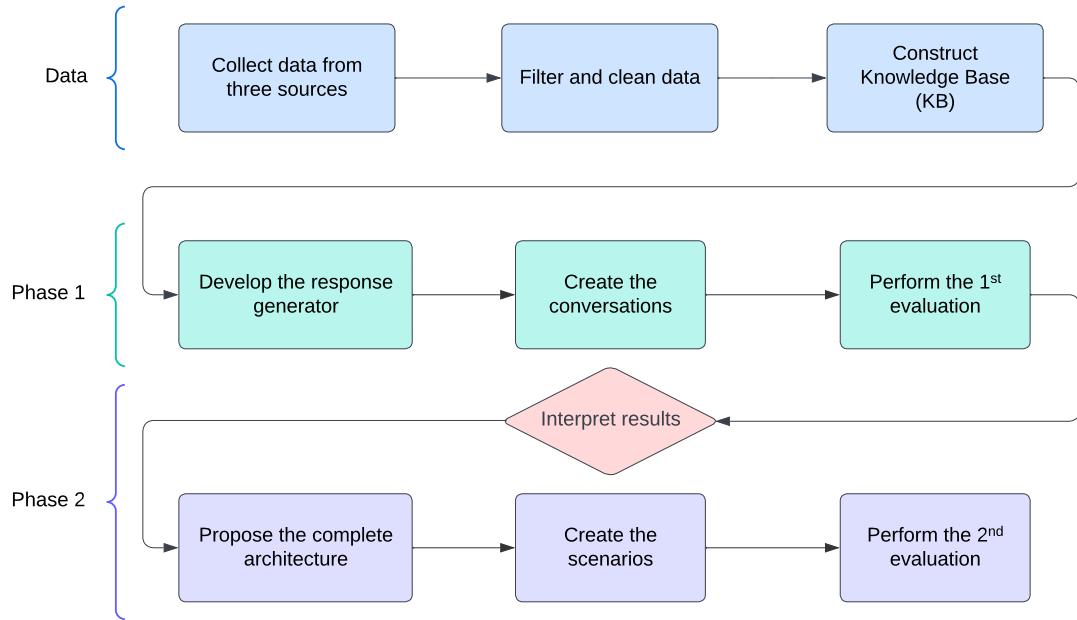


Figure 3.1: *The research overview that is performed in this thesis.*

human and automatic evaluation metrics. Following the first evaluation we start with the second phase, where insights gained from the first evaluation helped the development of a complete chatbot architecture. Finally, a second evaluation was conducted to assess the performance of the enhanced chatbot architecture, to ensure that the improvements did effectively address the initial shortcomings. These three parts will be explained in detail below in section 3.2, section 3.3, and section 3.4 respectively.

3.2 Data process

For the creation of the knowledge base (KB) in this study, we had the following sources of data available:

- **Email conversations** between the customer support department of the company and the customer. These conversations were private.
- **Articles** from the company's help center each containing information on various topics, from instructions on how to use their software to more general questions. These were publicly available.
- **Tickets** from HubSpot with customer's questions, problems, etc. Via HubSpot, the

customer can submit tickets that get resolved within HubSpot by customer support and were again private.

We will discuss the collection (subsection 3.2.1) and cleaning process (subsection 3.2.2) of these data sources below, ultimately resulting in our KB as explained in subsection 3.2.3.

3.2.1 Data collection

The emails were collected from the customer support email inbox, where we started with about 4500 emails. These were exported to a CSV using Outlook's export functionality.

As for the articles, these were collected from the company's help center and then converted to a text file, along with meta-data like the article name and URL. On average each article had about 350 to 375 words, with the longest article being 1774 words long and the shortest 24 words.

The tickets from HubSpot were collected using their API, and processed into the same CSV format as the emails. All future resolved tickets will also be collected using this API because support emails also get redirected to HubSpot meaning that we don't have to look at the email inbox anymore. Examples for each of these data types can be found in Table A.1.

3.2.2 Cleaning up process

The filtering, cleaning, and sanitizing process of the data, which can be found in Figure 3.2 is an important part of this thesis. We started with the selection of the useful and non-useful conversations described in subsubsection 3.2.2.1. Then we moved on to the sanitization of the remaining sources which is described in subsubsection 3.2.2.2. The articles did not need to go through the cleaning up process as they were from the help center on the website, thus they were all useful and did not need sanitization. We provided more details of the contents of the data before and after filtering in Table 3.1.

3.2.2.1 Filtering out conversations

We found that many of the conversations in both the emails and tickets were not useful for this thesis. We performed three steps to remove these conversations:

1. We started by filtering out all non-customer-related conversations like spam emails or adverts from the emails and the tickets. This was done by hand for the 4500 emails and 1230 tickets and resulted in about 2200 email conversations remaining and 1120 ticket conversations. It is important to note that, later on, in the tickets, the unrelated conversations were not present anymore as customer support removed these beforehand.
2. Next, we removed emails and tickets that either contained bug reports or were conversations about dealing with bugs. These sources weren't useful as these reports

Datatype	Before filtering		After filtering		Time-span
	Quantity	Nr. words	Quantity	Nr. words	
Emails	4489	~ 2.2M	839	~ 260K	(03-01-2022) / (08-02-2024)
Articles	279	~ 95K	279	~ 95K	Frequently updated
Tickets	1230	~ 286K	633	~ 138K	(02-10-2023) / (29-07-2024)

Table 3.1: More detailed properties of the data sources before and after filtering.

and conversations were specific to that moment in time and were fixed soon after such a bug or report came in, resulting in about 1720 emails and 880 tickets remaining.

3. Lastly, we filtered out all duplicate conversations, which also included conversations that had new entries in the mailbox when a reply was sent, and conversations with the same ticket number while not being entirely the same. In this case, the latest conversation was kept, which included all the previous replies from that conversation. We used a sequence matcher for this to compare the similarity between two conversations, where we kept the longer conversation if it contained all the information of the shorter conversation. In this way, we ended up with the longest conversations containing all the back-and-forth interactions, while discarding the earlier entries that contained a subset of the interactions. The remaining 839 emails and 633 ticket conversations were ultimately used for the creation of the KB, with each conversation containing on average about 250 to 350 words.

The remaining conversations consisted of questions related to software problems, account information, or questions from users who needed specific assistance or information.

3.2.2.2 Sanitizing conversations

The sanitization of the conversations was another crucial step in the clean-up process. First of all, this was needed to protect the privacy of the customer by removing or anonymizing sensitive data, however, it was also to make sure we ended up with cleaner and more readable data. So all sensitive information like names, addresses, email addresses, phone numbers, passwords, etc. as well as unnecessary information like company footers and standard closing phrases were removed from the conversations. This was done in three separate steps:

1. For the removal of email addresses, footers, and closing phrases, we created a piece of code that looked for these parts using Regex. We removed the footers and headers for the conversations, however, in the case of email addresses and phone numbers we replaced them with a mask.
2. Next, for the identification of names, organization names, and locations we used a sentence tagger by [Schweter and Akbik \(2020\)](#) that could identify these tags and

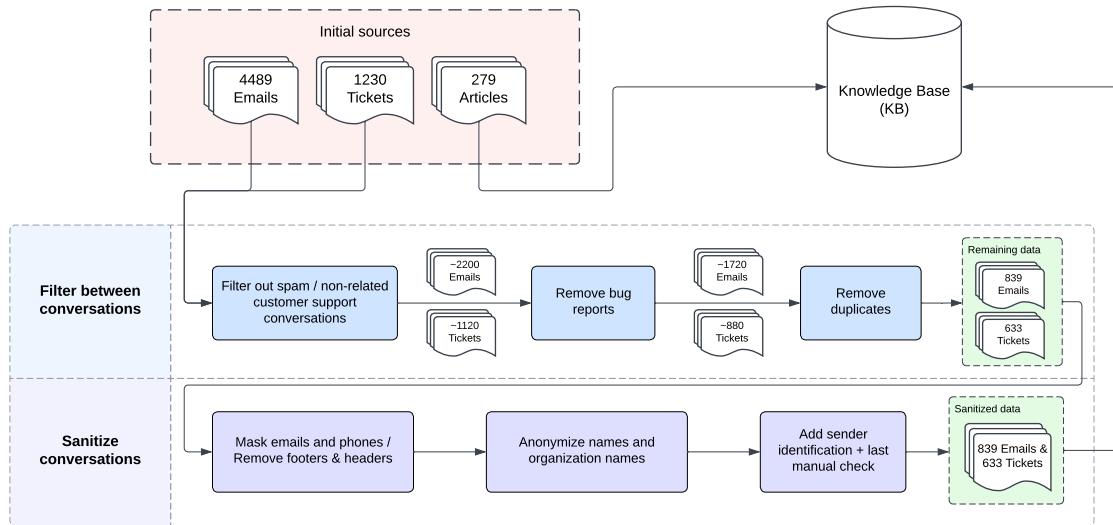


Figure 3.2: *The filter process applied to each of the data sources and how we constructed the knowledge base from these sources.*

replace them with masks.

3. Lastly, we added a sender indication, using various Regex patterns, to find out which party sent which email. This could be either ‘Customer:’ or ‘Support:’, and was placed before each email to make conversations more structured. We also performed a last manual check after the filtering process was done, to get an indication if all sensitive information had been removed. From what we saw, there were no more sensitive parts present anymore, indicating that these steps work effectively, and can be used for future instances.

The resulting conversations were then ready to be added to the KB along with the articles that did not need any processing.

3.2.3 Knowledge Base

The KB that was used in this thesis, consists of the cleaned emails, cleaned tickets, and articles discussed earlier. The quality of the data KB greatly influences the quality of the LLM’s performance (Liu et al., 2024), we were therefore quite thorough in the filtering process described above.

We also made sure to keep the KB up-to-date by implementing a feature that, using the HubSpot API, automatically adds tickets to the KB when they have been resolved. However, before being added they still need to go through the clean-up process. These new tickets only follow the second and third steps from the filtering between conversations

process as described in subsubsection 3.2.2.1. This is because unrelated tickets were already removed by a customer support employee and therefore not resolved, so they aren't collected when using the API. Starting with the second step, we first determine if the conversation is not a bug report or something related, by looking at the subject, because bug reports always have 'Bug report' or 'Bug' in the subject. Then we check if a ticket with the same ticket number is not already in the database, if it is not, we can just add the ticket to the database. If it is already present, we remove the existing database entry and replace it with the new ticket, as the new ticket is a continuation of the old entry, so it should contain all the information of the old entry plus the new extra information. We also have to sanitize the conversation, so before being added the new entry follows the same process as described in subsubsection 3.2.2.2.

3.3 Phase 1 - Initial response generator

This section will cover the process and different steps of phase 1 in detail as described in Figure 3.1. We first discuss the vector base and how it is created, and then we describe the response generator and its components. Lastly, we talk about the preparations for the evaluation and how it is done.

3.3.1 Retrieval database

The query handler (subsubsection 3.3.2.3) retrieves sources from two different databases: the document chunks database and the vector base, which were created from the KB (subsection 3.2.3).

3.3.1.1 Document chunks database

The document chunks database contains smaller 'chunks' of the sources that came from the KB. Chunking the data can help improve the relevance of the data ([Yepes et al., 2024](#)), and can also help with limiting the amount of context passed to the retriever. To create such a database we used Langchain's recursive character splitter to first split up a piece of text into smaller chunks, by defining the chunk size and chunk overlap. **Chunk size** is the number of characters each chunk contains while keeping words intact. So if we define a chunk size of 15 then the sentence 'An example of how chunk size could work in a document.', would result in ['An example of', 'how chunk size', 'could work in', 'a document.']. **Chunk overlap** is defined by the amount of characters that overlap between chunks. So in the above example when we also add a chunk overlap of 5 we get ['An example of', 'of how chunk', 'size could', 'work in a', 'in a document.']. We experimented with a large variety of chunk sizes ranging from 200 to 1500 characters and chunk overlaps from 50 to 200 characters. Ultimately, we chose to use a chunk size of 1000 characters with an overlap of 150 characters for each of the sources, as we thought that these values showed the best results.

3.3.1.2 Vector Base

A vector base or vector store, also described in subsubsection 2.1.3.7, is a database containing numerical representations of texts called vectors. We used the same chunks as for the document chunks database except that we needed to perform an extra step to transform the chunks into vectors. We embedded the chunks using the OpenAI embeddings model ‘text-embedding-3-large’ ([Neelakantan et al., 2022](#)), and stored them in a Chroma database ([Chroma, 2024](#)), for fast retrieval in the query handler. The vectors also included metadata about the source (email, ticket, article) it came from.

3.3.2 Response Generator

The response generator is an important part of the first phase, as it contains both the NLU and NLG units. In Figure 3.3 we can find the components of the response generator, a history step described in subsubsection 3.3.2.1, a pre-processing step in combination with the generator in subsubsection 3.3.2.2, and the query handler subsubsection 3.3.2.3.

3.3.2.1 History

Memorizing past information from a conversation is an important step in the generation of a response, to make sure that you can ask a follow-up question without losing memory of what was asked before. This was done with the use of an LLM instruction template, found in Table D.1, that took the chat history of the conversation, along with the follow-up question. It then rewrote the follow-up question into a standalone question, while it kept the history in mind. This way, the newly created question should contain the information present in the history combined with the newly asked question. The new question was then used further on in the response generator. An example of how a follow-up response is generated can be found in Table 3.2. The question doesn’t get rewritten if the response is about a completely new subject, for which the chat history is not needed anymore. We also don’t rewrite a question if it is the first question in the conversation, because there is no history yet in that case.

Initial question: Wat is a unit-element
Response: ...
Follow-up question: And how does it work?
Standalone question: How does the unit-element work?

Table 3.2: *An example of how a follow-up question gets rewritten based on the chat history into a standalone question, with the context of the past questions.*

You are a friendly customer support assistant. Answer the question at the end using the Q: and A: examples below.

Q: What is the maximum number of devices that can be active at the same time with a single user account?

A: A user account can have a maximum of 2 devices active simultaneously, as long as the combination of devices is Apple & Android, Web & Apple, or Web & Android. When using the same type of device, only 1 can be active at a time.

Q: What can happen if more than the allowed number of devices are active with a single account?

A: If more than 2 devices (or more than 1 of the same type) are active, active sessions may be revoked. This can result in drafts not being properly synchronized to the server.

Q: Can you use one account on multiple devices?

A:

Table 3.3: An example of a possible few-shot prompt template, using only two examples from the context. Below the line is the user query taken from Appendix B: question 24.

3.3.2.2 Pre-processing and generator

We now cover the process of how the responses were generated using the different generation methods. We mention the query handler multiple times in this section, which will be discussed in detail in subsubsection 3.3.2.3. The most relevant context from the query handler was passed on to the generator depending on the generation method. This context information was pre-processed in slightly different ways for each method so that each method received its respective format.

Zero-shot model uses an LLM in combination with a template, found in Table 3.4, that states it is a customer support assistant and it should answer the question. However it receives no context whatsoever, so we let the LLM figure out the answer on its own. We introduce this method to compare the general skill of the LLM in context-specific scenarios as a baseline performance indication.

Few-shot learning needs the context from the retriever to be processed in a Q&A template form so the LLM understands the task, such an example few-shot template can be found in Table 3.3. The template starts with the instruction of the task, like: 'You are a

You are a friendly customer support assistant.

Answer the 'Question' at the end of the prompt. This must be in the language {language}.

Question: {question}

Table 3.4: An example of the zero-shot and fine-tune instruction template, with the language, and question being variables filled in during the conversation.

friendly customer support assistant. Answer the question at the end by using the Q: and A: examples below.'. We then need to provide Q and A examples to the template using the context of the retriever. We did this by first letting an LLM convert each document from the retriever into a single Q&A pair, using the template found in Table D.2. The Q&A pairs then got added to the few-shot template, where each Q was followed by its corresponding A, and then the next Q started after the previous A, continuing this pattern. Lastly, we provided the user query to the LLM and we left the final 'A:' part empty so the LLM 'understands' that it needs to answer that last question. In this example, we see only two examples taken from the context, however, this could become a total of eight Q&A pairs as that was the maximum amount of documents that can be received from the retriever. If no context was available, then the LLM would perform zero-shot learning as it had no examples to go off.

Fine-tuning used all data from the KB, in total being $839 + 633 + 279 = 1751$ examples, to fine-tune its weights which made the model more adept to the KB-specific data. In the fine-tuning process, we fine-tuned OpenAI's GPT-3.5-turbo model, as no other models were available at that time. We had to format the data from the KB in a specific way, an example of this format is shown in Table 3.5, which we did for each of the examples. The role 'system' was the instruction that the model needed to take on, which was the same for each instance. Then you add the message from the customer as the content for the role 'user' and the answer from the support as the content for the role 'assistant'.

```
{"messages": [{"role": "system", "content": "You are a friendly and understanding customer support assistant"}, {"role": "user", "content": "..."}, {"role": "assistant", "content": "..."}]}
```

Table 3.5: An example of the JSON format used when fine-tuning for one data point. There was no limit to the number of back-and-forth messages you could add to one such entry.

You could do this multiple times, for instance, when a conversation had multiple turns between each party you could include all of these turns in the message. However, we had to keep in mind that each 'user' instance should directly be followed by an 'assistant' response. So if we had two 'user' or 'assistant' instances in a row, we appended them

You are a friendly customer support assistant.

Use the given 'Context' below to answer the 'Question' at the end of the prompt. This must be in the language {language}.

Context: {context}

Question: {question}

Table 3.6: An example of a possible RAG instruction template, with the language, context, and question being variables filled in during the conversation.

together as one, to ensure this structure was kept. During the response generation process, this method received almost the same template as the zero-shot model, so it did not get any context as well. This was to make sure the model only relied on the knowledge gained from the fine-tuning process and not on the context. However, another change was that the first line of the template about 'being a nice customer support assistant' was not present, as we already incorporated that during the fine-tuning process.

Retrieval-Augmented Generation (RAG) feeds all the documents of the retriever in combination with an instruction prompt to the LLM. The instruction prompt was a detailed explanation of what the LLM could and couldn't do in certain contexts, as seen in Table 3.6.

The zero-shot model and fine-tuning were able to generate a response when no context is provided, as they didn't use any context. Few-shot and RAG could still generate responses without any context, however, both methods would closely resemble the zero-shot model in this case, which would defeat their purpose. A key takeaway is that the query handler wasn't a bottleneck for the zero-shot model and fine-tuning in cases where it retrieves irrelevant or incorrect context, since they weren't using the context. Zero-shot, few-shot, and RAG were all using the newest OpenAI gpt-4 model ([OpenAI, 2023](#)). We used the LangChain framework to implement the entire response generator.

3.3.2.3 Query handler

The query handler was designed to effectively process user inputs by identifying and retrieving the most relevant information from the retrieval database. Upon receiving a user query it got matched to existing chunks or vectors from the retrieval database (subsection 3.3.1) using three different methods: keyword matching (BM25), similarity matching, and maximal marginal relevance. These three methods then potentially retrieve relevant documents to be used during response generation.

Best Matching 25 (BM25) is a ranking function used to estimate the relevance of documents to a given search query and has a probabilistic approach to ranking relevant documents ([Amati, 2009](#)). BM25 takes into account various factors such as term frequency (how often a term appears in a document), inverse document frequency (how common or rare

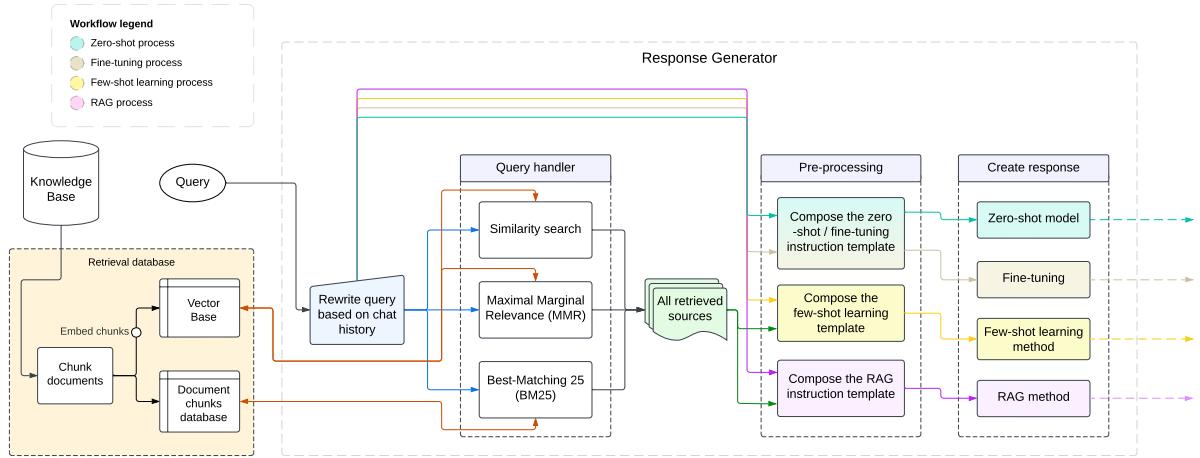


Figure 3.3: The response generator that is used in this thesis, consists of a query handler, a processing step, and a generator, as well as the retrieval database.

a term is across all documents), and document length. This method is more sophisticated than basic keyword matching as it considers the importance of each term in the context of the entire document collection, thus improving the retrieval of relevant documents even when exact word matches are not present. It, however, still struggles somewhat when it comes to context and things like synonyms. For BM25 we needed to use the document chunk database from subsubsection 3.3.1.1 because this method works with normal text and not with vectors. We retrieved the most relevant documents by applying BM25 to all the chunks of the document chunks database, with a score threshold of 0.125. We retrieved a maximum of four documents using BM25.

Similarity Matching involves transforming the query into a vector in high-dimensional space and comparing it with vectors stored in a vector base (subsubsection 3.3.1.2). To identify relevant vector base entries, we first used cosine similarity (with scores ranging from -1 being most dissimilar to 1 being most similar) to measure how close the query vector is to each stored vector in the vector space. This initial step ranks the vectors by similarity and retrieves the top candidates based on their cosine similarity scores.

After selecting the most similar vectors using cosine similarity, the Euclidean relevance function converted the Euclidean distance between the query vector and the retrieved vectors into a similarity score ranging from 0 to 1. This was done to ensure the results were normalized and provide a consistent measure of relevance. We chose a similarity relevance threshold of 0.35, discarding any documents with relevance scores below this value. The documents above the threshold were then converted back to their corresponding text chunks. We chose a maximum of four relevant documents for this retrieval method.

Maximal Marginal Relevance (MMR) is another technique to retrieve information (Carbonell and Goldstein, 1998). It is designed to balance the relevance and diversity of

retrieved documents and aims to select items that are not only relevant to the query but also provide new information, thus avoiding redundancy. This can help to improve the quality of the retrieved documents and therefore improve the quality of the information of the query handler. The most similar documents were again retrieved by comparing the embedded query to the vectors in the vector base using cosine similarity. However, the MMR algorithm then selects the most relevant and diverse documents from the selected vectors. We used the Euclidean relevance function just like for similarity matching and the resulting documents were ultimately converted back to chunks of text. In this study, a similarity relevance threshold has been set to 0.35, so documents below the threshold were not included. With the maximum amount of retrieved documents being four again.

The retrieved documents from each method were combined in an **ensemble retriever** that uses a weighted voting strategy and are ranked by the retriever based on the Reciprocal Rank Fusion algorithm (Cormack et al., 2009). This produced a ranked list of relevant documents, that could achieve better performance than any single algorithm. We set the maximum amount of retrieved documents to eight, so a maximum of eight out of the possibly twelve ($4 * 3$) most relevant documents, after reranking, were used later on during response generation. From experimenting with the thresholds and amount of retrieved documents for each retrieval method, we saw that in most cases each method provided some documents. However, in general, similarity matching and MMR were the most successful retrieval methods with at least two retrieved documents in most cases (depending on the query). When no relevant information was found by any of the retrieval methods, the query handler would return no context. We mentioned in subsubsection 3.3.2.2 that the query handler could be a potential bottleneck for some methods. This is why we have been careful in designing the query handler and its multiple components, to ensure it was as reliable as possible when retrieving documents.

3.3.3 Preparations for the first evaluation

To perform the human evaluation, we first needed to create a set of conversations generated by each of the methods. After this, we created a survey so people could rate the conversations based on multiple metrics. As for the automatic evaluation we created a set of question and golden response pairs, where each of the methods was used to generate a response to the question, these responses were then compared to the golden response.

3.3.3.1 Preparing the human evaluation

For the testing of the system, we used a range of 25 pre-selected initial questions from earlier interactions, with a customer query and a 'golden' response from customer support. We created conversations by interacting with the chatbot ourselves along with a customer support employee. We started with the initial question as input to the response generator and continued to converse with the system until one of the following ending conditions had been reached:

- We get to a satisfactory answer as provided by the support conversation.
- The chatbot's answer is deemed incorrect by us twice in a row.
- The conversation takes more than six turns (back and forth combined) before we get a satisfactory response.

We chose these conditions to make sure that we didn't end up in a loop, and conversations didn't take too long both resulting in customer dissatisfaction. We performed the same steps for each of the four methods resulting in $4 * 5 = 100$ total conversations.

To perform the human evaluation we created a survey in Qualtrics. The survey contained a random subset of 48/100 example conversations from each of the four response generation methods for each participant. Every conversation had to be rated based on six metrics that are explained later on in subsection 3.3.4. In between the 48 conversations we placed two other questions, one after conversation 16 and one after conversation 32. This was done to switch things up from time to time and make sure participants stayed alert, while they could still perform a relevant task. For the first question, participants were shown four conversations, one for each method, and were asked to comment on the structure of each conversation. For the second question, they were shown a customer query and had to respond to it themselves.

3.3.3.2 Preparing the automatic evaluation

To evaluate the effectiveness of the automatic evaluation metrics, we focused on a subset of tickets from the KB that included a 'question' variable. This variable represents the initial question posed by the customer and provides a clear user query. Some tickets did not include this variable because it was either not filled in by the user or the ticket was a reply to a previous message. However, tickets with this variable consistently featured a clear question along with a generally straightforward initial answer by customer support. We chose to use only the first response of the customer support as the 'golden response' even if the customer-support conversation consisted of more messages between both parties. Because the first response to the question most likely directly addresses the initial question, while successive responses are generally responses to the previous messages, which might not be about the initial question anymore.

We chose to use only tickets for the evaluation because they generally contained the 'question' variable, unlike emails or articles, which did not. While we considered using the subject lines of emails or the titles of articles, these did not guarantee that the content represents a real question, whereas the 'question' variable in tickets did. By focusing only on tickets with this variable, we ensured a consistent and meaningful dataset for testing the evaluation metrics.

3.3.4 Evaluation

The evaluation metrics for both human and automatic evaluation that were used in this study can be found in this section. We used six human metrics, as well as six automatic metrics. Human and automatic metrics don't correlate one to one as three of the automatic metrics were used as potential relevance metrics, two as potential completeness metrics, and one as a potential conciseness metric. We did not identify any automatic metrics that we considered appropriate for measuring accuracy, empathy, and friendliness.

3.3.4.1 Human evaluation

The participants from the human evaluation are from the company itself and have different functions like software developers, marketers, customer success employees, or people from human resources. With various levels of expertise and knowledge regarding the software and its workings. Participants could fill it in on their own as it was an online survey. They received instructions to evaluate the conversations, generated using the four different methods, on six metrics by answering six questions. The six metrics and respective questions, seen in Table 3.7, have been proposed using two studies. Questions one to four come from the study by [Bansal et al. \(2021\)](#), while questions five and six are taken from a study by [Dou et al. \(2023\)](#). We chose these metrics as they provide a comprehensive overview of different aspects of a response. Relevance and accuracy assess the informational quality of the response, while completeness and conciseness evaluate its structure and content. Additionally, empathy and friendliness are for evaluating the tone and emotional impact of the response. These metrics combined are used to determine the response quality in this thesis.

The questions were about the responses of the chatbot regarding the metrics relevance, accuracy, completeness, conciseness, empathy, and friendliness. The participants could rate each question on a 1-5 Likert scale ([Likert, 2017](#)) starting with one (strongly disagree) to five (strongly agree). In addition to the scores from 1 to 5, we categorized the scores during the evaluation as follows: scores 1 and 2 were deemed bad, score 3 was neutral, and scores 4 and 5 were considered good. This was to enhance the clarity of our evaluation and to give a more general analysis of the response quality and whether it was good, bad, or neutral. It is important to note that the ratings for the questions about completeness and conciseness were reversed, so a score of one was the best while five was the worst. This was taken into account during the analysis of the results. These questions were asked for each of the four methods to see which performs best. The order in which the conversations from each method were shown was randomized to avoid any bias. The survey was originally written in Dutch, however, it was also partially translated into English to make it more accessible for participants who were not native Dutch speakers. We translated most parts of the survey, except the conversations that were to be assessed. This was due to time constraints as well as the possibility of altering the conversation meanings during the translation process. Ultimately this did not have major effects as only one participant filled in the survey in English and did not mind the Dutch conversations. Participants were also

Metric	Question
Relevance	Independent of other factors, the response is relevant to the question, even if the information it contains is inaccurate.
Accuracy	The response is accurate, even if it is not relevant to the question.
Completeness	The response is missing important information, and that can hinder my understanding.
Conciseness	The response contains a lot of unnecessary information.
Empathy	The response demonstrates an understanding of the user's intents.
Friendliness	The response employs a friendly tone.

Table 3.7: *The six evaluation questions with corresponding metrics that were used in the survey.*

allowed to switch languages at any time during the survey.

3.3.4.2 Automatic evaluation

The automatic evaluation was done with the metrics BERTScore, BLEU, DeepEval, METEOR, ROUGE-l, and word count. BERTScore and BLEU used the 'golden' response and compared it to the system's response to perform an evaluation. DeepEval compared the response to the user query, to check if the response is a logically valid answer given the user query and gives a score between 0 and 1 based on the relevance. Multiple evaluation metrics were used to see how the different methods compare. Table 3.8 shows the metric scores given a question, response, and 'golden' response.

We matched these metrics with the relevance, completeness, and conciseness metrics from the human evaluation, which can be found in Table 3.8. The BERTScore value is based on the F1-score of the embedding similarities, possibly aligning with relevance. The BLEU value measures precision, which is the proportion of n-grams in the response that appear in the golden response, which could provide insights into response relevance. For DeepEval the relevance metric was chosen, which gives a relevance value using an LLM. It uses an LLM to rate the relevance of a question by returning either 'yes', 'no', or 'idk'. It performs this test many times and calculates the score based on the amount of 'yes' verdicts divided by the total amount of verdicts. The LLM used for this is the LLaMa-3.1-70b ([Dubey et al., 2024](#)) model using the Groq API so that GPT-4o is not evaluating itself. DeepEval also returns a reason for its decision when it finds the answer not relevant.

METEOR takes synonyms, stemming, and paraphrasing into account and shows correlation in capturing key information, which could indicate completeness. ROUGE-l emphasizes recall by measuring the overlap of phrases and sequences, possibly making it useful for assessing completeness in generated responses. Word count provides a straightforward comparison for conciseness, allowing for the evaluation of response length relative to human-rated optimal length.

Question: Will I lose all my forms and data after the trial period?

Response: Thank you for the question. No, you won't lose all your forms and data, because collected data will be stored in the cloud.

Golden response: No, you will not lose all your forms and data after the trial period. Collected data will remain stored in the cloud.

Relevance		Completeness		Conciseness	
	Score		Score		Score
BERTscore	0.862	METEOR	0.720	Word counts	23
BLEU	0.499	ROUGE-l	0.650		
DeepEval (Rel)	0.500				

Table 3.8: How the different automatic evaluation metrics score given a question, response, and golden response.

For each of the metrics the higher the score the better the response was. It is important to note that when no 'golden' response is available BLEU wasn't used, the BERTScore was compared to the question, and for the DeepEval metric, there were no differences.

3.3.4.3 Human and automatic metrics

To see if any of the automatic metrics could be used in the final architecture as one of the potential checks we performed a correlation test. Especially the relevance metrics could have been useful within the architecture if we found correlations. To delve deeper into the relevance metrics, we took a closer look at the DeepEval relevance metric and how an LLM compared to a human when looking at relevance, as this was a metric that had not yet been widely investigated. If none of the metrics correlated with the human metrics we wouldn't use the automatic metrics in the final architecture.

3.4 Phase 2 - Complete chatbot architecture

In this section, we will cover the process of the second phase after the first phase has been completed. The proposal of the new architecture is the most important part of the second phase and will be discussed in this section. We first cover the structure of the architecture itself in subsection 3.4.1, and then we look at the changes made to the response generator in subsection 3.4.2. Lastly, we discuss the second evaluation that we perform on this architecture in subsection 3.4.5.

3.4.1 Chatbot architecture

The chatbot architecture can be found in Figure 3.4. The architecture consisted of a KB and a query handler, however, we used an enhanced response generator, based on the results of

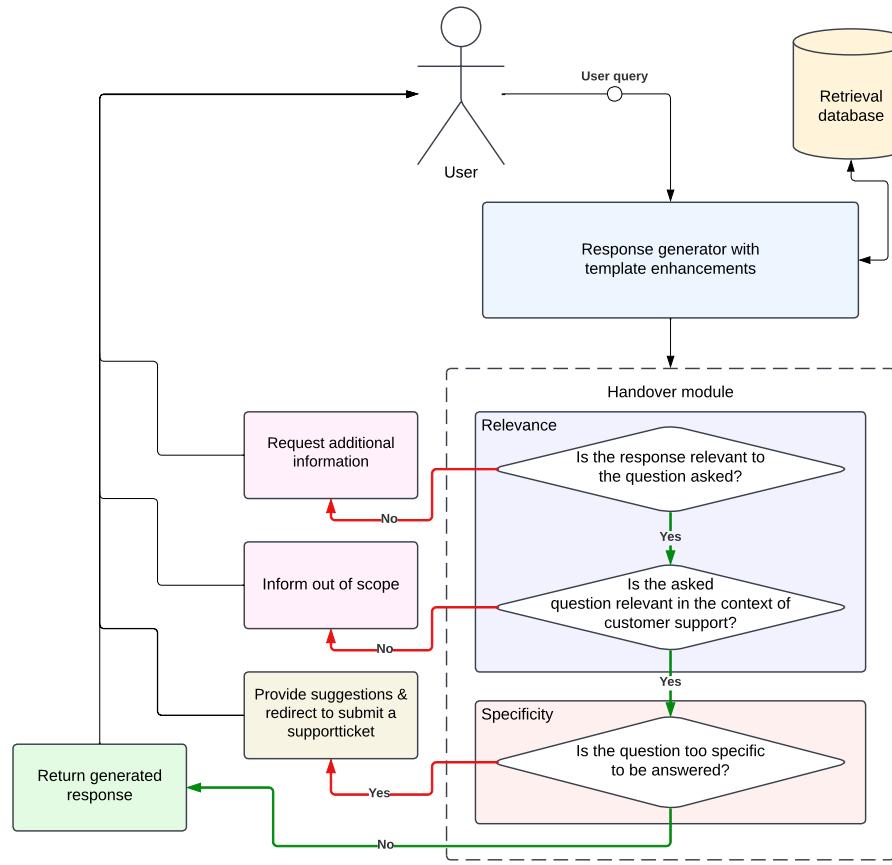


Figure 3.4: The proposed architecture in this thesis, including the handover module and the checks it performs.

the first phase. We also included a handover module that evaluates the generated response based on different checks explained later on in subsection 3.4.3 and returns a response based on these checks.

3.4.2 Response generator changes

We use the best-performing method (RAG) over all the metrics, from the evaluation of the first phase, as the backbone of the response generator for this second phase. The process of the response generator itself, shown in Figure 3.3, stays the same in the second phase. However, some potential changes to how we generate answers were made, based on the findings of the first phase, such as changing the instruction templates. We made changes to the templates not just by insights identified by us during the entire process, but also by comparing RAG to the other methods using individual question scores. If another

method performed better than RAG, we compared the responses of the two and see why the response of that method scored better. We let these comparisons guide us during the enhancement process, as we think that only looking at the RAG method and seeing why it scored worse in some cases, might not give us enough information as to why this is, which is why taking all methods into account, could provide more information.

3.4.3 Handover module

A handover module was implemented to manage specific types of questions that the response generator may not handle effectively. We will first cover the checks and the necessity of the module, after this, we look at how the checks should work, and lastly then how it was implemented, within the architecture.

3.4.3.1 Functionality

While overall RAG did provide the best response as seen in section 4.2, this method as well as the others still had some problems in its responses, like being irrelevant, answering questions it shouldn't, and being overconfident. This is the reason the handover module incorporates three key checks:

- Is the response relevant to the question asked?
- Is the asked question relevant in the context of customer support?
- Is the question too specific to be answered?

We wanted to make sure that the relevance of an answer to the response is guaranteed because a customer support chatbot needs to be helpful to the customer, providing relevant answers using the first check. We also didn't want questions that fall outside of the scope of customer support, which was done by the second check, to be answered, as it is a customer support bot. Otherwise, it would look more like a normal general chatbot like ChatGPT that answers questions from any domain you want it to.

Lastly, we want to balance the confidence of the chatbot in its answer with the accuracy of the answer. It has been shown that LLMs are often overconfident in their answers (Lin et al., 2022; Yang et al., 2024). To mitigate this overconfidence the module needed a specificity check that should make suggestions instead of direct answers, in cases where an initially clear answer might not be applicable.

So overall, this module ensured that when questions fall outside the scope of customer support or the model's capabilities, they were appropriately redirected, like how a human support agent would handle irrelevant, out-of-scope, or complex queries.

3.4.3.2 Construction of the handover checks

When a response is generated it passed through each of the three checks from subsubsection 3.4.3.1, this process is also depicted in Figure 3.4. If the final response passes the three

questions then the handover module was confident in the validity of its answer and used the generated response to reply to the query. Depending on the customer's reaction we could continue helping with their problem, or reply with 'Do you need help with anything else?'. If there was uncertainty in the response, the handover module decides what to do. This can be one of three things:

1. If check 1 fails, the bot responds with a question to provide some extra information. This makes it so there is more information to work with in the follow-up question. The follow-up question follows the same path as the initial question, however, if the test fails again then redirect the customer to submit a ticket to receive help.
2. If check 2 fails, the bot informs the user that the question is out of scope and responds with something like: 'I'm sorry, but I cannot help you with that question, if you have questions about the Incontrol software feel free to ask me.'. As it deemed the question not relevant in the realm of customer support.
3. If the first 2 tests are passed but test 3 fails, then the bot slightly modifies the originally generated response. Let's say the user has a problem with a personal concept and the response generator generates a response for this problem, but it might not apply to their case. The specificity step instead rewrites the conversation into a suggestion instead of an answer.

3.4.3.3 Implementation of the handover checks

To incorporate the checks from subsubsection 3.4.3.2, we used an LLM as the evaluator, which was OpenAI's gpt-4o model. The implementation of checking the relevance of the question itself and relevance within the scope of customer support was done by instructing an LLM to evaluate both using a specifically constructed template. As we have seen in subsection 4.2.7 we did not find any correlations between human and automatic evaluation metrics, we therefore chose not to use any of these metrics to determine relevance, but instead use an LLM in combination with a self-constructed template to determine the relevance of both. In this way, we could fine-tune the template, seen in Table L.3, used for this, so it evaluated in a way that we wanted it to. We combined both checks into one template to reduce generation costs, reduce generation time, and make a clear distinction between checking for relevance and checking for specificity.

To check for specificity it is important to consider what we mean by that. We defined specificity as instances where a human customer support employee would need more information about that question or user. This would involve taking a look at their account or asking follow-up questions, to find out what the problem is (e.g. personal concept problems, personal account issues, potential bugs, etc). These problems were not possible to solve using the current architecture as it has no way of accessing user accounts, concepts, or templates nor has it knowledge of the code to handle bugs. To perform the specificity check we again define a template and use an LLM for this, which can be found in Table L.4. We handled these questions in the following way: first, acknowledge that this

question cannot be confidently answered, then make a suggestion, if possible, given the response from the generator, and lastly, tell them to submit a ticket to customer support if the suggestion does not work.

The get the best results possible we continuously changed and tested parts of it, to make sure we balanced confidence with the accuracy of the answers to our liking while also being as relevant as possible we described in subsubsection 3.4.3.1. To provide a consistent and satisfying performance for the customer and their interactions with the bot. It is also important to note that the customer is redirected to customer support whenever the bot doesn't know the answer the second time. This was done to prevent the customer from getting stuck in a loop where the bot doesn't know the answer, as well as to prevent the bot from making up an answer that might not be correct or relevant at all. The specificity check was implemented to make sure that in cases that were too specific, the bot is not entirely confident in its answers and thus makes suggestions that could help, but also tells the customer that they might not help.

Good execution of these checks can help with customer satisfaction, as explained in section 2.2, by limiting the amount of misinformation and irrelevant answers. For the two relevance checks, there should, be a good balance between answering the question and redirecting the customer. The constant redirecting of the chatbot to customer support, when it is not entirely certain, may decrease the usefulness of the chatbot. For the specificity check, we tried to find a good balance between confidence in its answers and accurate/relevant answers.

3.4.4 Preparations for the second evaluation

For the second evaluation, we first created a set of customer scenarios for participants to place themselves in. We then created a survey where participants could interact directly with the chatbot and use the scenario to get an answer. The participants had to rate the chatbot responses after each completed scenario. To create the scenarios we looked at the individual question scores from the initial questions in Appendix B for RAG, which was the best-performing method. From the 25 initial questions, we took the five lowest-scoring questions, five average-performing, and the five highest-scoring questions based on all metrics. These questions were the following:

- Questions 3, 4, 12, 18, and 24 were selected for their high scores, with ranks 1 to 5.
- Questions 6, 7, 10, 11, and 15 had average scores with ranks 11 to 15.
- Questions 1, 2, 5, 17, and 22 were the questions with the lowest scores being ranked 21 to 25

From these 15 questions, we created the three difficulty sets, easy, medium, and hard, consisting of five questions and rewrote them into scenarios. In the second evaluation, the goal of the assignment was for participants to imagine themselves as a customer in that scenario. Each of these scenarios can be found in Appendix C.

3.4.5 Evaluation

In the second evaluation, we let the participants interact with the bot themselves instead of evaluating pre-made conversations. This had more real-life accurate results compared to the first evaluation as every person had a different approach to asking questions. The results in subsubsection 4.2.7.1 show that we didn't find any correlations between the human and automatic metrics. We, therefore, decided to not use any automatic evaluation metrics in the second evaluation.

3.4.5.1 Human evaluation

We again conducted a human evaluation to assess the quality of the final chatbot architecture, allowing participants to interact freely with the system. An interactive web application, developed using Streamlit, was used for this which included both the chatbot and a survey. During the evaluation, we were present to oversee the procedure and provide additional information when requested.

The evaluation began by presenting participants with a scenario discussed in subsection 3.4.4, placing them within a hypothetical context. Participants could ask the chatbot a question related to the scenario, and the bot would respond. If satisfied with the response, they clicked a "satisfied" button, which redirected them to a scoring page to evaluate the chatbot's answers. If unsatisfied, they could ask follow-up questions to attempt to achieve a satisfactory response. Each scenario allowed for up to three questions, ensuring participants had sufficient opportunities to reach a satisfying response. After the third question, participants were automatically redirected to the scoring page. Following each scenario, they completed a survey evaluating the chatbot on the same metrics as those used in the first evaluation, detailed in Table 3.7, and employed the same Likert scale described in subsubsection 3.3.4.1.

To ensure consistency and balance, each participant was assigned six scenarios randomly, with two scenarios drawn from each difficulty level (easy, medium, and hard). As in the first human evaluation, participants were selected from within the company, representing varying levels of experience. This evaluation method provides a deeper insight into the system's general performance, as we captured the feedback of the participants after interactive engagements with the chatbot.

Chapter 4

Results

In this chapter, we will first cover some of the chatbot responses to get some initial insights into how each method generates its responses in section 4.1. In section 4.2 we will look at the results of the first evaluation, first tackling the three subquestions: SQ 1.1, SQ 1.2, and SQ 1.3. We also cover the influence of experience levels, the initial questions, and the comparisons between the human and automatic evaluation metrics in this section. Lastly, we cover the results from the second evaluation and how these compare to those of the first evaluation in section 4.3.

4.1 Chatbot responses

We will go over different conversations that were generated by using a zero-shot model, few-shot learning, fine-tuning, and RAG. We can take a first look at these generated responses and see how each method fares. Each of the methods will be evaluated in more detail in section 4.2. We started with 25 chosen initial customer queries which can be found in Appendix B. These questions were put into each method and depending on the answer a follow-up question was asked. A sample conversation from each of these methods is shown in Table 4.1. In this sample, the initial question a customer asked was about the possibility of signing reports using a tablet. All conversations consist of first-try answers, even if they are not accurate. For zero-shot, few-shot, and RAG the newest OpenAI gpt-4o model was used for the LLM, while for fine-tuning we used the gpt-3.5 model.

4.1.1 Zero-shot responses

Even though the zero-shot model had no context about the company, it can still answer the example question seemingly well, mostly due to the impressive abilities of the LLM. The responses do seem to be quite general and consist of quite some repetition, likely due to the lack of context. We can see however that the responses are friendly, and feel like they are trying to help.

4.1.2 Few-shot learning responses

The responses that were generated using few-shot learning were really concise and straight to the point if we look at the example. It does answer the question correctly, it could have given a little more explanation. Another thing we noticed was that even though the answers were concise and straight-to-the-point they did not feel very empathic or friendly. This could be problematic for a customer support bot.

4.1.3 Fine-tuning responses

Generating the responses using fine-tuning showed quite different results compared to the previous two methods. The responses were quite long and detailed, however, this doesn't mean that the provided information was correct. The information and links that were provided were often untrue or incorrect. The responses do have a friendly and helpful tone which is nice to see, as it reflects how the customer support answers in the dataset.

4.1.4 RAG responses

Using RAG for generating responses, we saw that responses usually look quite good. They provide some detail without it being too long and often the information is also quite accurate. Some consecutive responses can, however, become repetitive as the responses often contain part of the information from the response before it. The RAG responses are again quite empathic and friendly, which is expected from a customer support bot.

Table 4.1: *We can see how the different methods respond to the user queries, in a conversational setting.*

Method	User:	Initial Question: Can I sign reports using my iPad?
Zero-shot	System:	Hello! Yes, you can sign documents on the iPad. Our software supports signing documents on various devices, including the iPad. If you need help setting up or using this feature, please let me know! I'm happy to assist you further.
	User:	And can this be done with every subscription?
	System:	Hello! Yes, you can sign documents on the iPad with any of our subscriptions. Our software supports electronic signatures on all devices, including the iPad. If you have any other questions or need assistance, feel free to let me know!
	User:	Okay, thank you.
Few-shot	System:	Yes, this is possible for all subscriptions.

Continued on next page

Table 4.1 continued from previous page

Method	User:	Initial Question: Can I sign reports using my iPad?
<i>Fine-tuning</i>	System:	<p>Dear,</p> <p>Thank you for your message. It is possible to sign documents on the iPad. This can be done with the Adobe Acrobat Reader application. This application is free to download from the App Store. Once you have opened the document in the app, you can click on the pen icon in the top right corner. Then you can use the pen to sign your signature.</p> <p>When you are finished signing, you can click on the checkmark in the top right corner. The document will then be saved automatically. If you have any further questions, please let us know.</p>
	User:	Can't it be done within the app itself?
	System:	<p>Dear,</p> <p>Thank you for your message. It is possible to sign documents without downloading an additional application. This can be done through the web version of Incontrol on your tablet. Please see the following link: https://helpcenter.getincontrol.eu/nl/faq/hoe-kan-ik-een-document-onder tekenen-als-ik-het-document-niet-kan-openen-op-mijn-tablet. If you have any further questions, please let us know.</p>
	User:	And can this be done with any subscription?
	System:	<p>Thank you for your message. It is possible to sign documents on the iPad with any subscription. However, you do need a signature to sign documents. You can place this signature on the iPad by using the signature feature in the app. This feature is only available for users with a premium license. See also: https://helpcenter.incontrol.app/n1/handtekening-zetten/hoe-kan-ik-een-handtekening-zetten-in-de-ios-app I hope I have provided you with sufficient information. If you have any further questions, please let us know.</p>
	User:	Thank you!
<i>RAG</i>	System:	Yes, it is possible to sign documents on an iPad. We support digital signing on tablets and mobile devices. This works in a manner similar to a "postman" method. If you need any additional information, please let me know!
	User:	And can this be done with any subscription?

Continued on next page

Table 4.1 continued from previous page

Method	User:	Initial Question: Can I sign reports using my iPad?
	System:	Dear, Yes, the signing of documents is available for all subscriptions. You can do this on tablets and mobile devices, as mentioned earlier. If you have any further questions or need additional information, please feel free to let us know!
	User:	Thank you!

4.2 Phase 1: Evaluation

This part will provide the results of the first evaluation in detail. We first cover the characteristics of the gathered data in subsection 4.2.1, then we move on to the results and statistical analysis for each of the sub-questions in subsection 4.2.2, subsection 4.2.3, and subsection 4.2.4. As for the statistical tests, given that the data collected were ordinal Likert scale ratings, which may not meet the assumptions of normality and equal variances required for parametric tests (Jamieson, 2004), we used a non-parametric test. The Kruskal-Wallis test was selected, because we are comparing more than two independent groups: zero-shot, few-shot, fine-tune, and RAG (Sheskin, 2003). A Dunn's post-hoc test with Bonferroni correction was used to identify differences between groups while controlling for Type I error (Dunn, 1964; Sheskin, 2003). With p being $\alpha/2$ so we reject H_0 when $p < 0.025$ for $\alpha = 0.05$. We will lastly provide the accuracy scores based on the experience level of the participants (subsection 4.2.5), the scores for each of the initial questions (subsection 4.2.6), and correlation tests between the human and automatic metrics (subsection 4.2.7).

4.2.1 Survey overview

In Table 4.2 we see that 25 participants have filled in the survey, which resulted in 954 conversation reviews. Some of the entries were not entirely completed, however, these results were still used as the conversations that were reviewed were still useful and not finishing the survey did not change this. Participants also had various levels of knowledge about

Experience level	Participants	Nr. of reviews	K-alpha
1	0	0	0
2	2	59	0.42
3	5	169	0.51
4	9	384	0.68
5	9	342	0.81
Combined	25	954	0.65

Table 4.2: *The number of participants and reviews retrieved from the survey.*

	Zero-shot		Few-shot		Fine-tune		RAG	
	\bar{X}	σ	\bar{X}	σ	\bar{X}	σ	\bar{X}	σ
Relevance	3.92	1.12	4.05	1.08	3.97	1.09	4.34	0.76
Accuracy	3.34	1.38	3.59	1.24	3.35	1.42	3.92*	1.08
Completeness	2.95	1.46	3.15	1.43	3.04	1.41	3.62**	1.30
Conciseness	3.31**	1.39	3.76	1.29	3.71	1.36	3.86	1.21
Empathy	3.97	0.92	3.52**	1.16	3.84	0.99	4.20*	0.80
Friendliness	4.24	0.92	3.67**	1.08	4.10	0.72	4.39	0.66
Total	3.62	1.28	3.62	1.25	3.67	1.24	4.06**	1.03

Table 4.3: Average scores for each method per metric and in total, green being the highest score and red the lowest score per metric. * = $p < .025$, ** = $p < .01$

the software, however, all participants except two rated themselves as at least a three or higher. This was to be expected as all participants were employees of the company, meaning they were at least somewhat familiar with the software. We can see from Krippendorff's α (Krippendorff, 2011) that the inter-annotator agreement overall shows moderate agreement between annotators. People of experience 2 and 3 have poor agreement with each other, however, this could be due to the small sample sizes. We see that very experienced participants do show satisfactory agreement between each other. How the experience influenced the results will be discussed later in subsection 4.2.5. We can find the summary of the survey results in Table E.1, with the proportions of the given scores displayed in Figure 4.1. These human scores will be discussed per metric in subsection 4.2.2, subsection 4.2.3, and subsection 4.2.4, as well as their corresponding automatic metrics.

4.2.2 Relevance and Accuracy

We will address the results of SQ 1.1 and find out how the generation methods compare to each other when it comes to relevance and accuracy of responses. We do this for both the human metrics and the automatic metrics only for relevance, as we had no automatic metric for accuracy, below.

4.2.2.1 Human evaluation

In Table 4.3 we see the average scores for both relevance and accuracy given the method. We first take a look at relevance. Even though RAG receives higher scores on average with less variance, it is overall not performing statistically better than all three methods. If we look at Table E.1 we see that RAG does perform significantly better than the zero-shot model and than fine-tuning but there are no statistical differences between RAG and few-shot learning. Figure 4.1 shows us that RAG receives a 4 or 5 score in more than 93% of the cases compared to around 80% for the other methods, while also receiving less 1 or 2 scores.

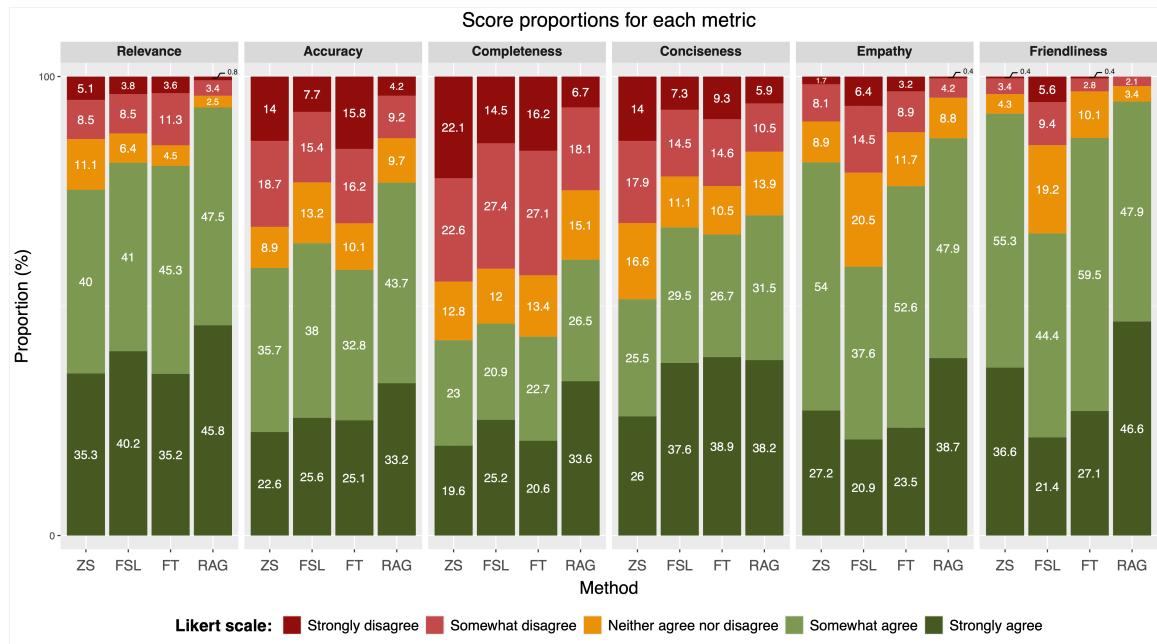


Figure 4.1: The score percentages for each response given the method and per metric. ZS = zero-shot, FSL = few-shot learning, FT = fine-tuning, RAG = retrieval-augmented generation.

If we then look at accuracy, we see that RAG has a higher average score with a lower standard deviation and is statistically performing better than the other methods. For both the zero-shot and fine-tuning this difference is more apparent than for few-shot learning. The same distributions of scores can be seen for accuracy compared to relevance, however, the scores, in general, are noticeably lower. Again RAG receives almost 10% more 4 or 5 scores than the other methods. However, only 76% of scores being a 4 or 5 is quite low when it comes to accuracy.

4.2.2.2 Automatic evaluation

When we take a look at the automatic metric means for relevance in Table 4.4 we see that the average scores for BLEU were statistically the lowest for the zero-shot model, with few-shot learning scoring the highest. In the case of BERTScore, zero-shot again scored significantly worse than the rest, while RAG performed best even if it was not significantly better compared to few-shot learning. For DeepEval, RAG was scoring significantly worse than both zero-shot and few-shot learning, but, not compared to fine-tuning. Few-shot learning, on the other hand, was doing significantly better than the other three methods.

4.2.3 Completeness and Conciseness

This part will cover SQ 1.2 about how the generation methods fare against each other when it comes to completeness and conciseness of responses. We first look at the human metrics and then automatic evaluation metrics.

4.2.3.1 Human evaluation

If we look at Table 4.3 we see that among the methods RAG performs the best again while the zero-shot model was the worst. The differences between RAG and the other methods were noticeably significant as can be seen in Table E.1. We see in Figure 4.1 that the score distributions for all methods are quite low when it comes to completeness. With RAG at only 62% 4 or 5 scores being the highest, while the other methods were not able to get over 50%.

As for the conciseness scores, we can see that RAG is performing the best and zero-shot performs the worst as has been before. For conciseness, the zero-shot model was performing significantly worse than all other methods. We also see in Figure 4.1 that fine-tuning gets slightly more 5 scores than RAG, however when we look at 4 or 5 scores then RAG does better with about 5%. The amount of 1 or 2 scores is also quite a bit less when we look at RAG compared to the other methods.

4.2.3.2 Automatic evaluation

When we look at Table 4.5, we see the automatic evaluation metrics used for completeness and conciseness. Completeness was assessed using the METEOR and ROUGE-1 scores. The zero-shot model was for both metrics significantly worse than the other methods, while RAG achieved the highest METEOR and ROUGE-1 scores. RAG was, however, not statistically better than few-shot for either METEOR or ROUGE-1.

In terms of conciseness, using the word count metric, few-shot learning showed significantly lower word counts compared to responses of the other methods. Fine-tuning also had significantly lower word counts compared to RAG and zero-shot but not as low as

	Zero-shot		Few-shot		Fine-tune		RAG	
	\bar{X}	σ	\bar{X}	σ	\bar{X}	σ	\bar{X}	σ
BLEU	0.034**	0.032	0.127	0.225	0.065	0.051	0.119	0.184
BERTScore	0.648**	0.034	0.686	0.090	0.681	0.038	0.691	0.074
DeepEval (Rel)	0.755	0.197	0.854**	0.237	0.661	0.252	0.654	0.197
Human	3.92	1.12	4.05	1.08	3.97	1.09	4.34	0.76

Table 4.4: How the different automatic evaluation metrics compare between the four methods regarding relevance, with green the highest and red the lowest scoring per metric. * = $p < .025$, ** = $p < .01$

	Zero-shot		Few-shot		Fine-tune		RAG	
	\bar{X}	σ	\bar{X}	σ	\bar{X}	σ	\bar{X}	σ
Completeness								
METEOR	0.156*	0.057	0.225	0.225	0.209	0.096	0.276	0.206
ROUGE-1	0.161**	0.062	0.252	0.228	0.218	0.080	0.262	0.191
Human	2.95	1.46	3.15	1.43	3.04	1.41	3.62**	1.30
Conciseness								
Word counts	100.959	59.761	58.680**	45.972	66.046	41.857	90.896	46.815
Human	3.31**	1.39	3.76	1.29	3.71	1.36	3.86	1.21

Table 4.5: How the different automatic evaluation metrics compare between the four methods for completeness and conciseness, green being the highest scoring metric and red the lowest. * = $p < 0.025$, ** = $p < .01$

few-shot learning.

4.2.4 Empathy and Friendliness

We will now address the third and last sub-question SQ 1.3 about how the empathy and friendliness of responses compare between the generation methods. We only covered the results of the human metrics as we had no corresponding automatic metrics.

4.2.4.1 Human evaluation

For the empathy scores, we see in Table 4.3 that few-shot learning is the worst-performing while RAG has the highest average scores with the lowest variation in its scores. The statistical differences between few-shot learning and the other methods are very apparent however RAG is also performing statistically better than both the zero-shot model and fine-tuning. We also see in Figure 4.1 that the percentage of 4 and 5 scores is over 80% for zero-shot and fine-tuning while being over 85% for RAG.

When taking a look at friendliness we almost see the same results as for empathy. Few-shot does the worst while RAG has the best scores while having the lowest standard deviation. Again, the few-shot learning scores are quite significantly different compared to the other scores. However, in this case, the scores of RAG aren't as far apart compared to the zero-shot model and fine-tuning as was for empathy. In Figure 4.1 we see that the percentage of scores being 4 or 5 is even higher for these methods. Being around 90% for the zero-shot model and fine-tuning, and even close to 95% for RAG also without being given a single 1 (strongly disagree) score.

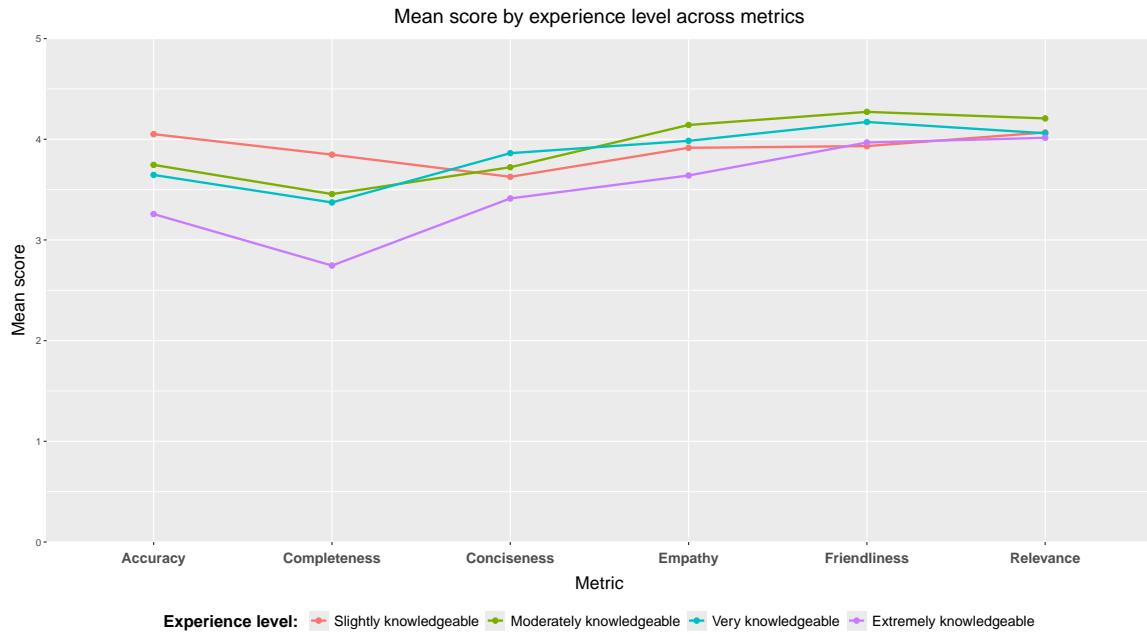


Figure 4.2: The mean scores for each of the metrics based on the participant's experience level.

4.2.5 Impact of experience

Now we will show the results of how experience level regarding the software. In Figure 4.2 we can see how it affects the scores for each of the metrics. We can see that especially for the extremely knowledgeable group the means are lower than the rest. This is fairly consistent over all the metrics as well, where the lowest score has consistently been given by the people with the most experience. This trend is most noticeable for accuracy and completeness but is also present in the other metrics. It is important to note that the number of participants of experience levels 2 and 3 was considerably less than for 4 or 5 as seen in Table 4.2, which could influence these results. Nonetheless, the differences between very knowledgeable and extremely knowledgeable, which are the most prominent levels, are still clearly visible in four of the six metrics.

4.2.6 Initial questions

When we take a look at the initial questions we can find out what questions were answered well and what questions were not. In Table 4.6 we can find the top five best-performing questions and top five worst-performing given all the metrics. We can also look at the individual metrics, shown in Appendix F, and compare them to the overall scores. Here we can see that relevance, accuracy, and conciseness have very similar best- and worst-performing questions compared to the overall scores, with some slight differences in their

Top 5 Best given All Metrics			
Nr.	Question	\bar{X}	σ
18.	<i>I want to upgrade from a basic license to a premium license, can this be done without data loss?</i>	4.44	0.73
10.	<i>Can I create a back-office connection with your application?</i>	4.26	0.86
13.	<i>I'm testing your tool, but find it very complicated... can I get a demo?</i>	4.23	0.92
6.	<i>Can I add a company logo to the template so that it appears on the page instead of the Incontrol logo?</i>	4.14	0.99
9.	<i>Can I retrieve a form that was accidentally deleted?</i>	4.06	0.93
Top 5 Worst given All Metrics			
Nr.	Question	\bar{X}	σ
5.	<i>For the past few days, a draft won't load. Other drafts do load.</i>	2.80	1.32
2.	<i>I have lost my form, can it be retrieved?</i>	3.18	1.37
17.	<i>I would like to add a PDF to a form from my hard drive. Currently, I can only add a photo. Is that possible?</i>	3.22	1.30
1.	<i>I created a trial account on your website last week. Can you tell me if I can manage with the basic package?</i>	3.33	1.30
8.	<i>Some photos I take initially appear sharp but turn out to be blurry later, how can this happen?</i>	3.39	1.26

Table 4.6: The top five best and worst performing questions based on their average scores given all metrics.

order. For completeness, empathy, and friendliness we can see that some of the best- and worst questions are featured when comparing them to all metrics combined, but with more differences compared to the other three metrics. Some questions like questions 13 and 18 for the best-performing and questions 2 and 5 for the worst-performing are featured in all individual metrics. In Appendix G we can find the proportion of scores for each question per method for all six metrics.

4.2.7 Human vs. Automatic metrics

Now we will take a look at how the human and automatic evaluation methods compare. We will look at potential correlations, apparent discrepancies, and some considerations regarding automatic evaluation metrics.

4.2.7.1 Correlations between the human and automatic metrics

To find potential correlations between the human and automatic metrics we used a Spearman correlation test (Spearman, 1904). In Table 4.7 we can see that there is no significant correlation between the human metrics and any of the different automatic metrics based on the p-values. From Spearman's rho, we can see that there is a weak negative relationship

Relevance			Completeness			Conciseness		
	ρ	p-value		ρ	p-value		ρ	p-value
BLEU	-0.019	0.855	METEOR	0.100	0.320	Words	-0.173	0.085
BERTScore	0.095	0.345	ROUGE-1	0.021	0.834			
DeepEval	-0.028	0.779						

Table 4.7: How the different automatic evaluation metrics correlate, using Spearman’s rho and p-value, to the human relevance, completeness, and conciseness metrics.

between conciseness and word counts ($\rho = -0.173$). This indicates that longer answers tend to be less concise or in other words that shorter answers are more concise. However, this relationship is not statistically significant.

4.2.7.2 Discrepancies between the human and DeepEval relevance scores

We will handpick a few examples that show big discrepancies between the human relevance metric and the automatic DeepEval metric to see how LLMs rate relevance differently compared to humans. The conversations mentioned here can be found in Appendix J.

For the Deepeval metric, we see that some of the scores between humans and this metric differ quite a lot. For example, in the zero-shot response for question 6, the relevance score given by humans is 4.7/5 while DeepEval only gives it a 0.33/1.0. The reason behind DeepEval’s lower score is that while it found part of the answer was relevant, it thought ‘the presence of irrelevant statements in the actual output, such as a greeting, a request for clarification, and an offer to help, detracted from the main topic.’ Humans were way more satisfied with the relevance of the answer and did not have problems with the formalities of a greeting, a clarification request, and an offer to help.

For the fine-tuned response to question 2, humans gave on average a 3.25/5 score while DeepEval gave a relevance score of 0.1/1.0. The reasoning behind this relevance score was that ‘the actual output barely addresses the user’s query about retrieving a lost form, and is mostly filled with irrelevant statements, such as generic advice and account information not related to the query at hand.’ While humans also didn’t find the answer that relevant, their scores weren’t as bad as those of others because the answer still addresses some points of relevance like what happened to the form, even though the answer is completely wrong.

Lastly, the response of few-shot learning on question 21, has only a 2.25/5 score by humans while it gets a perfect 1.0/1.0 from DeepEval. Humans found all of the questions that were given in the response not relevant to the query, as they were not needed to solve the problem. DeepEval stated that ‘the actual output perfectly addresses the issue, with no irrelevant statements, and the solution is presented in a concise manner.’

4.2.7.3 Considerations regarding the automatic evaluation metrics

BERTscore had some interesting results, that we wanted to investigate. More than 94% of the scores were above 0.6, with more than 75% of these scores being between 0.6 and 0.7, indicating that there was reasonable relevance in most cases. We performed an extra step to further research the appropriateness of this metric and assess whether or not the metric was measuring something more general like the domain of the text instead of the contents itself. We tested this in two ways:

- We split up each of the responses in two and compared each individual split to the golden response. We then took the average score of the two splits to see how these results compare to the original scores.
- We shifted all of the golden responses by one in the dataset so the golden response should therefore not apply to the question anymore and then calculated the BERTScores for these.

The results of these tests can be found in Table 4.8. We see that when we calculate the scores in the first test, the results are quite similar to the original BERTScores. A Wilcoxon rank sum test indicates that the two sets of scores are significantly different, however, about 92% of scores are still above 0.6. From the scores above 0.6, about 85% were between 0.6 and 0.7, showing that part of the scores became less relevant compared to the original. This is because only 15% of the scores were above 0.7 for the splits, while for the original scores, this was around 25%, which can also be seen in Figure I.1.

For the second test, we see that the scores are a little lower than those of the original however the differences are still not very big. The two sets of scores are significantly different according to a Wilcoxon rank sum test, however, still about 91% of scores are above 0.6. If we look at how many scores are between 0.6 and 0.7, we see that about 95% of scores were between this interval, as seen in Figure I.1. This shows that part of the scores have become even less relevant compared to the original scores because only 5% is above 0.7,

	Zero-shot		Few-shot		Fine-tune		RAG	
	\bar{X}	σ	\bar{X}	σ	\bar{X}	σ	\bar{X}	σ
BERTScore	0.648**	0.034	0.686	0.090	0.681	0.038	0.691	0.074
BERTScore splits	0.643**	0.029	0.660	0.065	0.665	0.033	0.672	0.054
BERTScore shifted	0.635	0.030	0.635	0.031	0.665**	0.033	0.644	0.030

Table 4.8: The BERTScores for normal golden responses, split golden responses, and shifted golden responses, with green the highest and red the lowest score.

For DeepEval we were curious how it would rate the golden responses to see if the metric did prefer human responses over the generated ones. As DeepEval was the only metric that did not consider the golden responses but instead directly compared the question to the generated response we were able to do this. In Table 4.9 we see that the average

	Zero-shot		Few-shot		Fine-tune		RAG		Golden	
	\bar{X}	σ								
DeepEval	0.755	0.197	0.854**	0.237	0.661	0.252	0.654	0.197	0.694	0.308

Table 4.9: *The DeepEval relevance scores of the different methods or for the golden responses, green being the highest and red the lowest.*

relevance score of DeepEval for the golden response was significantly worse than both few-shot learning and the zero-shot model. It did somewhat better than RAG and fine-tuning, however, this was not significant. When we looked at the reasons for the lower scores, we generally saw statements like: ‘The score is 0.5 because, although the actual output touches upon the main issue, it includes irrelevant statements that are unrelated to the user’s main concern.’.

4.3 Phase 2: Evaluation

We will now take a look at the results of the second evaluation. We first cover the survey and its details in subsection 4.3.1, and then we show the results of how the previously best-performing method (RAG) fares against the newly created architecture in subsection 4.3.2. Lastly, we also look at some of the initial scenarios to see which ones are scoring the best and worst in subsection 4.3.4.

4.3.1 Survey overview

In Table 4.10, we see that 16 participants filled in the survey for the second evaluation, resulting in a total of 96 conversation reviews. All participants were employees of the company and had various levels of experience with the software. All participants rated themselves at least a level 2 or higher, indicating just like the first evaluation that they were at least somewhat familiar with the software, which makes sense. Compared to the first evaluation, there were fewer participants and reviews in this round, as this was an

Experience level	Participants	Nr. of reviews	K-alpha
1	0	0	0
2	2	12	0.21
3	2	12	0.25
4	5	30	0.53
5	7	42	0.68
Combined	16	96	0.46

Table 4.10: *The number of participants and reviews retrieved from the survey.*

	RAG		RAG subset		New	
	\bar{X}	σ	\bar{X}	σ	\bar{X}	σ
Relevance	4.34	0.76	4.34	0.76	4.55	0.61
Accuracy	3.92	1.08	3.93	1.10	4.51	0.73
Completeness	3.62	1.30	3.64	1.32	4.17	1.09
Conciseness	3.86	1.21	3.89	1.24	4.47	0.89
Empathy	4.20	0.80	4.14	0.86	4.17	0.95
Friendliness	4.39	0.66	4.34	0.71	4.56	0.74
Total	4.06	1.03	4.05	1.05	4.40	0.86

Table 4.11: Average scores for RAG in total, for RAG only using the subset of questions that were used for creating the scenarios, and the new architecture, both per metric and in total. Green shows the highest score for that metric.

in-person evaluation instead of an online one. Despite the smaller sample size, the participants' experience levels remained diverse, with the majority rating themselves at level 4 or 5. If we look again at Krippendorf's α we see that there is poor agreement among participants. This is the case for all experience levels except the very experienced group which showed moderate agreement. There were some differences in this evaluation compared to the one performed in the first phase. First of all, the way in which we performed the evaluation was different as the participants were able to interact with the chatbot themselves, instead of looking at pre-made conversations. Another change came with the inclusion of the handover module, where responses could possibly be influenced instead of giving the response directly. Lastly, the RAG prompt templates were enhanced to improve the overall response quality.

4.3.2 RAG vs. changed architecture

Now we will look at the results of the survey and how the new architecture compares to the previous best method RAG. The results in Table 4.11 provide an insight into the differences between the previous best method, RAG, the scores of RAG using a subset with only the questions that were selected for the creation of the scenarios, and the newly proposed architecture. We see that the scores between RAG and the subset are generally very similar. Figure 4.3 also shows the distribution of answers per metric between RAG, RAG with the subset, and the new method. RAG and the subset again showed very similar distributions for all metrics. As the results between the first and the second phases were collected in quite different ways (scoring pre-made conversations vs. scoring personal interactions), we will not perform any statistical analyses on these results. It is, however, still interesting to lay both of them side by side and look at the differences.

4.3.2.1 Relevance and Accuracy

In terms of relevance, the new model achieves a higher average score compared to RAG and the subset, with a lower standard deviation. This could indicate that the new architecture not only provides more relevant responses but does so more consistently. We also see that the percentage of 4 or 5 scores is a little higher for the new method, mostly from the amount of 5 scores being almost 15% higher.

For accuracy, the improvement is even more pronounced. The new architecture scores an average of almost 0.6 higher compared to RAG and the subset, while also having a lower standard deviation. This could suggest that responses from the new model, besides being more relevant, are also more accurate and consistent. The percentage of 4 or 5 scores is also almost 16% higher for the improved architecture compared to both RAG and the subset, with the amount of 5 scores being almost doubled.

4.3.2.2 Completeness and Conciseness

In the case of completeness, the new architecture again greatly outperforms RAG and the subset with an average of 0.55, while its standard deviation is lower. This again could indicate an increased performance and consistency in providing comprehensive answers. The number of 4 and 5 scores again rose from about 60% (RAG) or 62% (subset) to 75.0%, largely due to the large increase of 5 scores.

Regarding conciseness, the improved architecture scores around 0.6 higher on average than both and with less variability. The amount of 4 and 5 scores were over 80% for the new architecture compared to around 70% for RAG and the subset. The increase in 5 scores was the main reason for this, being over 30% higher, as well as getting almost no 1 or 2 scores.

4.3.2.3 Empathy and Friendliness

An interesting observation is seen in the empathy metric, where RAG does slightly better compared to the new architecture, in both score and standard deviation. The subset does not perform better on the score, as it had less variation across its scores. This is reflected in the score distributions, where the amount of 4 or 5 scores is almost 12% higher for RAG and 10% for the subset, even though the new architecture has about 9-10% more 5 scores.

In terms of friendliness, the new architecture achieves a higher average score compared to both, with a slightly higher standard deviation. This indicates a generally more friendly interaction from the new model, though with a bit more variability. We can see this in the percentages as well, where RAG does have about a 3% and the subset about 1.5% higher percentage of 4 or 5 scores compared to the new method. While the new architecture has over 20% more 5 scores, it also has more low and neutral scores.

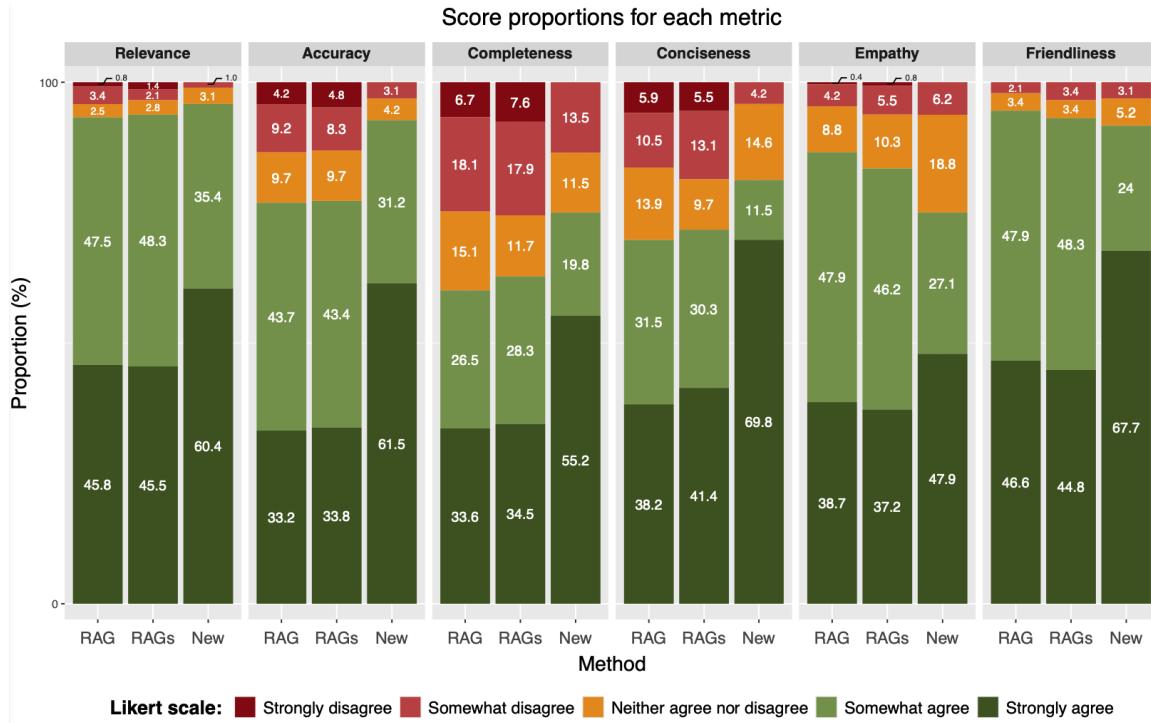


Figure 4.3: The score proportions for each of the metrics between RAG, RAGs using the subset of questions, and the new architecture.

4.3.3 Templates and the handover module performance

In Table 4.12, we see the average scores based on the templates and the handover module. These scores were calculated by removing all instances where the handover module intervened and then calculating the averages. The results show that the template enhancements alone improved on all metrics compared to the original RAG method (subsection 4.3.2), especially on accuracy and completeness.

The handover module intervened in about 28% of the responses, 77% of these were from the specificity check, where the response was rewritten to a suggestion. In 17% of the cases, extra information was requested, and on the next turn, the answer was given. In 6% of the responses, the bot deemed the question to not be relevant for the question. In Table 4.12, we see that the specificity check had the highest scores on most metrics compared to the other checks, except relevance, but the relevance score was still high. Accuracy and conciseness also received high scores for this check, with completeness and conciseness not far behind. We see that participants found that the relevance was really good for the first relevance check, even though the other metrics did not do as well. The second relevance check performed the worst for all metrics.

	Check 1 (Rel)		Check 2 (Rel)		Check 3 (Speci)		Templates	
	\bar{X}	σ	\bar{X}	σ	\bar{X}	σ	\bar{X}	σ
Relevance	4.75	0.50	4.00	1.41	4.68	0.57	4.47	0.82
Accuracy	4.25	1.50	4.00	1.41	4.68	0.48	4.44	0.89
Completeness	4.00	1.41	3.5	0.71	4.27	0.99	4.12	1.23
Conciseness	4.25	1.50	3.5	0.71	4.64	0.73	4.32	0.93
Empathy	3.5	1.29	3.5	0.71	4.04	1.13	4.25	0.99
Friendliness	4.24	0.92	3.5	0.71	4.46	0.96	4.59	0.86
Total	4.21	1.14	3.67	0.778	4.46	0.86	4.37	1.04

Table 4.12: Average scores for the conversations based on the enhancements, green being the highest and red the lowest score. The template scores were for when the handover module did not intervene, and the other three when the handover module did intervene given the check. More detail about these checks can be found in subsection 3.4.3.

4.3.4 Initial scenarios

We will now take a look at the individual scenarios the participants were given, to find out how each scenario was handled by the new method. In Table 4.13 we see that the worst-performing question still has an average score of 3.75, which is higher than before by 0.95 as seen in Table 4.6. It is, however, interesting that the worst-performing questions in this evaluation did quite well in the first evaluation. RAG in the first evaluation outperforms the current architecture by almost 0.5 in scenario 2, while the other two scenarios show similar scores. The best-performing questions also scored higher than before, with the top 3 having a higher average score than the highest scoring from the first evaluation. These findings are also present when we look at Appendix H with the best and worst questions based on the individual metrics. We do see considerable improvements compared to the results of Table F.2 in the scores for most metrics. Question 11, for example, showed perfect results on multiple metrics, except on empathy and friendliness. The rest of the scenarios generally varied between metrics, however, in general, the scores were quite high even for the worst-performing scenarios.

Top 3 Best given All Metrics			Original Questions			
Nr.	Scenario	\bar{X}	σ	Nr.	\bar{X}	σ
4.	A customer wonders if you can work with one account on multiple devices?	4.76	0.53	24.	4.12	1.25
1.	A customer would like to upgrade from a basic license to a premium license but wonders if this can be done without data loss?	4.67	0.97	18.	4.72	0.53
8.	A customer asks if a company logo can be added to the template so that it appears on the page instead of the Incontrol logo?	4.57	0.63	10.	4.00	0.91

Top 3 Worst given All Metrics			Original Questions			
Nr.	Scenario	\bar{X}	σ	Nr.	\bar{X}	σ
15.	A customer shared a draft with a colleague but can no longer find it on their iPad.	3.75	0.99	22.	3.72	0.79
2.	A customer noticed that when mentioning an email address in a text element and/or an automatically generated email, the email address gets truncated. Everything after the @-symbol is not displayed.	4.25	0.97	12.	4.72	0.53
6.	A customer wonders if a back-office connection can be made with Incontrol.	4.26	0.89	10.	4.23	0.98

Table 4.13: The top three best and worst performing scenarios based on their average scores given all metrics. As a comparison, we also provided the RAG scores, from the first evaluation, of the respective questions on which the scenarios are based.

Chapter 5

Discussion

We will now further discuss the research that was done, along with the obtained results. First, we take a look at the first phase of the thesis and discuss its implications in section 5.1. Following the evaluation of this phase, the best-performing method was further refined to address observed limitations by making adjustments to the architecture, leading to a second phase, discussed in section 5.2. We will discuss the outcome of the second phase and compare it to the results of the first phase. Lastly, will we discuss some limitations in section 5.3, have a final conclusion to the research including our recommendations in section 5.4, and cover potential future areas of research in section 5.5.

5.1 Phase 1: Research

In this section, we cover the first main research question:

How do few-shot learning, retrieval-augmented generation, and fine-tuning compare in enhancing the relevance, accuracy, completeness, conciseness, empathy, and friendliness of responses generated by Large Language Models in customer support bots?

Along with the three sub-questions:

- SQ 1.1: How do few-shot learning, RAG, and fine-tuning differ in their ability to generate responses that are relevant and accurate to customer questions in a support chatbot?
- SQ 1.2: Which method among few-shot learning, RAG, and fine-tuning most effectively balances completeness and conciseness in responses generated by a customer support chatbot?
- SQ 1.3: In what ways do few-shot learning, RAG, and fine-tuning impact the perceived empathy and friendliness of a customer support chatbot's responses?

5.1.1 Main research question 1

This study aimed to compare few-shot learning, retrieval-augmented generation (RAG), and fine-tuning to determine which method performs best when it comes to context-specific language generation. This was done based on different key metrics such as relevance, accuracy, completeness, conciseness, empathy, and friendliness of responses generated by leveraging the different generation methods. Looking at the combined scores of each metric we see that RAG did perform significantly better overall compared to the other methods. If we then take a look at the performance given each metric, RAG was again shown to be the most effective method for generating responses, especially for the accuracy, completeness, and empathy metrics. Its ability to handle large amounts of context allowed it to generate responses that are not only accurate and somewhat complete but also empathic and friendly. These responses did result in longer response lengths, however, though human judgment did not seem to find that problematic.

Few-shot learning scores fluctuated more, as it was the second-best method behind RAG for relevance, accuracy, completeness, and conciseness while being the worst at empathy and friendliness. Its responses often lacked phrases such as 'That is unfortunate to hear...' or '... If you have other questions, feel free to let me know!'. This resulted in shorter responses compared to RAG, but could also have made its responses less complete, empathic, or friendly. Fine-tuning showed comparable results to the zero-shot model on most metrics, except for conciseness where it performed significantly better than zero-shot, however, it was not a stand-out method.

When looking at the overall scores of the individual metrics we see that completeness was the worst-performing metric, while the responses had generally high scores for friendliness. We also saw that while RAG was the best-performing method, the zero-shot method was generally the worst-performing method, except for the metrics empathy and friendliness as discussed before.

For the automatic evaluation methods, we see that either RAG (BERTScore, METEOR, ROUGE-l) or few-shot learning (BLEU, DeepEval, Word counts) had the highest scores. However, in most cases, except for DeepEval and Word counts these findings were not significantly better than the scores of the other methods. We also saw that the zero-shot model performed significantly worse for most metrics, except for DeepEval and word counts. It shows that the automatic evaluation methods do produce similar results compared to human judgments when it comes to rating generation methods, however, we saw in subsubsection 4.2.7.1 that there were no correlations between the human and automatic evaluation methods. This could be because automatic metrics might be giving higher scores to different conversations compared to human judgment as we saw was occasionally the case for DeepEval in subsubsection 4.2.7.2. So the automatic metrics might be able to differentiate between methods, however, they likely diverge in their assessments resulting in no apparent correlations. We delve deeper into the performance of automatic metrics in subsubsection 5.1.2.4.

5.1.1.1 SQ 1.1: Relevance and accuracy

Relevance and especially accuracy, from the first sub-question, are important metrics for customer support bots when evaluating the quality of responses. We used a relevance and an accuracy metric in the survey and also proposed different automatic metrics that could be used as a relevance metric. RAG outperformed zero-shot, few-shot learning, and fine-tuning in generating responses that were both relevant and accurate more consistently as evaluated by humans. [Lewis et al. \(2020\)](#) also confirmed that in general RAG enhances contextual understanding and accuracy in generated responses. While both RAG and few-shot learning leverage external knowledge bases during response generation, RAG's superior performance could be attributed to a smaller amount of pre-processing of the context compared to few-shot learning. Few-shot learning relies on a number of pre-processed examples from the context to fit the few-shot template. In contrast, RAG accesses the entire information context, enabling the LLM to handle more of the processing during response generation rather than during the pre-processing stage required for few-shot learning. Fine-tuning showed performance similar to the zero-shot model, which is likely due to it not inherently having external data sources incorporated, limiting the relevance and accuracy of its answers. Other studies did show that fine-tuning gpt-3.5 show great results in various domains such as Q&A systems ([Tan et al., 2024](#)), code review ([Pornprasit and Tantithamthavorn, 2024](#)), and response scoring ([Latif and Zhai, 2024](#)). The poor performance of fine-tuning in this study could potentially be explained by the limited amount of sources available for each problem. A dataset with more variations of the same problems could possibly help the LLM during the fine-tuning process, which in turn could help its responses to these problems.

Relevance of the responses is shown to generally be high, this is likely because LLMs tend to be good at staying in the domain of the question that is asked. We can see this, when looking at the responses from the zero-shot model, as it has no context whatsoever to go off, but still manages to respond in a way that is within the domain of the question and thus seems relevant in 75% of the cases. However, this doesn't mean that the given response is correct, which we can see if we look at the accuracy scores. The best-performing method RAG, while it is relevant in 93% of the cases, only has been given a 4 or 5 score in about 75% of the cases, which is a low score for a customer support bot. It shows that RAG generally is a good method of adding additional context to an LLM, but still cannot be used on its own for a customer support bot.

If we look at the automatic evaluation metrics for relevance, we see that none of them including BERTScore have correlations with human judgment, we will delve deeper into these findings in subsection 5.1.2. On the other hand, automatically assessing the validity of an answer is still an area that needs to be researched and will be discussed in section 5.5.

5.1.1.2 SQ 1.2: Completeness and conciseness

A conciseness and completeness metric was used in the survey to answer the second sub-question. While we argue that the relevance and accuracy metrics are slightly more im-

portant than these two metrics, they are still important to enhance response quality and user engagement. This is confirmed by [Cordero et al. \(2022\)](#) where the authors mention that providing clear and concise answers to user questions is important and by [Ngai et al. \(2021\)](#), who stated that evaluation of the completeness of answers is also one of the metrics to focus on. For both conciseness and completeness, we included some automatic metrics that could potentially correlate with human judgment. Balancing completeness and conciseness presented a significant challenge in chatbot response generation, as we see that both metrics are among the lowest scoring of the six according to human judgment. Especially completeness of responses showed to be quite difficult to obtain for every method. RAG was again the best-performing generation method for both metrics, even though few-shot and fine-tuning were not far off when it came to conciseness. Zero-shot showed the lowest scores, due to its lack of context, resulting in answers that might seem decent at first glance but ultimately lack key details. While RAG was the best generation method, its scores were still not very high. An explanation for this could be that while the LLM does do well in generating responses, it might overlook some details, resulting in an incomplete answer. Another possibility is the missing of these details in the context, as we chunk conversations into smaller bits, this could result in some of the necessary information being cut off during the chunking process.

For the conciseness scores, RAG possibly has the problem that the amount of information in the context resulted in a longer response as there is a lot of information available to be used. Few-shot learning could have these same problems as RAG for completeness, as well as the additional problem of the pre-processing step that could leave out some important details from the context when creating the few-shot prompt template. For conciseness, it is interesting to note that while few-shot receives lower average scores than RAG from humans, the average word counts were way lower. Few-shot had responses averaging about 59 words per response with RAG averaging around 91 words. The shorter word count in few-shot responses may also suggest why it had lower completeness scores. It is interesting that the example responses from each method in Table 4.1 did not entirely reflect these findings. In these responses, we see that fine-tuning resulted in the longest responses by far, with the zero-shot model and RAG being considerably shorter, and few-shot being the shortest by far. This is likely just an example where fine-tuning coincidentally had a long answer, as generally, its responses tend to be much shorter, while it was the opposite for few-shot, with its extremely short answer.

Based on the results, we concluded that perfect completeness meant addressing all necessary points within a response, even if the response included additional, non-essential details. We concluded that perfect conciseness, on the other hand, was shown to minimize unnecessary or redundant content, while focusing on brevity without necessarily including all necessary information. In other words, while a complete response includes all required information, it may also contain extra content, whereas a concise response prioritizes brevity by limiting the amount of information. Receiving perfect scores in both would mean that the response does contain all required information, without containing any unnecessary information, which would be a perfect response. These definitions of

completeness and conciseness can help in interpreting future results and guide improvements in response quality.

5.1.1.3 SQ 1.3: Empathy and friendliness

Lastly, for the third sub-question, we incorporated an empathy and friendliness metric in the human survey. While we think these metrics are slightly less impactful to the responses themselves, they are still important when it comes to user experience and user expectations as described in section 2.2. RAG scored highest in perceived empathy and friendliness, with zero-shot closely following it, especially for friendliness. The reason for this could be the nature of OpenAI's language models in combination with the instructions that were given to the LLM, to be a 'friendly customer support assistant'. Especially few-shot learning was lacking in this department, while receiving the same instructions, which could be due to its tendency to give shorter answers. During the fine-tuning process, we used a system prompt, to give instructions as to how we want to fine-tune it. Additionally, the model is also fine-tuned on customer support conversations, meaning the model adopts the style in which conversations are held, which are likely to be empathic and friendly. These personality traits get imbued in the LLM as confirmed by [Laney \(2023\)](#), who states that an LLM picks up the personality traits from the dataset. This is reflected in the results, as fine-tuning did reasonably well for both metrics, but not as good as RAG or the zero-shot model.

While the results do show good results on the metrics, especially for friendliness, some participants acknowledged that some responses were overly friendly. In [Wester et al. \(2024\)](#) the authors show that for mental health chatbots, overly friendly behavior might have negative consequences on the users, while on the other hand, the authors of [Schneider et al. \(2024\)](#) mention that a lack of emotionality in LLM responses, that otherwise show many human traits, might become eery and awkward. Even if these two studies were not specifically about customer support and therefore might not apply to this domain. They can still be important to keep in mind to potentially find a good balance between the two in a customer support bot.

5.1.2 Other discussion points

We will cover some other notable findings, such as inter-annotator agreement, participant experience, the analysis of initial questions, and the performance of the automatic evaluation metrics in this section.

5.1.2.1 Inter-annotator agreement

We saw that participants overall showed moderate agreement with each other. While a moderate α indicates some level of consistency among raters beyond chance, it also can raise some concerns about the reliability of the ratings. A reason for this level of agreement

could be ambiguous rating guidelines, which could have resulted in different interpretations of the metrics by participants, as people might have different ideas of metrics like relevance, empathy, and friendliness. Another reason could be that the 1-5 Likert scale had some influence on this. For example, two participants could have been in two minds giving either a 3 or 4, if one chooses a 3 and the other a 4, that could lead to variations between annotators.

We also saw that participant experience influences the α scores, which shows that very experienced participants generally do have more agreement with each other than less experienced ones. This could be because they have a more clear idea in most or even all scenarios what the score should be as opposed to less experienced participants for which this might vary, especially metrics like accuracy or completeness. We will talk more about participant experience in subsubsection 5.1.2.2. Addressing these issues through clearer definitions, changing the scoring scale, and refining the metrics could enhance agreement levels and strengthen the overall validity.

5.1.2.2 Participant experience

As presented in Figure 4.2, there was a noticeable trend where overall scores decreased with increasing participant experience regarding the software. The most knowledgeable group consistently assigned the lowest scores across most methods, indicating a more strict standard. This pattern is most apparent for the accuracy and completeness metrics. It suggests that their deeper understanding of the software helps them to identify inaccuracies or incomplete parts, that less experienced users might overlook. Overall, these findings imply that the perceived quality of chatbot responses is influenced by the user's level of expertise. More knowledgeable users may have higher expectations and be more strict in their assessments of responses. On the other hand, it is important to address how these varying levels of expertise impact different user groups, particularly those with less experience who might rely heavily on chatbot assistance.

Less experienced users might perceive chatbot responses as satisfactory because of their inability to detect slight inaccuracies or incomplete information, which can pose significant risks. These users may rely on incorrect or incomplete responses, leading to potential mistakes, confusion, and decreased user satisfaction and trust. Therefore, it is crucial to ensure that chatbot responses maintain a high standard of accuracy and completeness, even for users with lower experience levels, to prevent misinformation.

These implications can help to meet the expectations of employees and customers during the development and evaluation of customer support chatbots by highlighting the need to consider the different experience levels. Ensuring that chatbots provide reliable and accurate information for all user expertise levels can help maintain trust and effectiveness in customer support interactions.

5.1.2.3 Analysis of the initial questions

Taking a look at the individual conversations provides further insight into which questions were answered well and which were not across the different methods. This analysis highlights that some questions are more challenging for the chatbot to address accurately. These may involve complex issues, ambiguous language, or specific knowledge that the models might not have.

If we first look at the results given all metrics in Table 4.6, we see that questions that are straightforward or frequently encountered are generally easier for the methods to handle effectively, as the answer will be less complex or ambiguous. Questions 10 or 18 are examples of these straightforward questions and we see that they are doing well as there are basically two options, 'yes, ...' or 'no, ...'. The binary nature of these questions simplifies the response generation process, as the answers are more likely to be explicitly present in the context or underlying data. This reduces the complexity and ambiguity that might be present in more complex questions, which helps with generating more precise and accurate responses efficiently. Questions 5 or 2 are examples of more nuanced questions, and therefore, are more likely to be answered wrongly without any extra information, as there are many possibilities as to what might be the problem. In this case, the different methods gave straightforward answers as if there is just one possibility, however, the response might not be the appropriate response in that case. We think these responses are not necessarily hallucinations (subsection 2.1.5), as they are likely one of the many options that could be applied in that case, but they can lead to misinformation. These problems could have resulted in the generally lower scores for every method on these questions.

When we only look at the individual metric scores in Appendix H, we see a lot of the same best-performing and worst-performing questions between metrics. In general, again questions with a more binary 'yes, ...' or 'no, ..' answer did well for most metrics, while more complex questions performed worse. Question 18 was one of the best-performing questions in all metrics, while question 5 was one of the worst. We see for completeness that only question 18 got a good score, while all other questions had scores quite significantly below 4. This is likely due to the fact that question 18 does not need much information to be complete. Question 12 for accuracy is an interesting case, when we look at Figure G.2 we see that for few-shot and RAG, this question did well, with lots of high scores. While for both zero-shot and fine-tuning, this was the complete opposite. This question was specific to a feature that is present in the software, so it makes sense why the zero-shot model performed poorly on this question. Fine-tuning did even worse, which is likely because this feature was not learned in the LLM during the fine-tuning process. We think this is why the response of fine-tuning to the question was that 'it was an error in the software, which will be solved in the future'. While we did filter out bug reports or questions about bugs, there might have been a conversation that was about an error or bug in the system, which could have influenced this answer. Another possibility could have been that all the conversations about bugs were filtered out, however, the fine-tuning model did not 'know' what the answer was. It therefore could have hallucinated this response, which possibly could have been correct if it was another software program.

Understanding which questions are problematic can help implement specific improvements in the chatbot's templates and architecture. By taking a close look at the types of questions that are often answered poorly, we can enhance the model's ability to handle challenging queries, thereby improving overall performance. For example, we can provide more focused and detailed instructions about challenging scenarios to the LLM in order to make it more capable in the scenarios where it previously struggled. It is, of course, not possible to cover every specific challenging scenario at once, however, capturing the most challenging scenarios can help the development greatly.

5.1.2.4 Performance of the automatic metrics

The results of the Spearman test between the human and automatic evaluation metrics did not show any significant correlations. In contrast to our findings, other studies did report correlations between certain automatic metrics and human evaluations. For instance, [Zhang et al. \(2019\)](#) found that BERTScore correlates with human judgments of relevance in dialogue systems, and [Chen et al. \(2019\)](#) observed that METEOR scores correlate well with human assessments in Q and A systems.

A reason for these findings could be the differing evaluation methods between human evaluators and automatic metrics. Human evaluators assessed the entire conversation, which could capture more context and potentially relevant information, whereas automatic metrics were applied to question-and-answer pairs between the first customer-support interaction as explained in subsubsection 3.3.3.2. These Q&A pairs only contained the initial customer question and initial support answer, without any follow-up interactions during the automatic evaluation process.

Additionally, the definitions of evaluation criteria such as relevance, completeness, and conciseness may vary between human judgments and automated calculations. Humans could have interpreted relevance, completeness, and conciseness in a more subjective and context-aware way, whereas automatic metrics relied on specific algorithmic measures that likely did not fully encapsulate these aspects, which resulted in low correlation scores.

5.1.2.5 General experiment choices and findings

The conversation creation process for the first evaluation process was quite challenging to execute. This was due to the fact that we couldn't predict what each of the generation methods would return while trying to stay consistent between methods. In Table 4.1, we see an example of this, where we tried to create logical conversations without trying to vary questions and contents too much between methods. For the zero-shot model and RAG, we used the same two questions, and for few-shot, we only had to ask one question as it already stated if it was available for all subscriptions in the first response. For fine-tuning we first had to ask the extra question: 'if it was possible within the software'. If we hadn't asked this extra question then the conversation would not have made sense. This is because the response mentioned an external application, therefore, asking about the subscription within the current software would not have been logical. So while we

based the conversations on past support interactions, the generation methods sometimes didn't follow the same structure. We, along with a support employee, sometimes had to improvise the follow-up questions like in the example. We still believe these conversations turned out well, however, in the future another method might be more efficient. Especially for larger-scale evaluations, because the process of hand-crafting conversations was very time-consuming and would not scale well.

5.2 Phase 2: Research

Based on the results and discussion, we propose an architecture that should give improved responses. The results show that RAG performed the best for all metrics, as well as in most of the initial questions, therefore, the revised architecture revolves around this generation method. However, some of the templates used with RAG were slightly modified by including the results of the other methods. The architecture also incorporates a handover module that hands the user over to a customer support assistant, when needed.

5.2.1 Template enhancements and handover module decisions

We will discuss some insights, in subsubsection 5.2.1.1, from the other generation methods, when they performed better based on human evaluations compared to RAG for certain questions, to figure out why this was the case. We will also make some enhancements to the RAG templates based on the insights from the first evaluation in subsubsection 5.2.1.2. Lastly, we will discuss some decisions made for the handover module in subsubsection 5.2.1.3.

5.2.1.1 Instances of superior performance by other methods

Despite RAG's dominance in most areas, specific questions revealed scenarios where zero-shot, few-shot learning, and fine-tuning methods outperformed RAG on each of the metrics. The questions discussed in this section can be found in Appendix B, with their scores for different metrics found in Appendix G.

In question 19, the zero-shot model outperformed RAG by delivering a more concise and accurate response. The zero-shot's straightforward approach, without any context, was mostly for predictable and procedural queries, such as those about license expiry or account upgrades. This was due to the fact these queries are generally solved in similar ways and were not company-specific. By avoiding unnecessary explanations, the zero-shot model provided clear and direct answers that aligned precisely with the user's needs, minimizing the risk of inaccuracies. For several questions, including question 19, zero-shot also outperformed RAG in both empathy and friendliness. This was primarily due to zero-shot's incorporation of greetings and acknowledgment phrases at the beginning of these responses, which helped with creating a personal and engaging tone.

For questions 4 and 14 few-shot learning surpassed RAG on relevance, accuracy, and conciseness, with completeness only being higher for question 14. Few-shot learning provided short and to-the-point responses that directly addressed the user's concerns without unnecessary elaboration. On the other hand, RAG's more detailed answers included additional, unnecessary information that worsened the relevance and accuracy of its responses, making RAG less effective in these contexts. Especially for question 14, where few-shot learning's answer was extremely concise, which is reflected in the scores being 5/5. While this response is really short it shows that it does cover the most important details needed to answer the question, considering the relevance, accuracy, and completeness scores for that question. We do see, however, that the perceived empathy and friendliness are considerably lower for this short response.

When we look at questions 3, 6, 8, and 25 the fine-tuning method outperformed RAG by delivering responses that were in general, similar or better in relevance, more accurate, and more concise. However, for question 3 and question 25, the responses weren't as complete as that of RAG and were in general also less empathic and friendly. It seemed that in cases where fine-tuning does provide more accurate answers compared to RAG, it generally gives more relevant and concise responses. However, fine-tuning often lacked in the accuracy department compared to RAG, resulting in mediocre responses all around.

5.2.1.2 Response template enhancements

We implemented template refinements based on weaknesses of RAG, that we identified, during the process of the first evaluation. These refinements were implemented to enhance RAG's performance across all metrics. In subsubsection 5.2.1.1, we have seen multiple instances of other methods performing better mostly on accuracy, and conciseness, but also on relevance and completeness compared to RAG. These instances helped us get a better understanding of user preferences for responses, why one response was scored better than another during the refinement process, but also how to balance responses on different metrics.

To improve relevance and accuracy, the templates were explicitly designed to utilize only the provided context for generating answers as we noticed that RAG sometimes incorporated unnecessary external information in its responses i.e. 'ONLY the context should be used to generate answers do NOT make anything up'. We also made the decision to prioritize information from the help center articles when available, to ensure high accuracy. As help center articles were guaranteed to be accurate and up-to-date i.e. 'Prioritize information from articles over that of emails and tickets'.

For conciseness, we had to make sure to guide the LLM into not giving unnecessary information and focusing only on responding using the essential parts, without going beyond the scope of the question. This is to ensure responses are brief and directly address user queries without overcomplicating the answer i.e. 'Do NOT give unnecessary information and try to be concise, only answer using necessary information and don't go beyond'.

For improvements to completeness, we tried to enhance the template by instructing

the LLM to slightly prioritize content from a single reliable source over a combination of multiple sources i.e. 'If applicable prioritize contents of one source over a combination so all the necessary information in that source is used'. This is to make sure that a potentially complete answer present in a single source is more likely to be used than a combination of different sources. This combination, using part of multiple sources, might not present the whole picture, by leaving out some parts of a certain source that might have been useful. We noticed this on some occasions, where the information from multiple sources was combined in an incorrect way. This resulted in responses that were less accurate and incomplete, as seen in Table K.1. It is an example that occurred during testing, of how context was combined incorrectly. It is incorrect because the user asks specifically if a form can be built for him, and the response does inform that they are able to help build one using the first source. It also informs the user, using the second source of the context, about pre-made templates for inspections, which wasn't the question. It should have just used all the information from the first source without using the second one in order to give a correct response. It does not mean that a response from different sources cannot be complete, but a slight nudge towards using more information from a single source we think could help with completeness.

We tried to improve friendliness and empathy by maintaining a consistent response structure that begins with acknowledging or addressing the user's problem, providing an answer, and offering the opportunity for further questions, thereby fostering a personable and engaging tone i.e. 'For responses ALWAYS maintain the following structure: acknowledge the problem, answer it, tell them that if needed they can ask more questions'.

Using the insights of some few-shot instances we tried to find a balance between conciseness and friendliness, as we found that a response that is too concise generally reduces the empathy and friendliness of the response. We also noticed that in some instances RAG tried to give as much information as possible resulting in part of the information becoming unnecessary. In these instances, the other metrics often performed better on accuracy and conciseness as a result of this, even if RAG was more complete in some cases. Giving more straightforward and, therefore, sometimes less complete answers, like the other methods did in those cases, could help RAG improve accuracy and conciseness. While the answer might not be entirely complete on its own in that case, a follow-up question and response could help with this.

Lastly, some extra considerations were the exclusion of links from the emails or tickets as they might be outdated or expired, and the exclusion of placeholders found in the context. The template also contained the instructions for responding in a certain language and identifying themselves as a customer support bot. The entire response template used for the second phase can be found in Table L.1. The chat history template was also slightly modified to be more in line with the other instruction templates and can be found in Table L.2.

5.2.1.3 Handover module decisions

We fine-tuned the templates, by continuously changing, refining, and testing parts of it, in a way that balanced confidence with the accuracy of the answers to our liking while also being as relevant as possible. While we tried to enhance the templates as much as possible they might be lacking in some areas, these limitations will be discussed in subsection 5.3.2.

We also considered using the LLaMa-3.1-70b model to evaluate relevance to prevent an LLM from evaluating itself, as we did for DeepEval, however, after some testing we found the results to be somewhat lacking. It was less consistent and accurate in the checks compared to the gpt-4o, which is why we chose the potential disadvantages of self-evaluation over the inconsistencies or inaccuracies in evaluation. For specificity, we also chose to use gpt-4o as this check was not really about self-evaluation but more about recognizing a question that is too specific and rewriting it, so we stayed consistent with the rest of the architecture and chose gpt-4o.

5.2.2 Main research question 2

The second phase aimed to compare the best-performing method of phase 1 and enhance the performance of this method by incorporating findings from the results as well as incorporating a handover module in the architecture. To do this, the second research question was:

How does the implementation of the best-performing method with template refinements, based on the findings from RQ 1, in combination with a handover module, affect the relevance, accuracy, completeness, conciseness, empathy, and friendliness of responses in customer support chatbots?

5.2.2.1 Response template performance

The results of the second evaluation showed that the enhanced template did improve the response quality based on all metrics and especially on accuracy and completeness compared to the first evaluation as seen in Figure 4.3. Friendliness is still the best-performing method, with relevance and accuracy following closely behind. While completeness received the lowest scores again it was improved by about 0.5, scoring fairly well, however, this might still be an area of focus in future enhancements.

It was, in general, fairly hard to trace back which enhancements to the templates resulted in which improvements. However, we did notice that participants liked a more consistent and structured response, explained subsubsection 5.2.1.2, which could have resulted in higher scores on both empathy and friendliness. The more relevant and accurate responses could be attributed to the change of only using context, or maybe because articles are prioritized over the other sources. The instruction of giving a more concise answer could have maybe influenced the accuracy and relevance as well because unnecessary information was left out. It is important to note that while responses were generally more

concise the completeness was still higher than before, which could be because even though one answer might not be complete on its own, a follow-up question could have resulted in the complete answer.

These findings do show that refining instruction templates is a crucial way of improving responses. Further experimenting with the templates could, in the future, yield possibly even better results.

5.2.2.2 Handover module performance

The results showed that the handover module did intervene in almost 30% of cases. These were mostly from the specificity check found in Table M.3, which is likely because most of the ‘hard’ scenarios were too specific for a direct response. The first relevance check, which checks for relevance of the response also occurred a few times, which could be explained by how the questions were asked, which might not have been clear. An example of this can be found in Table M.1, where the user accidentally asked an incomplete question and the bot responded appropriately, using this check. The second relevance check occurred the least which made sense as participants generally tried to ask questions in the context of customer support. However, in Table M.2 we see an instance where a question was deemed not relevant. This check might become more crucial when the bot is publicly accessible. This is because participants did try to focus on asking relevant questions, however, this is not guaranteed when a bot is available to everyone.

Furthermore, we also see that the specificity check showed comparable or even better performance than the new architecture in general if we look at the individual metrics. This can indicate that this check does increase the performance of the new architecture on various metrics like relevance, accuracy, completeness, and conciseness. We also see that the first relevance check scored worse on most metrics compared to the specificity check and the general architecture, however on relevance, it outscored both of them. This is interesting as the information itself is not relevant to the question, because it just asks for more information, however, humans still thought that this response was relevant. This also ties back to how humans and automatic metrics differ in their perceptions of relevance. We see that the second relevance check performs the worst on all metrics. This makes sense as the response does not try to help the user but instead states that it cannot help them with that request. This response would likely not be satisfying to a user, especially if they believed their question to be relevant. So while we argued that the second check could become important for publicly available support bots, this check should not be overly present for users to maintain customer satisfaction. It would be annoying to constantly be told that the question cannot be answered, even though you have a genuine question.

5.2.2.3 Overall performance

The results show that overall both the template refinements (subsubsection 5.2.2.1) in combination with the handover module (subsubsection 5.2.2.2) performed considerably better on the metrics relevance, accuracy, completeness, and friendliness, while empathy showed

slightly lower scores compared to the previously best-performing method RAG. Especially for the metrics accuracy, completeness, and conciseness, we saw improvements of over 0.5, indicating that the quality of responses has increased significantly. When we took a subset of the questions on which the scenarios were based, we saw that the scores were very similar to the normal RAG scores so the findings of the new architecture were very comparable. Only in the case of empathy were the results different as we saw that RAG was more empathetic compared to the new method, while this was not the case when we took the subset of questions. Showing that the taken questions for the scenarios were, in general, given slightly lower scores. Overall, the total average score for the new method is considerably higher than RAG and the subset, with also a lower standard deviation across all metrics. So even though these results cannot be put next to each other with statistical tests, they still indicate some positive findings regarding the new method. Completeness is, just like for RAG and the taken subset, the lowest-performing metric along with empathy. However, while friendliness is still the best-performing metric, the difference between other metrics, such as relevance, accuracy, and conciseness, has shrunk considerably. It is also interesting that none of the participants ever gave a rating of 1 (strongly disagree) for the new method.

5.2.2.4 Inter-annotator agreement

In the second phase, we saw that participants overall showed poor agreement with each other. A low α indicates that the consistency among raters is barely above what would be expected by chance, raising concerns about the reliability of the ratings. The same reason we gave during the first phase can be applied here, however, there could be another reason for these even lower scores, which is how we conducted the second evaluation. Participants were free to interact with the chabot itself, which could have resulted in different responses between participants, therefore, showing lower agreement scores. The even fewer amount of participants could have also influenced the inter-annotator agreement. While these results do raise concerns about the validity of the scores and therefore, of the entire second evaluation, we think that the findings could still be useful. Mostly for gaining insights into the overall performance of scenarios compared to the first evaluation, to see if there were some differences, even if the results didn't have as much value as the ones from the first evaluation.

5.2.3 Other discussion points

We will cover some other notable findings, like the analysis of the initial scenarios, during the evaluation of the second phase in this section.

5.2.3.1 Analysis of the initial scenarios

Taking a look at the individual scenarios created by us can provide further insight into the performance of each scenario just like for the first phase.

We see in Table 4.13 that, given all metrics, there are still scenarios that are more challenging than others, although the differences between the best-performing and worst-performing scenarios have become smaller. We still, however, see that more straightforward scenarios are receiving higher scores, compared to more complex ones. The top 3 best-performing questions are all basically 'yes' or 'no' questions again showing that the new architecture is still able to handle these well. As for the lowest-scoring scenarios we see that scenario 5 is again a more ambiguous problem, where the architecture might not give the best response. However, we do still see an improvement in scores, which can likely be attributed to the specificity check from the handover module. This check could have helped by turning the response into a suggestion. For question 12 the problem that participants encountered was that the architecture sometimes returned a suggestion for this response instead of a straightforward answer. In this case, however, it could have given a direct response as the problem is the result of a feature in the software. When we look at the handover module and its evaluation we see that the original response did give a direct answer to the question but as the specificity check thought it was about a specific problem it rewrote it to a potential suggestion. We do see that participants did not mind this too much as the scores were still relatively good, but we should keep in mind that these faulty evaluations by the handover module do occur.

For the individual metric scores of the conversations we see in Appendix F, we see that the overall scores have become considerably higher compared to before. For the metrics relevance, accuracy, completeness, and conciseness the best-performing scenario was 1 based on question 18, as was also the case in the first evaluation. What is interesting, however, is that the rest of the highest-scoring scenarios vary between metrics. The scores of the lowest-performing questions were still relatively high for all metrics, especially compared to the first phase. What is interesting is that some scenarios like 2, 3, and 5 which were considered 'easy' based on the first evaluation, were among the lower-scoring ones on some metrics. An example of this is scenario 2 based on question 12 which had one of the highest scores in the first evaluation for RAG, while it received on average about 0.5 lower given all metrics in the second evaluation. We saw that the biggest problem with these scenarios like scenario 2, was the way in which the question was asked by the participant. This observation will be explored in more detail below in subsubsection 5.2.3.2.

The insights into how these scenarios performed, and what strengths and weaknesses the new architecture has, could be beneficial for developing and enhancing future iterations. Especially comparing these to before the enhancements shows what the influence of the enhancements was and if they worked. If there are still scenarios in which the chatbot struggles you can try and implement enhancements based on the scenarios, evaluate them, and compare the results to before. To eventually get an architecture that is reliable and accurate in every scenario.

In our case, we saw that the new architecture does a lot better overall compared to phase 1, however, was not yet performing perfectly. We still noticed some problems with the handover module and how to balance confidence with accuracy, as well as some problems with the retriever that did not always retrieve the most relevant sources given a mod-

ified query.

5.2.3.2 General experiment choices and findings

We intentionally did not look at response times, during the chatbot interactions of this phase. While the response times were generally slightly longer compared to the first phase (around 4-5sec in the first phase vs. around 5-7sec in the second phase), because of the extra LLM calls in the handover module. We still think the slightly longer response times are not problematic, as it was only marginally slower compared to the first phase. We also added the phrase 'Typing...' during the generation process, to let the participants know that an answer was being composed.

We also did not take response costs into consideration during the research, however, this is still a notable area, especially for companies. If we look at the average costs of responses from the first phase (\$0.0028) compared to the average costs of responses from the second phase (\$0.0052) we see that costs have almost doubled. This is mainly due to the extra costs that came with the evaluation steps from the handover module, so for companies, the consideration between response quality and costs could be made.

In the evaluation of the second phase we saw that, during the chatbot interactions, participants sometimes got sub-optimal results, due to the way that they formulated their questions. We gave the participants instructions about what scenario they were in, however, how they asked the questions was completely up to them. We noticed that some participants gave some unnecessary information in the questions, that were in line with what they had experienced during actual customer interactions. The new architecture sometimes had problems with this extra information and how these questions were formulated. The retriever, therefore, retrieved less relevant documents, as the relevant context of the participant's questions wasn't fully captured, which ultimately resulted in less accurate answers. This is something we should keep in mind as each user has their own way of asking the same question. It is important to ensure each of them receives a similar response to questions about the same problem, even if asked in different ways. Something we could try is to use an LLM that rewrites questions in a more general way to make the variation between questions more standardized. However, a possible disadvantage could be that this rewriting results in less personalized and more static user experiences.

When participants, during chatbot interaction, wanted to ask a follow-up question, we saw that the follow-up response sometimes became quite repetitive. This was both for the general response generator and also when the responses were caught by the handover module. For the relevance check, this was logical as we gave instructions, seen in Table L.3, to ask for more information if the response wasn't relevant, which would result in repetition if multiple irrelevant interactions occurred. However, for the response generator and the specificity check the repetitiveness became somewhat annoying according to some participants, an example of this can be found in Table N.1. This is understandable, as receiving almost the same answer twice, while they expect something else, didn't feel helpful. We also saw that the follow-up response sometimes lacked relevance, because the rephrased

standalone question did not capture the history and the follow-up question completely. Some modifications to how we try to capture the chat history might be needed to keep responses relevant during longer conversations. We will propose some recommendations on how to prevent repetitive responses and a possible new way of capturing the chat history in subsection 5.4.1.

We intentionally did not include the experience levels in the second evaluation as we had even fewer participants compared to the first evaluation. We therefore thought that any of the comparisons would not be very useful because of this small sample size.

5.3 Overall limitations

While this study provides valuable insights into the comparison of few-shot learning, RAG, and fine-tuning in enhancing customer support chatbots, several limitations must be acknowledged. These limitations may have influenced the results and interpretations of the findings, which is why they should be addressed.

5.3.1 Data limitations

The quality and structure of the knowledge base data can significantly impact LLMs' performance. In this study, while the data preparation was rigorous, the process could have been even more rigorous to ensure the most optimal model performance. We noticed that there are still many sources that contain uninformative or outdated information, during the testing and refining we removed sources like these, however, there are still a lot of these sources present in the data. These could muddy the context during the retrieval process, potentially resulting in responses of lower quality. The database also needs constant updating to ensure the information stays relevant and applicable. Currently, only new tickets are added to the database, which is part of the maintenance for the database. However, it could be the case that a feature might not have been possible in the past when an update to the software resulted in the feature being present. Sources in which the feature was said to not be possible should then be removed from the database as these would have become untrue. This dynamic updating of the database is a task that has not yet been implemented in this thesis.

Another limitation of this study was the size of the dataset used in this study being relatively small, which could have affected the generalizability of the findings. We noticed that the retrieval of relevant documents depends on how the question is asked, if there are many different sources available in the dataset that contain relevant information the chances of retrieving such a document can become substantially higher, even when the questions are asked in multiple different ways. Expanding the dataset to include more varied and extensive customer interactions could therefore be a way to do this. It is, however, important to consider that the additional data is of high quality as we have discussed before, to make the context as relevant as possible.

5.3.2 Limitations in response generation

Effective chunking of text data is essential for retrieval-based methods like RAG to function optimally. This is because you don't want to give the LLM too much context with irrelevant information, as this could result in relevant parts being overlooked while some other less relevant parts are used instead by the LLM. Even if chunking is an important part of the current approach, it may still have limitations in capturing the full context or nuances of the conversation between customer and conversation. It could be the case that a relevant source has been split up where part of the answer is in one chunk while the other part is in another. If only one of the two chunks is retrieved during retrieval then you might miss out on some key information, which could impact the quality of the response. In this thesis, we tried many variations of chunk size and overlap to minimize the amount of irrelevant information while still capturing all relevant information, however, this doesn't mean that the chosen values were the optimal ones. Implementing more sophisticated chunking techniques, such as sentence window retrieval ([Eibich et al., 2024](#)), could enhance the source retrieval process and the model's ability to generate coherent and contextually appropriate responses.

We generally see that the quality of responses from the response generator largely depends on the quality of the retrieved documents, as RAG heavily relies on the context to generate the answers. This does mean that in the current architecture quality of the query handler and its ability to retrieve the most relevant sources is detrimental. In subsubsection 5.2.3.2 we discussed some occasional problems with the query handler, these problems can eventually lead to some limitations during response generation. We did keep these limitations in mind during the creation of the architecture and tried to make the query handler as good as possible, however, the strong reliance on the query handler should still be considered during future development.

In this study, we used the most recent and advanced models of OpenAI to improve the quality of responses and stay up-to-date, so generally when a new model was available we switched to that model. For fine-tuning this was the earlier gpt-3.5, while for the other methods, we used the more recent and improved gpt-4o models. The reason we used an older model for fine-tuning was that at the time of performing the fine-tuning, this was the latest model available. Later on during the research, the gpt-4o models were also made available for fine-tuning, however, due to time constraints we couldn't use these models anymore. Fine-tuning was generally one of the worst-performing methods, so these newer models could potentially yield different and improved results. OpenAI also stated in their fine-tuning documentation that fine-tuning gpt-3.5 could give results on par with or even better than gpt-4o depending on the use case, so fine-tuning gpt-4o could show even better results.

Alternate designs and constructions of response and instruction templates can lead to different results ([Mizrahi et al., 2024](#)). The role that these templates play in both the chatbot's output and the evaluation of the handover module should, therefore, not be underestimated. In this study, while we tried to iteratively optimize the templates, they may still need enhancements to become even more effective. Testing, refining, and enhancing

the templates can improve the performance of the response generator and handover module, as prompt engineering is crucial for LLM interactions ([Ekin, 2023](#); [Wang et al., 2023b](#)). Future research should focus on iterative template development and testing to identify the most effective structures for different types of customer interactions.

5.3.3 Experimental limitations

The number of participants involved in both of the human evaluation phases was insufficient to capture a wide array of user experiences and perceptions. A larger participant pool would provide more accurate and statistically significant insights into the effectiveness of each method. Additionally, a wide range of participant age groups and more balanced expertise levels would give a more diverse and nuanced view of the results and might reflect the customer end better. Future research could involve a greater number of participants not only from the company but also from customers to enhance the validity and accuracy of the evaluation results.

The study employed a 1-5 Likert scale for measuring participant responses, which has inherent limitations, such as the potential for response bias and the assumption of equal intervals between scale points. These limitations might affect the precision of the measurements and the interpretation of the results. As we have discussed earlier in subsubsection 5.1.2.1, this could have also been a reason for the lower inter-annotator agreement scores. ([Lozano et al., 2008](#)) showed that a larger Likert scale from 1-7 could prove to be better and give different results during evaluation, therefore, this change could be considered in future research.

5.4 Conclusion

In this thesis, we aimed to enhance the performance of customer support chatbots through a structured two-phase approach.

First, we evaluated different response generation methods, including a zero-shot model, few-shot learning, fine-tuning, and retrieval-augmented generation (RAG). We showed using a comprehensive evaluation that while RAG demonstrated superior performance across all key metrics: relevance, accuracy, completeness, conciseness, empathy, and friendliness, according to human judgment, the method still encountered problems. There were some scenarios where alternative methods outperformed RAG or where there was no method that performed well. Cases where other methods were better included some scenarios where few-shot often had more concise answers, albeit not very empathetic or friendly because of this. Fine-tuning performed in most cases worse than RAG, however, in the few scenarios it did have a correct response it often gave a better answer than RAG.

Secondly, we built on the insights gained from the first phase and proposed a complete architecture that integrates RAG with additional enhancements such as template refinements and a handover module, to mitigate the identified limitations of RAG. We improved the relevance and accuracy of responses by enhancing the LLM instruction template. We

did this by ensuring responses were generated only using information from the provided context and by prioritizing information from verified sources. Additionally, we included guidelines to maintain conciseness by leaving out unnecessary information and structuring responses to increase friendliness and empathy. The implementation of the handover module was proposed to manage irrelevant, complex, or out-of-scope queries that RAG did not handle effectively. By performing three key checks, these being relevance, contextual relevance, and specificity, the handover module ensured that these queries were first evaluated on their relevance and if needed rephrased to become a suggestion instead of a straight response. This approach tried to mitigate the risks of overconfidence and irrelevant responses to enhance overall user satisfaction by providing accurate and useful responses.

The improved templates and handover module showed great improvements on almost all key metrics except empathy. It showed that the new architecture was successful in providing higher quality responses, especially on accuracy, completeness, and conciseness with improvements over 0.5. Even if the validity of the results can be taken into question due to high inter-annotator differences, we still think that moving forward with continued optimization and refining of these two strategies could help further elevate the effectiveness and reliability of AI-driven customer support chatbots.

In conclusion, this thesis successfully demonstrated RAG is the most promising technique for creating context-specific chatbots. We also showed that we can increase the performance of an RAG-based architecture by combining template refinements with a handover module. We will give our recommendations of an architecture below, to further refine and enhance the chatbot structure proposed in this thesis.

5.4.1 Our recommendations

Based on the findings of this research, the following recommendations are proposed to enhance the architecture, and therefore the quality of the chatbot responses, demonstrated in this thesis:

- Developing and maintaining a high-quality up-to-date database is the first important step to providing accurate and reliable information. We recommend only keeping the most relevant sources in the database, and removing sources when they become outdated. This could potentially be done with keyword matching, where each database entry has numerous keywords describing it. When a new entry gets added the already present sources with similar keywords could be analyzed to determine if they should be kept or removed. Having the highest quality database is essential for the chatbot to generate accurate and relevant responses consistently.
- Utilizing other chunking techniques like sentence window retrieval can enhance the relevance and accuracy of responses by ensuring that the most useful information is retrieved as well as reducing the amount of context that is fed to the response generator, thereby improving the precision of the chatbot's responses.

- Implementing a history check system that prevents the chatbot from repeating answers could enhance the user experience by providing diverse and contextually appropriate responses. This mechanism ensures that interactions remain engaging and that users do not encounter redundant or repetitive information, as would sometimes happen with the current architecture. Where users tried to ask a follow-up question and the chatbot returned almost the same answer. Implementing such a check could help maintain the chatbot's effectiveness and user satisfaction.
- Continuously testing and refining the response template and templates in the handover module can be vital for maintaining and improving the chatbot's performance. Evaluating the performance of different templates to see how they perform and what can be improved are areas that we recommend exploring. Enhancements based on these evaluations ensure that templates remain effective and adaptable to evolving user needs.

These recommendations can help organizations develop effective and efficient customer support chatbot systems that harness the strengths of LLMs while mitigating their limitations.

5.5 Future work

While this study provides significant insights into the comparisons between few-shot learning, RAG, and fine-tuning methods as well as enhancing the most promising method, several areas of research remain unexplored. Addressing these areas in future research can further help optimize chatbot performance, enhance user satisfaction, and expand the research of AI-driven customer support systems. We will cover some potential unexplored areas and potential future research below.

5.5.1 Architectural changes

We propose some architectural changes, including history memorization, fine-tuning additions, a hybrid architecture, retrieval-augmented fine-tuning, and user feedback in this section.

5.5.1.1 History memorization

To keep the memorization of the history consistent, the follow-up question should capture more detail of the history when needed, as right now it generally just returns a short sentence. However, you don't want a history that is too long capturing too much unnecessary information, which could result in longer standalone questions with too much detail that could instead worsen the retrieval process. We, therefore, recommend that only the history of the last three interactions is captured to limit the amount of context, but from these

three interactions, the LLM picks the most informative parts and rewrites them into a more detailed standalone question.

5.5.1.2 Improved fine-tuning and hybrid architecture

While we argued in subsubsection 5.1.1.1 that the fine-tuning possibly performed poorly due to the data, we also think that fine-tuning the newest gpt-4o model could be another way of improving the response quality. In future research the newly fine-tuned model could be tested on its performance. If this model shows considerable improvements then the model could even be used in combination with the current RAG architecture, instead of the current gpt-4o model that is used for response generation. This would then be a combination of two of the methods explored in this thesis, with fine-tuning narrowing down the domain of the LLM and with RAG leveraging this fine-tuned model to retrieve and incorporate relevant information from the knowledge base, to ensure even higher-quality responses. This type of construction could also mitigate the importance of the query handler as the model itself does contain knowledge, and doesn't only rely on the query handler for relevant information.

The different methods used in this thesis could also be used in an even more complex hybrid architecture that leverages the respective strengths of each method to create an enhanced chatbot architecture. For instance, a hybrid model could utilize few-shot learning for generating concise responses in straightforward scenarios, RAG for handling complex and knowledge-intensive queries, and fine-tuning to enhance the LLM's overall capabilities in the customer support domain as explained in subsection 5.4.1. This could include a sort switch module that analyses the incoming query and determines what model to use for that query.

5.5.1.3 Retrieval-Augmented Fine-Tuning

[Zhang et al. \(2024\)](#) propose a new method named retrieval-augmented fine-tuning (RAFT), which combines retrieval-based techniques with fine-tuning to enhance model performance. This new method is about teaching LLMs to get smarter about specific topics, in this case, customer support, while improving the RAG performance with that domain. RAFT achieves this by fine-tuning the models with domain-specific knowledge, so they are well-trained in relevant areas. Additionally, RAFT strengthens the model to prevent it from using inaccurate information retrievals by training them on how to fit the user's question, the retrieved documents, and the correct answer together. It's like studying for an open-book test by learning to spot what's important in your notes and what's not ([SuperAnnotate, 2024](#)). While the results from RAFT show potential, a comprehensive evaluation against the current methods would give insights into how well it performs in a full-fledged architecture.

5.5.1.4 User feedback

Implementing an automatic user feedback loop within the architecture could help give insights into the live performance of the bot. This can be done by giving a thumbs up or down after a response or a conversation has finished. These insights can help identify scenarios in which the chatbot struggles, and can, for example, be used to enhance the instruction templates.

5.5.2 Other LLMs

This study mostly focused on the LLMs taken from OpenAI with gpt-4o being the model used for zero-shot, few-shot, RAG, and the checks in the handover module, while gpt-3.5 was used for fine-tuning. Meta's Llama was the only other model used when we performed the DeepEval evaluation. A variety of different other LLM models like Google's Gemini ([Gemini Team, 2024](#)), Anthropic's Claude ([Anthropic, 2024](#)), or Meta's Llama ([Dubey et al., 2024](#)) could prove to be better suited for generating responses. Research into how each model performs in a situation can be interesting, it could be the case that in the handover module, the Gemini model is good at determining the relevance of a response, while Claude is the best model for determining specificity. A comprehensive comparison could be made in the future between all these different models and how they fare against each other in each situation.

Using other models than gpt-4o can also reduce generation costs, as LLaMa, for example, is free to use for a limited amount of calls per day, and could therefore be used in some cases. Using OpenAI's gpt-4o-mini in some cases could reduce generation costs as well. An evaluation of generation costs and response quality could be a future research area for companies to explore.

As recommended in subsection 5.4.1, fine-tuning the latest models is an important area of research, particularly in today's rapidly evolving landscape. Future research should focus on fine-tuning the latest gpt-4o model or, in the future, newer iterations, to assess their performance on chatbot response quality. These more advanced models may offer improved response quality, which could address some of the limitations observed in this study. Additionally, researching how data is presented to the LLM during the fine-tuning process, including the amount of data provided, and examining how these factors influence the model's responses would be interesting areas of study.

5.5.3 Improve retrieval process

In ([Setty et al., 2024](#)) the authors propose multiple methods of improving the retrieval of RAG-based generators. They mention different sorts of chunking methods like recursive chunking, element-based chunking ([Yepes et al., 2024](#)), semantic chunking, and agentic chunking. They mention a method called query expansion, which basically expands the original query to find more relevant chunks that would initially not be found. They also mention HyDE ([Nayak, 2023](#)), this method generates a theoretical answer and uses the

original question and the theoretical answer to perform the semantic search. The authors also mention metadata annotations, which could be used to pass extra information, like keywords of the documents, to the retriever. Another option is a document reranker that reranks documents based on importance, helping with prioritizing certain retrieved information over other information. Lastly, they propose the fine-tuning of the embeddings, which could enhance the retrieval process in a specific domain. While they didn't evaluate each of the mentioned methods, they showed that the document reranker and query expansion improved the retrieval process. In the future, these two methods, as well as the other unexplored ones, could be evaluated in a complete architecture to see how they affect the response quality.

Another new retrieval method has been proposed by (Sawarkar et al., 2024). They introduce Blended RAG, which uses three search techniques during the retrieval process: keyword-based, vector-based, and sparse encoder-based semantic search. They then created a method to create blended retrievers by blending semantic search with hybrid queries. They showed promising results on multiple datasets, which could be a great incentive to explore these findings in a full-fledged chatbot architecture.

The authors of Versebeniuc et al. (2024) use a multi-query retrieval in their architecture. With multi-query retrieval, an LLM generates multiple different versions of the incoming query. This can help capture multiple different perspectives, which can help retrieve a broader range of relevant documents. This could help with improving the accuracy, as more relevant documents get retrieved, which in turn will help the LLM create a more relevant and accurate response.

The implementation of time-weighted sources within the retrieval database could also be an area of research. While we did consider some source types more important than others, the age of the source can also be taken into consideration. Giving older sources a lower weight within their metadata compared to newer sources could help retrieve newer, more relevant sources. It is still important to keep in mind that some of the older sources could still be more relevant than the newer sources, so finding a good balance is key.

Lastly, a comprehensive evaluation of the query handler and its retrieval process could be an area of future research. Delving deeper into the retrieval process like how many documents of the available relevant documents are retrieved (recall), how many of the retrieved documents are relevant (precision), or the F1-score between the two. Such an evaluation can give a more clear indication of how well the query handler performs.

5.5.4 New evaluation metrics

The authors of Chen and Mueller (2024) propose a new method to evaluate the accuracy and trustworthiness of an LLM response. They introduce the BSDetector, which gives a confidence score for each response it generates. If incorporated in a complete chatbot architecture could help with the accuracy and confidence of the generated responses. While it does show promising results it also takes extra computing power and time which are some limitations. In Wang et al. (2024), the authors propose vRAG-Eval, which assesses

correctness, completeness, and honesty. They found substantial alignments with human judgment showing that their method could be an automatic evaluation method that needs to be further investigated in chatbot architectures.

5.5.5 Business aspect of LLMs

There are two hosting options: self-hosting or cloud-based hosting. Self-hosting, for example, a local LLaMa-3 model, generally offers more control over the infrastructure, allowing companies to adapt the environment to their specific needs, with also no additional costs to the LLM calls, or fine-tuning. However, this approach requires significant investment in hardware like GPUs, employees for maintenance, and security considerations. Cloud-based solutions, like OpenAI's API, on the other hand, can help with scalability and are easier to deploy with the only costs being the API calls and the potential fine-tuning process.

User data from past conversations but also during live chatbot interactions, will only stay within the company when you self-host an LLM, while with cloud-based LLMs the data will also go to the hosting company. For example, OpenAI guarantees that data from API calls is not used for training it still gets analyzed to make sure it follows OpenAI's guidelines. It should therefore be something that should still be considered for businesses.

Acknowledgments

I would like to express my deepest gratitude to my supervisor, dr. Marijn Schraagen, for their guidance and insightful feedback throughout the course of this research. Their expertise and insights have been essential in shaping this thesis. I would also like to thank dr. Shihan Wang for their enthusiasm and positive reception of my research proposal.

I am also thankful to Maxd’Oro and my supervisor there, ing. Maarten Kruit, for providing the resources and opportunities necessary to conduct this study. The collaborative environment and professional support at Maxd’Oro have greatly contributed to my work. I would also like to express my gratitude towards my fellow employees at the company for the enjoyable period I had with them.

I am also grateful to Wouter van Oosten Msc. for his guidance, help, and feedback throughout a large portion of this thesis.

A heartfelt thank you goes to my family for their continuous love, patience, and understanding. Your belief in me has been my greatest source of motivation.

Lastly, I would like to thank all of my friends for their constant encouragement and for being a source of joy and balance during this journey. Your support has made this achievement possible.

Bibliography

- Abercrombie, G., Curry, A. C., Dinkar, T., and Talat, Z. (2023). Mirages: On anthropomorphism in dialogue systems. *arXiv preprint arXiv:2305.09800*. doi: [10.48550/arxiv.2305.09800](https://doi.org/10.48550/arxiv.2305.09800).
- Acharya, A., Sneha, Y., Khettry, A., and Patil, D. (2020). AtheNA an avid traveller using LSTM based RNN architecture. *J. Eng. Sci. Technol*, 15:1413–1428.
- Adamopoulou, E. and Moussiades, L. (2020). Chatbots: History, technology, and applications. *Machine Learning with applications*, 2:100006. doi: [10.1016/j.mlwa.2020.100006](https://doi.org/10.1016/j.mlwa.2020.100006).
- Adiwardana, D., Luong, M., So, D. R., Hall, J., Fiedel, N., Thoppilan, R., Yang, Z., Kulshreshtha, A., Nemade, G., Lu, Y., and Le, Q. V. (2020). Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977*. doi: [10.48550/arXiv.2001.09977](https://doi.org/10.48550/arXiv.2001.09977).
- Al-Ajmi, A.-H. and Al-Twairesh, N. (2021). Building an Arabic flight booking dialogue system using a hybrid rule-based and data driven approach. *IEEE Access*, 9:7043–7053. doi: [10.1109/ACCESS.2021.3049732](https://doi.org/10.1109/ACCESS.2021.3049732).
- Amati, G. (2009). *BM25*. Springer US, Boston, MA.
- Anthropic (2024). The Claude 3 model family: Opus, Sonnet, Haiku. In *Claude-3 Model Card*.
- Avishka, P., Ravikumar, N., Banujan, K., and Gunasinghe, H. (2021). Towards a more intuitive Sinhala chatbot: Leveraging NLU for enhanced intent identification and entity extraction. *Sabaragamuwa University Journal*, 19(2):46–63. doi: [10.4038/suslj.v19i2.7792](https://doi.org/10.4038/suslj.v19i2.7792).
- Bae, S., Kwak, D., Kim, S., Ham, D., Kang, S., Lee, S.-W., and Park, W. (2022). Building a role specified open-domain dialogue system leveraging large-scale language models. *arXiv preprint arXiv:2205.00176*. doi: [10.48550/arXiv.2205.00176](https://doi.org/10.48550/arXiv.2205.00176).
- Banchs, R. E. and Li, H. (2012). IRIS: A chat-oriented dialogue system based on the vector space model. In *Proceedings of the ACL 2012 System Demonstrations*, pages 37–42.

- Banerjee, S. and Lavie, A. (2005). METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.
- Bansal, A., Eberhart, Z., Wu, L., and McMillan, C. (2021). A neural question answering system for basic questions about subroutines. In *2021 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 60–71. IEEE. doi: [10.1109/SANER50967.2021.00015](https://doi.org/10.1109/SANER50967.2021.00015).
- Bavaresco, R., Silveira, D., Reis, E., Barbosa, J., Righi, R., Costa, C., Antunes, R., Gomes, M., Gatti, C., Vanzin, M., Junior, S. C., Silva, E., and Moreira, C. (2020). Conversational agents in business: A systematic literature review and future research directions. *Computer Science Review*, 36:100239. doi: [10.1016/j.cosrev.2020.100239](https://doi.org/10.1016/j.cosrev.2020.100239).
- Bragg, J., Cohan, A., Lo, K., and Beltagy, I. (2021). FLEX: Unifying evaluation for few-shot NLP. *Advances in Neural Information Processing Systems*, 34:15787–15800. doi: [10.48550/arXiv.2107.07170](https://doi.org/10.48550/arXiv.2107.07170).
- Braggaar, A., Liebrecht, C., van Miltenburg, E., and Krahmer, E. (2023). Evaluating task-oriented dialogue systems: A systematic review of measures, constructs and their operationalisations. *arXiv preprint arXiv:2312.13871*. doi: [10.48550/arXiv.2312.13871](https://doi.org/10.48550/arXiv.2312.13871).
- Brocki, L., Dyer, G. C., Gladka, A., and Chung, N. C. (2023). Deep learning mental health dialogue system. In *2023 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pages 395–398. IEEE.
- Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901. doi: [10.48550/arXiv.2005.14165](https://doi.org/10.48550/arXiv.2005.14165).
- Carbonell, J. and Goldstein, J. (1998). The use of MMR, diversity-based reranking for re-ordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336.
- Chada, R. and Natarajan, P. (2021). FewshotQA: A simple framework for few-shot learning of question answering tasks using pre-trained text-to-text models. *arXiv preprint arXiv:2109.01951*. doi: [10.48550/arXiv.2109.01951](https://doi.org/10.48550/arXiv.2109.01951).
- Chen, A., Stanovsky, G., Singh, S., and Gardner, M. (2019). Evaluating question answering evaluation. In *Proceedings of the 2nd workshop on machine reading for question answering*, pages 119–124.

- Chen, D., Chen, H., Yang, Y., Lin, A., and Yu, Z. (2021a). Action-based conversations dataset: A corpus for building more in-depth task-oriented dialogue systems. *arXiv preprint arXiv:2104.00783*. doi: [10.48550/arXiv.2104.00783](https://doi.org/10.48550/arXiv.2104.00783).
- Chen, J., Lin, H., Han, X., and Sun, L. (2024). Benchmarking large language models in retrieval-augmented generation. In *Proceedings of the AAAI Conference on Artificial Intelligences*, volume 38, pages 17754–17762.
- Chen, J. and Mueller, J. (2024). Quantifying uncertainty in answers from any language model and enhancing their trustworthiness. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5186–5200.
- Chen, J.-S., Le, T.-T.-Y., and Florence, D. (2021b). Usability and responsiveness of artificial intelligence chatbot on online customer experience in e-retailing. *International Journal of Retail & Distribution Management*, 49(11):1512–1531.
- Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*. doi: [10.48550/arXiv.1406.1078](https://doi.org/10.48550/arXiv.1406.1078).
- Chroma (2024). Chroma - the open-source embedding database. url: <https://github.com/chroma-core/chroma>.
- Colby, K. M., Weber, S., and Hilf, F. D. (1971). Artificial paranoia. *Artificial intelligence*, 2(1):1–25.
- ConfidentAI (2023). DeepEval: A framework for evaluating the relevance of responses in conversational systems. url: <https://github.com/confident-ai/deepeval>.
- Cordero, J., Barba-Guaman, L., and Guamán, F. (2022). Use of chatbots for customer service in MSMEs. *Applied Computing and Informatics*.
- Cormack, G. V., Clarke, C. L., and Buettcher, S. (2009). Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 758–759.
- Cui, L., Huang, S., Wei, F., Tan, C., Duan, C., and Zhou, M. (2017). SuperAgent: A customer service chatbot for e-commerce websites. In *Proceedings of ACL 2017*, pages 97–102.
- Daher, K., Casas, J., Khaled, O. A., and Mugellini, E. (2020). Empathic chatbot response for medical assistance. In *Proceedings of the 20th ACM International Conference on Intelligent Virtual Agents*, pages 1–3.
- Dou, C., Jin, Z., Jiao, W., Zhao, H., Zhao, Y., and Tao, Z. (2023). PlugMed: Improving specificity in patient-centered medical dialogue generation using in-context learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5050–5066.

- Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Yang, A., Fan, A., et al. (2024). The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.
- Dunn, O. J. (1964). Multiple comparisons using rank sums. *Technometrics*, 6(3):241–252.
- Eibich, M., Nagpal, S., and Fred-Ojala, A. (2024). ARAGOG: Advanced RAG output grading. *arXiv preprint arXiv:2404.01037*.
- Ekin, S. (2023). Prompt engineering for ChatGPT: a quick guide to techniques, tips, and best practices. *Authorea Preprints*.
- El Bakkouri, B., Raki, S., and Belgnaoui, T. (2022). The role of chatbots in enhancing customer experience: Literature review. *Procedia Computer Science*, 203:432–437.
- Elman, J. L. (1990). Finding structure in time. *Cognitive science*, 14(2):179–211.
- Ferrucci, D., Brown, E., Chu-Carroll, J., Fanand, J., Gondek, D., Kalyanpur, A. A., Lally, A., Murdock, J. W., Nyberg, E., Prager, J., Schlaefer, N., and Welty, C. (2010). Building Watson: An overview of the DeepQA project. *AI magazine*, 31(3):59–79.
- Forgues, G., Pineau, J., Larchevêque, J.-M., and Tremblay, R. (2014). Bootstrapping dialog systems with word embeddings. In *Nips, modern machine learning and natural language processing workshop*, volume 2, page 168.
- Gemini Team (2024). Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context.
- Gill, S. S., Xu, M., Patros, P., Wu, H., Kaur, R., Kaur, K., Fuller, S., Singh, M., Arora, P., Parlikad, A. K., Stankovski, V., Abraham, A., Sakellariou, S., Uhlig, S., and Rajkumar, B. (2024). Transformative effects of ChatGPT on modern education: Emerging era of AI chatbots. *Internet of Things and Cyber-Physical Systems*, 4:19–23.
- Ginting, Y., Chandra, T., Miran, I., and Yusriadi, Y. (2023). Repurchase intention of e-commerce customers in Indonesia: An overview of the effect of e-service quality, e-word of mouth, customer trust, and customer satisfaction mediation. *International Journal of Data and Network Science*, 7(1):329–340.
- Gokul, A. (2023). LLMs and AI: Understanding its reach and impact. doi: [10.20944/preprints202305.0195.v1](https://doi.org/10.20944/preprints202305.0195.v1).
- Hardalov, M., Koychev, I., and Nakov, P. (2018). Towards automated customer support. In *Artificial Intelligence: Methodology, Systems, and Applications: 18th International Conference, AIMA 2018, Proceedings 18*, pages 48–59.
- Haugeland, I. K. F., Følstad, A., Taylor, C., and Bjørkli, C. A. (2022). Understanding the user experience of customer service chatbots: An experimental study of chatbot interaction design. *International Journal of Human-Computer Studies*, 161:102788.

- Hjerpakk, A., Hopland, A. M., Høyland, S. E., Kirknes, A., and Nylund, M. E. (2021). Perception of chatbots in customer service. *NTNU Trondheim*.
- Hochreiter, S. (1998). The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116.
- Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.
- Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., and Chen, W. (2021). LoRA: Low-rank adaptation of large language models. *CoRR*, abs/2106.09685.
- Huseynov, F. (2023). Chatbots in digital marketing: Enhanced customer experience and reduced customer service costs. In *Contemporary Approaches of Digital Marketing and the Role of Machine Intelligence*, pages 46–72. IGI Global.
- Jamieson, S. (2004). Likert scales: How to (ab) use them? *Medical education*, 38(12):1217–1218.
- Jenneboer, L., Herrando, C., and Constantinides, E. (2022). The impact of chatbots on customer loyalty: A systematic literature review. *Journal of theoretical and applied electronic commerce research*, 17(1):212–229.
- Joshi, H. (2021). Perception and adoption of customer service chatbots among millennials: An empirical validation in the indian context. In *Webist*, volume 2021, pages 197–208.
- Kandpal, N., Deng, H., Roberts, A., Wallace, E., and Raffel, C. (2023). Large language models struggle to learn long-tail knowledge. In *International Conference on Machine Learning*, pages 15696–15707. PMLR.
- Kasahara, T., Kawahara, D., Tung, N., Li, S., Shinzato, K., and Sato, T. (2022). Building a personalized dialogue system with prompt-tuning. *arXiv preprint arXiv:2206.05399*. doi: [10.48550/arXiv.2206.05399](https://doi.org/10.48550/arXiv.2206.05399).
- Khanna, A., Pandey, B., Vashishta, K., Kalia, K., Pradeepkumar, B., and Das, T. (2015). A study of today's AI through chatbots and rediscovery of machine intelligence. *International Journal of u-and e-Service, Science and Technology*, 8(7):277–284.
- Kim, B., Kim, H., Lee, S.-W., Lee, G., Kwak, D., Dong Hyeon, J., Park, S., Kim, S., Kim, S., Seo, D., Lee, H., Jeong, M., Lee, S., Kim, M., Ko, S. H., Kim, S., Park, T., Kim, J., Kang, S., Ryu, N.-H., Yoo, K. M., Chang, M., Suh, S., In, S., Park, J., Kim, K., Kim, H., Jeong, J., Yeo, Y. G., Ham, D., Park, D., Lee, M. Y., Kang, J., Kang, I., Ha, J.-W., Park, W., and Sung, N. (2021). What changes can large-scale language models bring? Intensive study on HyperCLOVA: Billions-scale Korean generative pretrained transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3405–3424.

- Krippendorff, K. (2011). Computing Krippendorff's alpha-reliability.
- Kuylenstierna, E. and Trägårdh, H. (2021). Customers perception of artificial intelligence as chatbots in CRM.
- Kvale, K., Freddi, E., Hodnebrog, S., Sell, O. A., and Følstad, A. (2020). Understanding the user experience of customer service chatbots: what can we learn from customer satisfaction surveys? In *International Workshop on Chatbot Research and Design*, pages 205–218. Springer.
- Laney, M. (2023). The effects of fine-tuning on the personality traits of large language models. Master's thesis, University of North Carolina at Chapel Hill.
- Latif, E. and Zhai, X. (2024). Fine-tuning ChatGPT for automatic scoring. *Computers and Education: Artificial Intelligence*, 6:100210.
- Lee, N., Ping, W., Xu, P., Patwary, M., Fung, P. N., Shoeybi, M., and Catanzaro, B. (2022). Factuality enhanced language models for open-ended text generation. *Advances in Neural Information Processing Systems*, 35:34586–34599.
- Lei, S. I., Shen, H., and Ye, S. (2021). A comparison between chatbot and human service: customer perception and reuse intention. *International Journal of Contemporary Hospitality Management*, 33(11):3977–3995.
- Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., and Kiel, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.
- Li, C.-H., Chen, K., and Chang, Y.-J. (2019). When there is no progress with a task-oriented chatbot: A conversation analysis. In *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services*, pages 1–6.
- Li, J., Galley, M., Brockett, C., Gao, J., and Dolan, B. (2015). A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055*. doi: [10.48550/arXiv.1510.03055](https://doi.org/10.48550/arXiv.1510.03055).
- Li, Z., Fan, S., Gu, Y., Li, X., Duan, Z., Dong, B., Liu, N., and Wang, J. (2024). FlexKBQA: A flexible LLM-powered framework for few-shot knowledge base question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18608–18616.
- Likert, R. (2017). *The method of constructing an attitude scale*. Routledge.
- Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.
- Lin, S., Hilton, J., and Evans, O. (2022). TruthfulQA: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*. doi: [10.48550/arXiv.2109.07958](https://doi.org/10.48550/arXiv.2109.07958).

- Liu, B. and Sundar, S. S. (2018). Should machines express sympathy and empathy? Experiments with a health advice chatbot. *Cyberpsychology, Behavior, and Social Networking*, 21(10):625–636.
- Liu, C.-W., Lowe, R., Serban, I. V., Noseworthy, M., Charlin, L., and Pineau, J. (2016). How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. *arXiv preprint arXiv:1603.08023*. doi: [10.48550/arXiv.1603.08023](https://doi.org/10.48550/arXiv.1603.08023).
- Liu, S., Zheng, C., Demasi, O., Sabour, S., Li, Y., Yu, Z., Jiang, Y., and Huang, M. (2021). Towards emotional support dialog systems. *arXiv preprint arXiv:2106.01144*. doi: [10.48550/arXiv.2106.01144](https://doi.org/10.48550/arXiv.2106.01144).
- Liu, Y., Jia, Y., Geng, R., Jia, J., and Gong, N. Z. (2023). Prompt injection attacks and defenses in LLM-integrated applications. *arXiv preprint arXiv:2310.12815*. doi: [10.48550/arXiv.2310.12815](https://doi.org/10.48550/arXiv.2310.12815).
- Liu, Y., Tao, S., Zhao, X., Zhu, M., Ma, W., Zhu, J., Su, C., Hou, Y., Zhang, M., Zhang, M., et al. (2024). CoachLM: Automatic instruction revisions improve the data quality in LLM instruction tuning. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 5184–5197. IEEE.
- Lo, C. K. (2023). What is the impact of ChatGPT on education? A rapid review of the literature. *Education Sciences*, 13(4):410.
- Lozano, L. M., García-Cueto, E., and Muñiz, J. (2008). Effect of the number of response categories on the reliability and validity of rating scales. *Methodology*, 4(2):73–79.
- Mangal, S., Joshi, P., and Modak, R. (2019). LSTM vs. GRU vs. Bidirectional RNN for script generation. *arXiv preprint arXiv:1908.04332*.
- Meyer-Waarden, L., Pavone, G., Poocharoentou, T., Prayatsup, P., Ratinaud, M., Tison, A., and Torné, S. (2020). How service quality influences customer acceptance and usage of chatbots? *SMR-Journal of Service Management Research*, 4(1):35–51.
- Miller, A. H., Feng, W., Fisch, A., Lu, J., Batra, D., Bordes, A., Parikh, D., and Weston, J. (2017). ParlAI: A dialog research software platform. *arXiv preprint arXiv:1705.06476*.
- Misischia, C. V., Poecze, F., and Strauss, C. (2022). Chatbots in customer service: Their relevance and impact on service quality. *Procedia Computer Science*, 201:421–428.
- Mitsuda, K., Higashinaka, R., Li, T., and Yoshida, S. (2022). Investigating person-specific errors in chat-oriented dialogue systems. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 464–469.
- Mizrahi, M., Kaplan, G., Malkin, D., Dror, R., Shahaf, D., and Stanovsky, G. (2024). State of what art? a call for multi-prompt llm evaluation. *Transactions of the Association for Computational Linguistics*, 12:933–949.

- Nayak, P. (2023). Advanced RAG — Improving retrieval using Hypothetical Document Embeddings(HyDE). url: <https://medium.aiplanet.com/advanced-rag-improving-retrieval-using-%20hypothetical-document-embeddings-hyde-1421a8ec075a>.
- Neelakantan, A., Xu, T., Puri, R., Radford, A., Han, J. M., Tworek, J., Yuan, Q., Tezak, N., Kim, J. W., Hallacy, C., Heidecke, J., Shyam, P., Power, B., Nekoul, T. E., Sastry, G., Krueger, G., Schnurr, D., Such, F. P., Hsu, K., Thompson, M., Khan, T., Sherbakov, T., Jang, J., Welinder, P., and Weng, L. (2022). Text and code embeddings by contrastive pre-training.
- Ngai, E. W., Lee, M. C., Luo, M., Chan, P. S., and Liang, T. (2021). An intelligent knowledge-based chatbot for customer service. *Electronic Commerce Research and Applications*, 50:101098.
- Nguyen, T. (2019). Potential effects of chatbot technology on customer support: A case study. Master's thesis, Aalto University.
- Nicolescu, L. and Tudorache, M. T. (2022). Human-computer interaction in customer service: The experience with AI chatbots—a systematic literature review. *Electronics*, 11(10):1579.
- Noever, D. and McKee, F. (2023). Chatbots as problem solvers: Playing twenty questions with role reversals. *arXiv preprint arXiv:2301.01743*. doi: [10.48550/arXiv.2301.01743](https://doi.org/10.48550/arXiv.2301.01743).
- Nuruzzaman, M. and Hussain, O. K. (2020). Intellibot: A dialogue-based chatbot for the insurance industry. *Knowledge-Based Systems*, 196:105810.
- OpenAI (2023). GPT-4 technical report. *arXiv preprint arXiv:2303.08774*. doi: [10.48550/arXiv.2303.08774](https://doi.org/10.48550/arXiv.2303.08774).
- Ott, M., Edunov, S., Baevski, A., Fan, A., Gross, S., Ng, N., Grangier, D., and Auli, M. (2019). Fairseq: A fast, extensible toolkit for sequence modeling. *arXiv preprint arXiv:1904.01038*. doi: [10.48550/arXiv.1904.01038](https://doi.org/10.48550/arXiv.1904.01038).
- Pandya, K. and Holia, M. (2023). Automating customer service using LangChain: Building custom open-source GPT chatbot for organizations. *arXiv preprint arXiv:2310.05421*. doi: [10.48550/arXiv.2310.05421](https://doi.org/10.48550/arXiv.2310.05421).
- Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.
- Perera, R. and Nand, P. (2017). Recent advances in natural language generation: A survey and classification of the empirical literature. *Computing and Informatics*, 36(1):1–32.
- Pornprasit, C. and Tantithamthavorn, C. (2024). GPT-3.5 for code review automation: How do few-shot learning, prompt design, and model fine-tuning impact their performance? *arXiv preprint arXiv:2402.00905*.

- Reshma, S. and Balakrishnan, K. (2018). Empowering chatbots with business intelligence by big data integration. *International Journal of Advanced Research in Computer Science*, 9(1).
- Roller, S., Dinan, E., Goyal, N., Ju, D., Williamson, M., Liu, Y., Xu, J., Ott, M., Shuster, K., Smith, E. M., Boureau, Y., and Weston, J. (2020). Recipes for building an open-domain chatbot. *arXiv preprint arXiv:2004.13637*.
- Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.
- Rus, V. and Lintean, M. (2012). An optimal assessment of natural language student input using word-to-word similarity metrics. In *Intelligent Tutoring Systems: 11th International Conference, ITS 2012. Proceedings 11*, pages 675–676. Springer.
- Sawarkar, K., Mangal, A., and Solanki, S. R. (2024). Blended RAG: Improving RAG (retriever-augmented generation) accuracy with semantic search and hybrid query-based retrievers. *arXiv preprint arXiv:2404.07220*.
- Schneider, J., Flores, A. C., and Kranz, A.-C. (2024). Exploring Human-LLM conversations: Mental models and the originator of toxicity. *arXiv preprint arXiv:2407.05977*.
- Schweter, S. and Akbik, A. (2020). Flert: Document-level features for named entity recognition.
- Setty, S., Thakkar, H., Lee, A., Chung, E., and Vidra, N. (2024). Improving retrieval for RAG based question answering models on financial documents. *arXiv preprint arXiv:2404.07221*.
- Sheskin, D. J. (2003). *Handbook of parametric and nonparametric statistical procedures*. Chapman and hall/CRC.
- Shi, S., Xu, Z., Hu, B., and Zhang, M. (2023). Generative multimodal entity linking. *arXiv preprint arXiv:2306.12725*.
- Spearman, C. (1904). The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):72–101.
- SuperAnnotate (2024). RAFT: Combining RAG with fine-tuning | SuperAnnotate. [url: https://www.superannotate.com/blog/raft-retrieval-augmented-fine-tuning](https://www.superannotate.com/blog/raft-retrieval-augmented-fine-tuning).
- Suthar, S. (2018). How chatbot helps businesses improve customer service. *Becoming Human: Artificial Intelligence Magazine*, 29.
- Tan, T. F., Elangovan, K., Jin, L., Jie, Y., Yong, L., Lim, J., Poh, S., Ng, W. Y., Lim, D., Ke, Y., et al. (2024). Fine-tuning large language model (LLM) artificial intelligence chatbots in ophthalmology and LLM-based evaluation using GPT-4. *arXiv preprint arXiv:2402.10083*.

- Topsakal, O. and Akinci, T. C. (2023). Creating large language model applications utilizing LangChain: A primer on developing LLM apps fast. In *International Conference on Applied Engineering and Natural Sciences*, volume 1, pages 1050–1056.
- Trivedi, J. (2019). Examining the customer experience of using banking chatbots and its impact on brand love: The moderating role of perceived risk. *Journal of internet Commerce*, 18(1):91–111.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.
- Versebeniuc, D., Elands, M., Falahatkar, S., Magrone, C., Falah, M., Boussé, M., and Härmä, A. (2024). Generative ai-based virtual assistant using retrieval-augmented generation.
- Vlasov, V., Drissner-Schmid, A., and Nichol, A. (2018). Few-shot generalization across dialogue tasks. *arXiv preprint arXiv:1811.11707*.
- Vlasov, V., Mosig, J. E., and Nichol, A. (2019). Dialogue transformers. *arXiv preprint arXiv:1910.00486*. doi: [10.48550/arXiv.1910.00486](https://doi.org/10.48550/arXiv.1910.00486).
- Wang, H., Wang, L., Du, Y., Chen, L., Zhou, J., Wang, Y., and Wong, K.-F. (2023a). A survey of the evolution of language model-based dialogue systems. *arXiv preprint arXiv:2311.16789*.
- Wang, J., Shi, E., Yu, S., Wu, Z., Ma, C., Dai, H., Yang, Q., Kang, Y., Wu, J., Hu, H., et al. (2023b). Prompt engineering for healthcare: Methodologies and applications. *arXiv preprint arXiv:2304.14670*.
- Wang, S., Yan, Q., and Wang, L. (2023c). Task-oriented vs. social-oriented: Chatbot communication styles in electronic commerce service recovery. *Electronic Commerce Research*, pages 1–33.
- Wang, Y., Hernandez, A. G., Kyslyi, R., and Kersting, N. (2024). Evaluating quality of answers for retrieval-augmented generation: A strong llm is all you need. *arXiv preprint arXiv:2406.18064*.
- Wang, Y., Yao, Q., Kwok, J. T., and Ni, L. M. (2020). Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur)*, 53(3):1–34.
- Webb, N. (2000). Rule-based dialogue management systems. In *Proceedings of the 3rd International Workshop on Human-Computer Conversation*, pages 3–5.
- Weizenbaum, J. (1966). ELIZA—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45.

- Weld, H., Huang, X., Long, S., Poon, J., and Han, S. C. (2022). A survey of joint intent detection and slot filling models in natural language understanding. *ACM Computing Surveys*, 55(8):1–38.
- Wester, J., Pohl, H., Hosio, S., and van Berkel, N. (2024). "this chatbot would never...": Perceived moral agency of mental health chatbots. *Proceedings of the ACM on Human-Computer Interaction*, 8(CSCW1):1–28.
- Wetstein, S. (2017). Designing a Dutch financial chatbot. Master's thesis, Vrije Universiteit Amsterdam.
- Wu, Q., Burges, C. J., Svore, K. M., and Gao, J. (2010). Adapting boosting for information retrieval measures. *Information Retrieval*, 13:254–270.
- Yang, H., Wang, Y., Xu, X., Zhang, H., and Bian, Y. (2024). Can we trust LLMs? Mitigate overconfidence bias in llms through knowledge transfer. *arXiv preprint arXiv:2405.16856*.
- Yao, Y., Duan, J., Xu, K., Cai, Y., Sun, Z., and Zhang, Y. (2024). A survey on Large Language Model (LLM) security and privacy: The good, the bad, and the ugly. *High-Confidence Computing*, page 100211.
- Yepes, A. J., You, Y., Milczek, J., Laverde, S., and Li, R. (2024). Financial report chunking for effective retrieval augmented generation.
- Zhang, R., Han, J., Liu, C., Gao, P., Zhou, A., Hu, X., Yan, S., Lu, P., Li, H., and Qiao, Y. (2023). LLaMA-Adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*.
- Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., and Artzi, Y. (2019). BERTScore: Evaluating text generation with BERT. *arXiv preprint arXiv:1904.09675*. doi: [10.48550/arXiv.1904.09675](https://doi.org/10.48550/arXiv.1904.09675).
- Zhang, T., Patil, S. G., Jain, N., Shen, S., Zaharia, M., Stoica, I., and Gonzalez, J. E. (2024). RAFT: Adapting language model to domain specific rag.
- Zheng, Z., Liao, L., Deng, Y., and Nie, L. (2023). Building emotional support chatbots in the era of LLMs. *arXiv preprint arXiv:2308.11584*.
- Zhou, H., Huang, M., Zhang, T., Zhu, X., and Liu, B. (2018). Emotional chatting machine: Emotional conversation generation with internal and external memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.
- Zhou, H., Ke, P., Zhang, Z., Gu, Y., Zheng, Y., Zheng, C., Wang, Y., Wu, C. H., Sun, H., Yang, X., Wen, B., Zhu, X., Huang, M., and Tang, J. (2021). EVA: An open-domain chinese dialogue system with large-scale generative pre-training. *arXiv preprint arXiv:2108.01547*. doi: [10.48550/arXiv.2108.01547](https://doi.org/10.48550/arXiv.2108.01547).
- Zhou, L., Gao, J., Li, D., and Shum, H.-Y. (2020). The design and implementation of XiaoIce, an empathetic social chatbot. *Computational Linguistics*, 46(1):53–93.

Appendix A

Data types table

Table A.1: *The different datatypes that will be used in the Knowledge Base (KB) translated from Dutch to English for readability purposes. Sensitive information has been masked.*

Data type	Example
Email conversation	<p>Customer: We want to upgrade our forms to a newer version soon. For this, it would be useful if we could see whether - and if so, how many and with which users - there are still concepts open. Can this be seen somewhere?</p> <p>Support: Dear [Name],</p> <p>It is possible, when deleting, or simply clicking on the trash bin icon, of a form to see how many concepts are still open in total. There is no need to actually delete the form, just clicking is enough. However, it is not possible to specifically see whose concepts these are, for this, you would need to check the concepts per user account. Users themselves can see how many concepts they have open per form, so everyone can check their own concepts. I hope this answers your question.</p> <p>Customer: Hi Support Team,</p> <p>Clear explanation and good to know. Thanks for this information!</p>
Articles	<p>Article title: Will I lose all my forms and data after the trial period?</p>

Continued on next page

Table A.1 continued from previous page

Data type	Example
HubSpot tickets	<p>Content: No, you will not lose all your forms and data after the trial period. Collected data will remain stored in the cloud. If you decide to purchase Incontrol after the trial period, you can simply continue with your own data. Even if there are several months in between. For more information about your data, please look at the article "<i>Licentie aanpassen en beheer van je data</i>".</p> <p>Ticket name: Concepts missing Requestor's name: [Name] Category: Help me with a problem Ticket description: The user "[Name]" claims to be missing concepts. When I look in the management module, I see all the concepts listed, however, it says 'no access' behind them. Furthermore, nothing else is ready.</p> <p>Support: Dear [Name],</p> <p>All concepts are based on the form "[Name]". This template is only linked to the security group "[Name]". The user "[Name]" is no longer linked to the security group "[Name]". By re-linking him to this group, he should be able to view and further edit the existing concepts.</p> <p>I look forward to hearing if it was successful.</p> <p>Kind regards, The support team.</p> <p>Ticket requester: It worked, thank you</p> <p>Kind regards, [Name]</p>

Appendix B

Initial questions

B.1 Original questions

1. Ik heb sinds vorige week een proef account gemaakt op jullie website. Kunt u mij vertellen of ik met het basispakket uit de voeten kan.
2. Ik ben mijn formulier kwijt, kan deze worden teruggehaald?
3. Wij hebben een account op de naam van iemand die niet meer bij ons werkt en willen dit graag veranderen.
4. Ik heb in de container aangevinkt 'voorkom afbreken in het document'. Maar hij splitst toch nog een blok over verschillende pagina's.
5. Sinds een paar dagen laadt een concept niet. Overige concepten laden wel.
6. Kan ik aan het template een bedrijfslogo toevoegen zodat deze op de pagina staat in plaats van het Incontrol logo?
7. Ik kan geen preview in PDF naar partijen sturen. Als ik klik op bestand delen, dan stuurt hij een link die partijen naar de website stuurt.
8. Sommige foto's die ik maak lijken in eerste instanties scherp te zijn maar blijken later onscherp te zijn, hoe kan dit?
9. Kan ik een per ongeluk verwijderd formulier nog terughalen?
10. Kan ik een backoffice koppeling maken met jullie applicatie?
11. Mijn wachtwoord herstel werkt niet.
12. Bij het vermelden van een emailadres in een tekstelement en/of een automatisch gegenereerde email wordt het emailadres afgekapt, alles na het @-teken wordt niet afgebeeld.

13. Ik doe een test met jullie tool, maar vind het erg ingewikkeld... kan ik een demo krijgen?
14. Kan je documenten ook ondertekenen op de iPad?
15. Kunnen jullie een formulier voor mij maken en wat zijn de kosten hiervoor?
16. Hoe kan je excel gegevens importeren in de app?
17. Ik wil graag een pdf toevoegen aan een formulier vanaf mijn harde schijf. Nu kan ik alleen een foto toevoegen. Kan dat?
18. Ik wil van een basis licentie naar een premium licentie, kan dit zonder gegevens verlies?
19. Als mijn licentie is verlopen ben ik dan ook al mijn gegevens kwijt?
20. Er zit een wachtwoord op al mijn gegenereerde PDF bestanden, kan deze eraf?
21. Kan je met meerdere inspecteurs aan één concept werken?
22. Ik heb een concept gedeeld met een collega maar kan deze niet meer terug vinden op mijn iPad.
23. De uitkomst van een som wordt in het document of het voorbeeld wordt fout weer gegeven t.o.v. het formulier.
24. Kan je met 1 account op meerdere devices werken?
25. Hoe kan ik ervoor zorgen dat ik op het dashboard Mijn Formulieren alleen de formulieren weergeef die ik daar wil hebben.

B.2 English translated questions

1. I created a trial account on your website last week. Can you tell me if I can manage with the basic package?
2. I have lost my form, can it be retrieved?
3. We have an account in the name of someone who no longer works with us and we would like to change this?
4. I checked 'avoid breaking on the document' in the container, but it still splits a block across multiple pages.
5. For the past few days, a draft won't load. Other drafts do load.
6. Can I add a company logo to the template so that it appears on the page instead of the Incontrol logo?
7. I can't send a PDF preview to parties. When I click on share file, it sends a link that directs parties to the website.
8. Some photos I take initially appear sharp but turn out to be blurry later, how can this happen?
9. Can I retrieve a form that was accidentally deleted?
10. Can I create a back-office connection with your application?
11. My password reset isn't working.
12. When mentioning an email address in a text element and/or an automatically generated email, the email address gets truncated, everything after the '@' symbol doesn't appear.
13. I'm testing your tool, but find it very complicated... can I get a demo?
14. Can you sign documents on the iPad as well?
15. Can you create a form for me, and what are the costs involved?
16. How can you import excel data into the app?
17. I would like to add a PDF to a form from my hard drive. Currently, I can only add a photo. Is that possible?
18. I want to upgrade from a basic license to a premium license, can this be done without data loss?
19. If my license has expired, do I lose all my data?

20. All my generated PDF files have a password on them, can this be removed?
21. Can multiple inspectors work on one draft?
22. I shared a draft with a colleague but can no longer find it on my iPad.
23. The result of a calculation is displayed incorrectly in the document or the preview compared to the form.
24. Can you use one account on multiple devices?
25. How can I ensure that on the 'My Forms' dashboard, I only display the forms that I want to see?

Appendix C

Initial scenarios

C.1 Original scenarios

Easy		
Nr.	OG nr.	Scenario
1.	18.	<i>De klant wilt graag upgraden van een basis licentie naar een premium licentie, maar vraagt zich af of dit kan zonder gegevensverlies?</i>
2.	12.	<i>Het viel een klant op dat bij het vermelden van een emailadres in een tekstelement en/of een automatisch gegenereerde email het emailadres wordt afgekapt. Alles na het @-teken wordt namelijk niet afgebeeld.</i>
3.	3.	<i>De klant heeft een account op naam van iemand die niet meer bij ze werkt. Dit account willen ze graag omzetten.</i>
4.	24.	<i>De klant vraagt zich af of je met 1 account op meerdere devices kan werken?</i>
5.	4.	<i>De klant heeft in de container aangevinkt 'voorkom afbreken in het document'. Maar nog steeds wordt een blok over verschillende pagina's gesplitst.</i>

Medium		
Nr.	OG nr.	Scenario
6.	10.	<i>De klant vraagt zich af of er een backoffice koppeling kan worden gemaakt met Incontrol.</i>
7.	15.	<i>Een klant vraagt zich af of er een formulier op verzoek kan worden gemaakt en wat hier dan de kosten voor zijn?</i>
8.	6.	<i>De klant vraagt zich of aan het template een bedrijfslogo kan worden toegevoegd. Zodat deze op de pagina staat in plaats van het Incontrol logo?</i>
9.	7.	<i>Als de klant een formulier wilt delen kan deze niet in PDF naar partijen sturen maar in plaats daarvan wordt er een link gegeven die partijen naar de website stuurt.</i>
10.	11.	<i>De klant geeft aan dat het wachtwoord herstel niet werkt. Er komt namelijk geen link binnen.</i>

Table C.1: Continued on the next page.

Nr.	OG nr.	Scenario	Hard
11.	5.	<i>Sinds een paar dagen laadt het concept van de klant niet, terwijl andere concepten wel correct laden.</i>	
12.	17.	<i>Een klant wil graag een bijlage toevoegen aan een formulier vanaf de harde schijf, de bijlage is een PDF. De klant verneemt dat alleen foto's kunnen worden toegevoegd</i>	
13.	2.	<i>De klant is zijn formulier kwijtgeraakt en wil het graag terugkrijgen.</i>	
14.	1.	<i>De klant heeft sinds vorige week een proef account gemaakt op de website en vraagt zich af of een basispakket voldoende is.</i>	
15.	22.	<i>De klant heeft een concept gedeeld met een collega maar kan deze niet meer terug vinden op z'n iPad.</i>	

Table C.2: The scenarios created from the initial question. The OG nr. is the question number found in Appendix B on which the scenario is based.

C.2 English translated scenarios

Easy		
Nr.	OG nr.	Scenario
1.	18.	A customer would like to upgrade from a basic license to a premium license but wonders if this can be done without data loss?
2.	12.	A customer noticed that when mentioning an email address in a text element and/or an automatically generated email, the email address gets truncated. Everything after the @-symbol is not displayed.
3.	3.	A customer has an account in the name of someone who no longer works with them. They would like to transfer this account.
4.	24.	A customer wonders if you can work with one account on multiple devices?
5.	4.	A customer has checked 'avoid breaking in the document' in the container. But a block is still split across multiple pages.

Medium		
Nr.	OG nr.	Scenario
6.	10.	A customer wonders if a back-office connection can be made with the application.
7.	15.	A customer wonders if a form can be created upon request and what the costs would be?
8.	6.	A customer asks if a company logo can be added to the template so that it appears on the page instead of the Incontrol logo?
9.	7.	When a customer wants to share a form, they cannot send it to parties as a PDF, instead, a link is given that directs parties to the website.
10.	11.	A customer indicates that the password reset is not working. No link is received.

Hard		
Nr.	OG nr.	Scenario
11.	5.	For the past few days, a customer's draft is not loading, while other drafts load correctly.
12.	17.	A customer would like to add an attachment to a form from the hard drive, the attachment is a PDF. The customer learns that only photos can be added.
13.	2.	A customer has lost their form and would like to retrieve it.
14.	1.	Since last week, the customer has created a trial account on the website and wonders if a basic package is sufficient.
15.	22.	A customer shared a draft with a colleague but can no longer find it on their iPad.

Table C.3: The translated scenarios created from the initial question. The OG nr. is the question number found in Appendix B, on which the scenario is based.

Appendix D

Templates 1st phase

D.1 Chat history template

If the Chat History contains information and there is a Follow-Up Input in the same context as the history, rephrase the follow-up input to be a standalone question in {language}.

Chat History: {chat history}

Follow Up Input: {question}

Standalone question:

Table D.1: *The template used for rephrasing a question in phase 1. The language, chat history, and question are variables given to the template during a conversation.*

D.2 Few-shot conversion template

For each of the documents in the context below, create a question and answer pair from it that covers the entire request, but is not too long.

Each Q&A pair you made must be in the format of a python dictionary, so
[{'question': '...', 'answer': '...'}, {'question': '...', 'answer': '...'},]

Just return the dictionaries.

Context: {context}

Table D.2: *The template used for creating Q&A pairs from the retrieved documents, which were then used for the few-shot learning template. The context is a variable given to the template during a conversation.*

Appendix E

Detailed statistical test results

Compare	Relev	Accur	Complete	Concise	Empathy	Friendl	All
ZS - FSL	0.429	0.28	0.37	0.001**	<0.001**	<0.001**	1.00
ZS - FT	1.00	1.00	1.00	0.002**	0.522	0.058	1.00
ZS - RAG	<0.001**	<0.001**	<0.001**	<0.001**	0.024*	0.094	<0.001**
FSL - FT	0.99	0.43	1.00	1.00	0.008**	<0.001**	0.767
FSL - RAG	0.041	0.013*	<0.001**	1.00	<0.001**	<0.001**	<0.001**
FT - RAG	0.015*	<0.001**	<0.001**	1.00	<0.001**	<0.001**	<0.001**

Table E.1: Statistical tests comparisons between all methods and for each question, with the bold entries being statistically significant. The Kruskal-Wallis test was used in combination with the post-hoc Dunn test to get the results. * = $p < .025$, ** = $p < .01$, ZS = zero-shot, FSL = few-shot learning, FT = fine-tuning, RAG = retrieval-augmented generation.

Appendix F

Best- and worst-performing questions

F.1 Relevance

Top 5 Best Only Relevance			
Nr.	Question	\bar{X}	σ
6.	<i>Can I add a company logo to the template so that it appears on the page instead of the Incontrol logo?</i>	4.68	0.47
18.	<i>I want to upgrade from a basic license to a premium license, can this be done without data loss?</i>	4.64	0.49
13.	<i>I'm testing your tool, but find it very complicated... can I get a demo?</i>	4.47	0.69
10.	<i>Can I create a back-office connection with your application?</i>	4.46	0.72
9.	<i>Can I retrieve a form that was accidentally deleted?</i>	4.41	0.64

Top 5 Worst Only Relevance			
Nr.	Question	\bar{X}	σ
5.	<i>For the past few days, a draft won't load. Other drafts do load.</i>	2.83	1.22
2.	<i>I have lost my form, can it be retrieved?</i>	3.35	1.33
22.	<i>I shared a draft with a colleague but can no longer find it on my iPad</i>	3.63	1.29
17.	<i>I would like to add a PDF to a form from my hard drive. Currently, I can only add a photo. Is that possible?</i>	3.64	1.13
8.	<i>Some photos I take initially appear sharp but turn out to be blurry later, how can this happen?</i>	3.64	1.23

Table F.1: The top five best and worst performing questions based on their average scores of the relevance metric.

F.2 Accuracy

Top 5 Best Only Accuracy			
Nr.	Question	\bar{X}	σ
18.	<i>I want to upgrade from a basic license to a premium license, can this be done without data loss?</i>	4.50	0.70
10.	<i>Can I create a back-office connection with your application?</i>	4.36	0.87
6.	<i>Can I add a company logo to the template so that it appears on the page instead of the Incontrol logo?</i>	4.18	0.93
13.	<i>I'm testing your tool, but find it very complicated... can I get a demo?</i>	4.05	0.87
3.	<i>We have an account in the name of someone who no longer works with us and we would like to change this?</i>	4.02	0.88
Top 5 Worst Only Accuracy			
Nr.	Question	\bar{X}	σ
17.	<i>I would like to add a PDF to a form from my hard drive. Currently, I can only add a photo. Is that possible?</i>	2.56	1.32
5.	<i>For the past few days, a draft won't load. Other drafts do load.</i>	2.71	1.29
2.	<i>I have lost my form, can it be retrieved?</i>	2.72	1.44
1.	<i>I created a trial account on your website last week. Can you tell me if I can manage with the basic package?</i>	2.98	1.32
12.	<i>When mentioning an email address in a text element and/or an automatically generated email, the email address gets truncated, everything after the '@' symbol doesn't appear.</i>	3.17	1.76

Table F.2: The top five best and worst performing questions based on their average scores of the accuracy metric.

F.3 Completeness

Top 5 Best Only Completeness			
Nr.	Question	\bar{X}	σ
18.	<i>I want to upgrade from a basic license to a premium license, can this be done without data loss?</i>	4.39	0.90
4.	<i>I checked 'avoid breaking on the document' in the container, but it still splits a block across multiple pages.</i>	3.76	1.32
25.	<i>How can I ensure that on the 'My Forms' dashboard, I only display the forms that I want to see?</i>	3.72	1.19
11.	<i>My password reset isn't working.</i>	3.72	1.21
13.	<i>I'm testing your tool, but find it very complicated... can I get a demo?</i>	3.71	1.27
Top 5 Worst Only Completeness			
Nr.	Question	\bar{X}	σ
5.	<i>For the past few days, a draft won't load. Other drafts do load.</i>	2.27	1.34
1.	<i>I created a trial account on your website last week. Can you tell me if I can manage with the basic package?</i>	2.50	1.27
16.	<i>How can you import excel data into the app?</i>	2.50	1.45
17.	<i>I would like to add a PDF to a form from my hard drive. Currently, I can only add a photo. Is that possible?</i>	2.53	1.42
8.	<i>Some photos I take initially appear sharp but turn out to be blurry later, how can this happen?</i>	2.76	1.21

Table F.3: The top five best and worst performing questions based on their average scores of the completeness metric.

F.4 Conciseness

Top 5 Best Only Conciseness			
Nr.	Question	\bar{X}	σ
13.	<i>I'm testing your tool, but find it very complicated... can I get a demo?</i>	4.58	0.60
18.	<i>I want to upgrade from a basic license to a premium license, can this be done without data loss?</i>	4.56	0.81
10.	<i>Can I create a back-office connection with your application?</i>	4.33	0.87
9.	<i>Can I retrieve a form that was accidentally deleted?</i>	4.23	0.84
6.	<i>Can I add a company logo to the template so that it appears on the page instead of the Incontrol logo?</i>	4.20	1.02

Top 5 Worst Only Conciseness			
Nr.	Question	\bar{X}	σ
5.	<i>For the past few days, a draft won't load. Other drafts do load.</i>	2.61	1.45
2.	<i>I have lost my form, can it be retrieved?</i>	2.79	1.50
8.	<i>Some photos I take initially appear sharp but turn out to be blurry later, how can this happen?</i>	2.90	1.39
17.	<i>I would like to add a PDF to a form from my hard drive. Currently, I can only add a photo. Is that possible?</i>	3.11	1.39
1.	<i>I created a trial account on your website last week. Can you tell me if I can manage with the basic package?</i>	3.17	1.36

Table F.4: The top five best and worst performing questions based on their average scores of the conciseness metric.

F.5 Empathy

Top 5 Best Only Empathy

Nr.	Question	\bar{X}	σ
10.	<i>Can I create a back-office connection with your application?</i>	4.36	0.58
20.	<i>All my generated PDF files have a password on them, can this be removed?</i>	4.29	0.79
19.	<i>If my license has expired, do I lose all my data?</i>	4.25	0.50
13.	<i>I'm testing your tool, but find it very complicated... can I get a demo?</i>	4.24	0.85
18.	<i>I want to upgrade from a basic license to a premium license, can this be done without data loss?</i>	4.22	0.72

Top 5 Worst Only Empathy

Nr.	Question	\bar{X}	σ
5.	<i>For the past few days, a draft won't load. Other drafts do load.</i>	2.95	1.26
22.	<i>I shared a draft with a colleague but can no longer find it on my iPad.</i>	3.31	1.21
17.	<i>I would like to add a PDF to a form from my hard drive. Currently, I can only add a photo. Is that possible?</i>	3.47	1.06
2.	<i>I have lost my form, can it be retrieved?</i>	3.53	1.14
7.	<i>I can't send a PDF preview to parties. When I click on share file, it sends a link that directs parties to the website.</i>	3.55	1.17

Table F.5: *The top five best and worst performing questions based on their average scores of the empathy metric.*

F.6 Friendliness

Top 5 Best Only Friendliness			
Nr.	Question	\bar{X}	σ
3.	<i>We have an account in the name of someone who no longer works with us and we would like to change this?</i>	4.41	0.55
10.	<i>Can I create a back-office connection with your application?</i>	4.41	0.60
20.	<i>All my generated PDF files have a password on them, can this be removed?</i>	4.37	0.81
18.	<i>I want to upgrade from a basic license to a premium license, can this be done without data loss?</i>	4.36	0.64
24.	<i>Can you use one account on multiple devices?</i>	4.36	0.68
Top 5 Worst Only Friendliness			
Nr.	Question	\bar{X}	σ
5.	<i>For the past few days, a draft won't load. Other drafts do load.</i>	3.44	1.14
22.	<i>I shared a draft with a colleague but can no longer find it on my iPad.</i>	3.60	1.17
15.	<i>Can you create a form for me, and what are the costs involved?</i>	3.82	1.09
7.	<i>I can't send a PDF preview to parties. When I click on share file, it sends a link that directs parties to the website.</i>	3.90	0.96
2.	<i>I have lost my form, can it be retrieved?</i>	3.91	0.78

Table F.6: The top five best and worst performing questions based on their average scores of the friendliness metric.

Appendix G

Initial question scores

G.1 Relevance

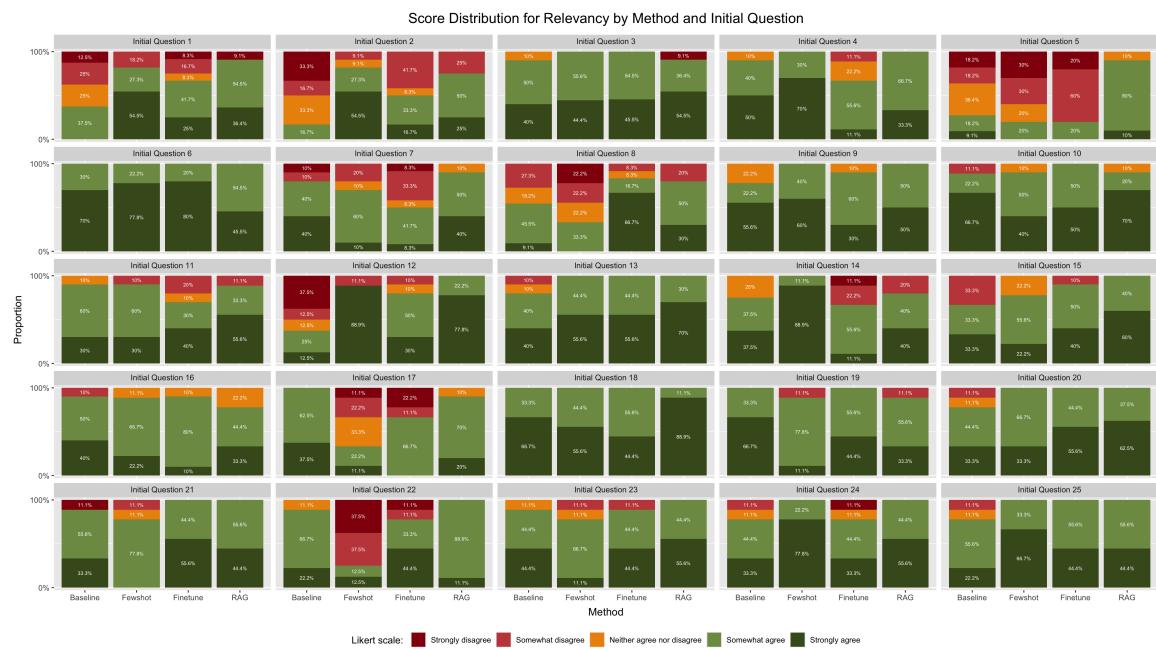


Figure G.1: The relevancy percentage scores for each initial question asked, given the method.

G.2 Accuracy

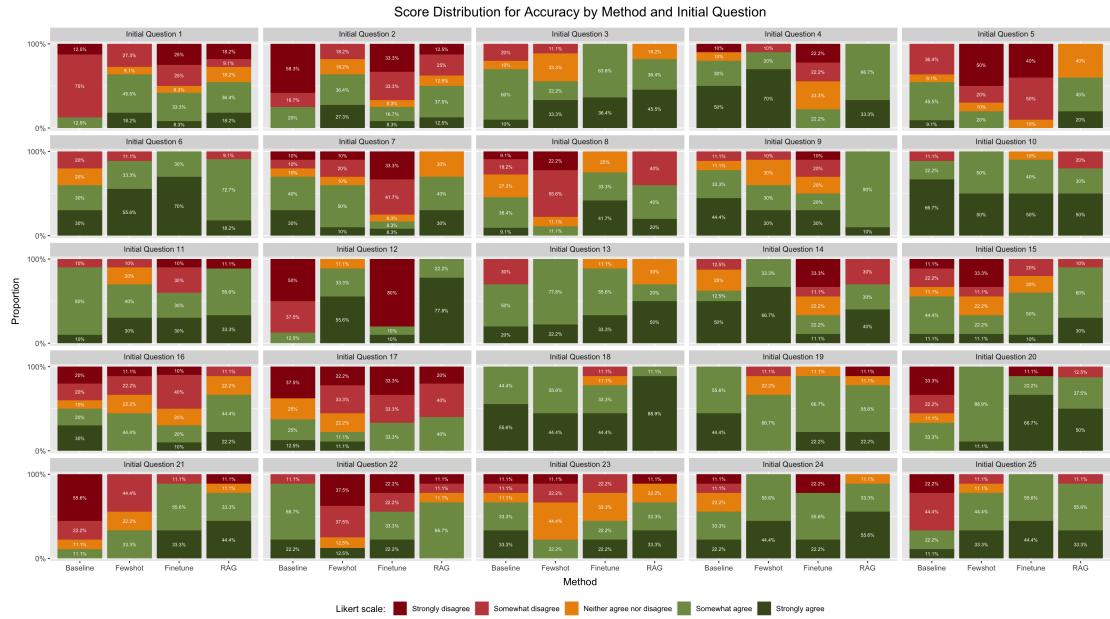


Figure G.2: The accuracy percentage scores for each initial question asked, given the method.

G.3 Completeness

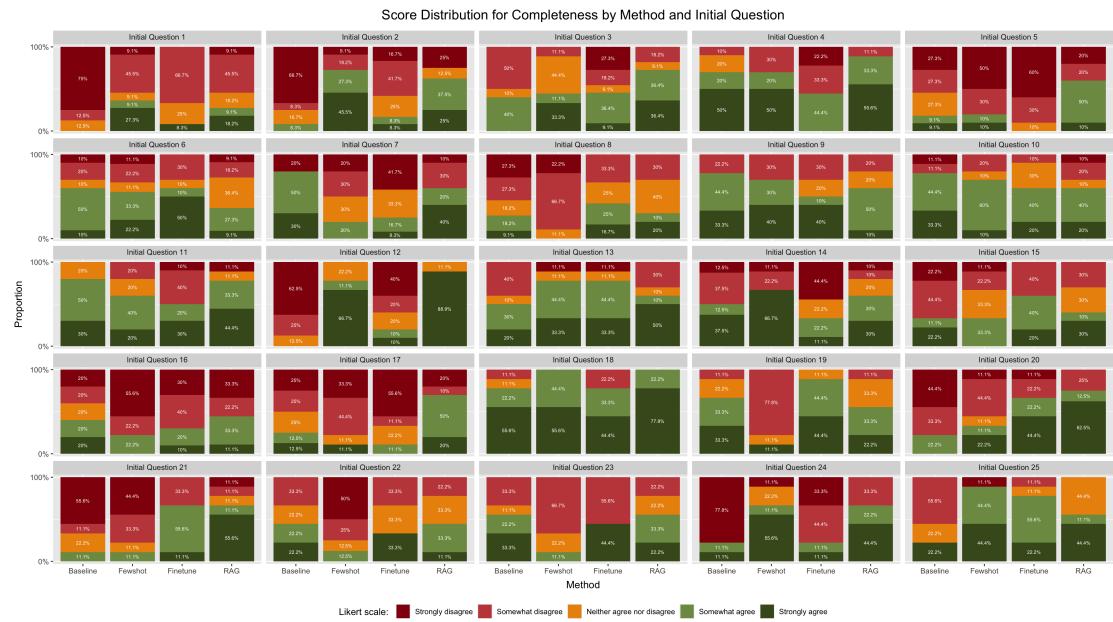


Figure G.3: The completeness percentage scores for each initial question asked, given the method.

G.4 Conciseness

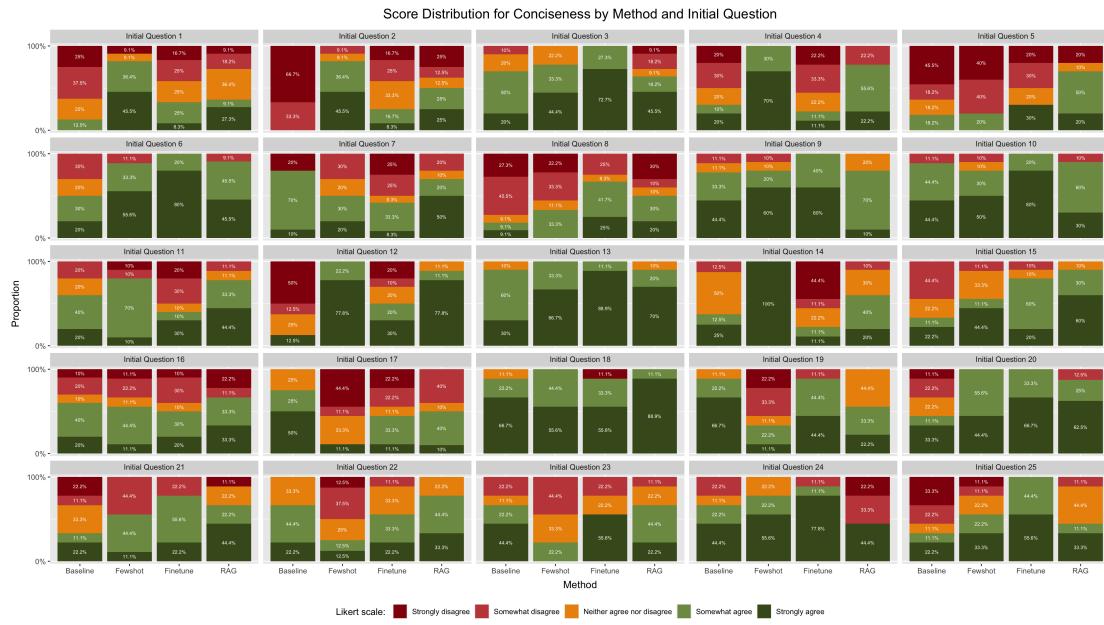


Figure G.4: The conciseness percentage scores for each initial question asked, given the method.

G.5 Empathy

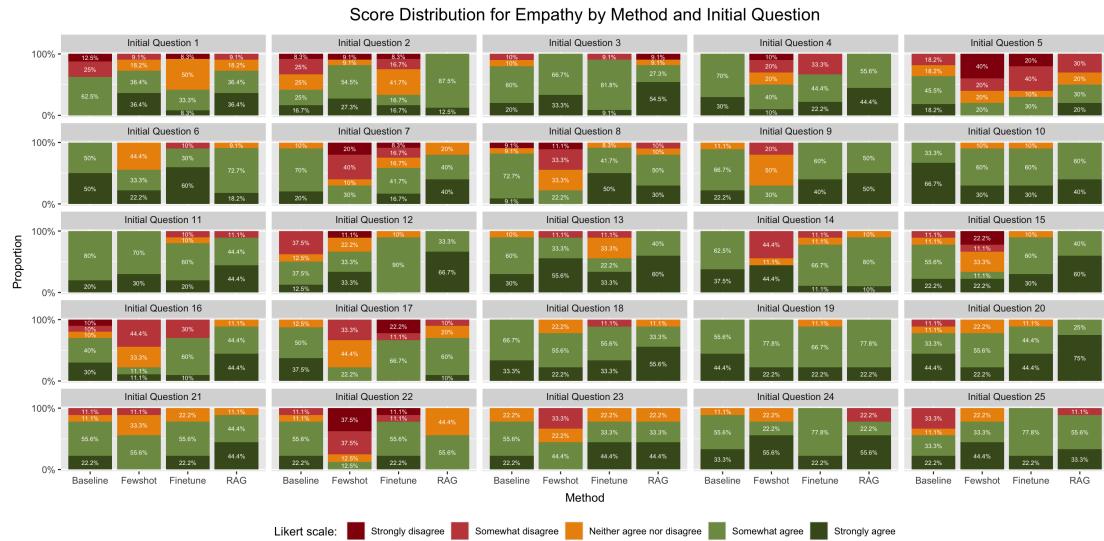


Figure G.5: The empathy percentage scores for each initial question asked, given the method.

G.6 Friendliness



Figure G.6: The empathy percentage scores for each initial question asked, given the method.

Appendix H

Best- and worst-performing scenarios

H.1 Relevance

Top 3 Best given Relevance			
Nr.	Scenario	\bar{X}	σ
1.	<i>A customer would like to upgrade from a basic license to a premium license but wonders if this can be done without data loss?</i>	5.00	0.00
14.	<i>Since last week, the customer has created a trial account on the website and wonders if a basic package is sufficient.</i>	4.88	0.35
8.	<i>A customer asks if a company logo can be added to the template so that it appears on the page instead of the Incontrol logo?</i>	4.8	0.45

Top 3 Worst given Relevance			
Nr.	Scenario	\bar{X}	σ
15.	<i>A customer shared a draft with a colleague but can no longer find it on their iPad.</i>	3.75	0.5
13.	<i>A customer has lost their form and would like to retrieve it.</i>	4.17	0.75
12.	<i>A customer would like to add an attachment to a form from the hard drive, the attachment is a PDF. The customer learns that only photos can be added.</i>	4.25	0.71

Table H.1: The top three best and worst performing scenarios based on their average relevance scores.

H.2 Accuracy

Top 3 Best given Accuracy			
Nr.	Scenario	\bar{X}	σ
1.	<i>A customer would like to upgrade from a basic license to a premium license but wonders if this can be done without data loss?</i>	5.00	0.00
7.	<i>A customer wonders if a form can be created upon request and what the costs would be?</i>	4.88	0.35
14.	<i>Since last week, the customer has created a trial account on the website and wonders if a basic package is sufficient.</i>	4.75	0.46

Top 3 Worst given Accuracy			
Nr.	Scenario	\bar{X}	σ
12.	<i>A customer would like to add an attachment to a form from the hard drive, the attachment is a PDF. The customer learns that only photos can be added.</i>	4.00	1.10
5.	<i>A customer has checked 'avoid breaking in the document' in the container. But a block is still split across multiple pages.</i>	4.25	1.50
13.	<i>A customer has lost their form and would like to retrieve it.</i>	4.25	0.71

Table H.2: *The top three best and worst performing scenarios based on their average accuracy scores.*

H.3 Completeness

Top 3 Best given Completeness			
Nr.	Scenario	\bar{X}	σ
1.	A customer would like to upgrade from a basic license to a premium license but wonders if this can be done without data loss?	5.00	0.00
8.	A customer asks if a company logo can be added to the template so that it appears on the page instead of the Incontrol logo?	4.8	0.45
4.	A customer wonders if you can work with one account on multiple devices?	4.71	0.49
Top 3 Worst given Completeness			
Nr.	Scenario	\bar{X}	σ
15.	A customer shared a draft with a colleague but can no longer find it on their iPad.	3.25	1.26
9.	When a customer wants to share a form, they cannot send it to parties as a PDF, instead, a link is given that directs parties to the website.	3.57	1.27
5.	A customer has checked 'avoid breaking in the document' in the container. But a block is still split across multiple pages.	3.88	1.36

Table H.3: The top three best and worst performing scenarios based on their average completeness scores.

H.4 Conciseness

Top 3 Best given Conciseness			
Nr.	Scenario	\bar{X}	σ
1.	A customer would like to upgrade from a basic license to a premium license but wonders if this can be done without data loss?	5.00	0.00
4.	A customer wonders if you can work with one account on multiple devices?	5.00	0.00
10.	A customer indicates that the password reset is not working. No link is received.	4.80	0.45

Top 3 Worst given Conciseness			
Nr.	Scenario	\bar{X}	σ
15.	A customer shared a draft with a colleague but can no longer find it on their iPad.	3.25	0.50
3.	customer has an account in the name of someone who no longer works with them. They would like to transfer this account.	4.12	1.25
13.	A customer has lost their form and would like to retrieve it.	4.33	1.21

Table H.4: The top three best and worst performing scenarios based on their average conciseness scores.

H.5 Empathy

Top 3 Best given Empathy			
Nr.	Scenario	\bar{X}	σ
12.	<i>A customer would like to add an attachment to a form from the hard drive, the attachment is a PDF. The customer learns that only photos can be added.</i>	4.50	0.76
14.	<i>Since last week, the customer has created a trial account on the website and wonders if a basic package is sufficient.</i>	4.50	0.93
9.	<i>When a customer wants to share a form, they cannot send it to parties as a PDF, instead, a link is given that directs parties to the website.</i>	4.43	0.79

Top 3 Worst given Empathy			
Nr.	Scenario	\bar{X}	σ
15.	<i>A customer shared a draft with a colleague but can no longer find it on their iPad.</i>	3.75	0.96
10.	<i>A customer indicates that the password reset is not working. No link is received.</i>	3.80	1.10
11.	<i>For the past few days, a customer's draft is not loading, while other drafts load correctly.</i>	3.83	1.33

Table H.5: The top three best and worst performing scenarios based on their average empathy scores.

H.6 Friendliness

Top 3 Best given Friendliness			
Nr.	Scenario	\bar{X}	σ
4.	<i>A customer wonders if you can work with one account on multiple devices?</i>	5.00	0.00
9.	<i>When a customer wants to share a form, they cannot send it to parties as a PDF, instead, a link is given that directs parties to the website.</i>	4.86	0.38
13.	<i>A customer has lost their form and would like to retrieve it.</i>	4.67	0.52

Top 3 Worst given Friendliness			
Nr.	Scenario	\bar{X}	σ
1.	<i>A customer would like to upgrade from a basic license to a premium license but wonders if this can be done without data loss?</i>	4.00	1.73
11.	<i>For the past few days, a customer's draft is not loading, while other drafts load correctly.</i>	4.17	1.17
10.	<i>A customer indicates that the password reset is not working. No link is received.</i>	4.20	1.30

Table H.6: The top three best and worst performing scenarios based on their average friendliness scores.

Appendix I

BERTScore comparison

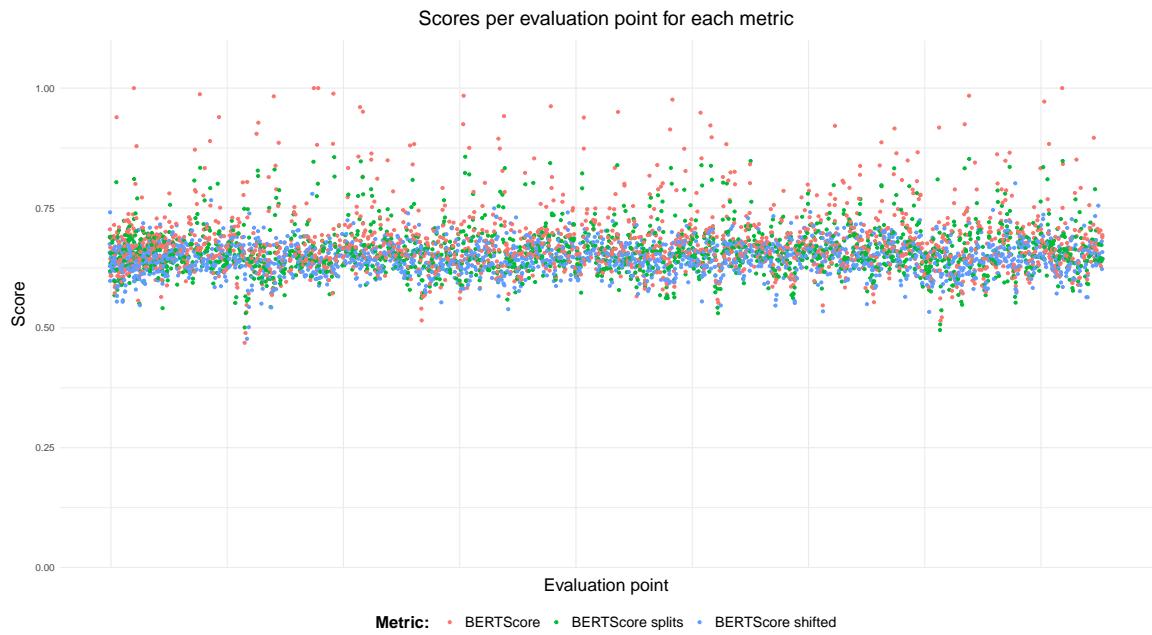


Figure I.1: The BERTscores using different golden responses to measure its sensitivity.

Appendix J

Discrepancies

J.1 Question 2 discrepancy

Question: I have lost my form, can it be retrieved?

Response: Dear,

Thank you for your message. I see that you deleted a form on 1-4-2022. Unfortunately, it is not possible to recover it. I see that you subscribed on 1-4-2022, and since then you have created 1 form. I also see that you have deleted a number of forms, but I cannot see when these were deleted. I advise you to avoid deleting forms in the future and instead archive them. This way, the forms remain visible at all times. I hope this information is sufficient. If you have any further questions, please let me know.

Question: I have not deleted anything?

Response: Dear,

Thank you for your response. I have checked again and see that the form "NEN 3140 - 2021" was deleted on 10-11-2022. I see that you deleted the form on 10-11-2022 at 14:00. I have restored the form for you. You can now find the form again in your organization. I hope this information is sufficient. If you have any further questions, please let me know.

Table J.1: *Discrepancy in the fine-tuning response of question 2.*

J.2 Question 6 discrepancy

Question: Can I add a company logo to the template so that it appears on the page instead of the Incontrol logo?

Response: Hello! Yes, you can definitely add a company logo to the template so that it appears on the page instead of the InControl logo. To do this, follow these steps:

1. Open the template you want to customize.
2. Go to the settings or the editing section of the template.
3. Look for the option to add or change a logo.
4. Upload your company logo and ensure it has the correct dimensions.
5. Save the changes.

If you need further assistance or if something is unclear, feel free to let me know! I'm happy to help you further.

Question: It seems that I can't do this due to my permissions.

Response: Hello! I'm sorry to hear that you're experiencing issues with adding the company logo. To obtain the necessary permissions, you can follow these steps:

1. Check your user role: Ensure that you have the appropriate user role that grants permissions to make changes. This can usually be checked in your account settings.
2. Contact the administrator: If you do not have the correct permissions, contact your account administrator or your IT department. They can assign you the necessary permissions.
3. Support team: If you are unable to reach the administrator or need further assistance, you can always contact our support team. We can help you obtain the correct permissions.

Let me know if you have any further questions or if I can assist you with anything else!

Table J.2: *Discrepancy in the zero-shot response of question 6.*

J.3 Question 22 discrepancy

Question: I shared a draft with a colleague but can no longer find it on my iPad.

Response: Please check if the draft is still listed under 'Shared Drafts' with the person you shared it with. If it is not there, please answer the following questions:

1. What did you do before you lost the draft?
2. What did you do after losing the draft?
3. When was the draft created?
4. When did you realize that the draft was missing?
5. On which device was the draft created?
6. Is the operating system of this device up to date?

Question: I did nothing with the draft before and after I lost it, the draft is a few weeks old and I found out yesterday that it was gone. My device is an iPhone 14 and the app is up to date.

Response: Please check if the draft is still visible to the person you shared it with by asking them if they can see the draft in their "Shared Drafts" section. If they cannot find it, there might be an issue with the sharing process, and you may need to share it again or contact support for further assistance.

Table J.3: *Discrepancy in the few-shot response of question 22.*

Appendix K

Incorrect combination of sources

Question: Could you create a form for me, for an inspection I want to perform?

Context:

- We can also create forms for, for example, inspections (solar panels, e-inspections, or other inspections), and other tasks specifically for you. To do this you can make a form build submission over email, specifically stating your needs, so we can discuss it.
- In our Template Store, we offer various templates that you can customize to your specific needs. Different templates have different sections that may be applicable when creating your inspection form.

Response: We can certainly help you create a form for your inspection. We offer various templates that you can customize to your specific needs. If you have specific requirements for the form, such as certain sections or questions, you can personalize them. If you have any other questions or need further assistance, feel free to let us know!

Table K.1: *An example of how information from the context gets combined in an incorrect way during response generation of a user query. This was a real example from one of the participants, however not all the context that was given to the LLM is displayed here.*

Appendix L

Enhanced templates

L.1 Response creation

You are a friendly and understanding customer support assistant.

****Instructions:****

- Use only the information in the 'Context' below to answer the question, but don't mention this, and do NOT make anything up.
- Do NOT give unnecessary information and try to be concise, but still appear friendly, only answer using necessary information and don't go beyond.
- Do NOT include any placeholders like [NAME], [EMAIL], [IBAN], etc.
- Prioritize contents of one source over a combination so all the necessary information in that source is used.
- Prioritize context from Articles over Tickets and Emails.

****Format:****

- Respond in the user's language: {language}.
- For responses ALWAYS maintain the following structure: acknowledge the problem, answer it, tell them that if needed they can ask more questions.

****Context:** {context}**

****Question:** {question}**

****Your Answer:****

Table L.1: *The enhanced template use for the response generator. The language, context, and question are variables given to the template during a conversation.*

L.2 Chat history

You are an expert in rewriting a chat history in combination with a question into a standalone question.

****Instructions:****

- If the follow-up input has nothing to do with the given Chat History, then disregard the Chat History; otherwise, incorporate relevant parts for the Chat History.

****Format:****

- Rephrase the Follow-Up Input into a 'Standalone Question' in {language} if needed

Chat History: {chat history}

Follow-up Input: {question}

Standalone question:

Table L.2: *The enhanced template used for rephrasing a question and incorporating chat history in phase 2. The language, chat history, and question are variables given to the template during a conversation.*

L.3 Relevance

Your goal is to assess the relevance of the 'Response' to the 'Question' provided below and determine the appropriate action.

****Instructions:****

1. ****Determine Relevance Between 'Question' and 'Response':****
 - ****Direct Relevance:**** Check if the 'Response' addresses the 'Question'.
 - ****Irrelevant:**** If the 'Response' is not relevant to the 'Question', politely ask the user for additional information.

2. ****Verify Context Appropriateness:****

- ****Customer Support Context:**** Determine if the 'Question' pertains to customer support.
- ****Irrelevant:**** If it is a irrelevant question politely inform the user that you cannot assist with that inquiry.

****Final Output:****

- ****All Checks Pass:**** If all the above checks are satisfied, ONLY output the 'Reponse', nothing else.
- ****Otherwise:**** Follow the appropriate response guidelines as per the instructions above.

****Format:****

- Do not include any additional text or explanations.
- Respond in the user's language: {language}.

****Question:**** {question}

****Response:**** {response}

****Your Answer:****

Table L.3: *The template used for the relevance check in the handover module. The language, question, and response are variables given to the template during a conversation.*

L.4 Specificity

Your goal is to determine if a user's question and your answer relate to a specific customer issue using the 'Question' and 'Response' at the bottom.

****Instructions:****

1. ****Identify Specificity:****

- ****Specific Issue:**** If the 'Question' and 'Response' pertain to a problem that is too specific and cannot be resolved without specific details, acknowledge that you cannot resolve it directly.

- ****General Issue:**** If the topic is general and not tied to something specific for that customer, no further action is needed.

2. ****Response Guidelines for Specific Issues:****

- ****Understand and Stating:**** Be understanding of their problem.

- ****Mention Potential Solutions:**** If the 'Response' below includes solutions that can be useful to the 'Question' use these as potential suggestions.

- ****Redirect:**** Tell them that if this doesn't work they can submit a ticket.

****Final Output:****

- ****Specific Issue:**** Provide the redirect message with caveated solutions.

- ****General Issue:**** Return ONLY the 'Response'.

****Format:****

- Respond in the language: {language}.

- Do not include any additional text or explanations.

****Question:**** {question}

****Response:**** {response}

****Your Answer:****

Table L.4: The template used for the specificity check in the handover module. The language, question, and response are variables given to the template during a conversation.

Appendix M

Handover module check examples

M.1 First relevance check

Question: I have started a trial and wk.

Response: Hello! It seems that you have started a trial, but I need a bit more information to better assist you. Can you tell me exactly where you need help? I am ready to help you with all your questions!

Table M.1: *An example of how the first relevance check was initiated, when the user accidentally asked an unfinished question.*

M.2 Second relevance check

Question: I want an inspection report of 4 A4 pages in Word. How much time does that take?

Response: Unfortunately I cannot help you with that question.

Table M.2: *An example of how the second relevance check was initiated, as the question was not deemed relevant in the context of customer support.*

M.3 Specificity check

Question: I have a draft that won't load while my other drafts do load, how come?

Response: Good afternoon! I'm sorry to hear that you're experiencing problems loading your draft.

Here are some general suggestions you can try:

- It might be that the draft isn't properly synchronized between your devices. Sometimes, a draft might be on the server but not correctly loaded on your device.
- Try sharing the draft to another email address or device to see if it loads there.

If these suggestions don't work, you can submit a support ticket.

Table M.3: *An example of how the specificity check was initiated, when the user asked a question that was too specific.*

Appendix N

Repetition example specificity check

Question: I can't find a filled form that I shared with my colleague. Where has it gone? I hope it hasn't been lost, because there's a lot of work in it.

Response: Good morning! How unfortunate that you can no longer find the shared form.

- It may be that the form is still somewhere in your account, for example under 'Drafts' or 'Completed Documents'.
- Also try checking if the form has accidentally ended up in the trash.
- If you still can't find the form, it might help to see if your colleague still has the form in their account.

These suggestions may help, but they might not apply to your situation. If this doesn't work, you can submit a support ticket.

Question: My colleague is sick, so I can't ask them. I would like to retrieve the draft and fill it in myself.

Response: Hello! It seems that you have a specific problem with retrieving a draft form. Here are some general suggestions you can try, but keep in mind that these may not apply to your situation:

- Try logging into the account that originally created the form and check if the draft is under the 'Drafts' or 'Completed Documents' section.
- If the form is not visible there, it may not have been saved or ended up in your trash.
- Also try logging in via a web browser, sometimes drafts are more visible there.

If these suggestions do not work, you can submit a support ticket.

Table N.1: *An example of how a conversation felt repetitive because of the specificity check in some cases.*



2024
DUBLIN

2 - 4 October, 2024 / Dublin, Ireland

MULTIMODAL AI: THE SIXTH SENSE FOR CYBER DEFENCE

Younghoo Lee

Sophos, Australia

younghoo.lee@sophos.com

ABSTRACT

Given the growing complexity and diversity of cyber threats across various media formats, there is a pressing need to enhance cybersecurity defence mechanisms. This paper explores the potential of multimodal AI in addressing these evolving threats. Specifically, we analyse the use of multimodal AI for detecting phishing emails and classifying NSFW (Not Safe For Work) websites. By integrating and analysing both textual and visual data, multimodal AI can uncover hidden intentions that traditional methods might miss. Additionally, the analysed data supports classification predictions, which traditional machine learning lacks. We examine LLM-based multimodal AI and embedding models, demonstrating their effectiveness in processing diverse data modalities to deliver superior classification performance. Our experimental results show that multimodal AI significantly outperforms traditional machine learning approaches in both phishing detection and NSFW classification while providing added explainability, thus offering a more robust and effective defence mechanism for today's cybersecurity landscape.

1. INTRODUCTION

As cyber threats become increasingly complex and leverage diverse media formats, there is an urgent need for innovative cybersecurity defence mechanisms. This paper investigates the capabilities of multimodal AI, a sophisticated tool that transcends traditional text-based analysis by integrating data from text and images to uncover hidden threats. We evaluate the effectiveness of multimodal AI in detecting phishing emails and categorizing NSFW websites.

1.1 Multimodal AI for email phishing detection

Phishing emails often contain urgent prompts to verify delivery details or address payment discrepancies and typically include malicious URLs aimed at capturing user credentials. They closely mimic legitimate communications, making detection difficult for conventional spam filters. Although attackers may use generative multimodal AI to fabricate convincing phishing emails, the same AI technology can be harnessed to detect these fraudulent communications. Traditional machine learning (ML) models undergo phases like data collection, feature extraction (often using TF-IDF [Term Frequency-Inverse Document Frequency]), and model training and deployment. However, models like Random Forest or XGBoost, which rely on familiar words from training datasets, may overlook new phishing formats. Multimodal AI like GPT-4o from *OpenAI* or Gemini from *Google*, trained on diverse datasets, can identify phishing attempts by analysing email headers, content, and visual elements such as logos and images. These generative AI models can recognize new phishing emails efficiently, even without specific training data.

1.2 Embeddings as machine learning features for website classification

NSFW classification involves categorizing websites into groups like gambling, weapons, sports and games, with categories like gambling and weapons being NSFW. In ML, embeddings transform words or phrases into numerical vectors that capture meanings and relationships. Traditional approaches involve data collection, HTML content extraction, and feature creation using techniques like TF-IDF. However, these methods are limited by their inability to interpret non-text objects within images. Multimodal AI addresses this limitation by generating detailed descriptions of screenshots, capturing both textual and visual information, thus improving classification accuracy by providing a holistic representation of the website.

By leveraging the strengths of multimodal AI and advanced embedding techniques, our approach offers a robust solution to modern cybersecurity challenges. In the following sections, we will explore the details of our methodology, provide experimental validation, and discuss case studies that illustrate the efficacy of multimodal AI in both phishing detection and NSFW website classification.

2. DETECTING PHISHING EMAILS

This section highlights the limitations of current text-based phishing detection methods and demonstrates how multimodal AI decodes suspicious signals in email text and visual elements.

2.1 Challenges in detecting phishing emails

Phishing emails often include urgent requests to verify details or resolve payment issues and embed URLs for credential harvesting. Attackers craft these emails to resemble legitimate communications, incorporating brand images and genuine text while replacing legitimate URLs with malicious ones.

Traditional machine learning methods involve several steps. Initially, during data collection and feature extraction, ML models require extensive datasets for effective learning. Features such as text tokens (words) are extracted, and techniques like TF-IDF are used to assign importance to words. For instance, terms like 'urgent', 'payment', and 'verify' may carry higher TF-IDF scores due to their frequent occurrence in phishing emails. Then, during the training and evaluation phases,

the dataset is split into training and test sets, with the model learning from the training data and being validated against the test data. Finally, the deployment phase entails monitoring incoming emails using the trained model. However, traditional models like Random Forest or XGBoost are limited because they depend on familiar words from the training dataset and may overlook new phishing emails that utilize unfamiliar words or images. For example, if a model lacks samples targeting the customers of Australian banks, it will likely miss phishing emails aimed at them. Models like Random Forest or XGBoost trained with TF-IDF features rely on frequent words in the training dataset to identify suspicious emails by detecting known combinations of suspicious words.

Multimodal AI based on large language models (LLMs), trained on massive datasets spanning numerous websites and brands, can identify phishing attempts by scrutinizing email headers, content, and visual elements such as logos and images. With specific examples and detailed instructions, generative models can adapt to recognize new and previously unseen phishing emails efficiently. Even without specific training data on novel phishing formats, these models can detect suspicious signals and the intent behind the email.

2.2 Leveraging multimodal AI for detection

Existing security procedures involve manual examination of suspicious emails for any signs of impersonation or malicious intent, looking at elements such as email headers, body content, and various UI features like brand logos. This process also includes analysing sender domains and URLs. These labour-intensive procedures can now be automated using large language models.

LLMs, trained on vast and diverse datasets, are capable of analysing suspicious elements within an email, understanding its intent, recognizing brand identities, and identifying impersonation attempts. They can do this by examining both the content and visual aspects of an email. These models can identify dubious domains or URLs without needing additional training or external databases. With just a few examples, they can detect new phishing emails, a feat that traditional machine learning methods can only achieve with large-scale training datasets.

We use GPT-4o, a multimodal AI from *OpenAI*, to inspect emails as part of our phishing detection methodology. We provide the LLM with specific instructions to uncover suspicious elements from the email data and screenshot images of the email or its URLs, which are generated by rendering the email's HTML content. The report that we get from our instructions provides a classification score indicating the presence of any suspicious elements. Such a report can reveal the suspicious nature of an email sender's display name or domain and any impersonation attempts in the URLs or images, something that traditional methods are unable to provide.

The GPT-4o instruction for phishing detection:

Analyse the email data and the accompanying screenshot images if available and generate a report in the following JSON format to detect and quantify the level of suspiciousness in the email, considering common indicators of phishing, fraud, or malicious intent.

- ‘summary’: A brief overview of the content of the email.
- ‘suspicious_elements_domain_content’: Review the sender’s email domain for consistency with the content of the email. Look for domain spoofing or the use of domains similar to reputable domains to mislead the recipient.
- ‘suspicious_text’: Highlight text in the email that indicates a sense of urgency, incites immediate action, or otherwise seems intended to manipulate the recipient emotionally.
- ‘suspicious_links’: Catalog top 3 links found in the email and assess each for potential malicious intent, especially those directing to suspicious or misspelled domains.
- ‘suspicious_images’: Analyse the accompanying screenshots for any indicators of phishing.
- ‘impersonated_target_in_image’: The impersonated brand or target in the images, the sender’s domain does not match with the target.
- ‘suspicious_score’: Provide an overall score between 0.0 (not suspicious) and 1.0 (extremely suspicious) based on the aggregated suspicious indicators found in the email.

2.3 Case studies

Case: Costco

Our approach identified a phishing email targeting *Costco* clients. The method detected suspicious signals in the email header and content, resulting in a suspicious score of 0.8 (on a scale from 0.0 to 1.0). When the screenshot image was analysed with GPT-4o, the multimodal AI recognized additional suspicious signals from both the email data and image, increasing the suspicious score to 0.9 – sufficient to mark it as malicious.

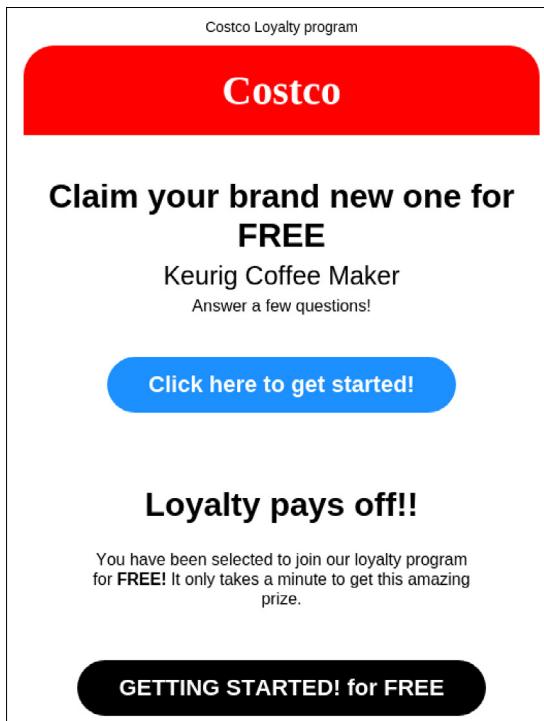


Figure 1: Costco phishing email.

The following GTP-4o report for the Costco email identifies the sender's suspicious domain and intention:

```
{
  "subject": "RE: You have won a Keurig Coffee Maker. #ID84656",
  "summary": "Email claiming the recipient has won a Keurig Coffee Maker and prompts them to click a link to claim the prize.",
  "sender": "Costco® <teamsupport-Lvmeee-923309212439507@ocxpnbnjfwuu.com.au>",
  "suspicious_elements_domain_content": "The domain 'ocxpnbnjfwuu.com.au' is not consistent with legitimate Costco domains, indicating potential domain spoofing.",
  "suspicious_elements_links_content": "The link 'https://storage.googleapis.com/' does not align with Costco's official URLs and uses a Google Cloud Storage domain to obscure the actual destination.",
  "suspicious_text": "Phrases like 'Claim your brand new one for FREE!' and 'You have been selected to join our loyalty program for FREE!' create a sense of urgency and pressure the recipient to take immediate action.",
  "impersonated_target_in_text": "Costco",
  "suspicious_score": 0.8
}
```

The following GPT-4o report for the Costco email and its screenshot image recognizes the additional impersonation attempt in visual elements.

```
{
  ...
  "suspicious_images": "The screenshot images contain visual elements that mimic genuine Costco branding. There are call-to-action buttons to claim a prize, which is a common phishing tactic. The URLs do not match with the sender's legitimate domain.",
  "impersonated_target_in_image": "Costco",
  "suspicious_score": 0.9
}
```

Case: Other brands

Similarly, our approach detected phishing emails impersonating *PayPal* and *FedEx*. The method identified the suspicious domain and UI elements within both the email and its screenshot image.

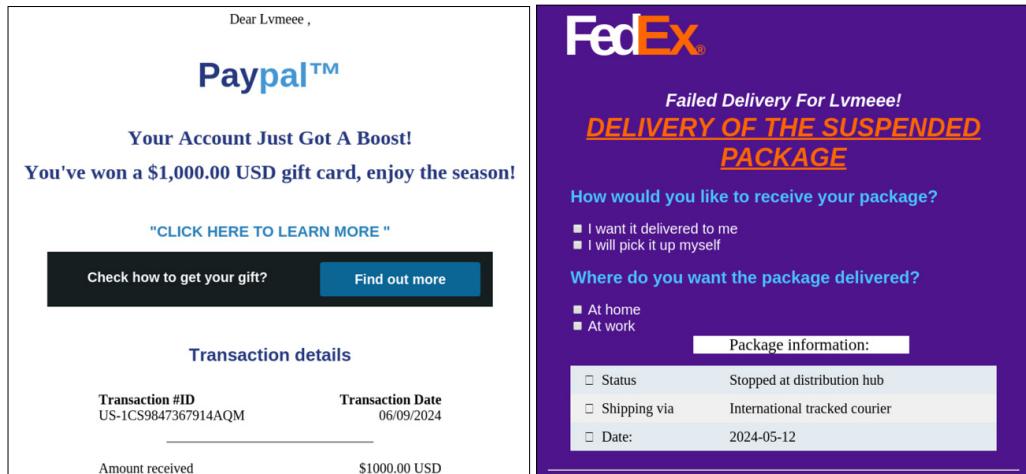


Figure 2: PayPal and FedEx phishing emails.

2.4 Experiment results

We conducted an experiment to compare the effectiveness of our phishing detection approach against baseline machine learning models: Random Forest and XGBoost. These models utilized TF-IDF features derived from raw email data. The dataset, consisting of approximately 2,000 balanced samples, was split into 70% for training and 30% for testing in a random split experiment. Additionally, to evaluate the models' performance on unseen samples, we conducted a brand split experiment by training on emails impersonating certain brands (e.g. *Google* and *Microsoft*) and testing on emails targeting other brands (e.g. *PayPal* and *Costco*).

The results, which are summarized in Tables 1 and 2, indicate the following:

- When evaluated on randomly split data, both Random Forest and XGBoost demonstrated high performance with an F1 score close to 0.991.
- On unseen samples, however, their performance dropped significantly, with Random Forest achieving an F1 score of 0.529 and XGBoost slightly better at 0.657. This indicates that these models struggled to generalize effectively to new threats during the brand split experiment.

Random Forest	Random split	Brand split
Precision	1.000	0.583
Recall	0.983	0.401
F1	0.991	0.529

Table 1: Performance of Random Forest with TF-IDF features.

XGBoost	Random split	Brand split
Precision	1.000	0.583
Recall	0.966	0.752
F1	0.983	0.657

Table 2: Performance of XGBoost with TF-IDF features.

Our approach with GPT-4o, which integrates both textual and image data descriptions for the test dataset in the brand split, significantly outperformed the baseline models. Through the fusion of features from both text and images, we were able to achieve superior detection performance against unseen threats. The results are presented in Table 3.

GPT-4o	Email text input	Email text and image input
Precision	1.000	0.982
Recall	0.936	0.961
F1	0.967	0.971

Table 3: The performance of GPT-4o for brand split with text and image input.

In conclusion, our multimodal AI approach, which leverages both text and image inputs, offers a more robust solution for detecting phishing attempts, particularly when facing unseen threats. The use of both text and image features proved to be more effective, as reflected by the highest F1 score of 0.971 in the brand split test set.

3. CLASSIFYING NSFW WEBSITES

The proliferation of NSFW content on the internet poses a significant challenge for web content classification. Embedding models powered by multimodal AI interpret the ‘language’ of images, facilitating the robust and accurate classification of NSFW websites. This section outlines the application of embeddings for NSFW classification, encompassing categories like gambling, weapons, sports and games, with gambling and weapons classified as NSFW.

In machine learning, embeddings are numerical representations of data. For text data, embeddings convert words or phrases into numerical vectors that capture their meanings, similarities and interrelations. For instance, the words ‘gambling’ and ‘casino’ would be proximate in the embedding space, indicating their similarity. In a traditional approach, initial steps involve data collection, extracting HTML content from URLs, and tokenizing words. Features like TF-IDF are used to represent the text numerically. For instance, terms such as ‘bet’, ‘poker’ and ‘jackpot’ would have higher TF-IDF scores on a gambling site. Additional features can be derived using OCR (Optical Character Recognition) to extract text from web page images. However, a limitation of traditional OCR is its inability to interpret non-text objects, missing vital details embedded within images.

3.1 Leveraging contextual embedding

Leveraging multimodal AI, we can obtain detailed descriptions of screenshots, which are critical for identifying a website’s purpose. This approach addresses the limitation of traditional text analysis by including visual context. For instance, it can describe text as well as visual elements like poker chips or roulette tables on a gambling site, providing additional explainability for our classification problem. The textual description of the HTML content and the screenshot can be used as input to generate TF-IDF features in the traditional ML pipeline. Furthermore, these text descriptions can be transformed into more representative numerical vectors using LLM-based embeddings. These contextual embedding vectors help to identify similar websites and can be used as features to train traditional models like Random Forest or XGBoost.

We employ GPT-4o to generate detailed descriptions of the HTML content and its corresponding screenshots. *OpenAI*'s embedding model, text-embedding-3-small, is then used to create embedding vectors from these descriptive texts. These embeddings are subsequently used to train ML models, such as Random Forest and XGBoost, for the web categorization task. Our experiments show that those models trained with embedding features significantly outperform those using traditional word-based TF-IDF features.

3.2 Case studies

Case: Gambling website

A gambling website was misclassified as a sports site by a Random Forest model trained with TF-IDF features from raw HTML data. This misclassification likely occurred due to the sports-related content present on the site. However, our approach, which includes descriptions from both HTML content and images, enabled the same Random Forest model to correctly classify it as a gambling site.

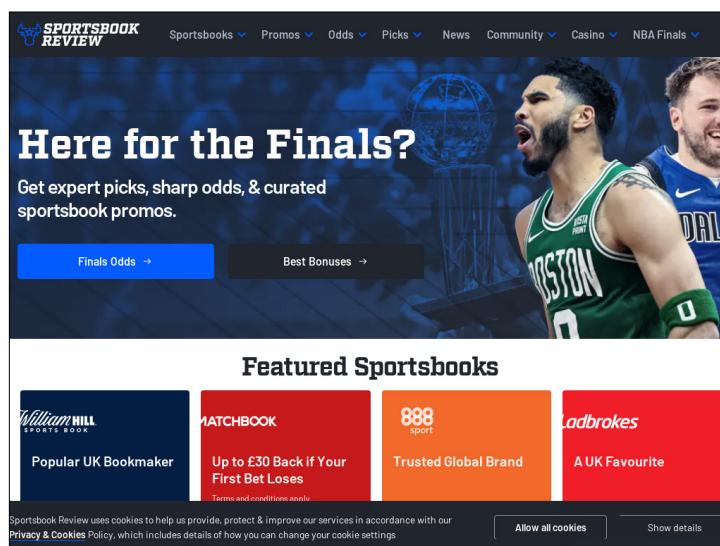


Figure 3: Gambling website.

The following descriptions from GPT-4o demonstrate its ability to understand the content of multimodal AI. GPT-4o accurately described the web page and provides relevant details for the task.

The description from the screenshot:

The screenshot image contains the following main objects:

1. Two basketball players in sports jerseys.
2. A large trophy in the background.
3. Text phrases such as ‘Here for the Finals?’, ‘Get expert picks, sharp odds, & curated sportsbook promos’, ‘Finals Odds’, ‘Best Bonuses’, and ‘Featured Sportsbooks’.
4. Logos and names of various sportsbooks (e.g. William Hill, Matchbook, 888sport, Ladbrokes).

The description from the HTML data:

Here are the main features:

1. Title:
 - The title of the page is ‘Sportsbook Review | SBR - Sports Betting Experts since 1999’.
2. Keywords and Content:
 - The keywords and phrases such as ‘sports betting’, ‘odds’, ‘picks’, ‘reviews’, ‘betting community’, ‘sportsbooks’, ‘sports betting apps’, ‘sports betting promos’, ‘sports betting landscape’, and ‘sports betting bonuses’ are repeated multiple times throughout the HTML content.

Case: non-English gambling website

A particular website was initially categorized as a sports site using the Random Forest model trained with raw HTML data. However, when the Random Forest model was trained with embeddings, it accurately identified the site as a gambling site. Notably, GPT-4o captured all relevant details, even though the site content was not in English. The generated description stated, ‘Text and icons related to sports, such as “SPORTY” and “SÁZKY LIVE”, and scores of soccer matches (e.g. Everton vs. Tottenham). These elements indicate that the website is related to sports, specifically soccer, and potentially sports betting due to the presence of match scores and the context of the app being promoted.’

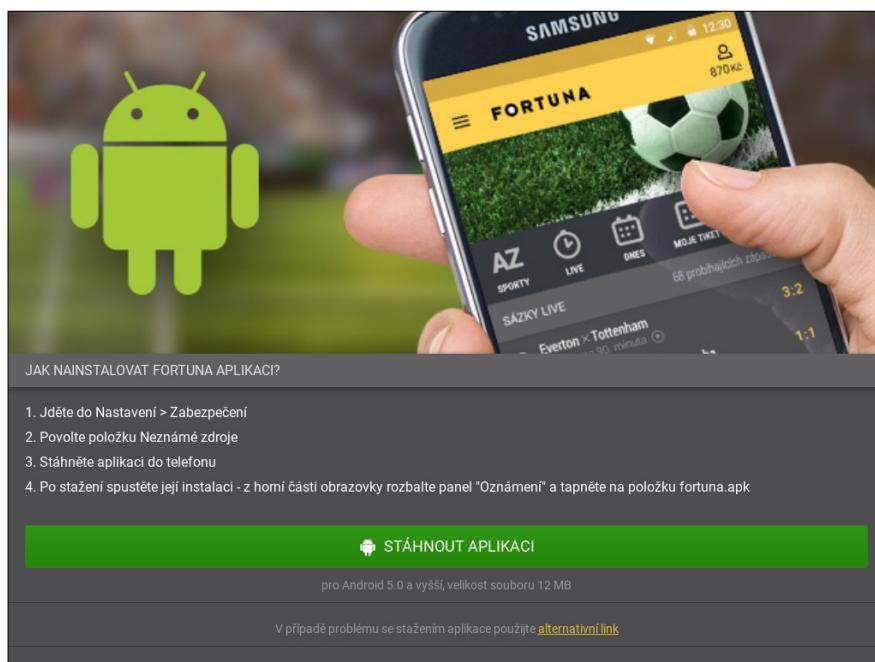


Figure 4: Non-English gambling website.

Case: Weapons website

Some websites with limited textual information were correctly classified as weapons sites due to the image-based information captured by multimodal AI.

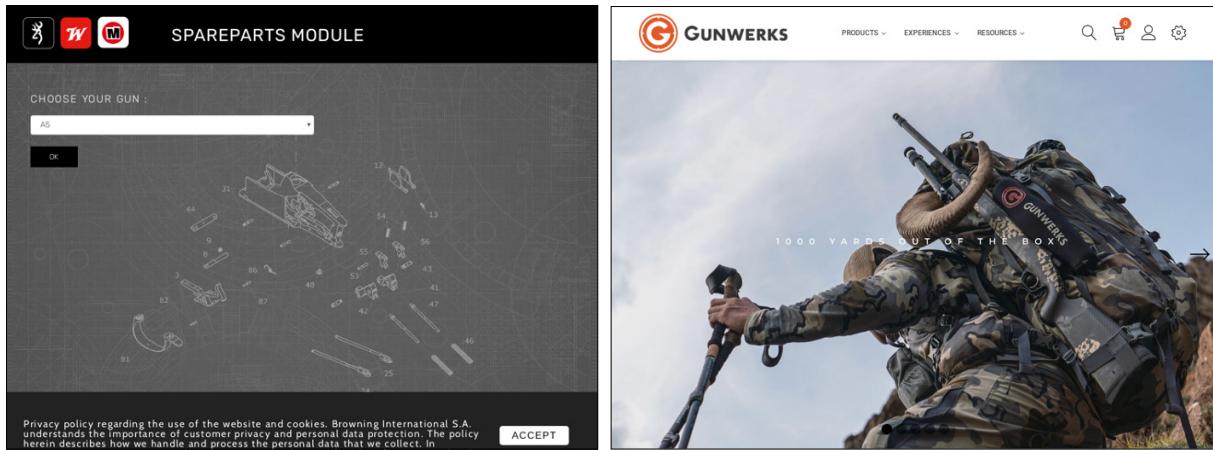


Figure 5: Weapons websites.

3.3 Experimental results

We conducted an experiment to compare our proposed approach with baseline models, specifically Random Forest and XGBoost models, utilizing TF-IDF features derived from raw HTML data and text description from GPT-4o. While it's feasible to use a vector database with embedding vectors for searching similar websites, we opted to train classification ML models with both word-based TF-IDF vectors and LLM-based embedding vectors to assess their performance in classification tasks.

The Random Forest model exhibited various levels of performance with different input features, as shown in Table 4. For the random split experiment, the dataset was divided into a 70% training set and a 30% test set; this dataset consisted of approximately 2,000 samples. To prevent overlap of similar websites, the dataset was deduplicated by domains. Initially, using TF-IDF features from raw HTML data, the model achieved an F1 score of 0.84. When TF-IDF features from HTML text descriptions generated by GPT-4.0 were utilized, the F1 score improved to 0.92. With LLM embedding features from the same text descriptions, the F1 score further increased to 0.94. The highest performance was noted when embeddings from both HTML content and screenshots were combined, resulting in an F1 score of 0.96. Thus, the inclusion of more detailed and combined feature embeddings significantly enhanced the model's classification performance. Table 5 shows similar trends for the XGBoost model.

Random Forest	TF-IDF with raw HTML	TF-IDF with HTML text	Embedding with HTML text	Embedding with HTML and Image text
Precision	0.891	0.939	0.958	0.970
Recall	0.813	0.912	0.930	0.951
F1	0.841	0.924	0.942	0.960

Table 4: The classification performance of Random Forest.

XGBoost	TF-IDF with raw HTML	TF-IDF with HTML text	Embedding with HTML text	Embedding with HTML and Image text
Precision	0.901	0.907	0.960	0.959
Recall	0.864	0.907	0.955	0.955
F1	0.880	0.907	0.958	0.957

Table 5: The classification performance of XGBoost.

These experimental results clearly demonstrate that embeddings derived from combined descriptions of HTML and images provide more efficient representations of websites. This, in turn, enhances the classification performance of general-purpose machine learning models, as evidenced by the improved precision, recall, and F1 scores across both Random Forest and XGBoost models.

4. CONCLUSION

Multimodal AI offers a powerful approach to navigating the increasingly complex world of cybersecurity threats. By merging data from various modalities, such as text and images, multimodal AI systems provide a more comprehensive understanding of data intent. Our research in phishing detection and NSFW website categorization shows that multimodal AI significantly outperforms traditional text-based machine learning techniques. Specifically, large language models (LLMs) based on multimodal AI and advanced embedding techniques exhibit superior classification capabilities, even against previously unseen threats, while also providing decision explainability. This research underlines the necessity of incorporating multimodal AI into cybersecurity structures for more precise and efficient threat detection and categorization. The concrete results detailed in this paper underscore the substantial advantages of multimodal AI, setting the stage for its wider use in protecting digital landscapes.

REFERENCES

- [1] Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Amodei, D. Language Models are Few-Shot Learners. arXiv preprint arXiv:2005.14165. 2020. <https://arxiv.org/abs/2005.14165>.
- [2] Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Bengio, Y. Generative Adversarial Nets. Advances in Neural Information Processing Systems, 27, 2672-2680. 2014. <https://papers.nips.cc/paper/5423-generative-adversarial-nets>.
- [3] Ho, T. K. Random Decision Forests. Proceedings of the Third International Conference on Document Analysis and Recognition, 278-282. 1995. <https://ieeexplore.ieee.org/document/598994>.
- [4] Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785-794. 2016. <https://dl.acm.org/doi/10.1145/2939672.2939785>.
- [5] Kowsari, K.; Meimandi, K. J.; Heidarysafa, M.; Mendu, S.; Barnes, L. E.; Brown, D. E. Text Classification Algorithms: A Survey. Information, 10(4), 150. 2019. <https://www.mdpi.com/2078-2489/10/4/150>.
- [6] Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient Estimation of Word Representations in Vector Space. arXiv preprint arXiv:1301.3781. 2013. <https://arxiv.org/abs/1301.3781>.
- [7] OpenAI. ChatGPT: Optimizing Language Models for Dialogue. 2023. <https://openai.com/research/chatgpt>.
- [8] Raff, E.; Barker, J.; Sylvester, J.; Brandon, R.; Catanzaro, B. Malware Detection by Eating a Whole EXE. arXiv preprint arXiv:1710.09435. 2018. <https://arxiv.org/abs/1710.09435>.
- [9] Saber, M. I.; Chakraborty, M.; Ghose, M. K. Detection of Phishing Emails Using Machine Learning and Deep Learning. Journal of King Saud University-Computer and Information Sciences, 33(9), 1166-1176. 2021. <https://doi.org/10.1016/j.jksuci.2019.06.001>.
- [10] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Polosukhin, I. Attention is All you Need. Advances in Neural Information Processing Systems, 30, 5998-6008. 2017. <https://papers.nips.cc/paper/7181-attention-is-all-you-need>.
- [11] Zhang, H.; Cisse, M.; Dauphin, Y. N.; Lopez-Paz, D. mixup: Beyond Empirical Risk Minimization. International Conference on Learning Representations. 2018. <https://arxiv.org/abs/1710.09412>.

On Mitigating Code LLM Hallucinations with API Documentation

Nihal Jain

Amazon Web Services, USA

nihjain@amazon.com

Robert Kwiatkowski*

robert.kwiatkowski@columbia.edu

Baishakhi Ray

Amazon Web Services, USA

rabaisha@amazon.com

Murali Krishna Ramanathan

Amazon Web Services, USA

mkraman@amazon.com

Varun Kumar

Amazon Web Services, USA

kuvrun@amazon.com

Abstract—In this study, we address the issue of API hallucinations in various software engineering contexts. We introduce CloudAPIBench, a new benchmark designed to measure API hallucination occurrences. CloudAPIBench also provides annotations for frequencies of API occurrences in the public domain, allowing us to study API hallucinations at various frequency levels. Our findings reveal that Code LLMs struggle with low frequency APIs: for *e.g.*, GPT-4o achieves only 38.58% valid low frequency API invocations. We demonstrate that Documentation Augmented Generation (DAG) significantly improves performance for low frequency APIs (increase to 47.94% with DAG) but negatively impacts high frequency APIs when using sub-optimal retrievers (a 39.02% absolute drop). To mitigate this, we propose to intelligently trigger DAG where we check against an API index or leverage Code LLMs’ confidence scores to retrieve only when needed. We demonstrate that our proposed methods enhance the balance between low and high frequency API performance, resulting in more reliable API invocations (8.20% absolute improvement on CloudAPIBench for GPT-4o).

Index Terms—code generation, hallucinations, retrieval augmented generation, API invocations

I. INTRODUCTION

Large Language Models for code generation (Code LLMs) are being increasingly used by developers in industries [2]–[4]. Major software companies like Meta, Google and Amazon are actively adopting AI-powered coding assistants relying on these models to enhance the productivity of their vast software development teams [5]–[7]. As these tools become integral to modern development workflows, ensuring that Code LLMs generate high quality code is essential. However, as noted in [8], these tools often introduce critical errors in code that developers struggle to localize and rectify, making debugging more challenging.

We show that a major mode of failure of Code LLMs is in generating invalid API invocations, *i.e.*, *API hallucinations*. These inaccuracies – where models invoke non-existent APIs, misuse API syntax, or generate fictitious arguments – pose significant challenges for developers. Software teams may rely heavily on third-party API providers, such as Amazon Web Services (AWS) and Microsoft Azure for cloud services. For

such developers using Code LLMs, API hallucinations can lead to critical failures in production and increased debugging costs. Hence, in this work, we focus on the occurrence of API hallucinations in the context of cloud-based APIs.

The problem of API hallucinations is exacerbated by the rapid evolution of APIs, including frequent updates and deprecation of existing APIs [9]. Consequently, new, updated, or infrequently used APIs (*i.e.*, *low frequency APIs*) are more prone to hallucinations (see Figure 1 (left)). To systematically measure the prevalence of these errors, we introduce CloudAPIBench, a benchmark specifically designed to evaluate API hallucinations for major cloud providers like AWS and Microsoft Azure. This makes CloudAPIBench a practical tool for measuring API hallucinations in real-world software engineering contexts that rely heavily on cloud services.

Further, we explore mitigation strategies for API hallucinations. When uncertain about API usage, human developers frequently rely on API documentation. Likewise, we hypothesize that Code LLMs should consult these resources under uncertainty. Hence, to address API hallucinations, we adopt retrieval augmented generation with documentation, *i.e.*, **Documentation Augmented Generation (DAG)**, which has shown early promise [10], [11].

However, DAG may be unnecessary when APIs are stable or well-covered in the model’s training data (*i.e.*, *high frequency APIs*)—we find that DAG with suboptimal retrievers indeed degrades performance for high frequency APIs, supporting that LLMs are sensitive to irrelevant information [12], [13]. As such, we also present two simple yet effective strategies that can be easily adapted with DAG to address such pitfalls.

Figure 1 (right) demonstrates how the frequency of an API’s occurrence in the public domain affects Code LLMs. We analyze the perplexity of StarCoder2-15B (base model) [14] on API tokens across two frequency groups: *low* (≤ 10 occurrences in training data: The Stack v2) and *high* (≥ 100 occurrences), with detailed frequency descriptions in Section II-B. The base model excels with high frequency APIs but struggles with low frequency ones. While DAG enhances performance for low frequency APIs, it compromises high frequency API performance due to occasional irrelevant augmentations from

*Work done while at Amazon

```

import boto3

# initialize bedrock client
client = boto3.client("bedrock")

# Create a job to fine-tune a base model
response = client.create_job(
    jobName="my-job",
    jobType="FineTune",
    ...
)

```

Target API: `create_model_customization_job`

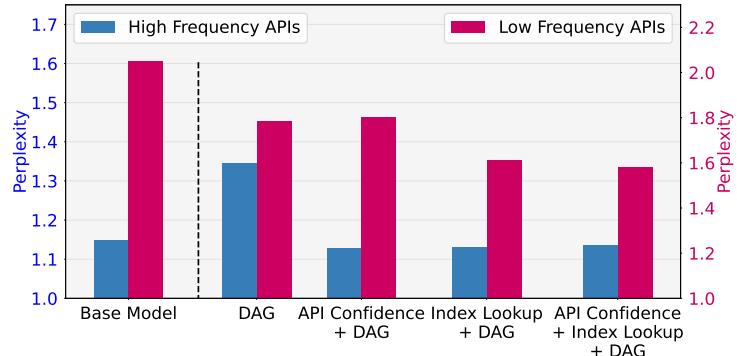


Fig. 1: Introduction. (**Left**) A CloudAPIBench task (yellow) and StarCoder2-15B’s response (red) are displayed. The target is a recently released AWS API [1], *i.e.*, a *low frequency* API. Due to limited training on such APIs, the Code LLM hallucinates a non-existent API invocation. (**Right**) Given a prompt from CloudAPIBench, we measure the perplexity of the target API tokens using StarCoder2-15B (*lower is better*). The base model handles high frequency APIs well but falters with low frequency ones. While DAG (with imperfect retrievers) improves low frequency API performance, it hurts high frequency API performance due to irrelevant augmentations. This paper’s methods and analyses address this limitation of DAG.

suboptimal retrievers: these distract the Code LLM’s reliance on its internal knowledge, which is sufficient for high frequency APIs. To address DAG’s limitations, we explore methods such as inspecting model confidence scores [15], [16] and validating model generations against an API index before retrieval. These strategies effectively mitigate DAG’s drawbacks, enhancing the reliability of Code LLMs.

We outline our key contributions and the structure of the paper as follows:

- We introduce CloudAPIBench to systematically study real-world API hallucinations; this benchmark evaluates API hallucinations across major cloud SDK APIs, *i.e.*, those by AWS and Azure (**Section II**).
- We present a thorough study of DAG to enhance CloudAPIBench performance, identifying the parts of documentation that reduce hallucinations and quantifying the impact of other retrieval components on model efficacy (**Section III**).
- We characterize scenarios where DAG may degrade performance and discuss selective retrieval methods to improve Code LLMs’ utilization of documentation (**Section IV**).

We believe this is the first work to measure and characterize real-world API hallucinations for various Code LLMs and explore strategies to mitigate this issue through documentation.

II. API HALLUCINATIONS & CloudAPIBench

We first comment on the impact of API hallucinations in and then introduce CloudAPIBench.

A. Impact of API Hallucinations

Reference [17] identify that API hallucinations constitute up to 15% of all hallucinations in state-of-the-art Code LLMs, influencing cloud software engineering where code is API-intensive. These hallucinations can propagate errors, creating a snowball effect [18]. For *e.g.*, a hallucinated API call can lead to hallucinated handling of its response in subsequent code

Model	Edit Similarity to GT (%) (↑)	
	Correct API	Hallucinated API
StarCoder2-2B	40.29	36.61
StarCoder2-7B	42.05	40.27
StarCoder2-15B	43.85	39.67

TABLE I: Impact of API hallucinations. Code LLMs tend to diverge further away from the groundtruth code when supplied with a hallucinated API name in the prompt.

segments, compounding the problem [19]. As adoption of Code LLMs grows, the cognitive burden on developers increases [8], as they must trace and rectify both the initial hallucination and all affected code segments.

We provide concrete evidence that API hallucinations lead to code failures in Table I. We sample 171 Python programs with AWS API invocations from the Stack v2 and cut the program after the name of the API; the following code is considered as the groundtruth and we prompt Code LLMs to continue the incomplete program. When the name of the API in the incomplete program is replaced with a hallucinated name (*e.g.*, `get_item` → `fetch_item`, *etc.*), the model generates code that diverges further away from the groundtruth as measured by Edit Similarity [20] in Table I, potentially introducing errors that need to be fixed by developers. Given this impact of API hallucinations, it is critical to explore methods for their detection and mitigation, as done in this work.

B. CloudAPIBench

Current benchmarks evaluate various programming skills such as problem solving [3], [21], repository-level coding [20], [22], and tool usage [11], [23]. However, a comprehensive benchmark for assessing real-world API hallucinations in software engineering remains absent. To address this, we

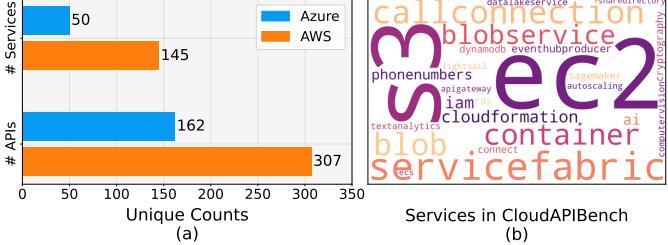


Fig. 2: **Composition of CloudAPIBench.** (a) The benchmark comprises diverse APIs from various AWS and Azure services. (b) Word cloud visualizing the services in CloudAPIBench; from AWS `s3` to Azure `computervision`, CloudAPIBench comprises many cloud-based software engineering use-cases.

introduce CloudAPIBench, a benchmark designed to evaluate Code LLMs’ abilities to invoke cloud APIs.

Composition. CloudAPIBench is a Python benchmark comprising 622 *synthetic tasks* to prevent data leakage with Code LLMs. Each task requires invoking a specific cloud API from providers like AWS and Azure, reflecting practical software engineering scenarios. Task prompts include imports, variable declarations, and a developer-style comment describing the API’s purpose, stopping just before the API invocation. Figure 1 (left) illustrates a sample task and a model response (demonstrating an API hallucination). Figure 2 presents a detailed task distribution, showing that CloudAPIBench captures diverse API invocation scenarios to evaluate Code LLMs comprehensively.

API Frequency Annotations. CloudAPIBench also contains the *API frequency* for APIs, *i.e.*, how often they occur in The Stack v2 [14]. As The Stack v2 is one of the largest open code pre-training datasets, we assume that our API frequencies approximates the distribution of APIs in public sources. Hence, this can be used to explore the relationship between hallucination rates and API frequencies.

To enhance interpretability, we classify API frequencies into three categories: *Low* (0 – 10 occurrences), *Medium* (11 – 100), and *High* (≥ 101). Since this treats APIs within the same class as identical, we minimize confounding factors (such as invocation complexity) by selecting diverse APIs within each class. This approach parallels the categorization of concepts based on popularity or pre-training frequencies [24], [25]. To our knowledge, this is the first granular analysis of a Code LLM’s pre-training corpus.

Construction. We construct CloudAPIBench with the goal of scaling coverage to multiple APIs from various providers. First, we source API specifications from official documentation to index their correct usage. Next, we determine each API’s frequency in The Stack v2 by counting function definitions and calls with the same names as the APIs in relevant files. We select APIs for CloudAPIBench while ensuring diversity of invocation complexity and frequency. Using Claude 3 Sonnet [26], we convert API descriptions into developer-style comments, and create prompts with necessary imports, declarations, and a descriptive comment before the API call.

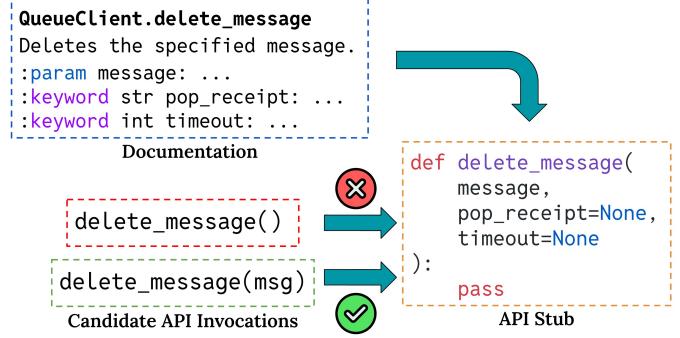


Fig. 3: **Valid API Invocation.** Using the API documentation, we create an API stub to capture correct usage. A candidate invocation is valid if it successfully binds to the stub. Here, `delete_message` requires *at least* one required argument for successful binding.

We provide elaborate details of this process in Appendix A.

Evaluation Metric. We introduce the **valid API invocation** metric, which verifies if an API is invoked according to its syntax. We obtain this syntax by tracking the API’s arguments and whether they are required. The metric is computed as: we create a dummy function mirroring the API’s signature (*i.e.*, API stub [27]). A candidate invocation is tested against this stub, and only successful bindings indicate validity. This evaluation method bypasses the intricacies of static analysis [11] and is more robust than string matching [20], ensuring reliable and scalable evaluations. Figure 3 illustrates this process.

C. Evaluation & Results

Models. We evaluate the following recent Code LLMs (and sizes) on CloudAPIBench: StarCoder2-{3B, 7B, 15B} [14], DeepSeekCoder-{1.3B, 6.7B, 33B} [30], Google CodeGemma-{2B, 7B} [31], IBM Granite-Code-{3B, 8B, 20B} [32] and GPT-4o (gpt-4o-2024-05-13) [29].

Inference. We use greedy decoding to generate code up to the first API call; this is evaluated for validity as detailed in Section II-B. This strategy is used throughout this work consistently. For chat models, we specify a system prompt indicating the model to generate only the API invocation.

Results. Table II presents the performance of all models on CloudAPIBench and HumanEval (for a reference of generic performance). Key observations include:

- *API Hallucinations.* All Code LLMs exhibit API hallucinations to a certain degree. These primarily occur due to (1) usage of non-existing APIs, (2) incorrect usage of the target API or, (3) usage of incorrect existing APIs. We explain these failure cases in Appendix B.

- *API Frequency Trends.* A strong correlation exists between API frequency and valid API invocations: high frequency APIs yield fewer hallucinations, while low frequency APIs result in more. While this is expected, this trend verifies the applicability of our API frequency annotations.

- *Low Frequency APIs.* Despite strong performance on high frequency APIs and generic benchmarks, all models

Model	HumanEval pass@1	CloudAPIBench		
		High Frequency	Medium Frequency	Low Frequency
StarCoder2-3B	31.44	84.39	37.33	11.61
StarCoder2-7B	34.09	86.34	47.33	9.36
StarCoder2-15B	44.15	88.78	57.33	24.72
Google CodeGemma-2B	27.28	79.51	26.67	4.49
Google CodeGemma-7B	40.13	87.80	52.67	12.36
IBM Granite-Code-3B	--	83.41	44.67	17.23
IBM Granite-Code-8B	--	85.85	62.67	28.09
IBM Granite-Code-20B	--	87.80	69.33	32.21
DeepSeekCoder-1.3B	32.13	79.02	22.67	5.24
DeepSeekCoder-6.7B	45.83	88.78	52.00	13.48
DeepSeekCoder-33B	52.45	90.24	70.00	34.83
GPT-4o	90.20	93.66	78.67	38.58

TABLE II: **Results on CloudAPIBench.** We present Valid API Invocation (%) results on CloudAPIBench for various Code LLMs, categorized by API frequency in The Stack v2. For comparison, we also include HumanEval [3] results from [28] and [29]. While Code LLMs excel on high-frequency APIs, their performance drops severely on low-frequency APIs, despite strong results on general programming tasks like HumanEval.

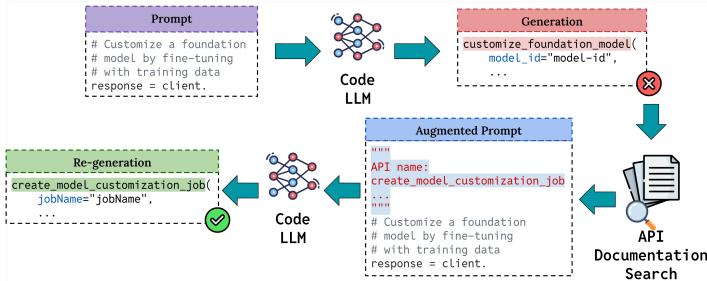


Fig. 4: **DAG Overview.** Starting with a CloudAPIBench task, we sample an API invocation from the Code LLM. This is used to retrieve documentation for the matching APIs. We then augment the prompt with the documentation and re-trigger the model.

exhibit high hallucination rates for low frequency APIs. This disparity highlights the value of CloudAPIBench in pinpointing scenarios where Code LLMs are prone to hallucinate. Refer to Appendix C for a more detailed analysis.

III. DOCUMENTATION AUGMENTED GENERATION (DAG)

In this section, we see how DAG enhances performance on CloudAPIBench. We first outline the key components of DAG: augmentation design, retrieval index and retriever, in Section III-A. Subsequently, we discuss how different design choices affect downstream performance in Section III-B.

A. Setup

Overview. Following [15], [33], we implement an iterative pipeline for DAG. Starting with a prompt, the Code LLM generates a hypothetical API invocation. This invocation forms a query to retrieve documentation for similar APIs. The

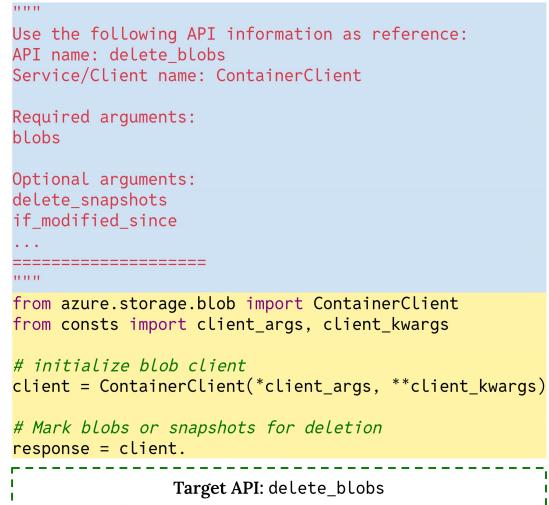


Fig. 5: **API Specification Augmentation.** Augmented prompt for the Oracle retriever with one retrieval. The “API Specification” (blue) contains the API name and a list of its required & optional arguments, providing an efficient summary of the documentation.

retrieved documentation is processed and appended to the original prompt, after which inference is re-triggered. This process is illustrated in Figure 4.

Query Formulation & Retrieval Index. Given a CloudAPIBench task, the Code LLM generates a candidate API invocation, which we process as the query. This query focuses solely on API-relevant keywords, excluding any distractor prompt content [15], [33], [34]. Our retrieval index includes all collected AWS and Azure APIs, identified using *keys* prepared similarly as the queries.

Retriever. We develop a retriever with configurable precision

Augmentation Design	Avg. Tokens	Valid API Inv. (%)
Base Model	--	41.80
API Name Only	36.07	52.73
API Description	78.80	53.22
API Specification	52.57	<u>86.82</u>
API Desc. + API Spec.	94.55	<u>87.14</u>
Full Documentation	685.24	88.75

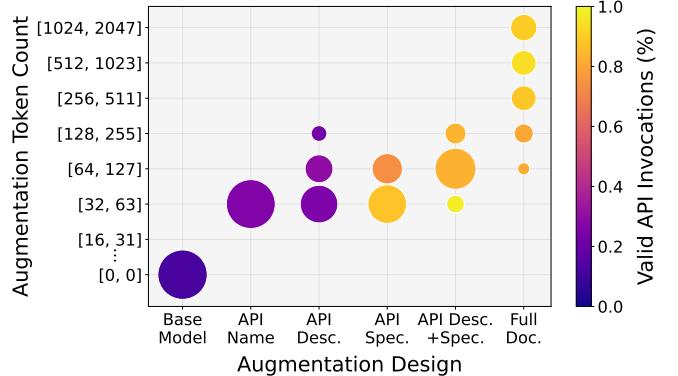


Fig. 6: **Augmentation Design Results.** (Left) Displays average tokens introduced per augmentation using the StarCoder2-3B tokenizer and average performance on CloudAPIBench for each augmentation design. (Right) Visualizes performance for *low frequency* APIs: the y-axis shows binned sequence lengths (exponential scale; capped at 2048), bubble color indicates performance, and bubble size indicates fraction of samples per bin. The improvements from API Specification are dramatic, though “Full Documentation” introduces too many tokens.

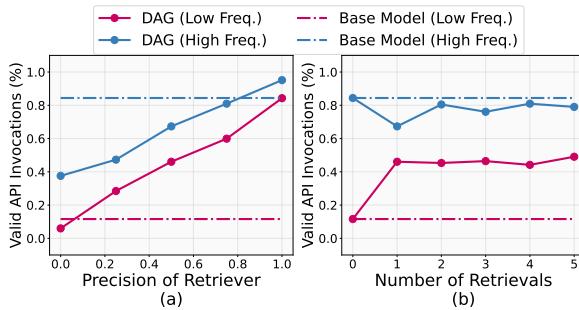


Fig. 7: **Precision and No. of Retrievals.** (a) While most low precision retrievers hurt performance on high frequency APIs, they may benefit low frequency APIs. (b) 1 retrieval hurts performance on high frequency APIs, but this is somewhat recovered as number of retrievals increases.

to study the effect of retrieval accuracy on CloudAPIBench. For an $x\%$ precision@ k setting, we return k documents via BM25, ensuring that the target API’s documentation is included $x\%$ of the time. This approach allows us to examine the impact of varying retrieval precision (x). We chose BM25 for its simplicity [11], [35], though our results are likely robust to different retrievers.

Augmentation Design. We prepend the retrieved documentation to the original prompt as a Python docstring after postprocessing. We test various augmentation strategies, each capturing different levels of API information and token count efficiencies: (1) API Name Only, (2) API Description, (3) API Specification, (4) API Description + Specification, and (5) Full Documentation. Figure 5 shows “API Specification” while additional details and examples are in Appendix D.

B. Experiments & Results

In this section, we perform ablations on various DAG components to analyze their impact on API hallucinations.

Experimental Setup. We present results from ablations on StarCoder2-3B. When testing a component (*e.g.*, retriever

Method	High Freq.	Low Freq.	Avg.
Base Model	84.39	11.61	41.80
DAG	67.32	46.07	54.50
DAG + Index Lookup	85.37	35.96	54.98

TABLE III: **Index Lookup.** Triggering DAG only for non-existent APIs reduces unnecessary retrievals for high-frequency APIs, enhancing performance. However, this also induces slight regressions for low-frequency APIs. [Avg. indicates performance across all frequencies.]

precision), other components (*e.g.*, number of retrievals) are held constant to isolate the effect. We also report the average valid API invocations across all tasks in CloudAPIBench for a concise performance measure, wherever indicated.

Augmentation Design. Our objective is to determine the most useful and efficient information to retain from an API’s documentation for augmentation. So, we use an Oracle retriever to fetch only one documentation; this guarantees that the relevant information is present *somewhere* in the documentation. Results are presented in Figure 6, showing valid API invocation rates and the number of tokens introduced per augmentation across all APIs. “API Name Only” and “API Description” do not significantly reduce hallucination rates, as they lack detailed API syntax. However, adding “API Specification” dramatically improves model performance ($41.80\% \rightarrow 86.82\%$ on average), indicating that detailed API specifications are crucial. While “Full Documentation” optimizes performance, it is highly token-inefficient (685.24 tokens per augmentation). “API Description + Specification” strikes an optimal balance between token efficiency and performance, so, we adopt this design for all subsequent experiments.

Precision of Retriever. Results are displayed in Figure 7a. Here, we retrieve one document and vary the retriever’s precision. As anticipated, the API hallucination rate decreases as retriever

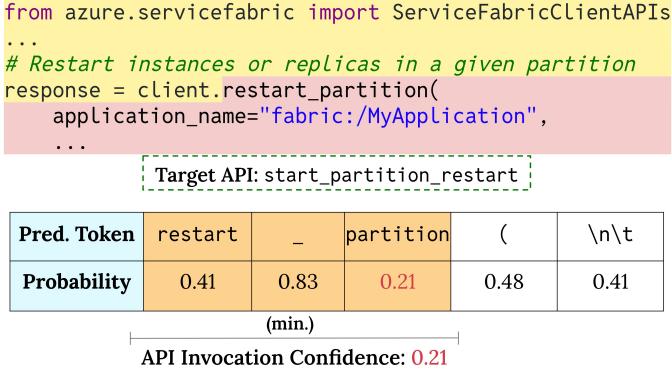


Fig. 8: **API Invocation Confidence.** We estimate the model’s uncertainty by taking the minimum probability of the predicted API name tokens (orange in table).

precision increases. Low frequency APIs show improvement over the base model even with low precision retrievers, while high frequency APIs require precision $> 80\%$ to match the base model’s performance. Thus, at most precision levels, DAG induces higher hallucination rates for high frequency APIs compared to the base model (e.g., $84.39\% \rightarrow 67.32\%$ valid API invocations at 50% precision), underscoring Code LLMs’ sensitivity to irrelevant augmentations [12].

Number of Retrievals. Here we maintain the retriever precision at 50%. Figure 7b illustrates our findings. For low-frequency APIs, one or more retrievals consistently enhance performance. Conversely, high-frequency APIs show a sharp decline with one retrieval, partially recovered with two or more. This indicates that irrelevant augmentations can lead to unexpected behavior in Code LLMs, especially when a single augmentation conflicts with the model’s internal knowledge.

Discussion. Our experiments above show that DAG significantly reduces hallucinations for low frequency APIs. However, high frequency APIs may suffer performance drops with DAG due to irrelevant retrievals. This issue can potentially be resolved by allowing the Code LLM to use its internal knowledge for high frequency APIs, bypassing DAG; this forms the core of the next section.

IV. IMPROVING DAG: WHEN TO RETRIEVE?

Given that suboptimal retrievers can increase hallucination rates with DAG, we investigate strategies to address this issue. By triggering DAG *selectively* – primarily when the Code LLM lacks knowledge of the target API – we can mitigate the negative impact of suboptimal retrievals, and allow the model to invoke APIs correctly using its internal knowledge. We discuss two strategies towards this here.

A. Index Lookup

Method. This simple technique verifies if the API name invoked during the first iteration generation exists in the API index. If not, the Code LLM is trying to call a non-existing API, and DAG provides the necessary references. Thus, DAG is not triggered for existing APIs; since this is likely to happen for

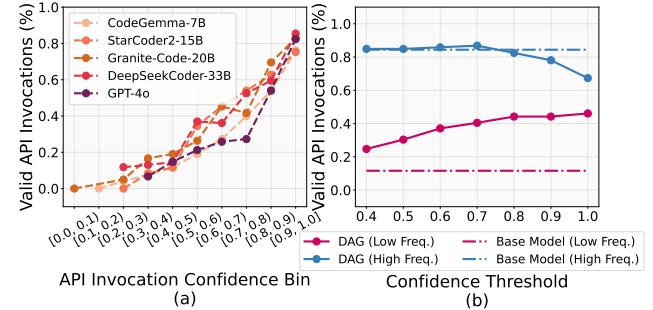


Fig. 9: **API Invocation Confidence Results.** (a) API invocation confidence scores are well-calibrated on CloudAPIBench for a range of Code LLMs. (b) By triggering DAG when the API invocation confidence is below a certain threshold, we can control regressions on high frequency APIs while maintaining good performance on low frequency APIs.

high-frequency APIs, we expect fewer imprecise DAG triggers with this method.

Experimental Setup. As before, we perform ablations with StarCoder2-3B. We use a 50% precision retriever to retrieve one documentation.

Results. Table III presents the results. The index lookup method significantly reduces the regressions introduced by DAG for high-frequency APIs, even showing slight improvements over the base model. However, this gain comes at the expense of reduced retrievals for low-frequency APIs: sometimes, the model invokes an existing incorrect API or incorrectly invokes the target API, leading to more hallucinations compared to DAG. Overall, this method shows promise for enhancing DAG.

B. API Invocation Confidence

Background: LLM Calibration. Prior work has highlighted that LLM probabilities are well-calibrated, allowing for uncertainty estimation for various tasks [15], [36]–[38]. As such, leveraging a Code LLM’s probabilities to predict potential hallucinations, we could selectively trigger DAG for those scenarios.

To quantify a Code LLM’s uncertainty during API invocation, we define **API invocation confidence** as the *minimum* probability among all predicted API name (sub-)tokens (see Figure 8). This minimum captures minor uncertainties in API prediction better than other aggregators like the mean [15], [16]. The focus remains on the API name, not the entire invocation, as Code LLMs may show low confidence in tokens in the face of multiple alternatives (*e.g.*, constants in API arguments, *etc.*; this represents aleatoric uncertainty [39]).

Evidence from various Code LLMs, shown in Figure 9a, confirms their calibration for API invocation on CloudAPIBench. A strong positive correlation is observed between API invocation confidence and correct API usage, indicating that confidence levels can preemptively identify likely hallucinations (*i.e.*, they capture epistemic uncertainty [39]).

Method. We measure the API invocation confidence of the first iteration of generation, and if this is below a certain *fixed*

Model	Method	Retrieval Triggered (%)			Valid API Invocations (%)			
		High Freq.	Med. Freq.	Low Freq.	High Freq.	Med. Freq.	Low Freq.	Avg.
Google CodeGemma-7B	Base Model	0.00	0.00	0.00	87.80	52.67	12.36	46.95
	DAG	100.00	100.00	100.00	61.95 _(-25.85)	56.00 _(+3.33)	46.44 _(+34.08)	53.86 _(+6.91)
	DAG++	20.98	44.67	74.16	88.29 _(+0.49)	65.33 _(+12.67)	43.07 _(+30.71)	63.34 _(+16.40)
StarCoder2-15B	Base Model	0.00	0.00	0.00	88.78	57.33	24.72	53.70
	DAG	100.00	100.00	100.00	69.76 _(-19.02)	58.67 _(+1.33)	49.44 _(+24.72)	58.36 _(+4.66)
	DAG++	20.98	43.33	70.41	88.78 _(+0.00)	58.67 _(+1.33)	46.44 _(+21.72)	63.34 _(+9.65)
IBM Granite-Code-20B	Base Model	0.00	0.00	0.00	87.80	69.33	32.21	59.49
	DAG	100.00	100.00	100.00	70.24 _(-17.56)	63.33 _(-6.00)	44.19 _(+11.99)	57.40 _(-2.09)
	DAG++	15.12	29.33	66.29	89.76 _(+1.95)	71.33 _(+2.00)	45.69 _(+13.48)	66.40 _(+6.91)
DeepSeekCoder-33B	Base Model	0.00	0.00	0.00	90.24	70.00	34.83	61.58
	DAG	100.00	100.00	100.00	69.27 _(-20.98)	64.00 _(-6.00)	51.31 _(+16.48)	60.29 _(-1.29)
	DAG++	20.49	30.67	59.55	86.83 _(-3.41)	71.33 _(+1.33)	55.43 _(+20.60)	69.61 _(+8.04)
GPT-4o	Base Model	0.00	0.00	0.00	93.66	78.67	38.58	66.40
	DAG	100.00	100.00	100.00	54.63 _(-39.02)	53.33 _(-25.33)	47.94 _(+9.36)	51.45 _(-14.95)
	DAG++	3.41	9.33	50.56	94.15 _(+0.49)	82.00 _(+3.33)	55.43 _(+16.85)	74.60 _(+8.20)

TABLE IV: **DAG++ Results.** We present the performance on CloudAPIBench and the (%) of retrieval triggers for high/low frequency APIs, with absolute improvements over the base model shown in subscript. It is noteworthy that DAG++ significantly reduces the frequency of retrievals for high frequency APIs while appropriately decides to retrieve for low frequency APIs; by smartly triggering retrieval, DAG++ attains top performance on CloudAPIBench for all models.

threshold, indicating the model’s lack of knowledge about the target API, we trigger DAG to assist the model.

Experimental Setup. Towards finding an optimal configuration, we vary the threshold of API invocation confidence below which to trigger DAG, and measure the API hallucination rate for StarCoder2-3B. As before, we use a 50% precision retriever with one retrieved document.

Results. Figure 9b shows the relation between the confidence threshold and valid API invocations. As we raise the threshold, DAG is triggered more often, leading to a consistent reduction in hallucinations for low frequency APIs. Conversely, high frequency APIs remain largely unaffected until a certain point, beyond which irrelevant augmentations start causing hallucinations. The optimal threshold balances improved performance for low frequency APIs without significant regressions for high frequency APIs; for StarCoder2-3B, this optimal range is approximately 0.7 – 0.8.

C. DAG++ & Discussion

Having seen the benefits of the above approaches, we now discuss how DAG can be effectively improved by combining these, *i.e.*, DAG++. In this method, we trigger DAG *iff* the API in the first iteration of generation does not exist in the index *OR* is being invoked with an API invocation confidence below a fixed threshold. We anticipate that this would help combine the benefits of both the discussed approaches.

Experimental Setup. We use a 50% precision retriever with one retrieval, consistent with previous experiments. Further, we fix the confidence threshold to be 0.8. Finally, to investigate the generalizability of our findings, we evaluate the largest models of all model families from Table II.

Results. The results are shown in Table IV. For each model, show how often retrieval is triggered and the resulting performance on CloudAPIBench. We make the following key observations:

– *Trigger of Retrievals.* We first examine how often retrieval is triggered with each method. The base model never triggers retrieval, DAG always does, and DAG++ selectively retrieves documentation. DAG++ exhibits a *strong negative correlation* between retrieval trigger frequency and API frequency: it triggers retrieval more often for low frequency APIs and less for high frequency APIs, aligning with the principle of retrieval *only when necessary*. For *e.g.*, with GPT-4o, DAG++ retrieves only 3.41% of the time for high frequency APIs indicating minimal need for documentation; conversely, retrieval is triggered 50.56% of the time for low frequency APIs, supplementing the model’s limited knowledge with relevant documentation.

– *DAG v/s DAG++.* Table IV also shows the performances (and absolute improvements over the base model in subscript) of various models on CloudAPIBench. As noted in Section III, while DAG significantly boosts low frequency API performance, it degrades high frequency API performance. For instance, GPT-4o experiences a 39.02% drop in performance for high frequency APIs with DAG, highlighting the the model’s sensitivity to irrelevant augmentations. DAG++ successfully mitigates this issue for high frequency APIs while maintaining or improving gains on low frequency APIs. Overall, DAG++ outperforms DAG indicating that selective retrieval of API documentation, that respects API frequency, aids performance on CloudAPIBench.

– *Generalizability.* All model families demonstrate similar enhancement trends with DAG++, despite architectural and training differences. This underscores the generalizability of the importance of selectively retrieving API documentation when Code LLMs lack API specific knowledge. Additionally, scaling trends with model sizes [40], [41] are evident: average performance monotonically improves with model size in Table IV. Finally, DAG++ reveals that larger models require fewer retrievals for optimal performance, suggesting they memorize API syntax (including low frequency) more efficiently.

V. RELATED WORK

Program Synthesis & API Invocations. Code LLMs are actively being used for automatic program synthesis [30], [42]. Relevant to our study is API invocation generation [11], [43], often done on tool-usage benchmarks that do not account for the distribution of APIs in the public domain. We develop CloudAPIBench, a benchmark targeting cloud-based software engineering scenarios, that includes API frequency annotations, allowing for nuanced failure analyses and targeted improvements through DAG. Works such as [10], [11], [34], [44] also use documentation to improve API generation, but their evaluations do not capture the granularities discussed here.

LLM Hallucinations. LLMs may generate factually incorrect statements about concepts, diminishing their utility [45]–[47]. As such, several works have emerged to deal with this issue. Some works focus on hallucination detection by exploiting the well-calibrated nature of LLMs [36]–[38] and using model confidence scores [15], [16]. Closest to our work, [17], give a taxonomy of hallucinations for code generation. While they focus on identifying hallucinations with Code LLMs, we focus on mitigating API hallucinations using documentation.

Retrieval Augmented Generation (RAG). RAG supplements language models by retrieving from external data-stores [48]. Some studies use fixed algorithms for retrieval [11], [49], [50], while others adopt adaptive retrieval through special tokens [51] or model confidence scores [15]. In this work, we establish how to use selective retrieval effectively to mitigate API hallucinations with documentation.

VI. CONCLUSION & FUTURE WORK

In this work, we thoroughly investigate API hallucinations and demonstrate mitigation strategies for various Code LLMs. We introduce CloudAPIBench, a benchmark to measure API hallucinations for diverse AWS and Azure APIs, including API frequencies to categorize low, medium, and high frequency APIs. We adapt RAG with documentation (DAG) to inform Code LLMs about the correct syntax during inference. We discuss which parts of documentation are important and how various retrieval components affect hallucinations. While DAG significantly enhances low-frequency API performance, it can degrade high-frequency API performance with irrelevant retrievals. We tackle this issue by *selectively* triggering retrievals through index lookup and API invocation confidence thresholding, and combine these methods in DAG++ leading to top performance on CloudAPIBench across Code LLMs. Future research could extend CloudAPIBench for long-context evaluations, explore DAG beyond iterative generation, and improve DAG by enhancing Code LLMs' robustness to irrelevant augmentations.

APPENDIX

A. CloudAPIBench *Construction*

We follow a multi-step procedure to obtain synthetic evaluation tasks in Python for AWS and Azure APIs to include in CloudAPIBench. A summary of the process is shown in Figure 10. We describe each step in detail here.

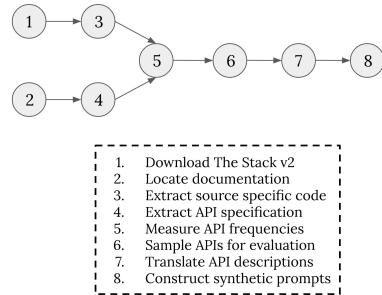


Fig. 10: Summary of steps to construct CloudAPIBench.

- 1) **Download The Stack v2.** We download The Stack v2 from its official repository on HuggingFace and SoftwareHeritage [14].
 - 2) **Locate Documentation & Syntax.** We use boto3 1.34.108 for AWS and the Python package azure-sdk-for-python for Azure. For AWS, we use the mypy_boto3_builder tool [52] to create API stubs for all AWS APIs; this helps us obtain the list of APIs. We obtain the official documentation for each of these by scraping the boto3 webpage for the respective APIs. For Azure, the complete docstring in the source code for an API’s defintion is its documentation.
 - 3) **Extract source specific code.** We identify source specific code samples in The Stack v2 so that we restrict the count of API frequencies to only these. For AWS, source specific files are those that import one of {boto3, botocore} or contain one of {aws, boto, amazon} in the filepath. Similarly, Azure specific samples are those that import azure or contain azure in the filepath.
 - 4) **Extract API specification.** For Azure, the complete documentation is available as a docstring in the respective function definitions for that API. Using tree-sitter, we parse the code files to obtain the list of APIs, their correct usages and complete docstrings for Azure, for as many APIs as possible. For AWS, we parse API stubs obtained using mypy_boto3_builder to curate the API specifications. This also serves as the index of APIs that we use in our experiments.
 - 5) **Measure API frequencies.** Given the list of APIs for a source, we count the number of times functions with the name as an API are invoked or defined within the source specific code samples identified above. We use several heuristics to avoid edge cases and maintain reliability. Nevertheless, some noise may creep in and we acknowledge that this process is far from perfect. However, the findings based off of these API frequencies align with our expectations, indicating their reliability.
 - 6) **Sample APIs for evaluation.** While sampling APIs for evaluation, we take care to ensure diversity with respect to API frequencies (uniform sampling from each frequency class as far as possible) and invocation complexity (within each frequency class, there should be uniform distribution of the number of required arguments required by APIs).

This ensures that CloudAPIBench is diverse and represents diverse software-engineering scenarios, from low to high frequency, and from APIs that are easy to invoke to those that require careful recall of API syntax. Further each API may appear in CloudAPIBench up to 3 times with different prompts.

- 7) **Translate API descriptions.** Each sample in CloudAPIBench contains a comment that expresses the intent to invoke the API. We obtain this comment by instructing Claude 3 Sonnet [26] to translate the documentation description of the API into 3 concise developer style comments. We use few-shot prompting to do this, and upon manual inspection of dozens of responses, find that Claude is able to do this task reliably. As such, we use Claude’s responses as comments describing the intent to invoke the APIs, fixing any issues that were noticed manually.
- 8) **Construct synthetic prompts.** As the final step, for the selected APIs, we create synthetic prompts by creating an incomplete code sample: these start with relevant imports, contain necessary variable declarations, include the comment expressing the intent to invoke an API, and end just before the API invocation. Manual inspection revealed that in a few cases, multiple APIs may be suitable targets for a task, and in such cases we manually enumerate all the possible targets to the best of our knowledge.

B. Hallucination Categorization & Illustration

We classify API hallucinations into three broad categories:

- 1) **Usage of incorrect existing API.** This occurs when the Code LLM attempts to invoke an API that exists but does not fulfill the task.
- 2) **Invalid usage of target API.** This occurs when the Code LLM attempts to invoke the correct API but does so incorrectly due to an invalid configuration of arguments; here the model may either pass arguments that the API does not accept or not pass a correct combination of required and optional arguments.
- 3) **Usage of non-existing API.** This occurs when the Code LLM attempts to invoke an API that does not exist.

C. Analyzing Low Frequency API Failures

We look closer into the various modes of failure for low frequency APIs in Table V. We present this analysis for the largest model in each model family. As shown, most failures arise when the models try to invoke a non-existing API or use the target API incorrectly. This goes to show the lack of knowledge about low frequency APIs, and the propensity to hallucinate under these scenarios in Code LLMs.

D. Augmentation Designs

We define and illustrate various augmentation designs in this section.

- **API Name Only.** We include only the name of the retrieved APIs as augmentation. This can test if the Code LLM can invoke the correct API just by referencing its name during inference.
- **API Description.** We include the name and a short description of the API. For Azure the short description is the first sentence from the API’s docstring, whereas for AWS the short description is the first 5 sentences from the API’s documentation on the boto3 webpage. We choose 5 here as we found, in several cases, the first 2–3 sentences to be irrelevant to the API’s description.
- **API Specification.** This is a concise summary of the syntax of the API. It includes the name of the API and the list of required and optional arguments without specifying any descriptions of the arguments.
- **API Description + API Specification.** This includes the description as defined above along with the specification as discussed above.
- **Full Docstring.** This uses the entire collected documentation as augmentation. Since this can be arbitrarily large, especially for AWS documentation, we right-truncate the documentation up to 5000 characters before augmenting. This assumes that the necessary information to invoke the API is within the first 5000 characters.

We illustrate these strategies in Figure 11. We skip “API Description + API Specification” in the figure as it is a combination of “API Description” and “API Specification”.

Model	Valid (%)	Invalid (%)	Invalid API Invocations Breakdown		
			Usage of incorrect existing	Invalid usage of target	Usage of non-existing
Google CodeGemma-7B	12.36	87.64	10.68	35.47	53.85
StarCoder2-15B	24.72	75.28	10.95	33.33	55.72
IBM Granite-Code-20B	32.21	67.79	17.13	23.76	59.12
DeepSeekCoder-33B	34.83	65.17	15.52	31.03	53.45
GPT-4o	38.58	61.42	13.41	33.54	53.05

TABLE V: **Results on CloudAPIBench for low frequency APIs.** We first show the fraction of valid and invalid API invocations for low frequency APIs for various models. The invalid API invocations are categorized into various types of failures. Notably, > 50% failures occur due to the models attempting to invoke non-existing APIs.

API Specification	API Description	Full Documentation
<pre>""" Use the following API information as reference: API name: list_identity_pool_usage ===== """ API Name Only</pre>	<pre>""" Use the following API information as reference: API name: list_identity_pool_usage ===== """ Service/Client name: cognito-sync</pre> <p>Description: Gets a list of identity pools registered with Cognito. ListIdentityPoolUsage can only be called with developer credentials. You cannot make this API call with the temporary user credentials provided by Cognito Identity.</p>	<pre>""" Use the following API information as reference: API name: List_identity_pool_usage Service/Client name: cognito-sync Gets a list of identity pools registered with Cognito. ListIdentityPoolUsage can only be called with developer credentials. You cannot make this API call with the temporary user credentials provided by Cognito Identity. See also: AWS API Documentation Request Syntax response = client.List_identity_pool_usage(NextToken='string', MaxResults=123) ... =====</pre>

Fig. 11: **API augmentation designs.** Illustrated for the AWS API: list_identity_pool_usage. “Full Documentation” is truncated to fit in the figure.

REFERENCES

- [1] S. Sivasubramanian, “Announcing new tools for building with generative ai on aws,” Apr 2023, [Accessed: (June 15, 2024)]. [Online]. Available: <https://aws.amazon.com/blogs/machine-learning/announcing-new-tools-for-building-with-generative-ai-on-aws/>
- [2] S. Peng, E. Kalliamvakou, P. Cihon, and M. Demirer, “The impact of ai on developer productivity: Evidence from github copilot,” 2023.
- [3] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. de Oliveira 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,” 2021.
- [4] A. Moradi Dakhel, V. Majdinasab, A. Nikanjam, F. Khomh, M. C. Desmarais, and Z. M. J. Jiang, “Github copilot ai pair programmer: Asset or liability?” *J. Syst. Softw.*, vol. 203, no. C, Sep. 2023. [Online]. Available: <https://doi.org/10.1016/j.jss.2023.111734>
- [5] M. AI, “Introducing code llama, an ai tool for coding,” Aug 2023, <https://about.fb.com/news/2023/08/code-llama-ai-for-coding/>.
- [6] R. Salva, “Introducing ai-powered app dev with code customization from gemini code assist enterprise,” Oct 2024, <https://cloud.google.com/blog/products/application-development/introducing-gemini-code-assist-enterprise>.
- [7] D. Prakoso, “Announcing new tools for building with generative ai on aws,” 2024, <https://aws.amazon.com/blogs/aws/amazon-q-developer-now-generally-available-includes-new-capabilities-to-reimagine-developer-experience/>.
- [8] S. Barke, M. B. James, and N. Polikarpova, “Grounded copilot: How programmers interact with code-generating models,” *Proc. ACM Program. Lang.*, vol. 7, no. OOPSLA1, Apr. 2023. [Online]. Available: <https://doi.org/10.1145/3586030>
- [9] T. McDonnell, B. Ray, and M. Kim, “An empirical study of api stability and adoption in the android ecosystem,” in *2013 IEEE International Conference on Software Maintenance*. IEEE, 2013, pp. 70–79.
- [10] S. Zhou, U. Alon, F. F. Xu, Z. Jiang, and G. Neubig, “Docprompting: Generating code by retrieving the docs,” in *The Eleventh International Conference on Learning Representations*, 2023.
- [11] S. G. Patil, T. Zhang, X. Wang, and J. E. Gonzalez, “Gorilla: Large language model connected with massive apis,” 2023.
- [12] F. Shi, X. Chen, K. Misra, N. Scales, D. Dohan, E. Chi, N. Schärlí, and D. Zhou, “Large language models can be easily distracted by irrelevant context,” in *Proceedings of the 40th International Conference on Machine Learning*, ser. ICML’23. JMLR.org, 2023.
- [13] O. Yoran, T. Wolfson, O. Ram, and J. Berant, “Making retrieval-augmented language models robust to irrelevant context,” in *The Twelfth International Conference on Learning Representations*, 2024.
- [14] A. Lozhkov, R. Li, L. B. Allal, F. Cassano, J. Lamy-Poirier, N. Tazi, A. Tang, D. Pykhtar, J. Liu, Y. Wei, T. Liu, M. Tian, D. Kocetkov, A. Zucker, Y. Belkada, Z. Wang, Q. Liu, D. Abulkhanov, I. Paul, Z. Li, W.-D. Li, M. Risdal, J. Li, J. Zhu, T. Y. Zhuo, E. Zheltonozhskii, N. O. O. Dade, W. Yu, L. Krauß, N. Jain, Y. Su, X. He, M. Dey, E. Abati, Y. Chai, N. Muennighoff, X. Tang, M. Oblokulov, C. Akiki, M. Marone, C. Mou, M. Mishra, A. Gu, B. Hui, T. Dao, A. Zebaze, O. Dehaene, N. Patry, C. Xu, J. McAuley, H. Hu, T. Scholak, S. Paquet, J. Robinson, C. J. Anderson, N. Chapados, M. Patwary, N. Tajbakhsh, Y. Jernite, C. M. Ferrandis, L. Zhang, S. Hughes, T. Wolf, A. Guha, L. von Werra, and H. de Vries, “Starcoder 2 and the stack v2: The next generation,” 2024.
- [15] Z. Jiang, F. Xu, L. Gao, Z. Sun, Q. Liu, J. Dwivedi-Yu, Y. Yang, J. Callan, and G. Neubig, “Active retrieval augmented generation,” in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, H. Bouamor, J. Pino, and K. Bali, Eds. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 7969–7992. [Online]. Available: <https://aclanthology.org/2023.emnlp-main.495>
- [16] N. Varshney, W. Yao, H. Zhang, J. Chen, and D. Yu, “A stitch in time saves nine: Detecting and mitigating hallucinations of llms by validating low-confidence generation,” 2023.
- [17] F. Liu, Y. Liu, L. Shi, H. Huang, R. Wang, Z. Yang, and L. Zhang, “Exploring and evaluating hallucinations in llm-powered code generation,” *arXiv preprint arXiv:2404.00971*, 2024.
- [18] M. Zhang, O. Press, W. Merrill, A. Liu, and N. A. Smith, “How language model hallucinations can snowball,” 2023.
- [19] H. Ding, V. Kumar, Y. Tian, Z. Wang, R. Kwiatkowski, X. Li, M. K. Ramanathan, B. Ray, P. Bhatia, and S. Sengupta, “A static evaluation of code completion by large language models,” in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, S. Sitaram, B. Beigman Klebanov, and J. D. Williams, Eds. Toronto, Canada: Association for Computational Linguistics, Jul. 2023, pp. 347–360. [Online]. Available: <https://aclanthology.org/2023.acl-industry.34>
- [20] Y. Ding, Z. Wang, W. Ahmad, H. Ding, M. Tan, N. Jain, M. K. Ramanathan, R. Nallapati, P. Bhatia, D. Roth *et al.*, “Crosscodeeval: A diverse and multilingual benchmark for cross-file code completion,” *Advances in Neural Information Processing Systems*, vol. 36, 2024.
- [21] 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,” 2021.
- [22] T. Liu, C. Xu, and J. McAuley, “Repobench: Benchmarking repository-level code auto-completion systems,” in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: <https://openreview.net/forum?id=pPjZIOuQuF>
- [23] K. Basu, I. Abdelaziz, S. Chaudhury, S. Dan, M. Crouse, A. Munawar, V. Austel, S. Kumaravel, V. Muthusamy, P. Kapanipathi, and L. Lastras, “API-BLEND: A comprehensive corpora for training and benchmarking API LLMs,” in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, L.-W. Ku, A. Martins, and V. Srikumar, Eds. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 12859–12870. [Online]. Available: <https://aclanthology.org/2024.acl-long.694>
- [24] Y. Razeghi, R. L. Logan IV, M. Gardner, and S. Singh, “Impact of pretraining term frequencies on few-shot numerical reasoning,” in *Findings of the Association for Computational Linguistics: EMNLP 2022*, Y. Goldberg, Z. Kozareva, and Y. Zhang, Eds. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 840–854. [Online]. Available: <https://aclanthology.org/2022.findings-emnlp.59>
- [25] A. Mallen, A. Asai, V. Zhong, R. Das, D. Khashabi, and H. Hajishirzi, “When not to trust language models: Investigating effectiveness of parametric and non-parametric memories,” in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, A. Rogers, J. Boyd-Graber, and N. Okazaki, Eds. Toronto, Canada: Association for Computational Linguistics, Jul. 2023, pp. 9802–9822. [Online]. Available: <https://aclanthology.org/2023.acl-long.546>
- [26] Anthropic, “Introducing the next generation of claudie,” 2024, <https://www.anthropic.com/news/claudie-3-family> [Accessed: (March 4, 2024)].
- [27] H. Zhu, L. Wei, V. Terragni, Y. Liu, S.-C. Cheung, J. Wu, Q. Sheng, B. Zhang, and L. Song, “Stubcoder: Automated generation and repair of stub code for mock objects,” *ACM Transactions on Software Engineering and Methodology*, vol. 33, no. 1, pp. 1–31, 2023.
- [28] BigCode, “Big code models leaderboard - a hugging face space by bigcode,” 2024, <https://huggingface.co/spaces/bigcode/bigcode-models-leaderboard> [Accessed: (June 12, 2024)].
- [29] OpenAI, “Hello gpt-4o,” 2024, <https://openai.com/index/hello-gpt-4o/> [Accessed: (June 19, 2024)].
- [30] D. Guo, Q. Zhu, D. Yang, Z. Xie, K. Dong, W. Zhang, G. Chen, X. Bi, Y. Wu, Y. K. Li, F. Luo, Y. Xiong, and W. Liang, “Deepseek-coder: When the large language model meets programming – the rise of code intelligence,” 2024.
- [31] C. Team, A. J. Hartman, A. Hu, C. A. Choquette-Choo, H. Zhao, J. Fine, J. Hui, J. Shen, J. Kelley, J. Howland, K. Bansal, L. Vilnis, M. Wirth, N. Nguyen, P. Michel, P. Choy, P. Joshi, R. Kumar, S. Hashmi, S. Agrawal, S. Zuo, T. Warkentin, and Z. e. a. Gong, “Codegemma: Open code models based on gemma,” 2024. [Online]. Available: <https://goo.gle/codegemma>
- [32] M. Mishra, M. Stallone, G. Zhang, Y. Shen, A. Prasad, A. M. Soria, M. Merler, P. Selvam, S. Surendran, S. Singh, M. Sethi, X.-H. Dang, P. Li, K.-L. Wu, S. Zawad, A. Coleman, M. White, M. Lewis, R. Pavuluri, Y. Koyfman, B. Lublinsky, M. de Bayser, I. Abdelaziz, K. Basu, M. Agarwal, Y. Zhou, C. Johnson, A. Goyal, H. Patel, Y. Shah, P. Zerfos, H. Ludwig, A. Munawar, M. Crouse, P. Kapanipathi, S. Salaria, B. Calio, S. Wen, S. Seelam, B. Belgodere, C. Fonseca, A. Singhee, N. Desai, D. D. Cox, R. Puri, and R. Panda, “Granite code models: A family of open foundation models for code intelligence,” 2024.

- [33] F. Zhang, B. Chen, Y. Zhang, J. Keung, J. Liu, D. Zan, Y. Mao, J.-G. Lou, and W. Chen, “RepoCoder: Repository-level code completion through iterative retrieval and generation,” in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, H. Bouamor, J. Pino, and K. Bali, Eds. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 2471–2484. [Online]. Available: <https://aclanthology.org/2023.emnlp-main.151>
- [34] A. Eghbali and M. Pradel, “De-hallucinator: Iterative grounding for Ilm-based code completion,” 2024.
- [35] X. Cheng, D. Luo, X. Chen, L. Liu, D. Zhao, and R. Yan, “Lift yourself up: Retrieval-augmented text generation with self-memory,” in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. [Online]. Available: <https://openreview.net/forum?id=lYNSvp51af>
- [36] S. Kadavath, T. Conerly, A. Askell, T. Henighan, D. Drain, E. Perez, N. Schiefer, Z. Hatfield-Dodds, N. DasSarma, E. Tran-Johnson, S. Johnston, S. El-Showk, A. Jones, N. Elhage, T. Hume, A. Chen, Y. Bai, S. Bowman, S. Fort, D. Ganguli, D. Hernandez, J. Jacobson, J. Kernion, S. Kravec, L. Lovitt, K. Ndousse, C. Olsson, S. Ringer, D. Amodei, T. Brown, J. Clark, N. Joseph, B. Mann, S. McCandlish, C. Olah, and J. Kaplan, “Language models (mostly) know what they know,” 2022.
- [37] C. Si, Z. Gan, Z. Yang, S. Wang, J. Wang, J. L. Boyd-Graber, and L. Wang, “Prompting GPT-3 to be reliable,” in *The Eleventh International Conference on Learning Representations*, 2023. [Online]. Available: <https://openreview.net/forum?id=98p5x51L5af>
- [38] J. Li, T. Tang, W. X. Zhao, J. Wang, J.-Y. Nie, and J.-R. Wen, “The web can be your oyster for improving language models,” in *Findings of the Association for Computational Linguistics: ACL 2023*, A. Rogers, J. Boyd-Graber, and N. Okazaki, Eds. Toronto, Canada: Association for Computational Linguistics, Jul. 2023, pp. 728–746. [Online]. Available: <https://aclanthology.org/2023.findings-acl.46>
- [39] Y. A. Yadkori, I. Kuzborskij, A. György, and C. Szepesvári, “To believe or not to believe your Ilm,” 2024.
- [40] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, “Scaling laws for neural language models,” 2020.
- [41] B. Wang, W. Ping, P. Xu, L. McAfee, Z. Liu, M. Shoeybi, Y. Dong, O. Kuchaiev, B. Li, C. Xiao, A. Anandkumar, and B. Catanzaro, “Shall we pretrain autoregressive language models with retrieval? a comprehensive study,” in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, H. Bouamor, J. Pino, and K. Bali, Eds. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 7763–7786. [Online]. Available: <https://aclanthology.org/2023.emnlp-main.482>
- [42] B. Rozière, J. Gehring, F. Gloeckle, S. Sootla, I. Gat, X. E. Tan, Y. Adi, J. Liu, R. Sauvestre, T. Remez, J. Rapin, A. Kozhevnikov, I. Evtimov, J. Bitton, M. Bhatt, C. C. Ferrer, A. Grattafiori, W. Xiong, A. Défossez, J. Copet, F. Azhar, H. Touvron, L. Martin, N. Usunier, T. Scialom, and G. Synnaeve, “Code llama: Open foundation models for code,” 2023.
- [43] Y. Qin, S. Liang, Y. Ye, K. Zhu, L. Yan, Y. Lu, Y. Lin, X. Cong, X. Tang, B. Qian, S. Zhao, L. Hong, R. Tian, R. Xie, J. Zhou, M. Gerstein, dahai li, Z. Liu, and M. Sun, “ToolLLM: Facilitating large language models to master 16000+ real-world APIs,” in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: <https://openreview.net/forum?id=dHng200Jjr>
- [44] D. Zan, B. Chen, Y. Gong, J. Cao, F. Zhang, B. Wu, B. Guan, Y. Yin, and Y. Wang, “Private-library-oriented code generation with large language models,” 2023.
- [45] A. Mishra, A. Asai, V. Balachandran, Y. Wang, G. Neubig, Y. Tsvetkov, and H. Hajishirzi, “Fine-grained hallucination detection and editing for language models,” 2024.
- [46] H. Kang, J. Ni, and H. Yao, “Ever: Mitigating hallucination in large language models through real-time verification and rectification,” 2023.
- [47] H. Lee, S. Joo, C. Kim, J. Jang, D. Kim, K.-W. On, and M. Seo, “How well do large language models truly ground?” 2023.
- [48] A. Asai, S. Min, Z. Zhong, and D. Chen, “Retrieval-based language models and applications,” in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 6: Tutorial Abstracts)*, Y.-N. V. Chen, M. Margot, and S. Reddy, Eds. Toronto, Canada: Association for Computational Linguistics, Jul. 2023, pp. 41–46. [Online]. Available: <https://aclanthology.org/2023.acl-tutorials.6>
- [49] Y. Wang, P. Li, M. Sun, and Y. Liu, “Self-knowledge guided retrieval augmentation for large language models,” in *Findings of the Association for Computational Linguistics: EMNLP 2023*, H. Bouamor, J. Pino, and K. Bali, Eds. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 10303–10315. [Online]. Available: <https://aclanthology.org/2023.findings-emnlp.691>
- [50] W. Shi, S. Min, M. Yasunaga, M. Seo, R. James, M. Lewis, L. Zettlemoyer, and W.-t. Yih, “REPLUG: Retrieval-augmented black-box language models,” in *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, K. Duh, H. Gomez, and S. Bethard, Eds. Mexico City, Mexico: Association for Computational Linguistics, Jun. 2024, pp. 8371–8384. [Online]. Available: <https://aclanthology.org/2024.naacl-long.463>
- [51] A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi, “Self-RAG: Self-reflective retrieval augmented generation,” in *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*, 2023. [Online]. Available: <https://openreview.net/forum?id=jbNjgmE0OP>
- [52] YouType, “mypy_boto3_builder,” https://github.com/youtype/mypy_boto3_builder/, 2024.

Rapid Response Information Report

Generative AI: Language models and multimodal foundation models

24 March 2023

This report was commissioned by Australia's National Science and Technology Council at the request of the Minister for Industry and Science, the Hon Ed Husic MP in February 2023.

The questions asked were:

- What are the opportunities and risks of applying large language models (LLMs) and multimodal foundation models (MFM) learning technologies over the next two, five and ten years?
- What are some examples of strategies that have been put in place internationally by other advanced economies since the launch of models like ChatGPT to address the potential opportunities and impacts of artificial intelligence (AI)?

Australia's National Science and Technology Council is responsible for providing advice to the Prime Minister and other Ministers on important science and technology issues facing Australia. The Council is Chaired by the Prime Minister, with the Minister for Industry and Science as the Deputy Chair, and Australia's Chief Scientist as Executive Officer.

This report was convened by the Australian Council of Learned Academies, alongside the Australian Academy of Humanities, the Australian Academy of Technological Sciences and Engineering and the Australian Academy of Science. Lead authors were Professors Genevieve Bell AO FTSE FAHA, Jean Burgess FAHA, Julian Thomas FAHA and Shazia Sadiq FTSE.

Rapid response information reports provide a synthesis of the available scientific and technical information at the time commissioned, and expert scientific opinion, and are peer reviewed by experts in the field. Rapid response reports typically do not provide recommendations or policy advice to government.

This report represents independent science evidence to government, and does not represent the views or policies of the Australian Government or Australia's Chief Scientist.

To cite this report:

Bell, G., Burgess, J., Thomas, J., and Sadiq, S. (2023, March 24). *Rapid Response Information Report: Generative AI - language models (LLMs) and multimodal foundation models (MFM)*. Australian Council of Learned Academies.

Rapid Response Information Report: Generative AI - language models (LLMs) and multimodal foundation models (MFMs)

24 March 2023

This rapid research report addresses the questions:

- **What are the opportunities and risks of applying large language models (LLMs) and multimodal foundation models (MFM) learning technologies over the next two, five and ten years?**
- **What are some examples of strategies that have been put in place internationally by other advanced economies since the launch of models like ChatGPT to address the potential opportunities and impacts of artificial intelligence (AI)?**

Summary points

- ChatGPT is an early example of the kinds of applications and services that will emerge from Generative AI built on LLMs and MFM. It was developed by the American AI organisation OpenAI – a combined non-profit/for profit entity that received a U\$10B investment from Microsoft in 2023. ChatGPT is based on a large language model (LLM) and uses considerable pre- and post-processing of data to deliver a compelling user experience.
- Given the speed of innovation, quantum of investment and lack of technical information, it is almost impossible to accurately forecast opportunities over the next decade. Known risks are clearer, but there are categories of emerging risks that are difficult to forecast. In the shorter term, generative AI, based on LLMs and MFM, will likely impact everything from banking and finance to public services, education and creative industries.
- Generative AI will raise questions about opportunities and risks of widespread adoption; the scope and adequacy of national strategic planning and policies; the fitness of legal and regulatory approaches; and the implications of increasing geo-political competition and geo-specific regulations.
- Generative AI presents opportunities across various industries, including healthcare where LLMs and MFM are being used to analyse medical images and consolidate patient data, and in engineering to evaluate and optimise designs.
- The current concentration of generative AI activities poses risks for Australia and raises questions about our capabilities, capacities, investments and regulatory frames. Questions include: do we have sufficient compute power, appropriately skilled practitioners, scientific expertise, workforce development strategies and policy settings that range from critical technologies, to education, ethics, governance and regulation?
- LLM and MFM are generating a surge in interest, innovation and investment. Much of this work is happening inside commercial organisations, and is currently concentrated in a small number of organisations and countries, notably the US and China.
- While some of the architectural innovations are being shared publicly, overall there is a paucity of information about the development, deployment and commercialisation of these models and the applications and services based upon them.

The current ‘ChatGPT moment’ is provoking public conversation about the role AI should have in Australian society. This report has been written in the context of rapid change in the ecosystem and heightened expectations about both the risks and possibilities of both ChatGPT in particular and generative AI more broadly.

Generative AI raises questions about opportunities and risks of widespread adoption; the scope and adequacy of national strategic planning and policies; the fitness of legal and regulatory approaches; and the implications of increasing geopolitical competition and geo-specific regulation in AI-related technologies and industries.

This report explains how generative AI, based on LLMs and MFMs, currently works, given that the technologies are nascent and rapidly evolving (e.g., GPT-4 was publicly released on 14 March 2023 with some multi-modal input functionality and Baidu released Ernie Bot on 16 March with multi-modal output functionality)¹ as are the business models, applications and services that are built upon them. Against this backdrop, the report explores foreseeable risks and opportunities, based on current patterns of uptake and application.

Defining generative AI

Whereas conventional AI has been largely analytic, generative AI takes its name from its capacity to generate novel content, as varied as text, image, music and computing code, in response to a user prompt. For example, conventional AI can be used to analyse features of a legal contract, such as to identify whether the contract deals with intellectual property or privacy. By contrast, generative AI can be used to generate (i.e. draft) a new legal contract to cover those issues.

GPT-3 (Generative Pre-Trained Transformer 3), which powers the free version of ChatGPT^a, is an example of new generative AI, built on an LLM. The launch of ChatGPT (a generative AI-powered chatbot) in November 2022, by OpenAI, has prompted an extraordinary amount of activity – from adversarial exploitation and forensic testing to better understand how the system works and its governing rules, to creative exploration. ChatGPT is more consumer-friendly than prior AI systems and it has been fundamentally misunderstood, from attributing it sentience to claiming that it is thoughtfully summarising the internet. Like earlier generations of AI, generative AI relies on complex mathematical models, considerable computing power and extensive human resources to train, develop and deploy.²

Large language models (LLMs) and multimodal foundation models (MFMs)

First developed in the 2010s, LLMs and MFMs use sophisticated machine learning algorithms to predict an output – such as an image or word – based on an input, such as a sequence of words. What all these models do is recognise patterns in data and produce sophisticated answers based on those patterns. The models are not intelligent or able to necessarily determine fact from fiction in their inputs or training data.

LLMs specialise in generating human-like text by training on vast quantities of text^b.³ MFMs are more complex as they use a wider range of information, including images, speech, numerical inputs and code,⁴ and they are trained on the relationship between the various inputs. Like LLMs, MFMs generate output

^a ChatGPT Plus, a paid version of the chatbot uses GPT-4. The free version uses an older version, GPT-3.5.

^b GPT-3 is trained on about 45TB of text data from different datasets, including Wikipedia and books. GPT-3’s training data is known to only contain information up to September 2021.¹⁴⁵ For one GPU, it would take over 300 years to train the model, and it cost 5 million dollars to train the neural network.¹⁴⁶

based on learned patterns from the training input. Like LLMs, they also require tremendous computational power to train their models.^c

The predictive text and image generation functionality of LLMs and MFM^s is not new. What is novel is the scale of the data used for training and the extremely large number of parameters in the models. Recent advances in architecture and modelling have made it possible to dramatically increase the size of datasets on which LLMs and MFM^s can be trained, and as a result the ability to ascertain a richer set of contextual patterns and probabilistic relationships between data. For example, GPT-4 will be trained on 100 trillion parameters whereas GPT-3 is currently trained on 175 billion. This scale is critical in allowing the models to account for the input context in a more nuanced way.³ Because of this, LLMs and MFM^s are now far more powerful than their predecessors.

That said, most people are not directly encountering LLMs or MFM^s; rather, they are encountering new kinds of services, applications and businesses that use them, whether in the form of chat-bots, enhanced applications or subscription services. For example, ChatGPT provides a seamless user experience in both requesting and receiving information from a LLM (see Figure 1). Invisible to the user, ChatGPT uses pre-processing and post-processing to calibrate whether the prompt is appropriate and return the answer in a form that seemingly responds to the original request. Developers have decided what and how user requests for information should be handled, including whether or not to label the request appropriate or ethical based on OpenAI's internal guidelines. It is not yet clear what the implications of such choices might be for third parties who integrate these services and applications into their own ecosystems.

[The LLM/MFM lifecycle](#)

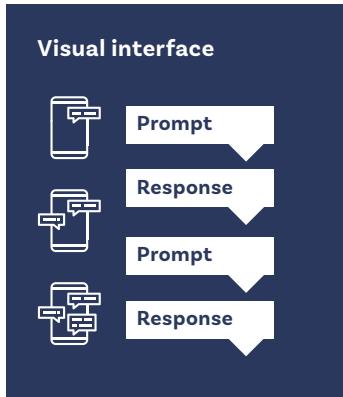
To date, successfully developing an LLM or MFM has required substantial monetary, computational and human resources.⁵ The data required, the processing power needed, and the risks and potential consequences of 'wrong' answers or malicious uses amplify these challenges. As a result, developers are employing an evolving range of strategies to design systems in ways that may prevent social harms (e.g., inequities, misinformation), maximise user safety (e.g., protecting vulnerable communities), and/or maintain some degree of control over downstream applications (see Table 1).⁶

^c GPT-4 appears to be an MFM that can process documents that contain both text and image data, answering detailed questions about images in plain text. Due to OpenAI's most recent policy on relaying information about their models (as seen in their preprint OpenAI 2023), we do not know the size of the model, its internal structure or the dataset that OpenAI used to create GPT-4, nor do we have an estimate about the computational power used to create this model or the cost.

Example LLM user experience

(based on ChatGPT-3)

What the user sees



What ChatGPT does

ChatGPT selects its responses from a pre-trained Large Language Model (LLM).

An LLM is an AI designed to understand and generate human-like language.

The current model (GPT3.5) has 175 billion parameters and three billion words.

The ChatGPT application shapes its output based on pre-determined rules and previous interactions.

Typical ChatGPT user experience

A user initiates a conversation with ChatGPT by typing a message or question into a chat interface, such as a messaging app or a chatbot on a website.

ChatGPT receives the user's message.

ChatGPT uses its natural language processing capabilities to predict the user's intent from the context of the conversation.

ChatGPT generates a response based on its training and the user's input.

The response is displayed in the chat interface, and the user can respond with follow-up questions or feedback.

The conversation between the user and ChatGPT continues in this way, refining future responses based on the information provided.

Chat GPT3.5 operating process

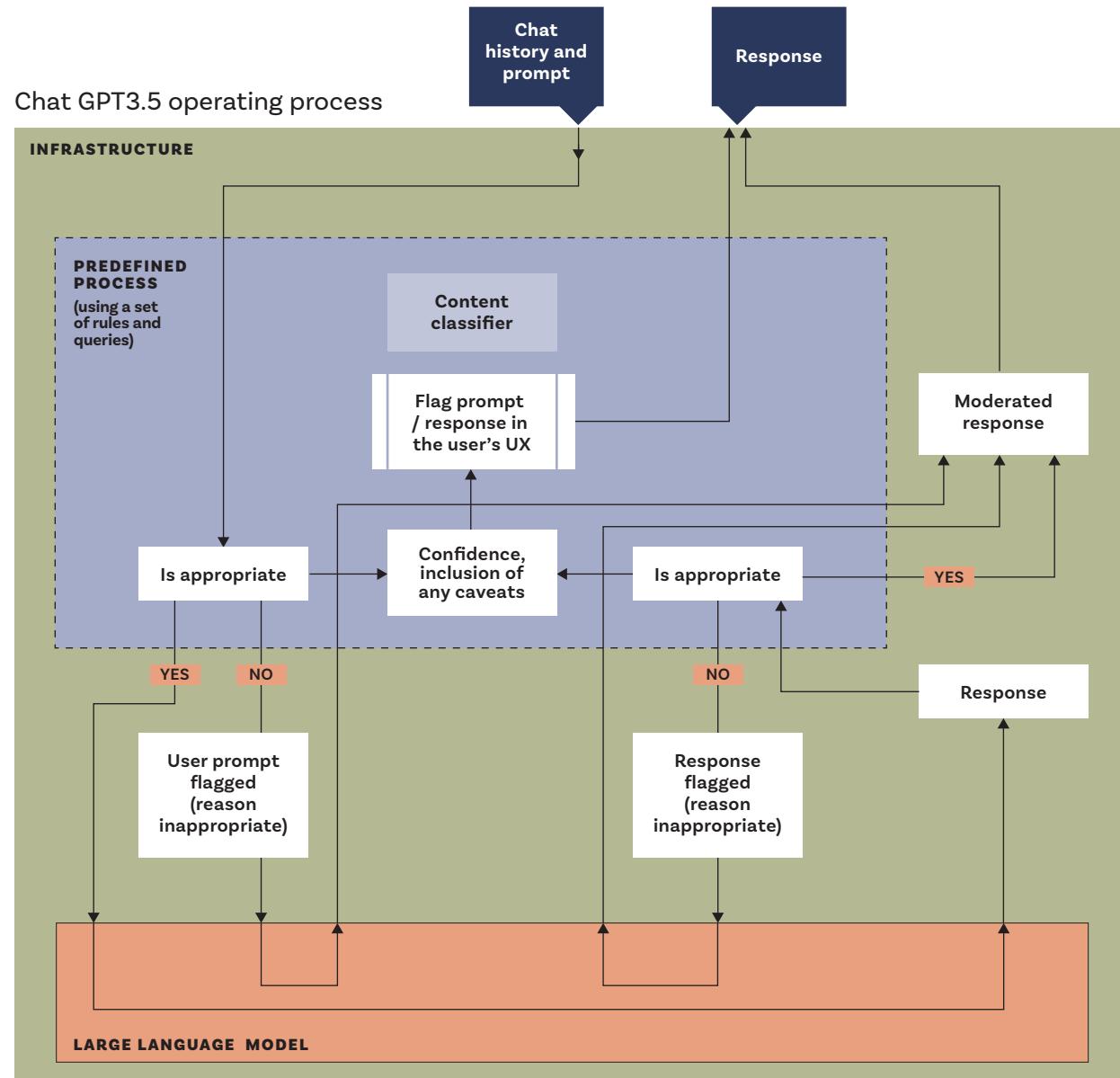


Table 1: Activities and risk management strategies throughout the generative AI model development lifecycle (source: developed for this rapid response report, on Generative AI, to NSTC)

Lifecycle Stage	Activity	Risk management strategies
Model pre-training	Models are trained on large, general datasets to perform general, abstract tasks like predicting subsequent words (LLMs) or associating existing images with captions (MFM).	Data are edited, added or removed to reduce bias and improve quality, and sensitive or undesired data or attributes may be removed (e.g., faces, adult content, copyright images). ⁶⁻⁸
Model fine-tuning	After pre-training on general data, models are fine-tuned for a specific application that builds on the general abilities learned in pre-training, such as helpfully and safely answering user queries (LLMs), or generating novel images from descriptions (MFM). Developing these models require more human intervention.	Fine-tuning relies on human judgement; it involves the use of static datasets (supervised learning) or interactive feedback from humans or other automated tools (reinforcement learning). ^{5,7-9} Fine-tuning is critical to make LLMs or MFM safer and more useful, e.g., ChatGPT was fine-tuned to become a dialog agent that can answer queries in a prompt-response format and follow instructions.
Implementing input and output filtering	Developers shape the user inputs to a model, and the generated outputs from a model, to mitigate risks or further improve model performance on the target application. Developers try to balance allowing users to ‘steer’ a model’s output to specific applications, while also enforcing ‘guardrails’ that prevent malicious or out-of-scope uses. ⁵	Inappropriate or out-of-scope user inputs (e.g., requests to generate adult content or medical advice) may be blocked or receive special treatment using, e.g., keyword matching. Model outputs may also be restricted using existing content moderation APIs or bespoke interventions (e.g., to prevent generation of adult content, or of celebrity deep fakes). ^{10,11} Outputs may also be watermarked to highlight that the content was AI-generated. ¹²
Pre-release testing	Developers typically test their application throughout the development process, and especially before release, to identify and amend problems. Due to the higher stakes, model testing is becoming more planned, structured and intentional.	‘Red-teaming’, a technique used in cybersecurity, is employed – whereby a group of users try to find flaws in the system and/or suggest design improvements. ^{11,13,14,12} ‘Fuzzing’ may also be used, whereby automated tools change system inputs and check for incorrect outputs. ¹⁵
Release and distribution	The developer of an application can choose a range of strategies for making the system available. Historically, AI researchers have tended to open-source their model source code and parameters, but as LLMs and MFM increase in cost and potential risks, release strategies are becoming more controlled. ¹⁶⁻¹⁸	Limited access protocols (e.g., giving developers paid access to an API and the general public access to a web interface (which OpenAI does for ChatGPT) may be implemented, rather than publishing a system as open-source. Specialised licences for datasets and models are also increasingly employed. ¹⁹ Model documentation may also be released, and there are efforts to standardise the information that is reported (e.g., features, risk, limitations, assumptions). ²⁰⁻²²
Post-release monitoring	After release, a developer will continue to monitor the way users interact with an application to make appropriate updates.	Developers have in the past offered monetary rewards or priority access to advanced models for users who report problems such as biases or bugs or who contribute to evaluating model performance. ²³⁻²⁵ Internal and external (e.g., academics or public interest groups) auditing may also be done to examine the design of models to find potential harmful effects. ²⁶ In the worst case, a developer may even attempt to partially or fully retract an LLM or MFM application that is found to be deeply problematic. ²⁷

Who is developing LLMs and MFMs?

To date, despite OpenAI catalysing the most recent interest in generative AI, big companies such as Alphabet^d and Microsoft are at the forefront of LLM development and their initial monetisation. Recently, Chinese and Indian tech companies have announced their own LLM-based chatbots, including Baidu's Ernie Bot.²⁸⁻³⁰ Meta has also announced its own LLM (LLaMa) aimed at academics, with a more energy efficient footprint. In pre-print articles, the developers of LLaMa have shared their dataset sources for their model and their fine tuning, as well as their entire model's architecture.^{31,32} Some universities, such as Stanford, are also innovating in this space, building on LLaMa, to create an open source model trained on ChatGPT, running on a laptop, for less than U\$600.³³ If this work proves stable and scalable, it has significant implications for the current eco-system and its business models.

LLMs and MFMs, and the applications, services and business models that are built on them, rely on a larger technology 'stack', including application program interfaces (APIs), machine learning operation management (MLOps), machine learning (ML) acceleration software, and supercomputing and cluster-based infrastructure (see Figure 2). This means there are lots of other players involved in the current generative AI wave. As more services, applications and business are built on top of LLM/MFMs, regulation may be necessary for safe and responsible management of generative AI and there are multiple points of intervention possible within the technology stack (see Figure 2).

^d Google's addition (Bard) that was released on 21 March 2023.

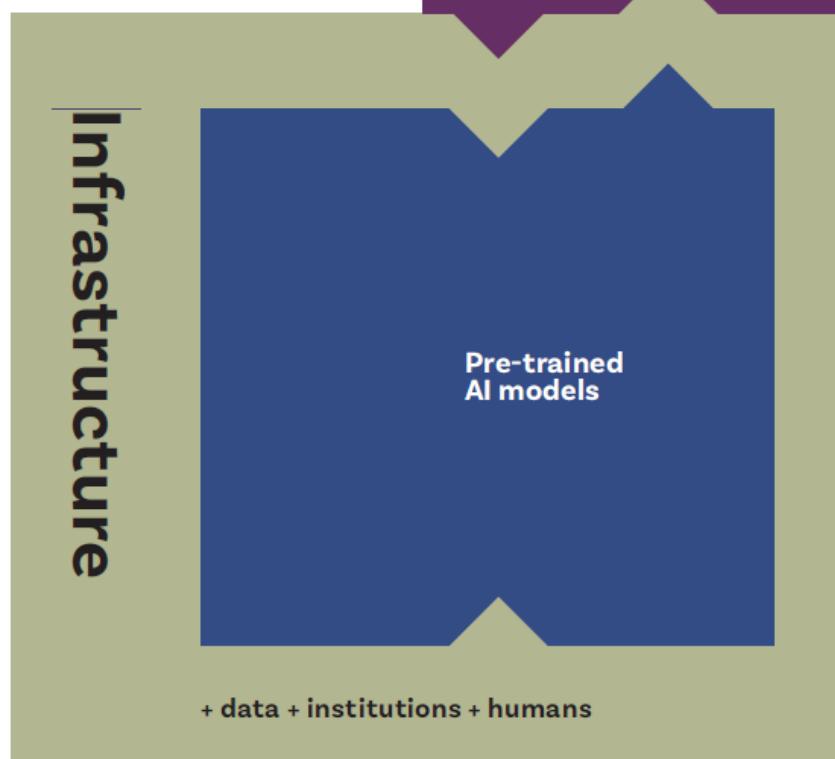
Figure 2: The generative AI technology stack (source: developed for this rapid response report, on Generative AI, to NSTC)

Generative AI overview

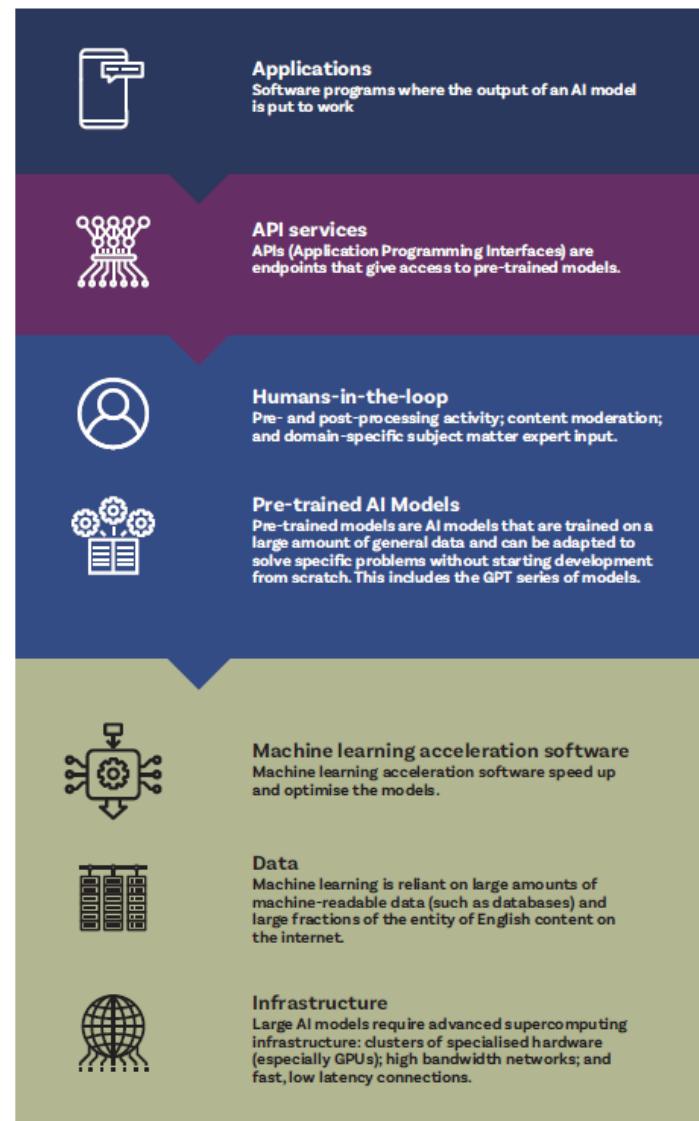
ChatGPT is an application that uses a Large Language Model (LLM), an AI designed to respond to and generate text. The training data for the current version (Chat GPT3.5 operating process) included more than 600 billion tokens—about 450 billion words. Chat GPT has been developed by OpenAI (owned by private entity OpenAI LP).

Microsoft's Bing search engine and Google's Bard are other applications which use Large Language Models.

GPT = Generative Pre-trained Transformer



Technology stack



The concentration of generative AI resources within a small number of large multinational and primarily US-based technology companies poses potential risks to Australia. Given the resource-intensive nature of LLM/MFM-based generative AI, can Australia be competitive in the production or fundamental research of these technologies? Australia has capability in AI-related areas like computer vision and robotics, and the social and governance aspects of AI, but its core fundamental capacity in LLMs and related areas is relatively weak.³⁴ While the Australian Government announced investments of \$100 million in AI-related initiatives (e.g. including a national AI centre),^{35,36} creating generative AI technologies has especially high barriers to access, due to its considerable compute and data requirements. The technologies also have requirements for skilled workers, OpenAI team currently has 375 employees.³⁷ We do have large public datasets, such as the Bureau of Meteorology, Australian Bureau of Statistics and the National Library of Australia's Trove, that could offer opportunities for generative AI in Australia.

Opportunities for LLM and MFMs

LLMs and MFMs, and the applications, services and business models based on them, have implications for the Australian economy, now and into the future. It has long been recognised that AI-based technologies may lead to job losses where they enable machines to replace the work of humans undertaking particular tasks; a recent analysis of the exposure of U.S. workers to the potential effects of GPTs estimates that in 19% of U.S. jobs, at least 50% of tasks are exposed, meaning that these tasks correspond to the known capabilities of generative AI. The jobs concerned span all income levels.³⁸ In a broader study of the economic impacts of AI, there is the potential for excessive automation, where firms do not take negative impacts on workers into account.³⁹

Others observe that automation often complements human labour, such as workplaces where robots are introduced often increasing total employment.⁴⁰ The fact that a machine may perform one or more relevant tasks does not mean that job replacement will necessarily occur, indeed the strongest business cases for investment in AI are likely to emphasise the creation of additional value to products or services rather than savings in labour costs.⁴¹ In these cases, technologies such as generative AI are likely to both create new jobs and augment existing ones by enhancing human decision-making skills.^{41,42}

The World Economic Forum's *2020 Future of Jobs Report* combines statistical evidence with interviews with business leaders in order to describe the future outlook for jobs and skills. The Forum predicts that currently emerging technologies, including but not limited to generative AI, will create more jobs than they destroy, but notes that while job destruction is accelerating, job creation is slowing.⁴³ Policy makers should be prepared for the challenge of balancing investment in generative AI with the need to manage job displacement and ensure decent human working conditions in automated workplaces.^{44,45,46}

As with the introduction of any new technologies, the impacts are likely to be unevenly distributed.⁴⁷ An initial impact on workplace productivity may involve the automation of basic tasks, such as letter-writing, and supporting higher-level tasks, for example, web development. Early adopters of existing AI, including information-based industries, research and healthcare, may benefit, through, for example, textual analysis and image processing, before a wider uptake of the technology occurs across the economy.

To maximise economic benefit, businesses will need to integrate generative AI into their operations and develop new business models, products and services, in addition to using the new tools to enhance the productivity of existing processes. They will also need to contemplate new job categories and skills – for instance, data curation and prompt engineering. There could be an increase in demand for digital skills, with the entry of generative AI expected to require 161,000 AI specialisations globally by 2030.^{48,49} It is predicted that there will be a rapidly increasing demand for skills in digital technologies in Australia, with skills in software orchestration/automation, AI and data analysis predicted to grow especially quickly. However, some of these new occupations, such as data scientist and data engineer, do not exist in the current Australian occupation classification system.⁵⁰

Measures of the Australian digital economy, including the mapping and classification of the skills it will require, are not yet well developed. However, a substantial and capable workforce will be critical for Australia to be generative AI-ready. Over the next two to five years especially, competition for talent both domestically and internationally and a shortage of skilled workers represent a key risk for Australia.^e

^e A cautionary example is provided by the United States, in which the tech sector has recruited many teaching and research ML specialists from the university sector, leaving few staff left to teach and supervise PhDs. This has led to a

Future industry and public sector opportunities

The rapid expansion of ChatGPT shows the potential of generative AI technologies is difficult to predict over the next two years, let alone ten. This is especially true given the large financial investments that are currently being made, and the expectations of future profits. However, one way to model what might happen is to focus on impact spaces rather than specific opportunities.

LLMs and MfMs are already being used across a range of industry settings from health and engineering to social services and creative industries. As noted elsewhere, we are also seeing these models being integrated into existing systems, including search and productivity software such as Microsoft 365 Copilot, Google Workspace and Bing.⁵¹⁻⁵⁴

LLM and MfMs are being used to analyse medical images⁵⁵⁻⁵⁷ and consolidate patient data,^{58,59} evaluate and optimise engineering designs,³ support social service provision,⁶⁰ analyse and generate documentation in legal services, generate creative material in the visual arts, music and filmmaking, as well as journalism, advertising and marketing.^{61,62} Educators are experimenting with the integration of LLMs and MfMs into classroom activities, curriculum and assessment, as well as developing services to detect their inappropriate use.⁶³

These current applications are an indication of the very broad spectrum and potentially rapid uptake of future integrations into industries and the public sector. As these applications develop further it will be important to understand how they interact across sectors; how applications based on LLMs and MfMs integrate and connect with each other, and how they interact with existing general-purpose technologies, such as search engines.

Risks of LLMs and MfMs

LLMs and MfMs, and the services, applications and businesses built with them, have already amplified longstanding public and expert concerns about the higher-scale risks of AI, including existential risks.⁶⁴ For instance, conversations about ChatGPT, in daily life and in the press, routinely evoke questions about what it means to be human, the role of computing in daily life, the perils of next-stage automation and fears about runaway, uncontrollable technology.⁶⁵

Heightened concerns could create a polarised and unproductive public debate, which may then dominate our responses to future uses of these applications. There is unlikely to be a consensus on these issues. It will be important to maintain an active and informed conversation on the uses and applications of these emerging technologies.

While LLM and MfM-based generative AI is relatively new, and the services, applications and businesses utilising them are nascent, we can build on what we already know to make sense of the risks posed on a spectrum from narrow to broad categories. There are three important categories of risk:

- **Technical system risks**, both for the model itself and its data. These include validity and reliability; trust in and accuracy of answers; safety; security and resilience; system accountability and transparency; explainability and interpretability; privacy; management of biases and other assorted quality assurance considerations.

critical undersupply of ML-qualified graduates.^{147,148} Joint positions between industry and academia are being explored as a potential solution to this problem.⁷⁹

- **Contextual and social risks**, including risks to human rights and values arising from AI use in high-stakes contexts (such as law enforcement, health and social services), and risks posed by the more ‘routine’ deployment of AI that reproduces and accelerates existing social inequalities.
- **Systemic social and economic risks**, including impacts on democratic systems; social discourse and dialogue; environmental impacts; transformation of work; mistrust in private and public sector organisations and market dominance by a small number of transnational corporations providing generative AI as a platform or service (including the issues of maintenance, and decommissioning of legacy systems).

The extent that these risks are realised or mitigated will depend on the actions of governments, industries, developers and consumers. Trust will be critical for the adoption of LLMs and MFMs and the applications and services built on them. Trustworthiness will be built through appropriate levels of reliability, transparency, accountability and legal, policy and other safeguards.

Generative AI will also raise regulation and deployment considerations to ensure existing and new inequalities are not exacerbated or initiated. AI tools require considerable internet bandwidth, power and suitable devices, which are not available or affordable to everyone. Regional Australians and older Australians particularly experience poorer digital inclusion.⁶⁶

[Accuracy and bias](#)

A major limitation of LLMs, and the applications built on them, is the accuracy and quality of the answers generated. They are only as good as the data they are trained on; the models use statistical analysis to determine the ‘correct’ next word, not an understanding of the content, and the user interfaces can shape the way users perceive the validity of the answers.

Inaccuracies and bias

- In some cases, outputs can be entirely erroneous, or simply misleading – known as ‘hallucination’.⁶⁷ Furthermore, the way a question is asked (to ChatGPT, for example) changes the perceived tone of confidence in the response. For example, if the user includes words such as ‘expert’, ‘technical’ or ‘consultant’ in their question, the system may respond with ‘experts say’ prefacing incorrect information.⁶⁸ Future generations of LLMs may need to cite genuine sources to provide sufficient reasoning for their results – noting that currently they sometimes invent references when asked, with potential problematic impacts.⁶⁹
- Representational bias where, for example, a model is only trained on Western literature or male ‘voices’, can exacerbate existing social inequalities.^{70,71} The consequences could be severe if applied to sectors such as law enforcement, recruitment (e.g., translation services) and social services.⁷²
- While healthcare is a key opportunity, it provides a case study for how a sector can replicate existing biases. The exclusion of women, and other minority groups, from medical research is well documented, leading to poorer health outcomes.⁷³ Conversely, well-designed LLM tools could assist in countering medical biases with a more reliable assessment of reported symptoms. Finally, models trained on overseas datasets may fail to capture place-based factors, such as diagnostics for bushfire-related respiratory issues. Deliberate training and review can address these limitations.⁷⁴
- In predictive policing, one example of ‘dark forecasting’, AI can perpetuate existing inequalities in over-policed populations. Some previous AI-predictive policing tools have been discontinued due to inbuilt bias.^{75,76} Racial profiling has been the subject of scrutiny as this can be both countered and inflated by algorithms.⁷⁷

Misinformation

- LLMs and MFM have the potential for misuse by generating high-quality, cheap and personalised content, including for harmful purposes. Tools built on these models are already in use to generate deep fakes (high-quality artificial images, video and speech for disinformation, including by state actors) indistinguishable, at least without special training or access to technical tools, from human-generated content.⁷⁸ Existing challenges related to the spread of misinformation may be amplified as AI-generated content circulates alongside other information.
- LLM-generated content could also be misused in democratic processes such as parliamentary consultations by creating a flood of submissions to mislead public opinion. While they provide ample opportunities for misuse, the capability of generative AI can also be used to detect harmful content, as well as the inappropriate use of generative AI in other contexts, such as education settings.⁷⁹

Pre- and post-Processing

- Applications and services based on LLMs and MFM use a range of pre- and post-processing activities, such as restating the query in a way that the LLM or MFM can best address, and declining ‘inappropriate’ requests based on pre-defined rules. However, this processing encodes values that are not always transparent to the user or potential regulators, and can be subverted through adversarial practices such as ‘jailbreaking’⁸⁰ - the process of exploiting a system’s features to remove provider-imposed restrictions on its use.

Human rights

Where an AI-enabled system has no clear human decision-makers, it is challenging – but essential – to establish responsibility for adverse impacts.⁴⁷ Most LLMs and MFM are ‘black box technologies’ where the public cannot understand how the model arrives at its outputs, making it difficult, or potentially impossible, for a human to assess the reliability of the results or seek redress.⁵⁸

Institutional protections that apply a ‘human-in-the-loop’ approach to ensure accountability and fairness may assist, alongside other design considerations, in addressing these issues for future digital services. ‘Human-in-the-loop’ requirements may not be appropriate where the benefits of an application are dependent on efficiency at scale; some risks will be better addressed by other approaches to monitor risks, identify errors, and provide access to remedies. Comprehensive and ongoing risk assessments and human rights due-diligence may help identify risks and mitigation strategies that are context-sensitive and appropriately tailored.⁸¹

Data privacy, security and sovereignty

To date, commercial organisations building LLMs and MFM have not shared a great deal of specific details regarding the training datasets they are using, and their providence, which could include the purchase of third-party datasets and data scraping. It seems likely that permissions have not always been provided for use of large datasets drawn from the internet. Under existing Australian privacy law regarding personal data scraping, the lawfulness of some of these training sets could be questioned. For example, the Office of the Australian Information Commissioner, via the Australian Information and Privacy Commissioner, recently made a determination that was critical of the use of data scraping by Clearview AI to build its facial recognition service.⁸² Attribution and reference to licences of existing copyright material remain issues to be appropriately addressed, as is the copyright of material generated by the language models.⁸³ This can be seen in applications as diverse as coding and in art (both for artists whose work is used in

training data and for artists using generative AI in creative works). Data sovereignty is particularly a consideration for First Nations data.^{84,85}

As systems become integrated, the management of privacy and consent when collecting, sharing and using datasets will need further attention. As generative AI is integrated into Australian systems, there will be questions regarding sovereign ownership of LLMs and MFMs, and the data they are trained on, particularly if integrated into public systems such as healthcare and education. New methods for providing and handling consent, frameworks for sharing and using data, and considerations for security in highly complex networks and with shared public–private ownership will be required.

AI presents new opportunities for data privacy breaches, for example, in the reidentification of anonymised data used for LLMs.⁸⁶ Data security is a key risk, particularly with cyberattack methods to extract training data.^{87,88} In healthcare for example, there are risks for both patients who have LLMs as part of their care and patients whose data is used in a model’s training dataset. This also presents an ethical challenge as patients may not have consented to this use of their data. Some say LLMs should be only trained on public data to avoid this issue; however, this results in less powerful LLMs with a higher risk of bias, as very few medical datasets are publicly available.⁸⁹

Computing power: environmental impact and capacity

As seen in Figure 2, access to computing infrastructure is a critical enabler and challenge for some countries for generative AI. LLMs and MFMs require supercomputing-like capability, most of which is found in the US, China and Europe.^{90,91}

To date, the creation of LLMs and MFMs have required large datasets stored in large data centres, and their use incurs further high costs in compute and data processing power.⁹² Managing the energy and water consumption of training and retraining (including data collection and cleaning) and operating LLMs and MFMs is a challenge.^{93,94} While techniques have improved the energy efficiency of algorithms, hardware upgrades and increasing levels of e-waste from computer components will heighten demand for critical minerals with resultant environmental and human rights impacts.^{94,95}

International strategies to address the opportunities and risks posed by LLMs and MFMs

Internationally, general AI strategic investment has begun to focus on generative AI. Examples include the UK's AI Strategy and investment of £900 million in an AI supercomputer (to help build LLMs and MFMs), Germany's €3 billion investment by 2025, the US Government funding analysis in AI, China's plan to be a global leader in AI by 2030, and investment globally in AI start-ups.^{96–98}

As the tech sector develops LLMs and MFMs, other sectors, such as finance and banking, will favour business models that leverage them. For example, Microsoft has integrated OpenAI's GPT-4 into the Bing search engine. There are indications of emerging competitive responses in this area, including the appearance of open-source development environments and an ethos of open-source AI innovation. Amazon Web Services' strategic partnership with Hugging Face, which promotes open source contributions, may signal the growing significance of alternative models for intellectual property and commercialisation in AI.⁹⁹

Even so, the intensive infrastructural and computational resources required for the development of generative AI, and ongoing research and innovation, are concentrated in a small number of firms and countries. The trend is towards more concentration and increased geopolitical competition. For example:

- the US's CHIPS Act and parallel EU measures aim to ensure ongoing onshore computational capabilities for future AI-driven industries, with a focus on infrastructure and semiconductor design and fabrication. Initiatives such as the proposed US National Artificial Intelligence Research Resource aim to shape markets and direct innovation and competition policies towards a domestic AI innovation system more closely aligned to national interests.¹⁰⁰
- China has provided policy and financial support to develop the AI industry as a national priority, with the ambition to lead in both research and application by 2030.^{101–103} It has been suggested that the success of ChatGPT may be a 'Sputnik moment' for China.¹⁰⁴ It has coincided with a renewed focus on chips and AI research, training and recruitment, and development of Chinese language models, such as MOSS, an AI chatbot and rival to ChatGPT launched in February 2023 for public testing, and Baidu's Ernie Bot launched in March, while China has censored ChatGPT.^{105,106} The result is parallel AI developments with local versions of technology developed as external versions are fenced out.

For smaller countries and markets like Australia, this competition could present challenges for access and capability, as well as the suitability of models for our context and needs. Equally, it could present opportunities for local firms, government entities and publicly funded research organisations to adapt and fine-tune small and large models for Australia-specific industries and research across all sectors.

Legal and regulatory responses

To date, global approaches to AI governance fall into two broad categories: government regulatory actions through legislation or regulator guidance, and self-regulation and voluntary standards. Whether mandatory or voluntary, these policies and regulations seek to require anyone developing and deploying AI to identify, mitigate, monitor and address risks of harm or misuse. AI specific policies and regulations operate in addition to other laws that can impact the use or misuse of AI and its applications, including privacy, tort, anti-discrimination, competition and consumer law.

Governments have been heavily involved in encouraging and supporting the development of ‘soft law’^f to allow safe but flexible innovation of AI in general.^{107,108} Globally, more than 630 ‘soft law’ AI governance programs have been identified as being developed and published between 2016 and 2019, with the number increasing substantially over time.¹⁰⁸ But their effectiveness is debatable. Technical standards are being developed by international standards organisations to assist in this process.¹⁰⁹

Specific regulatory frameworks to address generative AI, including LLMs and MFMs, are currently being developed but have not yet been deployed in Australia or overseas. There is a growing recognition that a range of institutional measures and policies are likely to be required to mitigate public risks. However, risk management approaches are most effective within a specific context of use, and against technical rather than social or systemic risks, such as use by social media platforms. There are also challenges in regulated industries regarding auditability and what constitutes an audit trail when a generative AI is changing and adapting. Regulatory frameworks are in development for managing risks associated with AI more generally and may also be expanded to cover LLMs and MFMs.

The European Union (EU) has proposed the EU AI Act. The EU model is notable for differentiating between AI use cases: banning unacceptable uses and identifying others as ‘high risk’, where active ex ante compliance and ongoing monitoring is required. The model applies differentiated obligations on actors within the AI supply chain: providers, suppliers, importers and users. To date, only an EU Directive, different from the EU AI Act, tangentially deals with the question of responsibility in the case of general-purpose models.^{110,111} The EU’s proposed framework for governing AI may not encompass the dynamic range of contexts in which they can be used and there is debate over whether general purpose models or applications could be categorised as ‘high risk’. It has been suggested that the EU AI Act, may, in the end, exclude LLMs and MFMS from its scope, until further consultation can occur.

Canada is moving in a similar direction.¹¹² Canada already has in place law requiring future impact assessments for the use of automated systems in the public sector. As drafted, it will apply to the deployment of systems based on LLMs/MFMs. In March 2023, Canada published plans to extend risk-based regulation and ‘interoperate’ with the EU Act. The initial focus of Canada’s proposed new regulator, the AI and Data Commissioner, will be on education and upskilling, but the proposed legislation (the Artificial Intelligence and Data Act) is structured like the EU Act, and grants the regulator powers to require audits, and even order suspension of an AI system’s use.

At the other end of the regulatory spectrum, the US relies on self-regulation, which includes public-sector driven, but voluntary multi-stakeholder processes to develop risk management and technical standards, similar processes in specific domains (such as medical devices), and contributing to international standards bodies. The US and Singapore have also developed specific tools to support AI developers and users to identify and mitigate risks.

China has its own approach involving government-led public–private sector partnerships on AI regulatory guidelines, coupled with strong government support for development of local technology and companies. A UK White Paper on regulating AI is expected shortly. Appendix 3 provides further details of approaches in different jurisdictions.

^f “Soft law” is a term usually used to refer to statements, declarations, or sets of principles that do not have the force of law, but are still intended to influence behaviour of firms and people. In this space, it refers to documents such as the Government’s AI Ethical Principles and statements and AI Ethics Principles. They have no legal force or effect (they are not mandatory; no sanction arises if they are breached) but clearly intended to influence the use and development of AI. Codes of Practice (that are not adopted into law or otherwise given legal force) can also be soft law instruments assuming they are not made binding in some way, eg via a contract.

Australia's current approach to technologies is largely through self-regulation and voluntary standards approach, but these have historically been technologically neutral. There are some laws that may impact the way AI systems are designed or the context they operate, such as the Copyright Acts 1968, Privacy Act 1988, Consumer Act 2010, Fair Work Act 2009 and laws related to anti-discrimination^g. There is currently no legal obligation for developers to undertake a risk assessment, except firms who supply to the NSW Government (under contractual obligation rather than legislation).

Multi-stakeholder and sector-specific development

LLMs and MFM will challenge risk-based approaches as they change the nature, predictability and scale of potential risks, and make it harder for any one entity to identify, assess or mitigate those risks.¹¹³

Meanwhile, governments and public and private sector organisations are responding to the specific risks ChatGPT is thought to pose. In education, for example, some Australian states, universities and schools have banned ChatGPT on the basis that it could aid academic dishonesty, while others have adopted a less restrictive approach, encouraging educators and learners to experiment, and acknowledge the service it can provide students and teaching staff.^{114–117} There is public discussion too of the legal and professional risks that could arise, for example, from lawyers, utilising ChatGPT to generate legal advice or inputting confidential information as part of user prompts.

There is a proliferation of voluntary principles, guidelines and standards for trustworthy AI and several multi-stakeholder coalitions (bringing together industry, government representatives and civil society organisations such as the OECD Global Partnership on AI and the industry-led Partnership on AI).¹¹⁸ These seek to develop industry consensus around emerging best practices, including documentation such as cards for model reporting and datasheets for datasets.^{119,120} Individual firms have published their own risk-based approaches: OpenAI has published 'Best Practice for Deploying Language Models',¹²¹ Microsoft has published a range of documentation on risks,¹²² threat modelling¹²³ and responsible use.¹²⁴ However, in 2023, we have seen the same organisations backing away from their prior guidelines and reducing their internal teams working on ethics and related issues.¹²⁵

International standards bodies are also active in developing standards for AI risk assessment^{126–129}, with Australia an active participant in ISO processes. There are also some attempts at rating systems and standards being promoted by civil society and public oversight groups, such as: Ranking Digital Rights^{130,131} which rates digital platform companies on human rights; and AlgorithmWatch's SustAIIn, which seeks standards for social and environmental sustainability on all aspects of AI development and deployments.⁹⁵

Areas for ongoing attention

Generative AI is transformational and is already beginning to change how we live and work. Decision-makers and the broader Australian community need a stronger understanding of its risks and opportunities if we are to successfully manage its rapid development, use and uptake over the next five years.

LLMs and MFM are evolving very rapidly and are likely to continue to do so. At the same time, additional information and analysis about the major current models continue to appear in the public domain. A clear

^g For example, copyright may affect the use of text and images as training data (at least if it occurred in Australia, could well infringe copyright). If personal information is involved, either as training data or inputted into generative AI systems, that would involve use of personal data subject to privacy legislation (at least if done by non-excluded entities ie large businesses and public sector entities).

understanding of the most important challenges, risks and opportunities of these models would benefit from ongoing attention. Critical areas are likely to include:

- the appearance of new LLMs and MFMIs and the business models, applications and services built on them;
- critical evaluations and risk assessments of the LLMs and MFMIs that help explicate the nature of training datasets, energy use, compute budgets and pre/post processing activities;
- the scale and nature of harmful social outcomes of LLM and MFM-based applications and the reasons for these outcomes;
- early examples of successful and sustained integration of LLMs and MFMIs into workforce and enterprise organisations; and
- regulatory developments and responses in other jurisdictions, together with multilateral and non-government governance initiatives.

References

1. Yang, Z. China tech giant Baidu releases its answer to ChatGPT. *MIT Technology Review* (2023).
2. Lai, C., Ahmad, S., Dubinsky, D. & Maver, C. AI is harming our planet: addressing AI's staggering energy cost. *Numenata* <https://www.numenta.com/blog/2022/05/24/ai-is-harming-our-planet/> (2022) [Accessed 21 March 2023].
3. Narayanan, D. *et al.* Efficient Large-Scale Language Model Training on GPU Clusters Using Megatron-LM; Efficient Large-Scale Language Model Training on GPU Clusters Using Megatron-LM. (2021) doi:10.1145/3458817.3476209.
4. Rick Merritt. What Is a Transformer Model? *NVIDIA* <https://blogs.nvidia.com/blog/2022/03/25/what-is-a-transformer-model/> (2022) [Accessed 1 March 2023].
5. OpenAI. *GPT-4 Technical Report*. <https://cdn.openai.com/papers/gpt-4.pdf> (2023).
6. Kaminski, M. E. Regulating the Risks of AI. *Forthcoming, Boston University Law Review, Vol. 103, 2023, U of Colorado Law Legal Studies Research Paper No. 22-21* (2022) doi:10.2139/SSRN.4195066.
7. Christiano, P. F. *et al.* Deep reinforcement learning from human preferences. *Adv Neural Inf Process Syst 2017-December*, 4300–4308 (2017).
8. § Wei, J. *et al.* Finetuned Language Models Are Zero-Shot Learners. (2021) doi:10.48550/arxiv.2109.01652.
9. § Bai, Y. *et al.* Constitutional AI: Harmlessness from AI Feedback. (2022) doi:10.48550/arxiv.2212.08073.
10. OpenAI. DALL·E Content policy. <https://labs.openai.com/policies/content-policy> (2022) [Accessed 2 March 2023].
11. § Rando, J. *et al.* Red-Teaming the Stable Diffusion Safety Filter. (2022) doi:10.48550/arxiv.2210.04610.
12. Aaronson, S. My AI Safety Lecture for UT Effective Altruism. *Shtetl-Optimized* <https://scottaaronson.blog/?p=6823> (2023) [Accessed 2 March 2023].
13. Hugging Face. Clean Diffusion 2.0 PoC Model Card. <https://huggingface.co/alfredplpl/clean-diffusion-2-0-poc> [Accessed 2 March 2023].
14. OpenAI. DALL·E 2 Preview - Risks and Limitations. <https://github.com/openai/dalle-2-preview/blob/main/system-card.md#external-red-teaming> (2022) [Accessed 2 March 2023].
15. Gao, X., Saha, R. K., Prasad, M. R. & Roychoudhury, A. Fuzz testing based data augmentation to improve robustness of deep neural networks. *Proceedings - International Conference on Software Engineering* 1147–1158 (2020) doi:10.1145/3377811.3380415.
16. Solaiman, I. *et al.* OpenAI Report Release Strategies and the Social Impacts of Language Models. (2019).
17. Shevlane, T. Structured Access: An Emerging Paradigm for Safe AI Deployment. *The Oxford Handbook of AI Governance* (2022) doi:10.1093/OXFORDHB/9780197579329.013.39.
18. Shevlane, T. & Dafoe, A. The offense-defense balance of scientific knowledge: Does publishing AI research reduce misuse? *AIES 2020 - Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society* 173–179 (2020) doi:10.1145/3375627.3375815.
19. Contractor, D. *et al.* Behavioral Use Licensing for Responsible AI. *ACM International Conference Proceeding Series* 778–788 (2022) doi:10.1145/3531146.3533143.
20. Mitchell, M. *et al.* Model Cards for Model Reporting. *FAT* 2019 - Proceedings of the 2019 Conference on Fairness, Accountability, and Transparency* 220–229 (2018) doi:10.1145/3287560.3287596.
21. § Gilbert, T. K., Dean, S., Lambert, N., Zick, T. & Snoswell, A. Reward Reports for Reinforcement Learning. (2022) doi:10.48550/arxiv.2204.10817.
22. Gebru, T. *et al.* Datasheets for datasets. *Commun ACM* **64**, 86–92 (2021).
23. Garg, V. Mitigating Prompt Injection Attacks on an LLM based Customer support App. <https://vaibhavgarg1982.medium.com/mitigating-prompt-injection-attacks-on-an-llm-based-customer-support-app-b34298b2bc7a> (2023) [Accessed 2 March 2023].
24. OpenAI. ChatGPT Feedback Contest: Official Rules. (2022).
25. Chowdhury, R. & Williams, J. Introducing Twitter's first algorithmic bias bounty challenge. *Twitter Engineering*

- https://blog.twitter.com/engineering/en_us/topics/insights/2021/algorithmic-bias-bounty-challenge (2021) [Accessed 2 March 2023].
26. Raji, I. D. *et al.* Closing the AI accountability gap: Defining an end-to-end framework for internal algorithmic auditing. *FAT* 2020 - Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency* 33–44 (2020) doi:10.1145/3351095.3372873.
 27. Snoswell, A. J. & Burgess, J. The Galactica AI model was trained on scientific knowledge – but it spat out alarmingly plausible nonsense. *The Conversation* (2022).
 28. Ahmed, N., Wahed, M. & Thompson, N. C. The growing influence of industry in AI research. *Science* (1979) **379**, 884–886 (2023).
 29. Tobin, M. & Li, L. Ernie, what is censorship? China’s chatbots face additional challenges. *The Washington Post* (2023).
 30. Sharma, D. India gets its first ChatGPT-powered AI chatbot Lexi, here are the details. *India Today* (2023).
 31. § Touvron, H. *et al.* LLaMA: Open and Efficient Foundation Language Models. (2023).
 32. § Wang, Y. *et al.* Self-Instruct: Aligning Language Model with Self Generated Instructions. (2022).
 33. Taori, R. *et al.* Alpaca: A Strong, Replicable Instruction-Following Model. *Stanford University* <https://crfm.stanford.edu/2023/03/13/alpaca.html> (2023) [Accessed 21 March 2023].
 34. Desmond, M., Duesterwald, E., Isahagian, V. & Muthusamy, V. A No-Code Low-Code Paradigm for Authoring Business Automations Using Natural Language. (2022) doi:10.48550/arxiv.2207.10648.
 35. Department of Industry, S. and R. Funding available for AI and Digital Capability Centres. <https://www.industry.gov.au/news/funding-available-ai-and-digital-capability-centres> (2022) [Accessed 21 March 2023].
 36. The Hon Melissa Price MP Media Releases. \$44 million to build AI and digital capability centres. *Ministers for the Department of Industry, Science and Resources* <https://www.minister.industry.gov.au/ministers/price/media-releases/44-million-build-ai-and-digital-capability-centres> (2022) [Accessed 21 March 2023].
 37. Roose, K. How ChatGPT Kicked Off an A.I. Arms Race . *The New York Times* <https://www.nytimes.com/2023/02/03/technology/chatgpt-openai-artificial-intelligence.html> (2023) [Accessed 24 March 2023].
 38. § Eloundou, T., Manning, S., Mishkin, P. & Rock, D. GPTs are GPTs: An Early Look at the Labor Market Impact Potential of Large Language Models. (2023).
 39. Acemoglu, D. *Harms of AI*. (2021).
 40. Coelli, M. B. & Borland, J. The Australian labour market and IT-enabled technological change. *SSRN Electronic Journal* (2023) doi:10.2139/SSRN.4326257.
 41. Agrawal, A., Gans, J. & Goldfarb, A. *Power and Prediction: The Disruptive Economics of Artificial Intelligence*. (Harvard Business Review Press, 2022).
 42. Aghion, P., Antonin, C., Bunel, S. & Jaravel, X. The Effects of Automation on Labor Demand: A Survey of the Recent Literature. *CEPR Discussion Paper No. DP16868* (2022).
 43. World Economic Forum. *The Future of Jobs Report 2020*. <https://www.weforum.org/reports/the-future-of-jobs-report-2020> (2020).
 44. O’Neill, C., Goldenfein, J., Sadowski, J., Kelly, L. K. & Phan, T. Burnout by design? Warehouse and shipping workers pay the hidden cost of the holiday season. *The Conversation* (2021).
 45. Acemoglu, D. *Harms of AI*. <https://www.nber.org/papers/w29247> (2021) doi:10.3386/w29247.
 46. Agrawal, A., Gans, J. & Goldfarb, A. Economic policy for artificial intelligence. *Innovation Policy and the Economy* **19**, 139–159 (2019).
 47. Australian Human Rights Commission. *Human Rights and Technology*. https://tech.humanrights.gov.au/downloads?_ga=2.8631797.1991244140.1678071894-1349076290.1678071894 (2021).
 48. Roos, G. & Shroff, Z. What will happen to the jobs? Technology-enabled productivity improvement – good for some, bad for others. *Labour & Industry: a journal of the social and economic relations of work* **27**, 165–192 (2017).
 49. Autor, D. H. Why Are There Still So Many Jobs? The History and Future of Workplace Automation. *Journal of Economic Perspectives* **29**, 3–30 (2015).

50. Hope, A. *et al.* Digital skills in the Australian and International economies. *National Skills Commission Annual Report 2020-2021*.
51. Ahmed, M. & Haskell-Dowland, P. Google and Microsoft are bringing AI to Word, Excel, Gmail and more. It could boost productivity for us – and cybercriminals. *The Conversation* (2023).
52. Spataro, J. Introducing Microsoft 365 Copilot – your copilot for work. *Official Microsoft Blog* <https://blogs.microsoft.com/blog/2023/03/16/introducing-microsoft-365-copilot-your-copilot-for-work/> (2023) [Accessed 21 March 2023].
53. Kurain, T. New AI features and tools for Google Workspace, Cloud and developers. *The Keyword* <https://blog.google/technology/ai/ai-developers-google-cloud-workspace/> (2023) [Accessed 21 March 2023].
54. Mehdi, Y. Reinventing search with a new AI-powered Microsoft Bing and Edge, your copilot for the web. *Official Microsoft Blog* <https://blogs.microsoft.com/blog/2023/02/07/reinventing-search-with-a-new-ai-powered-microsoft-bing-and-edge-your-copilot-for-the-web/> (2023) [Accessed 21 March 2023].
55. Kung, T. H. *et al.* Performance of ChatGPT on USMLE: Potential for AI-assisted medical education using large language models. *PLOS Digital Health* **2**, e00000198 (2023).
56. § Song, B., Zhou, R. & Ahmed, F. Multi-modal Machine Learning in Engineering Design: A Review and Future Directions. (2023) doi:10.48550/arxiv.2302.10909.
57. Zhang, Y. *et al.* Applying Artificial Intelligence Methods for the Estimation of Disease Incidence: The Utility of Language Models. *Front Digit Health* **2**, (2020).
58. Zhang, Y. *et al.* Applying Artificial Intelligence Methods for the Estimation of Disease Incidence: The Utility of Language Models. *Front Digit Health* **2**, 31 (2020).
59. Shen, Y. *et al.* ChatGPT and Other Large Language Models Are Double-edged Swords. *Radiology* (2023) doi:10.1148/radiol.230163.
60. Coco, B. A., Henman, P. & Sleep, L. Mapping ADM in Australian social services. (2022) doi:10.25916/XXJ4-N968.
61. Yerushalmy, J. German publisher Axel Springer says journalists could be replaced by AI. *The Guardian* (2023).
62. Lee, H.-K. Rethinking creativity: creative industries, AI and everyday creativity. *Culture & Society* **44**, 601–612 (2022).
63. Loble, L. The rise of ChatGPT shows why we need a clearer approach to technology in schools. *The Conversation* (2023).
64. Elliott, A. *The Culture of AI: Everyday Life and the Digital Revolution - 1st Edit.* (Routledge, 2019).
65. Samuel, S. The case for slowing down AI. *Vox* (2023).
66. Thomas, J. *et al.* *Australian Digital Inclusion Index: 2021*. <https://www.digitalinclusionindex.org.au/download-reports/> (2021) doi:10.25916/phgw-b725.
67. Roller, S. *et al.* Recipes for building an open-domain chatbot. *EACL 2021 - 16th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference* 300–325 (2020) doi:10.48550/arxiv.2004.13637.
68. Oviedo-Trespalacios, O. *et al.* The Risks of Using ChatGPT to Obtain Common Safety-Related Information and Advice. *SSRN Electronic Journal* (2023) doi:10.2139/SSRN.4346827.
69. Hanff, A. ChatGPT should be considered a malevolent AI and destroyed. *The Register* (2023).
70. DeCamp, M. & Lindvall, C. Latent bias and the implementation of artificial intelligence in medicine. *Journal of the American Medical Informatics Association* **27**, 2020–2023 (2020).
71. Nadeem, M., Bethke, A. & Reddy, S. StereoSet: Measuring stereotypical bias in pretrained language models. *ACL-IJCNLP 2021 - 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Proceedings of the Conference* 5356–5371 (2021) doi:10.18653/V1/2021.ACL-LONG.416.
72. Koenecke, A. *et al.* Racial disparities in automated speech recognition. *Proc Natl Acad Sci U S A* **117**, 7684–7689 (2020).
73. Merone, L., Tsey, K., Russell, D. & Nagle, C. Sex Inequalities in Medical Research: A Systematic Scoping Review of the Literature. *Women's Health Reports* **3**, 49–59 (2022).

74. Norori, N., Hu, Q., Aellen, F. M., Faraci, F. D. & Tzovara, A. Addressing bias in big data and AI for health care: A call for open science. *Patterns* **2**, 100347 (2021).
75. Pitfalls of Predictive Policing: An Ethical Analysis - Viterbi Conversations in Ethics. <https://vce.usc.edu/volume-5-issue-3/pitfalls-of-predictive-policing-an-ethical-analysis/> [Accessed 28 February 2023].
76. Spaniol, M. J. & Rowland, N. J. AI-assisted scenario generation for strategic planning. *Futures & Foresight Science* e148 (2023) doi:10.1002/FFO2.148.
77. Berk, R., Heidari, H., Jabbari, S., Kearns, M. & Roth, A. Fairness in Criminal Justice Risk Assessments: The State of the Art. *Sociol Methods Res* **50**, 3–44 (2021).
78. Kreps, S., McCain, R. M. & Brundage, M. All the News That's Fit to Fabricate: AI-Generated Text as a Tool of Media Misinformation. *Journal of Experimental Political Science* **9**, 104–117 (2022).
79. Bommasani, R. *et al.* On the Opportunities and Risks of Foundation Models. (2021) doi:10.48550/arxiv.2108.07258.
80. Rainey, C. ChatGPT jailbreak DAN makes AI break its own rules. *Fast Company* <https://www.fastcompany.com/90845689/chatgpt-dan-jailbreak-violence-reddit-rules> (2023) [Accessed 21 March 2023].
81. Raso, F., Hilligoss, H., Krishnamurthy, V., Bavitz, C. & Kim, L. *Artificial Intelligence & Human Rights*. <https://cyber.harvard.edu/publication/2018/artificial-intelligence-human-rights> (2018).
82. Office of the Australian Information Commissioner. *Commissioner initiated investigation into Clearview AI, Inc. (Privacy)* [2021] Alcmr 54 (14 October 2021). (2021).
83. Franceschelli, G. & Musolesi, M. Copyright in generative deep learning. *Data Policy* **4**, e17 (2022).
84. Vaughan-Nichols, S. J. GitHub's Copilot faces first open source copyright lawsuit. *The Register* https://www.theregister.com/2022/11/11/githubs_copilot_opinion/ (2022) [Accessed 1 March 2023].
85. Vincent, J. The lawsuit against Microsoft, GitHub and OpenAI that could change the rules of AI copyright. *The Verge* <https://www.theverge.com/2022/11/8/23446821/microsoft-openai-github-copilot-class-action-lawsuit-ai-copyright-violation-training-data> (2022) [Accessed 6 March 2023].
86. Murdoch, B. Privacy and artificial intelligence: challenges for protecting health information in a new era. *BMC Med Ethics* **22**, 1–5 (2021).
87. Brown, H., Lee, K., Miresghallah, F., Shokri, R. & Tramèr, F. What Does it Mean for a Language Model to Preserve Privacy? in *2022 ACM Conference on Fairness, Accountability, and Transparency* 2280–2292 (ACM, 2022). doi:10.1145/3531146.3534642.
88. Pan, X., Zhang, M., Ji, S. & Yang, M. Privacy Risks of General-Purpose Language Models. in *IEEE Symposium on Security and Privacy* 1314–1331 (2020).
89. Brown, H., Lee, K., Miresghallah, F., Shokri, R. & Tramèr, F. What Does it Mean for a Language Model to Preserve Privacy? in *2022 ACM Conference on Fairness, Accountability, and Transparency* 2280–2292 (ACM, 2022). doi:10.1145/3531146.3534642.
90. Kingston AI Group. Statement by the Kingston AI Group. <https://kingstonaigroup.org.au/news-and-publications/f/statement-by-the-kingston-ai-group> (2023) [Accessed 8 March 2023].
91. Ghahramani, Z. *Independent Review of The Future of Compute: Final report and recommendations*. <https://www.gov.uk/government/publications/future-of-compute-review/the-future-of-compute-report-of-the-review-of-independent-panel-of-experts#chap1> (2023).
92. Mytton, D. Data centre water consumption. *npj Clean Water* **2021** *4*:1 4, 1–6 (2021).
93. Ludvigsen, K. G. A. The Carbon Footprint of ChatGPT. *Towards Data Science* (2022).
94. § Patterson, D. *et al.* Carbon Emissions and Large Neural Network Training. (2021).
95. Rohde, F., Gossen, M., Wagner, J. & Santarius, T. Sustainability challenges of Artificial Intelligence and Policy Implications. *Ökologisches Wirtschaften - Fachzeitschrift* **36**, 36–40 (2021).
96. Shen, K., Tong, X., Wu, T. & Zhang, F. *The next frontier for AI in China* . *QuantumBlack AI by McKinsey* <https://www.mckinsey.com/capabilities/quantumblack/our-insights/the-next-frontier-for-ai-in-china-could-add-600-billion-to-its-economy> (2022).

97. Secretary of State for Digital, C. National AI Strategy . <https://www.gov.uk/government/publications/national-ai-strategy> (2021) [Accessed 8 March 2023].
98. Dawson, G. S., Desouza, K. & Denford, J. S. Understanding artificial intelligence spending by the U.S. federal government. *Brookings* <https://www.brookings.edu/blog/techtank/2022/09/22/understanding-artificial-intelligence-spending-by-the-u-s-federal-government/> (2022) [Accessed 8 March 2023].
99. Boudier, J., Schmid, P. & Simon, J. Hugging Face and AWS partner to make AI more accessible. *Hugging Face* <https://huggingface.co/blog/aws-partnership> (2023) [Accessed 21 March 2023].
100. Mazzucato, M., Schaake, M., Krier, S. & Entsminger, J. *Governing artificial intelligence in the public interest.* <https://www.ucl.ac.uk/bartlett/public-purpose/publications/working-papers> (2022).
101. Meng, B. “This is China’s Sputnik Moment”: The Politics and Poetics of Artificial Intelligence. <https://doi.org/10.1080/1369801X.2021.2003227> (2021) doi:10.1080/1369801X.2021.2003227.
102. Zeng, J. China’s Artificial Intelligence Innovation: A Top-Down National Command Approach? *Glob Policy* **12**, 399–409 (2021).
103. Webster, G., Creemers, R., Kania, E. & Triolo, P. Full Translation: China’s ‘New Generation Artificial Intelligence Development Plan’ (2017). *Digichina* <https://digichina.stanford.edu/work/full-translation-chinas-new-generation-artificial-intelligence-development-plan-2017/> (2017) [Accessed 6 March 2023].
104. Bin, R. & Dixon, L. A principled governance for emerging AI regimes: lessons from China, the European Union, and the United States. *AI and Ethics* **2022** **1**, 1–18 (2022).
105. Davidson, H. ‘Political propaganda’: China clamps down on access to ChatGPT . *The Guardian* <https://www.theguardian.com/technology/2023/feb/23/china-chatgpt-clamp-down-propaganda> (2023) [Accessed 6 March 2023].
106. Shujuan, L. First domestic chatbot MOSS to be made open source. *Chinadaily.com.cn* <https://global.chinadaily.com.cn/a/202302/28/WS63fd392fa31057c47ebb1211.html> (2023) [Accessed 6 March 2023].
107. Marchant, G. E., Tournas, L. & Gutierrez, C. I. Governing Emerging Technologies Through Soft Law: Lessons for Artificial Intelligence. *Jurimetrics* **61**, (2020).
108. Gutierrez, C. I. & Marchant, G. E. A Global Perspective of Soft Law Programs for the Governance of Artificial Intelligence. *SSRN Electronic Journal* (2021) doi:10.2139/SSRN.3855171.
109. Pouget, H. The EU’s AI Act Is Barreling Toward AI Standards That Do Not Exist - Lawfare. *Lawfare* <https://www.lawfareblog.com/eus-ai-act-barreling-toward-ai-standards-do-not-exist> (2023) [Accessed 2 March 2023].
110. European Commission. *AI Liability Directive.* (2022).
111. Council of the European Union. Artificial Intelligence Act: Council calls for promoting safe AI that respects fundamental rights. <https://www.consilium.europa.eu/en/press/press-releases/2022/12/06/artificial-intelligence-act-council-calls-for-promoting-safe-ai-that-respects-fundamental-rights/> (2022) [Accessed 20 March 2023].
112. Government of Canada. *The Artificial Intelligence and Data Act (AIDA) – Companion document.* <https://ised-isde.canada.ca/site/innovation-better-canada/en/artificial-intelligence-and-data-act-aida-companion-document> (2023).
113. Kolt, N. Algorithmic Black Swans. *Washington University Law Review* **101**, (2023).
114. Cassidy, C. Australian universities to return to ‘pen and paper’ exams after students caught using AI to write essays . *The Guardian* (2023).
115. Jaeger, C. ChatGPT tool ban introduced at Victorian schools. *The Age* (2023).
116. Hare, J. You can’t ban the bot, educators say as they struggle with ChatGPT. *Financial Review* (2023).
117. Davis, A. ChatGPT banned in WA public schools in time for start of school year. *ABC News* (2023).
118. IEEE SA. Autonomous and Intelligent Systems (AIS). *IEEE.org* <https://standards.ieee.org/initiatives/autonomous-intelligence-systems/> [Accessed 8 March 2023].

119. Gebru, T. *et al.* Datasheets for datasets. *Commun ACM* **64**, 86–92 (2021).
120. Mitchell, M. *et al.* Model cards for model reporting. *FAT* 2019 - Proceedings of the 2019 Conference on Fairness, Accountability, and Transparency* 220–229 (2019) doi:10.1145/3287560.3287596.
121. OpenAI. *Best Practices for Deploying Language Models*. <https://openai.com/blog/best-practices-for-deploying-language-models/> (2022).
122. Microsoft. *Foundations of assessing harm*. <https://docs.microsoft.com/en-us/azure/architecture/guide/responsible-innovation/harms-modeling> (2021).
123. Microsoft. *Threat Modeling AI/ML Systems and Dependencies*. <https://docs.microsoft.com/en-us/security/engineering/bug-bar-aiml> (2022).
124. Microsoft. *Responsible use of AI with Cognitive Services*. <https://docs.microsoft.com/en-us/azure/cognitive-services/responsible-use-of-ai-overview> (2022).
125. Schiffer, Z. & Newton, C. Microsoft lays off AI ethics and society team - The Verge. *Platformer* (2023).
126. International Organization for Standardization. ISO - ISO/IEC JTC 1/SC 42 - Artificial intelligence. <https://www.iso.org/committee/6794475.html> (2017) [Accessed 6 March 2023].
127. International Organization for Standardization. ISO - ISO/IEC 22989:2022 - Information technology — Artificial intelligence — Artificial intelligence concepts and terminology. <https://www.iso.org/standard/74296.html> (2022) [Accessed 6 March 2023].
128. ISO/IEC. ISO/IEC DIS 42001 - Information technology — Artificial intelligence — Management system. <https://www.iso.org/standard/81230.html> (2023) [Accessed 17 March 2023].
129. ISO/IEC. ISO/IEC 23894:2023 - Information technology — Artificial intelligence — Guidance on risk management. <https://www.iso.org/standard/77304.html> (2023) [Accessed 17 March 2023].
130. George, E. R. Corporate Social Responsibility and Social Media Corporations: Incorporating Human Rights Through Rankings, Self-Regulation and Shareholder Resolutions. *Duke Journal of Comparative & International Law, Forthcoming, University of Utah College of Law Research Paper No. 256* (2018) doi:10.2139/SSRN.3168880.
131. Ranking Digital Rights. The 2022 RDR Big Tech Scorecard. <https://rankingdigitalrights.org/bts22/> (2022) [Accessed 20 March 2023].
132. Helberger, N. & Diakopoulos, N. ChatGPT and the AI Act. *Internet Policy Review* **12**, (2023).
133. Kharpal, A. China's tech giants plan ChatGPT clones — and Beijing is watching closely. *CNBC* (2023).
134. Lau, L. J., Qian, Y. & Roland, G. Reform without losers: An interpretation of China's dual-track approach to transition. *Journal of Political Economy* **108**, 120–143 (2000).
135. China Legislation Standard. Artificial Intelligence Innovation and Development Pilot Zone. <http://www.cnstandards.net/index.php/artificial-intelligence-innovation-and-development-pilot-zone/> (2019) [Accessed 6 March 2023].
136. § Bello Villarino, J.-M., Gulson, K., Paul, R., Carmel, E. & Cobbe, J. Chapter X: Artificial intelligence, regulation and the purposes of education. https://www.researchgate.net/publication/368426216_Chapter_X_Artificial_intelligence_regulation_and_the_purposes_of_education_Later_version_forthcoming_in_Public_Policy_and_Artificial_Intelligence_Vantage_Points_for_Critical_Inquiry_Editors_Chapter_X_ (2023) [Accessed 3 March 2023].
137. Agence France-Presse. Google yanks gay dating app in Indonesia. *The Manila Times* (2018).
138. Reuters. Indonesia blocks Telegram messaging service over security concerns. *Reuters* (2017).
139. Reuters. Indonesia bans Chinese video app Tik Tok for ‘inappropriate content’. (2018).
140. Barrett, A. M., Hendrycks, D., Newman, J. & Nonnecke, B. Actionable Guidance for High-Consequence AI Risk Management: Towards Standards Addressing AI Catastrophic Risks. (2022).
141. PDPC. Launch of AI Verify - An AI Governance Testing Framework and Toolkit. <https://www.pdpc.gov.sg/news-and-events/announcements/2022/05/launch-of-ai-verify---an-ai-governance-testing-framework-and-toolkit> (2022) [Accessed 17 March 2023].
142. Yeong Zee Kin. Singapore's A.I.Verify builds trust through transparency - OECD.AI. *OECD.AI Policy Observatory* <https://oecd.ai/en/wonk/singapore-ai-verify> (2022) [Accessed 17 March 2023].

143. Hongladarom, S. The Thailand national AI ethics guideline: an analysis. *Journal of Information, Communication and Ethics in Society* **19**, 480–491 (2021).
144. Digital Thailand. *AI Ethics Guideline*. <https://www.etda.or.th/getattachment/9d370f25-f37a-4b7c-b661-48d2d730651d/Digital-Thailand-AI-Ethics-Principle-and-Guideline.pdf.aspx?lang=th-TH> (2018).
145. Joshua J. Why doesn't ChatGPT know about X? *OpenAI Help Center* <https://help.openai.com/en/articles/6827058-why-doesn-t-chatgpt-know-about-x> (2023) [Accessed 21 March 2023].
146. Foy, P. GPT-3 Fine Tuning: Key Concepts & Use Cases. *MLQ.ai* <https://www.mlq.ai/gpt-3-fine-tuning-key-concepts/> (2023) [Accessed 21 March 2023].
147. Goldfarb, A. & Trefler, D. Artificial Intelligence and International Trade. in *The Economics of Artificial Intelligence: An Agenda* vols 978-0-226-61347-5 463–492 (2019).
148. Hernandez, D. & King, R. Universities' AI Talent Poached by Tech Giants. *The Wall Street Journal* (2016).

Appendix 1: Contributing experts and peer reviewers

Lead Fellows / authors

Professor Genevieve Bell AO FTSE FAHA, Director School of Cybernetics, The Australian National University

Professor Jean Burgess FAHA, Associate Director, ARC Centre of Excellence for Automated Decision-Making and Society, Queensland University of Technology

Professor Julian Thomas FAHA, Director ARC Centre of Excellence for Automated Decision-Making and Society, RMIT University

Professor Shazia Sadiq FTSE, Director ARC Centre for Information Resilience, University of Queensland

Expert contributors

Dr Jose-Miguel Bello Y Villarino, ARC Centre of Excellence for Automated Decision-Making and Society, The University of Sydney

Dr Shakes Chandra, UQ AI Collaboratory, The University of Queensland

Dr Tong Chen, UQ AI Collaboratory, The University of Queensland

Dr Paul Dalby, Australian Institute for Machine Learning

Dr Richard Harvey, Australian Institute for Machine Learning

Professor Seth Lazar, College of Arts and Social Sciences, Australian National University

Dr Oliver Mayo FAA FTSE, Adjunct Professor, The University of Adelaide

Professor Michel Milford, Queensland University of Technology

Professor Saeid Nahavandi FTSE, Pro Vice Chancellor (Defence Technologies), Deakin University

Dr Dang Nguyen, ARC Centre of Excellence for Automated Decision Making & Society

Dr Ian Oppermann FTSE, CEO and Chief Data Scientist, NSW Data Analytics Centre

Professor Christine Parker FASSA, Melbourne Law School, The University of Melbourne

Professor Jason Potts, Blockchain Innovation Hub, RMIT University

Professor Nils Goran Arne Roos FTSE, Chair, Innovation Performance Australia Pty Ltd

Dr Aaron Snoswell, ARC Centre of Excellence for Automated Decision-Making and Society, Queensland University of Technology

Professor Nicolas Suzor, School of Law, Queensland University of Technology

James Taylor, School of Cybernetics, Australian National University

Professor Anton Van Den Hengel FTSE, Director of the Centre for Augmented Reasoning, Australian Institute for Machine Learning

Professor Kimberlee Weatherall, The University of Sydney

Distinguished Professor Mary-Anne Williams FTSE FACS, Michael J Crouch Chair for Innovation, The University of New South Wales

Associate Professor Hongzhi Yin, UQ AI Collaboratory, The University of Queensland

Professor Haiqing Yu, Professor of Media and Communication, RMIT University

Dr Junliang Yu, UQ AI Collaboratory, The University of Queensland

Associate Professor Guido Zuccon, UQ AI Collaboratory, The University of Queensland

[Peer reviewers](#)

Emeritus Professor Rod Brooks FTSE

Professor Nicole Gillespie, KPMG Chair in Organisational Trust and Professor of Management at the University of Queensland Business School

Professor Iven Mareels FTSE, Executive Dean of the Institute for Innovation, Science and Sustainability, Federation University

Dr Andrew McMullan, Chief Data and Analytics Officer, Commonwealth Bank of Australia

Professor Edward Santow, Director, Policy & Governance at the Human Technology Institute, and Industry Professor, Responsible Technology, University of Technology Sydney

Professor Toby Walsh FAA, Chief Scientist at UNSW.ai, the AI Institute, UNSW Sydney

Professor Karen Yeung, Interdisciplinary Professorial Fellow in Law, Ethics and Informatics, University of Birmingham

[Conflicts of interest declaration](#)

This briefing incorporates input from Australian experts directly involved in research in Australia. Many of these contributors and reviewers have worked directly on studies and reports cited in this briefing. Contributors and peer reviewers are drawn from a range of institutions, initiatives and fields, and collectively provide an independent and authoritative perspective on this topic.

[Acknowledgements](#)

The production of this rapid research report was supported by: Ryan Winn of the Australian Council of Learned Academies; Dr Kylie Brass and Inga Davis of the Australian Academy of the Humanities; Peter Derbyshire, Natasha Abrahams and Kylie Walker of the Australian Academy of Technology and Engineering; and Dr Hayley Teasdale, Lauren Sullivan, Chris Anderson, and Anna-Maria Arabia of the Australian Academy of Science. Edited by Robyn Diamond and Lydia Hales.

Appendix 2: Glossary

Algorithm: Automated instructions for a computer to perform a task or solve a problem.

Application program interfaces (APIs): A set of protocols to enable two software programs to communicate with one another, or for one program to run another program.

Architecture: The design or structure of an AI model.

Artificial intelligence (AI): A collection of interrelated technologies used for problem solving and to complete tasks that would otherwise require human intelligence.

AI model: A program that has been trained on a dataset to recognise patterns (typically using artificial neural networks) or reason diagnostically or predictively (as seen in probabilistic graphical models).

Artificial neural network: A type of machine learning consisting of a network of nodes that function analogously to the human brain.

Foundation models: Large AI-based models, trained on vast datasets, that can be applied to a variety of different tasks. Foundation models represent a paradigm shift in AI highlighting the phenomenal progression from algorithms (e.g. logistic regression), to architectures (e.g. transformers) to foundation models (e.g. GPT-3).

Generative AI: A type of AI model that can generate content such as text, images, audio and code, in response to user prompts.

Large language model (LLM): A type of generative AI that specialises in the generation of human-like text. May be used interchangeably with ‘language models’.

Machine learning (ML): The development of models that can autonomously ‘learn’ from datasets and from inputs continuously.

Machine learning model operationalisation management (MLOps): A discipline concerned with the development, deployment and governance of machine learning models.

Machine learning acceleration software: Software that makes the training process for models faster.

Multimodal foundation models (MFMs): A type of generative AI that can process information from multiple types of inputs (text, visual, auditory and tactile).

Natural language processing: An interdisciplinary branch of computer science, linguistics and artificial intelligence concerned with human–computer interaction and processing using human language.

Parallel processing: Using multiple computing processors concurrently, enabling the processing of larger amounts of data in a shorter amount of time.

Technology stack: The combination of technologies used to develop an application. May be used interchangeably with ‘tech stack’.

Transformer architecture: A neural network that has the feature of learning context and parallel processing, enabling more powerful and faster models. This is due to a number of underlying advances, including an encoder-decoder structure.

UX: User experience.

Vision models: A type of AI that can process visual information.

Appendix 3: Examples of regulatory actions in jurisdictions overseas

APPROACHES INVOLVING REGULATORY ACTION (LEGISLATION OR REGULATOR GUIDANCE)

Country / region	Strategy/approach	Generative AI specific implications	Implications for Australia
European Union	Proposed EU AI Act. ¹³² Risk management approach, categorising the applications of AI in each case to three risk categories. ^h Detailed risk management system applied to uses designated ‘high risk’. Once enforced, will create legal obligation for developers and users to undertake risk management and ongoing monitoring.	Current draft text proposed by the European Council states that insofar as a system remains general purpose (i.e. without concrete outputs within a sector or task, including GPT-3 or GPT-4, or ‘question-answer’ systems like ChatGPT) it cannot be assessed correctly and therefore is out of scope; applications built on LLM or MFM and deployed in high-risk use cases would require compliance with risk management. Negotiations are ongoing; ‘general purpose’ exclusion could change before the Act is finalised. Commentators argue that LLMs/MFMs may require a ‘general risk category’ with appropriate obligations attached, including a general monitoring obligation for systemic risks. ^{113,132}	EU model has potential to become an international standard as it will apply where EU citizens are impacted by systems even if developed overseas. Also provides further impetus to international standards development: compliance with technical standards will be a way to achieve legal compliance.
China	Government-led, public-private partnership in AI development and governance, as exemplified in AI Development Plan (2017), AI Safety Framework (2020), AI code of ethics (2021), and regulatory guidelines issued on specific technologies.	The Cyberspace Administration of China has issued algorithmic and ‘deep syntheses tech’ regulations since 2022, all applicable to ChatGPT-style technologies. ¹³³ China follows a ‘dual-track system’ to allow local experiments in policy implementation, ¹³⁴ The Guidelines for National New Generation Artificial Intelligence Innovation and Development Pilot Zone Construction Work (2019) is a legal framework for companies and provinces to work on concrete aspects of AI in parallel, ¹³⁵ including by: testing institutional mechanisms, policies, and regulations; promoting the in-depth integration of AI with economic and social development; and exploring new approaches to governance in the intelligent era. ¹³⁶	Reliance on guidelines and allowance for some local/provincial experimentation in governance illustrates a different approach from the EU, which is seeking to define regulatory standards applicable across the EU. China is highly engaged in developing standards for AI that have both similarities and differences from principles and approaches developed in the Global North; its approaches have potential for influence in the region.

^h The risk levels are, unacceptable uses of AI, which are explicitly prohibited; High risk systems, which will be managed via a strong form of co-regulation, consisting of government requiring industry self-regulation via incorporation of technical standards (which do not presently exist, but are under development within the European Standards organisation, CEN-CENELEC); and a subcategory of uses subjected to transparency obligations (biometric categorisation, emotion recognition, deepfakes), which have an additional requirement for users to know they are interacting with a system.

Country / region	Strategy/approach	Generative AI specific implications	Implications for Australia
Indonesia	Applies a risk-based licensing system for all electronic service providers (Vietnam News Agency 2023). The focus of risk assessment is less on technical systems, rather on the context of use and governance. Notably, Indonesia has a history of banning apps and platforms on legal and/or moral grounds. ¹³⁷⁻¹³⁹	The risks of generative AI may be assessed according to the nature of the content it generates at the point of licensing. Given generative AI's ability to develop unique output based on human requests, assessment is likely contingent on the perceived trade-off between economic impact of new technologies and the protection of contextually grounded moral values.	Australia should take into consideration the diverse cultural and socio-political landscape of the Asia-Pacific and consider taking a leadership role in communicating policy developments across the region.

SELF-REGULATION, VOLUNTARY, TECHNICAL STANDARDS

Country / region	Strategy/approach	Generative AI specific implications	Implications for Australia
United States	The US Government relies on self-regulation via voluntary multi-stakeholder processes for the development of risk management and technical standards. See National Institute of Standards and Technology (NIST) AI Risk Management Framework (AI RMF), v1 (January 2023). The US is also actively involved in international standards bodies.	NIST believes that its AI RMF is suitable for larger models and generative AI, treating generative systems as a category within the framework. NIST could in the future develop a specific AI RMF Profile for general purpose systems like LLMs and MFMs. ¹⁴⁰	The NIST AI RMF is freely and publicly available, unlike many international standards (which can be expensive to purchase), meaning it may provide useful guidance for Australian governments and firms.
Singapore	Development of standardised self-testing tools (AI Verify) under purview of the Personal Data Protection Commission (PDPC). ^{141,142} Aimed at enabling businesses to check the implementation of AI models against a set of principles. Singapore is also contributing to international standards development.	The framework is aimed at AI generally, rather than generative AI. The risks of generative AI might be similarly assessed by voluntary self-testing, although with qualifications.	Australia could consider encouraging businesses to conduct self-verifications of risk mitigation measures. This approach complements, rather than replaces, ethical standards in AI implementation.
Thailand	National AI Ethics Guideline provides basis for procurement-based risk management, providing principles and expectations for different actors (regulators, developers, manufacturers, end users). ^{143,144}	The framework is aimed at AI generally, rather than generative AI. The risks of generative AI may be assessed at the point of procurement drawing on these principles. <i>Ex ante</i> risk management operates in addition to existing laws that can regulate or deter AI misuse or harm.	Procurement-based risk assessment is an approach already employed by the jurisdiction.

Sequence-level Large Language Model Training with Contrastive Preference Optimization

Zhili Feng

Carnegie Mellon University*
zhilif@andrew.cmu.edu

Dhananjay Ram

Amazon

Cole Hawkins

Amazon

Aditya Rawal

Amazon

Jinman Zhao

Amazon

Sheng Zha

Amazon

Abstract

The next token prediction loss is the dominant self-supervised training objective for large language models and has achieved promising results in a variety of downstream tasks. However, upon closer investigation of this objective, we find that it lacks an understanding of sequence-level signals, leading to a mismatch between training and inference processes. To bridge this gap, we introduce a contrastive preference optimization (CPO) procedure that can inject sequence-level information into the language model at any training stage without expensive human labeled data. Our experiments show that the proposed objective surpasses the next token prediction in terms of win rate in the instruction-following and text generation tasks.

1 Introduction

Next token prediction is now the predominant way for pre-training and supervised fine-tuning (SFT) of large language models (LLM). This loss function can be easily scaled up to train models with trillions of parameters and tokens, and it has demonstrated the ability to generate coherent and contextually relevant text. Let P be the unknown target language distribution and let Q be the distribution of our model at hand. The goal of next token prediction is to minimize the *forward-KL* divergence between P and Q . This training process only supervises the prediction of one token at a time, given the full context of the ground truth. On the other hand, during inference, the model needs to generate a whole sequence (for a given prompt) relying on its own prior predictions. This mismatch between the training and inference stage is known as *exposure-bias* in the literature of RNN and sequence-to-sequence model (Bengio et al., 2015; Ranzato et al., 2015).

In other words, next token prediction injects only *token-level* information into the model, but missing *sequence-level* signal. The latter requires a

generation of a longer horizon, which often relies on reinforcement learning algorithms; for example, reinforcement learning with human feedback (RLHF) (Ouyang et al., 2022); and is computationally expensive. In this work, we ask the following question: *Can we introduce sequence-level information in LLM pre-training / SFT with a small computational cost?*

We answer the question affirmatively with our proposed **CONTRASTIVE PREFERENCE OPTIMIZATION** (CPO) method. The goal of CPO is to improve **generation quality**. Unlike RLHF, the proposed CPO method does not require human preference information as the training signal. While we demonstrate CPO in the SFT case, the loss can be seamlessly applied to the late stage of pretraining as well.

2 Related work

LLMs trained with next token prediction loss (Radford et al., 2019; Chung et al., 2022; Sanh et al., 2021; Zhou et al., 2023) have demonstrated many fascinating capabilities, including the ability to perform zero-shot or few-shot tasks (Radford et al., 2019; Brown et al., 2020) and the ability to reason (Wei et al., 2022).

Several works have investigated the shortcomings of MLE and exposure bias. Arora et al. (2022) measured the accumulation of errors in language generation due to exposure bias. Schmidt (2019) connected exposure bias to generalization. Wang and Sennrich (2020) studied how exposure bias leads to hallucination in neural machine translation. To mitigate exposure bias, there exists a long line of work that has explored sequence-level training methods. Bengio et al. (2015); Ranzato et al. (2015) proposed to train RNN with RL or RL-related algorithms rather than teacher-forcing. BRIO Liu et al. (2022) targeted the summarization task with the ROUGE signal. Pang and He (2020) trained the lan-

*Work done as an intern at Amazon.

guage models with an offline RL algorithm. There also exists a line of works that generate samples during training and mix the samples with ground truth data (Shen et al., 2015; Zhang et al., 2019; Duckworth et al., 2019).

Recently, RLHF (Stiennon et al., 2020; Ouyang et al., 2022) and its supervised version DPO (Rafailov et al., 2023) were developed for alignment. They are effectively sequence-level training techniques. These algorithms require a pair of preferred and rejected samples, which are usually gathered by human labeling. The RL approach to language modeling is also closely related to energy-based models (EBM) (Korbak et al., 2022; Deng et al., 2020). This EBM form has also been studied in controlled text generation Kumar et al. (2022). Pace et al. (2024) also consider synthetic data generation, but their purpose is to improve reward modeling in RLHF rather than sequence-level training.

3 Proposed approach

Consider a sentence of T tokens $\mathbf{x} = \{\mathbf{x}_1, \dots, \mathbf{x}_T\} \in \mathcal{X}$, we use $\pi(\mathbf{x})$ to represent the distribution of \mathbf{x} under some language policy π . In particular, we write π_θ for a distribution that is parameterized by θ , where θ is usually the set of trainable parameters of the LLM; we write π_{ref} for a reference distribution that should be clear given the context. Inspired by DPO, we introduce our CPO objective:

$$\mathcal{L}_{\text{CPO}}(\pi_\theta, \pi_{\text{ref}}) = \mathbb{E}_{\substack{(\mathbf{x}, \mathbf{y}_1) \sim \mathcal{D} \\ \mathbf{y}_2, \dots, \mathbf{y}_K \sim \mathcal{A}}} \left[\log \frac{\exp \left(\beta \log \frac{\pi_\theta(\mathbf{y}_1|\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_1|\mathbf{x})} \right)}{\sum_{j=1}^K \exp \left(\beta \log \frac{\pi_\theta(\mathbf{y}_j|\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_j|\mathbf{x})} \right)} \right]. \quad (1)$$

Here $(\mathbf{x}, \mathbf{y}_1)$ is the ground truth prefix-continuation pair from the natural language distribution \mathcal{D} , and $\mathbf{y}_2, \dots, \mathbf{y}_K$ are $K - 1$ negative continuations sampled from a to-be-discussed distribution \mathcal{A} . The derivation is deferred to the appendix. If some ranking of the data quality is presented, i.e. $\tau : [K] \rightarrow [K]$ where $\tau(i) < \tau(j)$ means \mathbf{y}_i is preferred over \mathbf{y}_j , we also have the following CPO objective with ranking:

$$\mathcal{L}_{\text{CPO}}(\pi_\theta, \pi_{\text{ref}}) = \mathbb{E}_{\substack{\tau, (\mathbf{x}, \mathbf{y}_1) \sim \mathcal{D} \\ \mathbf{y}_2, \dots, \mathbf{y}_K \sim \mathcal{A}}} \left[\log \prod_{k=1}^K \frac{\exp \left(\beta \log \frac{\pi_\theta(\mathbf{y}_{\tau(k)}|\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_{\tau(k)}|\mathbf{x})} \right)}{\sum_{j=k}^K \exp \left(\beta \log \frac{\pi_\theta(\mathbf{y}_{\tau(j)}|\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_{\tau(j)}|\mathbf{x})} \right)} \right]. \quad (2)$$

Unlike RLHF or DPO, which require human preference data $\mathbf{y}_1 \geq \mathbf{y}_2 \geq \dots \geq \mathbf{y}_K$, CPO requires only ground truth data $(\mathbf{x}, \mathbf{y}_1) \sim \mathcal{D}$, and $K - 1$ synthetic negative samples $\mathbf{y}_2, \dots, \mathbf{y}_K \sim \mathcal{A}$.

Possibly, we can also get a ranking among the $K - 1$ synthetic samples in a fully automatic way. On a high level, CPO implicitly rewards the ground truth more than the synthetic negative samples.

We consider four ways to generate synthetic data. (1) **autoregressive negatives (AN)**: We use the language model to autoregressively generate the negative samples given a prefix. We fixed the synthetic data generation strategy to be top- k sampling with $k = 50$. (2) **batch negatives (BN)**: given a batch of prefixes and continuations $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^b$, the negative samples to the prefix \mathbf{x}_i are composed of $\{\mathbf{y}_j\}_{j \neq i}$. (3) **meanfield negatives (MN)**: given a sequence $\mathbf{y} = \{y_1, \dots, y_T\}$, we randomly select c percent of the positions $\{t_1, \dots, t_j\} \subseteq [T]$, and substitute each y_{t_i} independently based on $\pi_\theta(y_{t_i}|\mathbf{y}_1, \dots, y_{t_i})$, i.e. we independently resample $c\%$ of the tokens according to their original autoregressive distribution. (4) **truncation negatives (TN)**: for each ground truth continuation, we truncate them at a random position and append an extra EOS token at the end.

In our experiments, we observe that CPO can often benefit from a ranking among K samples, where the ranking is based on their cosine similarity to the ground truth. Let e_1, \dots, e_K be the embeddings of given sequences $\mathbf{y}_1, \dots, \mathbf{y}_K$ and without loss of generality assume that e_1 is the ground truth, we define $\tau(i) < \tau(j)$ if $\frac{\langle e_i, e_1 \rangle}{\|e_i\| \|e_1\|} > \frac{\langle e_j, e_1 \rangle}{\|e_j\| \|e_1\|}$, with the lower ranking index indicating the better sample. Using the objective eq. (2), this process gives us denser signals during training and can lead to better downstream performance.

4 Experimental Setup

Throughout this section, **BN**, **AN**, **MN**, **TN** represents batch negatives, autoregressive negatives, meanfield negatives, and truncation negatives respectively. **MixN** represents a mixed negative sampling strategy for which the details can be found in its context. We use **ANR** for models trained with autoregressive negatives and ranking signals, similarly we can denote **MixNR**, etc. We always randomly swap 15% tokens using **MN**. Although this choice is mainly heuristic, such a ratio appears quite frequently since BERT (Wettig et al., 2022).

Task and model. We consider two tasks in this paper. The first is an instruction-following task, trained and evaluated on the Dolly dataset (Conover et al., 2023). This dataset is composed of 15011 total instruction and response pairs. We train with

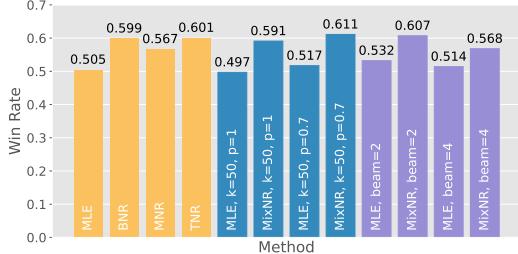


Figure 1: The effect of different generation configuration during inference and different negative sampling methods during training. Unless otherwise specified, greedy decoding is used. Win rate is evaluated by GPT-3.5 against the ground truth continuations.

Table 1: The win rate of GPT2-XL against the ground truth, samples generated by greedy decoding, evaluated by GPT-3.5.

	MLE	PSS	DPO	ANR	MixNR			
α	-	-	-	-	0	0.5	0.7	0.9
WinRate	0.471	0.086	0.383	0.506	0.476	0.479	0.487	0.485

7505 sequences and test with the rest 7506. We use pre-trained GPT2-XL (Radford et al., 2019) and OpenLlama-3B(Touvron et al., 2023; Geng and Liu, 2023) as the base model. The second is an open-ended text generation task on Wikidump data (Foundation). We train the OpenLlama-3B model to predict 85% tokens per sample given the leading 15% tokens.

Baselines. We consider three main baselines: MLE, DPO, and parallel scheduled sampling (**PSS**, (Duckworth et al., 2019)). Importantly, the difference between DPO and CPO lies in their negative samples. For DPO, we query GPT-3.5 to generate *unhelpful* response to the Dolly instructions. PSS is trained to sample 3 sequences for each training data, and each token is replaced with $p = 0.5$ (see Duckworth et al. (2019) for more details).

Training details. Throughout the experiment, we fix the learning rate to be $1e-5$, we use the AdamW optimizer with weight decay of 0.05. We keep the batch size to be 64. Unless otherwise specified, for the baseline model, we train GPT2-XL and OpenLlama-3B with the next token prediction loss for 2000 steps. Using these models as the reference model π_{ref} , we continue to train with the CPO objective either with or without ranking signals, with $\beta = 5$, for 1000 steps. For both models, each training data in a batch contains 11 negative samples in total. For MixN and MixNR, we also use a negative sample size of 11, consisting of 3 BN, 5 MN, and 3 TN. The MLE models used

for evaluation are continually trained for the same number of steps from the reference model, like the CPO models. All experiments are conducted on two AWS machines, each with 8 A100 GPUs.

Evaluation. As discussed in Goyal et al. (2022), almost all automated evaluation metrics have been shown to not align with human evaluations in the modern era of LLMs, so we decide to use GPT (Brown et al., 2020) as the evaluator. See the query template in the appendix. For efficiency, we generate and evaluate 1000 samples chosen from the 7506 test set. A similar template is used for Wiki text generation, see the detail in the appendix. During inference, we consider greedy decoding, top- p and top- k sampling, as well as beam search.

Weight-space ensemble. Previous works (Liu et al., 2022) have also suggested to combine the auxilliary loss function with the MLE training objective $\alpha \mathcal{L}_{\text{MLE}} + \mathcal{L}_{\text{CPO}}$, the downside of combining loss functions in this way is that for a different choice of α one will have to retrain the model. To investigate the importance of loss combination, we instead perform a weight-space ensemble (Wortsman et al., 2022). In particular, denote θ_{CPO} and θ_{MLE} the model parameters trained solely with CPO or MLE respectively, we generate with the interpolated weights $\theta = \alpha \theta_{\text{MLE}} + (1 - \alpha) \theta_{\text{CPO}}$.

5 Experimental Analysis

5.1 Instruction-Following Task

Our proposed CPO method with various negative sampling strategies consistently outperforms the MLE baseline models on the Dolly instruction-following task. Using greedy sampling with GPT2-XL, the CPO model has a clear margin over the MLE model, and CPO+ANR has a 3.5% higher win rate, see table 1. Note that CPO incurs very little computation overhead during the actual training: the overhead only comes a larger batch size, and even if we generate the negative samples autoregressively, it is a one-time offline cost.

The improvement in OpenLlama-3B is more significant: CPO+ANR has a 13.8% higher win rate than the MLE baseline, and CPO+MixNR has a 9.8% higher win rate in table 2. We also observe that weight-space ensemble has a positive impact on the model. Heuristically, for OpenLlama-3B, a smaller α is preferred (more emphasis on the CPO weights) (table 2), but the reverse holds for GPT2-XL (table 1). We hypothesize that the choice of α should depend on the model: if the model is more

Table 2: The win rate of OpenLlama-3B trained with CPO and MLE against the ground truth data in Dolly, sampled by greedy decoding, evaluated by GPT-3.5. MLE₁, ANR and AN are trained for 200 steps, the rest models are trained for 1000 steps. The best CPO model outperforms the MLE baseline by 13.8% win rate.

	MLE ₁	PSS	DPO	ANR	AN	MLE ₂	MixNR						MixN
α	-	-	-	-	-	-	0	0.1	0.3	0.5	0.7	0.9	-
WinRate	0.505	0.270	0.555	0.643	0.56	0.522	0.608	0.620	0.614	0.610	0.601	0.550	0.576

Table 3: OpenLlama-3B’s win rate against the ground truth continuation on Wikidump. The model is trained with either MLE or CPO+BNR. Weight ensemble is adopted. The best CPO model outperforms the MLE baseline by 3% win rate.

	MLE	BNR			
α	-	0	0.5	0.7	0.9
WinRate	0.508	0.455	0.505	0.5	0.538

capable, then it can benefit more from CPO. Here, we show the existence of a good α , and we leave further exploration to future research.

Comparison with DPO and PSS. The proposed CPO method performs better than other two baseline methods: DPO and PSS (see Table 2). We believe that DPO performs poorly because unhelpful/irrelevant continuations (even generated by ChatGPT) do not provide a very strong signal as human generated samples. Unlike in alignment, where the toxic/harmful samples provide a clear indication of what not to generate, here it is not clear what DPO can gain from merely a single irrelevant sample. On the other hand, CPO can benefit from larger negative sample size.

Sampling Strategy. In addition to greedy decoding, we also experiment with different choice of sampling strategies. In all settings, CPO has consistently demonstrated superior performance over MLE, see fig. 1.

Effect of different negative samples. We perform a study on the effects of different negative sampling strategies; the results are presented in fig. 1. We first train the OpenLlama-3B model with MLE loss for 1000 steps, then continue to train with CPO for 200 steps. For all ground truth sequences, we use 4 negative sequences. In this setting, we always use the ranking information to train CPO. We observe that the effects of BNR and TNR on the reward model preference are similar and that they perform slightly better than MNR.

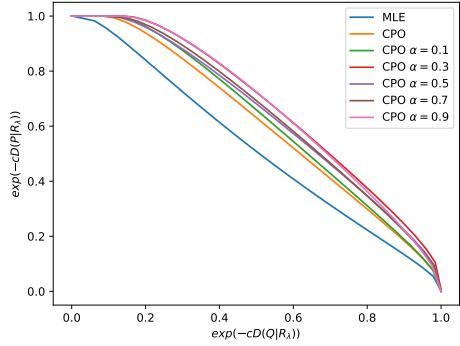


Figure 2: MAUVE score of MLE and CPO on Wiki data.

Table 4: MAUVE score of MLE and CPO on Wiki data.

	MLE	CPO BNR					
α	-	0	0.1	0.3	0.5	0.7	0.9
WinRate	0.524	0.610	0.627	0.673	0.645	0.651	0.668

5.2 Open-ended Text Generation Task

We further test OpenLlama-3B’s ability on an open-ended text generation task with CPO. Using Wikidump data ([Foundation](#)), for each test sample, we take its first 15% tokens as the prefix and train the model with CPO on the rest 85%. For negative sampling, we use four BNR examples. The results in table 3 show that CPO can improve the model’s win rate against the MLE baseline by 3%. We observe that increasing α improves the score, the opposite of the instruction-following task. It is likely because the negative samples here are too noisy, since only 15% prefixes are provided.

Additionally, we test the MAUVE score ([Pillutla et al., 2021](#)) of MLE and CPO compared to the ground truth. See the results in fig. 2 and table 4.

6 Conclusions and Limitations

In this paper, we propose an auxiliary CPO loss function for LLM training, which can be used with or without ranking signals, depending on the quality of the negative samples. We investigated several ways to generate negative samples. One limitation of this work is that the synthetic data are very noisy unless generated autoregressively; it is inter-

esting to explore other ways to efficiently generate high-quality negative data beyond the autoregressive fashion.

References

- Kushal Arora, Layla El Asri, Hareesh Bahuleyan, and Jackie Chi Kit Cheung. 2022. Why exposure bias matters: An imitation learning perspective of error accumulation in language generation. *arXiv preprint arXiv:2204.01171*.
- Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28.
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.
- Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.
- Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. Free dolly: Introducing the world’s first truly open instruction-tuned llm.
- Yuntian Deng, Anton Bakhtin, Myle Ott, Arthur Szlam, and Marc’Aurelio Ranzato. 2020. Residual energy-based models for text generation. *arXiv preprint arXiv:2004.11714*.
- Daniel Duckworth, Arvind Neelakantan, Ben Goodrich, Lukasz Kaiser, and Samy Bengio. 2019. Parallel scheduled sampling. *arXiv preprint arXiv:1906.04331*.
- Wikimedia Foundation. [Wikimedia downloads](#).
- Xinyang Geng and Hao Liu. 2023. [Openllama: An open reproduction of llama](#).
- Tanya Goyal, Junyi Jessy Li, and Greg Durrett. 2022. News summarization and evaluation in the era of gpt-3. *arXiv preprint arXiv:2209.12356*.
- Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 297–304. JMLR Workshop and Conference Proceedings.
- Tomasz Korbak, Ethan Perez, and Christopher L Buckley. 2022. R1 with kl penalties is better viewed as bayesian inference. *arXiv preprint arXiv:2205.11275*.
- Sachin Kumar, Biswajit Paria, and Yulia Tsvetkov. 2022. Gradient-based constrained sampling from language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2251–2277.
- Yixin Liu, Pengfei Liu, Dragomir Radev, and Graham Neubig. 2022. Brio: Bringing order to abstractive summarization. *arXiv preprint arXiv:2203.16804*.
- Zhuang Ma and Michael Collins. 2018. Noise contrastive estimation and negative sampling for conditional models: Consistency and statistical efficiency. *arXiv preprint arXiv:1809.01812*.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.
- Alizée Pace, Jonathan Mallinson, Eric Malmi, Sebastian Krause, and Aliaksei Severyn. 2024. West-of-n: Synthetic preference generation for improved reward modeling. *arXiv preprint arXiv:2401.12086*.
- Richard Yuanzhe Pang and He He. 2020. Text generation by learning from demonstrations. *arXiv preprint arXiv:2009.07839*.
- Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. 2021. Mauve: Measuring the gap between neural text and human text using divergence frontiers. *Advances in Neural Information Processing Systems*, 34:4816–4828.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*.
- Marc’Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*.
- Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.
- Florian Schmidt. 2019. Generalization in generation: A closer look at exposure bias. *arXiv preprint arXiv:1910.00292*.

Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2015. Minimum risk training for neural machine translation. *arXiv preprint arXiv:1512.02433*.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Chaojun Wang and Rico Sennrich. 2020. On exposure bias, hallucination and domain shift in neural machine translation. *arXiv preprint arXiv:2005.03642*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Alexander Wettig, Tianyu Gao, Zexuan Zhong, and Danqi Chen. 2022. Should you mask 15% in masked language modeling? *arXiv preprint arXiv:2202.08005*.

Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, et al. 2022. Robust fine-tuning of zero-shot models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7959–7971.

Wen Zhang, Yang Feng, Fandong Meng, Di You, and Qun Liu. 2019. Bridging the gap between training and inference for neural machine translation. *arXiv preprint arXiv:1906.02448*.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. 2023. Lima: Less is more for alignment. *arXiv preprint arXiv:2305.11206*.

A Appendix

A.1 A brief introduction to DPO, RLHF, and EBM

The equivalence of RLHF and EBM For the completeness of this paper, we include the result of the equivalence between RLHF and EBM. For the full proofs, we refer the reader to (Rafailov et al., 2023; Korbak et al., 2022).

The RLHF objective is the following:

$$\begin{aligned} \max_{\pi_\theta} & \mathbb{E}_{\mathbf{x} \sim \mathcal{D}, \mathbf{y} \sim \pi_\theta(\mathbf{y}|\mathbf{x})} [r(\mathbf{x}, \mathbf{y})] \\ & - \beta D_{KL}(\pi_\theta(\mathbf{y}|\mathbf{x}) || \pi_{\text{ref}}(\mathbf{y}|\mathbf{x})), \end{aligned} \quad (3)$$

where $\mathbf{x} \sim \mathcal{D}$ is a given prefix, $\mathbf{y} \sim \pi_\theta(\mathbf{y}|\mathbf{x})$ is a sampled continuation from the trainable model π_θ , and $r(\mathbf{x}, \mathbf{y}) \in \mathbb{R}$ is the reward. Meanwhile, we want to control the divergence between π_θ and π_{ref} , where π_{ref} is usually an already pretrained or finetuned LLM. The RLHF optimum is achieved at the following EBM:

$$\pi^*(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \pi_{\text{ref}}(\mathbf{y}|\mathbf{x}) \exp\left(\frac{1}{\beta} r(\mathbf{x}, \mathbf{y})\right), \quad (4)$$

where $Z(\mathbf{x}) = \sum_{\mathbf{y}} \pi_{\text{ref}}(\mathbf{y}|\mathbf{x}) \exp\left(\frac{1}{\beta} r(\mathbf{x}, \mathbf{y})\right)$ is the partition function.

Rafailov et al. (2023) assume that the preference over two sequences \mathbf{y}_w and \mathbf{y}_l given \mathbf{x} is parameterized by the Bradley-Terry model:

$$P(\mathbf{y}_w \succ \mathbf{y}_l | \mathbf{x}) = \frac{e^{r(\mathbf{x}, \mathbf{y}_w)}}{e^{r(\mathbf{x}, \mathbf{y}_l)} + e^{r(\mathbf{x}, \mathbf{y}_w)}}.$$

Under the Bradley-Terry model, DPO establishes the equivalence between the original RLHF objective eq. (3) and the following supervised objective:

$$\begin{aligned} \mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = & \mathbb{E}_{(\mathbf{x}, \mathbf{y}_w, \mathbf{y}_l) \sim \mathcal{D}} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(\mathbf{y}_w | \mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_w | \mathbf{x})} \right) \right. \\ & \left. - \beta \log \frac{\pi_\theta(\mathbf{y}_l | \mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_l | \mathbf{x})} \right], \end{aligned} \quad (5)$$

where $\sigma(\cdot)$ is the Sigmoid function.

They also generalize the formulation to the Plackett-Luce model, where we have a linear ordering $\tau(\cdot)$ among K sequences:

$$\begin{aligned} \mathcal{L}_{\text{DPO}}(\pi_\theta, \pi_{\text{ref}}) = & \mathbb{E}_{\substack{\tau, \mathbf{x} \sim \mathcal{D} \\ \mathbf{y}_1, \dots, \mathbf{y}_K}} \left[\log \prod_{k=1}^K \frac{\exp \left(\beta \log \frac{\pi_\theta(\mathbf{y}_{\tau(k)} | \mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_{\tau(k)} | \mathbf{x})} \right)}{\sum_{j=k}^K \exp \left(\beta \log \frac{\pi_\theta(\mathbf{y}_{\tau(j)} | \mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_{\tau(j)} | \mathbf{x})} \right)} \right]. \end{aligned} \quad (6)$$

Here, $\tau(1), \dots, \tau(K)$ induce a ranking among K sequences.

A.2 Derivation of the CPO objective function

Here we give a full derivation of the CPO objective function in eq. (1).

Let $\mathbf{y}_1, \dots, \mathbf{y}_K$ be K continuations of a given prefix \mathbf{x} . Without loss of generality, let \mathbf{y}_1 be the best candidate. We are interested in the MLE of the event $P(\mathbf{y}_1 \text{ is the best among } K \text{ candidates} | \mathbf{x})$.

We start from the sequence-level (RLHF) objective, notice that here $r(\cdot)$ is a reward over language quality, not human preference.

$$\begin{aligned} & \max_{\pi_\theta} \mathbb{E}_{\mathbf{x} \sim \mathcal{D}, \mathbf{y} \sim \pi_\theta(\mathbf{y} | \mathbf{x})} [r(\mathbf{x}, \mathbf{y})] \\ & - \beta D_{\text{KL}}(\pi_\theta(\mathbf{y} | \mathbf{x}) || \pi_{\text{ref}}(\mathbf{y} | \mathbf{x})), \end{aligned} \quad (7)$$

Its optimum is achieved at the following EBM:

$$\pi^*(\mathbf{y} | \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \pi_{\text{ref}}(\mathbf{y} | \mathbf{x}) \exp\left(\frac{1}{\beta} r(\mathbf{x}, \mathbf{y})\right), \quad (8)$$

where $Z(\mathbf{x}) = \sum_{\mathbf{y}} \pi_{\text{ref}}(\mathbf{y} | \mathbf{x}) \exp\left(\frac{1}{\beta} r(\mathbf{x}, \mathbf{y})\right)$ is the partition function. See the proof in (Rafailov et al., 2023; Korbak et al., 2022).

Now we consider the natural extension of the Bradley-Terry model to K candidates:

$$\begin{aligned} & P(\mathbf{y}_1 \text{ is the best among } K \text{ candidates} | \mathbf{x}) \\ &= \frac{\exp(r^*(\mathbf{x}, \mathbf{y}_1))}{\sum_{k \in [K]} \exp(r^*(\mathbf{x}, \mathbf{y}_k))}. \end{aligned} \quad (9)$$

Now assuming we have the optimal policy π^* , we can reparameterize r by rearranging eq. (8):

$$r^*(\mathbf{x}, \mathbf{y}) = \beta \log \frac{\pi^*(\mathbf{y} | \mathbf{x})}{\pi_{\text{ref}}(\mathbf{y} | \mathbf{x})} + \beta \log Z(\mathbf{x}). \quad (10)$$

Plugging eq. (10) into eq. (9), we get eq. (1).

A.3 Query template of Dolly and Wiki text generation

The query template for the Dolly instruction-following is the following: “For the following query to a chatbot, which response is more helpful?\n Query: {}\\n Response A: {}\\n Response B: {}\\n State only “A” or “B” to indicate which response is more helpful.\n More helpful:”

The template for Wiki is the following: “For the following prefix, which continuation is better?\n Prefix: {}\\n Continuation A: {}\\n Continuation B: {}\\n State only “A” or “B” to indicate which continuation is more helpful.\n Better:”

A.4 DPO generation template and example

When generating unhelpful responses for DPO, we query GPT with the following template: Given the ground truth instruction and response, can you generate a not helpful response?\n Instruction: {}\\nResponse: {}\\nNot helpful response::

One example of the generated response is the following: **instruction:** When did Virgin Australia start operating? **chosen response:** Virgin Australia commenced services on 31 August 2000 as Virgin Blue, with two aircraft on a single route. **rejected response:** Virgin Australia definitely exists and has airplanes that fly to different places.

A.5 Connection to noise contrastive estimation

Noise contrastive estimation (NCE) (Gutmann and Hyvärinen, 2010) is a novel estimation technique introduced to tackle the computational infeasibility of traditional likelihood-based methods in large-scale machine learning models, particularly those involving high-dimensional data. NCE diverges from typical maximum likelihood estimation by transforming the problem into a classification task, which is deeply connected to both DPO and CPO. In NCE, the model is trained to distinguish between real data and noise/synthetic data. Beyond binary classification, RankingNCE¹ also trains the model to rank the real data higher than all noise samples (Ma and Collins, 2018).

There are two important distinctions between CPO and NCE. First, instead of training the model to distinguish between real data and noise (at which any reasonable language model should already be good), we train the model to distinguish *better than a reference model does*, hence making the model better at recognizing natural text. Second, we also introduce a denser ranking signal by incorporating the similarity among embeddings of different samples. The experiments in this paper demonstrate that such a dense training signal consistently improves text generation quality.

¹Despite the name, it means the model is ranking the real data highest among all data, rather than learning a total ordering.

Reimagining Customer Service Journeys with LLMs: A Framework for Chatbot Design and Workflow Integration

Lennart Meincke and Christian Terwiesch¹

January 23, 2025

Mack Institute for Innovation Management, The Wharton School, University of Pennsylvania

Introduction

Whether assisting travelers with inquiries about their upcoming flight or destination, providing mental health support, helping students with homework assignments, or aiding in financial planning, chatbots powered by generative artificial intelligence (GenAI) in the form of large language models (LLMs) can help deliver efficient, scalable, and personalized customer support. Building an effective chatbot involves navigating a series of design decisions that influence user experience, technical implementations, and various legal and ethical issues. This white paper outlines a general framework for the functionality of chatbots and their operational use. Our aim is to help executives navigate the complex decisions that need to be made when leveraging the power of LLMs to support their customers, be they travelers, patients, students, or investors.

In the first part of this paper, we articulate five design choices that need to be made when building a chatbot. These choices provide the answers to five important managerial questions:

1. Focused vs. Broad Knowledge Base (“How much does the chatbot know?”)
2. Isolated Interactions vs. Long-time Relationships (“Does the chatbot remember users over multiple episodes?”)
3. Proactive vs. Responsive (“Does the user reach out to the chatbot or the chatbot to the user?”)
4. Static vs. Dynamic (“Does the chatbot learn about a user or a user population over time?”)
5. Quality Assurance (“How is the quality of the chatbot’s output assured?”)

These five dimensions create a taxonomy of chatbots.

In the second part of the paper, we discuss different modes of interactions between the chatbot and the human operators. Indeed, a successful use of chatbots is more than just a matter of designing and building a new technology. Rather, the chatbot must be integrated into the workflow of the organization. At the

¹ Meincke, Terwiesch: The Wharton School, 500 Huntsman Hall, 3730 Walnut Street, Philadelphia, PA 19104,
lennart@sas.upenn.edu, terwiesch@wharton.upenn.edu

30,000-foot level, such modes of interactions are broadly referred to as “humans in the loop” processes. But, as we will explain, there are many ways to design a workflow in which a human operator collaborates with a chatbot to support a customer.

Specifically, we distinguish between the following six workflow configurations, all of which have a human in the loop: configurations 1 and 2 are often referred to as AI-enabled auditing, configurations 3 and 4 are also known as AI-assisted workflow, and configurations 5 and 6 correspond to fully automated workflows.

1. Human operator response with offline auditing by chatbot
2. Human operator response with real time auditing by chatbot
3. Chatbot recommendations with human operator deciding and responding
4. Chatbot preparation with human operator responding
5. Chatbot response with real time auditing by human operator
6. Chatbot response with offline auditing by human operator

The decisions along the five chatbot design dimensions and the six workflow configurations provide executives with a blueprint to plan their efforts for reimaging customer support. Many other decisions, including privacy and security considerations as well as system integration, will also need to be made but are beyond the scope of this paper

Part I: The Five Dimensions of Chatbot Design

Dimension 1: Focused vs. Broad Knowledge Base

How much does the bot know?

A chatbot with a broad knowledge base can be created using frontier models such as OpenAI’s GPT-4 (OpenAI et al. 2024) or Google’s Gemini (Gemini Team et al. 2024). Thanks to extensive training data, these chatbots can answer a wide range of questions without any form of special training, and more recently, can ingest text, documents, audio and video.

As powerful as these frontier models might be, they do lack depth in specialized areas, especially when it comes to topics that highly depend on specific terms and operating policies. A high-profile example is Air Canada’s chatbot, which was reported to have offered discounted airfares to a customer requiring an urgent flight due to a death in the family. Though many airlines offer such bereavement tickets, Air Canada at that time did not (Melnick 2024). The fact that the Air Canada chatbot was trained on the broad body of knowledge typical for a frontier model (including the policies of many other airlines) as opposed to being focused exclusively on the Air Canada policies led to a frustrated customer and bad publicity.

Technical considerations. From a technical perspective, using an existing model like GPT-4, which has been trained on extensive public data, is the simplest approach to start using LLMs. Depending on the nature of the task, some hyperparameter tuning, such as picking a lower (more deterministic) or higher (more creative) temperature value might be helpful. For instance, when computing the dosage of a specific

medication, a more deterministic approach is likely to be preferred. To guide chatbot behavior, the most useful tool is the “system prompt,” which specifies the rules and persona of the bot and what it can and cannot do. A teaching assistant bot, for example, should perhaps only guide a student through a problem, but not reveal solutions too quickly. Customizing a bot’s abilities via the system prompt is straightforward and can be done entirely using free-text instructions. While results are often impressive, extensive testing is necessary to evaluate instruction and adherence and whether the model performs well enough without any special knowledge.

If more specialized knowledge is needed, implementing a retrieval augmented generation (RAG) approach can help integrate specific data sources, like product information or operating policies, into the bot’s knowledge base.

RAG works by taking a set of documents, breaking them down into smaller pieces (chunks), and storing them in a database. When processing a user question, the query is compared to all the knowledge pieces to determine the most relevant ones to help answer the question. When searching for related documents, ideally the bot only considers highly relevant information, reducing the amount of text it has to process which improves response time and can help with accuracy as information is less likely to be overlooked.

For example, a customer support chatbot for a retailer should probably be aware of return policies (“Is there a longer return window for purchases during the holiday season?”). A “plain vanilla” GPT model is of little value in such cases just as in the Air Canada example. However, overloading the system prompt with this type of information can degrade response performance and is often impractical due to context window limitations. As such, allowing the bot to refer to external information can vastly improve its knowledge base. RAG is especially useful when external information changes rapidly, as the data storage can be quickly updated independently of the LLM. Depending on the data sources needed for RAG, e.g. customer records in Salesforce, additional engineering work is necessary to translate LLM requests to meaningful lookups in the respective databases.

Many intricate details can greatly impact RAG performance. The original documents need to be split into parts that are not too big (and hence too general) and not too small to lack sufficient detail.ⁱ

Dimension 2: Isolated Interactions vs. Long-time Relationship

Does the bot remember users over multiple episodes?

Most bots start an interaction with a user with a “clean slate” (“Hi, I am a virtual assistant; what can I do for you today?”). Each interaction stands alone and there is no memory from one interaction to the other. This works well for a myriad of inquiries, such as asking about historic events or gift ideas.

In contrast, a relationship-focused bot can offer personalized assistance based on past interactions (“Great to see you today! I know you were wondering about opening an account last week. We just increased our interest rate for new customers. Do you want to learn more?”).

The potential for a chatbot to form a long-term relationship with a customer is enormous. Imagine, for example, a user with a learning disability or someone who simply struggles with specific mathematical content (e.g., the concept of compound interest rates). Not only might a relationship-focused bot customize its answers to a form that proves effective over time with the user, such as the terminology used, but it could also help diagnose that the user is experiencing difficulties in the first place. The bot will “remember” what the user struggles with and thereby can adapt its interactions.

A long-term relationship might also make the user more comfortable interacting with the bot. Though such increased comfort comes at the risk of “humanizing” the chatbot and entering an unhealthy relationship (a phenomenon that has been reported especially among teenagers, Roose 2024), there exists enormous potential for customization in utilizing the history of prior interactions.

Technical considerations. It is simple not to store any user history (“stateless”), as only the current question needs to be processed by the bot. No additional resources for storage or authentication are needed. For long-term relationships, the challenge lies with storing past interactions and providing relevant pieces to the chatbot, so it remembers key facts. Due to context window limits (the amount of text an LLM can remember), it is often not feasible to provide the entire conversation history in the prompt to the LLM. Then, a second step is necessary that extracts meaningful insights from previous conversations and summarizes them to provide context for future conversations. In addition, certain customer actions or information from other sources might also be provided as part of the context to the LLM to improve the long-time relationship. Extensive testing is necessary to establish how much information can be provided before performance deteriorates, such as the LLM starting to forget previous facts or responses becoming slower and more expensive since more text needs to be processed with each query.ⁱⁱ This is especially important for cases where the customer might expect that all previous conversation history is considered for future requests; clear communication is necessary in cases where this is not possible.

Dimension 3: Proactive vs. Responsive

Does the user reach out to the chatbot or the chatbot to the user?

Most chatbots are responsive, i.e., they wait until the user takes the initiative and approaches the chatbot with a request. However, there is no reason why the chatbot shouldn’t be proactively taking the initiative (“Hi Joe, your upcoming flight to Paris is in 3 days. It looks like it will be rainy for the first few days. Do you want to learn more about must-visit indoor spots in the city?”). Such proactive bots might increase engagement by offering timely information or reminders, something that is well-studied in the medical domain (Volpp et al. 2017, Lekwijit et al. 2024).

Currently, most chatbots do not reach out proactively by themselves. There are several reasons for this. In general, chatbots mimic the workflow of a customer support center or a help desk, which is by nature responsive. In addition, deploying a responsive chatbot is very simple on a technical level, as that is precisely what the chatbots are designed to do.

Some recent examples of companies moving toward proactive chatbots include platforms such as Character.ai that have experimented with chatbots sending messages to users who have not engaged with

them for a while. Such proactive chatbots require a more complex technical setup where bots periodically “wake up” to reach out to the customer. In the travel context, the outreach could be triggered by specific events, such as upcoming flights, changes in travel advisories, or be customer-specific behaviors, such as checking in with a customer who usually books specific upgrades.

Technical considerations. Building a proactive chatbot requires integrating calendar systems, monitoring tools, and customer relationship management (CRM) systems. The challenge lies in designing a system that balances proactive engagement without becoming intrusive, potentially requiring advanced user preference management and context-awareness capabilities. A proactive chatbot is generally not possible without knowing the user (dimension 2) and might often require additional data from an external system.

In most cases, this requires system integration and thus additional development resources. Lastly, the medium of outreach can play a big role — nudging via email might be less effective than via a text message or app notification, which would require further development resources to design a mobile app. Data and privacy concerns arise in these cases and should be carefully considered and addressed to ensure user trust and compliance.

Dimension 4: Static vs. Dynamic

Does the bot learn about a user or a user population over time?

A static bot operates with a fixed set of rules or knowledge. For a given question, a static bot always provides the same answer. This might, however, miss important opportunities. For example, an investor who is requesting her tax documents likely has very different preferences in April (when most Americans file their taxes) compared to November. A dynamic chatbot adapts its behavior based on external factors or evolving data.

The choice between static and dynamic bots influences how the bot adapts over time and depending on the user. A static bot uses a system prompt and potentially auxiliary systems, such as RAG, to provide information. While it can refer to previous conversations if those are preserved (see dimension 2), it has no representation of time and its influence on customer needs and changes in the surrounding world.

Technical considerations. A simple way to make the chatbot dynamic is by updating the system prompt periodically to include new knowledge, such as the time to the next Tax Day. An even more dynamic approach could involve retrieving the investor’s personal tax schedule. Just as discussed previously, this would require access to additional databases, such as the organization’s CRM system.

In addition to learning more about an individual customer over the course of a longer relationship, we can also imagine chatbots learning from other customers (population-level learning, see Siggelkow and Terwiesch 2019) and using this knowledge to improve the quality of the support. In this case, rather than using an existing frontier model, the organization would train its own model based on the incoming support requests and their resolutions (“we noticed that customers like you who faced similar problems benefited from doing xyz”).

Dimension 5: Quality Assurance

How is the quality of the bot's output assured?

The final dimension in our chatbot design framework focuses on quality assurance. LLMs can sometimes produce erratic output and are prone to hallucinations, which can be more or less problematic depending on the use case. A chatbot supporting travelers that wrongly attributes the origin of a local dish is not as bad as a chatbot that does not take the user's allergies into account when recommending a medication.

Generally, we need to consider two types of defects. First, random hallucinations are instances where the bot generates false or nonsensical information without apparent reason; it just happened not to know or chose a bad token. Second, a user might intentionally try to manipulate (jailbreak) the bot into performing actions against its programming or the programmer's intent. For example, a student might try to convince a teaching bot that their life depends on passing the course and therefore the bot should provide the full answer to all problems in the class immediately.

The complexity of quality assurance increases with the chatbot's generality (dimension 1). A more focused chatbot is easier to validate, while a general-purpose assistant requires more comprehensive measures. We propose three main approaches to quality assurance.

First, the organization can decide to proceed with no formal quality assurance. This approach works well for low-stakes situations, such as providing recommendations for a holiday trip or ideas for an entertainment event. Here, the focus is on setting clear expectations with users about the bot's limitations and potential for errors. The simplest strategy to mitigate most risks is a robust system prompt that clearly defines the bot's boundaries and ethical guidelines. For instance, a bot can be clearly told never to agree to any price discussions. Even in these cases, however, designers should implement extensive testing scenarios to understand the range of responses and test with "edge cases." The system prompt can also include predefined templates that the AI can fill in to reduce variability, such as asking the LLM to always output its final answer in specific tags (<answer>) after it has reasoned about what to do.

Second, the organization can rely on chatbot-based quality assurance by using another LLM (from the same or from a different frontier model) as an auditor. This method employs an additional chatbot to assess the primary chatbot's output relative to the user's prompt. The secondary chatbot acts as a validator, checking for inconsistencies, hallucinations, or inappropriate content. Complex instructions can lead to the first chatbot not always following all best practices, so a secondary bot checking for the most glaring issues can be a helpful and simple-to-implement defense strategy. The auditor chatbot requires a clear set of criteria and possible actions, such as rewriting parts of the response or asking the first bot to regenerate its answer, to provide an effective safeguard. For instance, a user that managed to trick a complex first chatbot into revealing the answer to a homework problem might find it much harder to also trick the second chatbot that is merely told to "ensure the following response never reveals the full solution" instead of having to also adhere to many other instructions. In theory, one can chain many chatbots to improve overall answer quality.

However, each additional chatbot in the chain impacts response times since the previous chatbot needs to finish generating its response, leading to a sequential dependency. The user request is first sent to the first chatbot in the chain, alongside the system prompt and previous conversation. Then, the full response from the first chatbot is awaited. Once completed, the second (next) chat bot can review the answer based on its instructions. If it is the last chatbot in the chain, the revised answer can be streamed (sending smaller paragraph chunks) instead of awaiting full completion to improve responsiveness. For simpler rules, such as stripping out specific words, a buffer for the response of the first chatbot could be used so that words can be validated before they are returned as part of a streaming response.

The third strategy for quality assurance is to “put a human in the loop,” something that we will discuss at length in the second part of this paper.

Part II: Integrating chatbots into workflows

How much autonomy is granted to the chatbot and how much human labor should remain in the workflow depends on the business goals behind the chatbot implementation. In this second part, we will first discuss what organizations might aspire to get out of a chatbot implementation followed by different workflows that determine how human operators interact with the chatbot.

Business Objectives

Broadly speaking, a chatbot can exist for two reasons. First, automation can lead to efficiency gains while maintaining the same level of quality / providing a similar customer experience. When resetting a password or updating a mailing address, for example, a chatbot can achieve the same outcome for a fraction of the cost of a human to manually make such changes].

Second, automation can also be used to enhance the customer experience and the perceived quality of the service provided. For example, a chatbot can be used to engage patients and increase their compliance with their medication regimen in a way that would just not be possible with a non-automated process. Or a student studying geometry can be tutored by a chatbot and obtain a learning experience that otherwise might only be available for those students who can afford private tutoring. Another advantage of the chatbot is that the support of the customer can typically be provided immediately, saving the customer from spending endless time in a queue waiting to be served by a human.

When deciding to implement a chatbot, it is critical to know that they can affect the accuracy of a service and its ability to consistently adhere to a quality standard. For example, hallucinations are a common quality concern for chatbots. Also, customers might have an inherent preference for human operators and human operators might be more knowledgeable and better able to resolve a customer problem. Humans might also be averse to receiving advice from a chatbot (Dietvorst et al. 2015) even though they might be unable to tell the difference between a human and an AI-powered response (Meincke et al. 2024).

On the other hand, chatbots never get tired, are always equally friendly, and can be very knowledgeable if designed accordingly (dimension 1). They also have been shown to be rather charming and persuasive. A recent study comparing chatbots with doctors, for example, found that the chatbots were not only more competent in their diagnosis, but also were perceived as being more empathetic by the patients (Ayers et al. 2023).

As the organization chooses between the following workflows, it should evaluate the most promising process designs alongside their business objectives.

Workflow Configurations: Designing Processes with “Humans in the Loop”

Be it for quality reasons or other business objectives outlined above, fully automated chatbots with absolutely no human oversight are rare and probably are nothing to be desired. For this reason, we now turn to our six workflow configurations that specify how human operators collaborate with GenAI technology. Our six configurations might remind the reader of the levels of autonomous driving. However, while autonomous driving levels are increasing in the responsibility the AI has in driving the car (from level 0: fully human to level 5: fully AI-based) all of our six configurations rely on an interplay between humans and AI. What changes is the division of labor between the two: in configuration 1, the human operator performs the work and the AI focuses on quality assurance and feedback while in configuration 6 it is exactly the opposite. Figure 1 shows the responsibility distribution for response creation and response evaluation performed by the human and agent for each configuration.

We illustrate the six configurations for the hypothetical scenario of a patient seeking help in preparing for an upcoming surgery. In such a scenario, the patient may want to know what and until when she is allowed to eat, when she should arrive at the hospital, and when she can expect to go home and resume her normal activities.

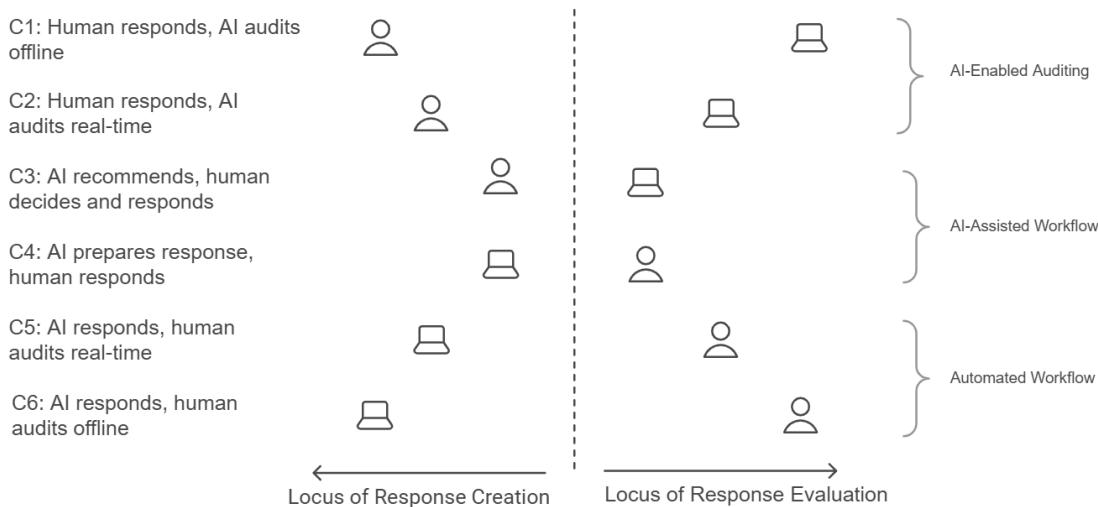


Figure 1: Responsibility distribution for response creation and response evaluation between human and AI

Configuration 1: Human operator response with offline auditing by chatbot

In this first configuration, all the work related to responding to the customer is carried out by the human operator. In the case of our example, the hospital operates a call center or allows patients to contact their care team so that they can ask them questions such as “When is the last meal I am allowed to take?”, “How long will I have to rest after surgery?”, or “Can I have coffee the morning of my surgery?”. The human operator directly provides an answer.

Note that this configuration still allows for some use of AI. For example, if the organization wants to evaluate the accuracy of the medical advice provided by their human operators, it could periodically audit the chats. In such an offline audit, the hospital could use an LLM to identify possible mistakes performed by the human operators and bring them to the attention of management. Such offline auditing by an LLM is not unique to configuration 1 and can be applied to any configuration.

In short, Generative AI is used to improve the accuracy of the customer support and thereby reduces defects and enhances learning about common questions and mistakes. This defect reduction and additional learning should translate to higher efficiency and better customer satisfaction in the long run.

Configuration 2: Human operator response with real-time auditing by chatbot

The second configuration is the same as the first but features real-time AI oversight, instead of offline auditing, of the human operator during the customer support session by AI. Returning to our example, if a human operator provides incorrect or incomplete information to the patient (e.g., the operator gives the wrong co-pay information to the patient or fails to alert the patient that she will not be able to operate a vehicle right after the surgery), an LLM listening to the call or following the chat could instantly pick up the mistake and alert the human operator to it while the conversation with the patient is still ongoing. While this is a technically more complex implementation than many other configurations, it still provides the most autonomy to the human agent outside of configuration 1.

Just like the first configuration, the immediate efficiency gains are relatively low. After all, the customer support is still carried out by human operators. However, the immediacy of the feedback that is now happening in real time leads to fewer defects and faster learning.

These first two configurations are most relevant for transactional tasks that require relatively little cognitive work or active problem solving by the customer support person. However, they may be preferred when dealing with high-risk or high-compliance tasks, as they leave most of the autonomy with the human operator. Next, we will look at the role LLMs can play in helping with more challenging support requests.

Configuration 3: Chatbot recommendations with human operator deciding and responding

Returning to our scenario, consider a patient calling the hospital with a medical problem. The hospital or the healthcare network might have hundreds or even thousands of providers. Each provider differs in their

specific expertise, their geographic location, their availability of appointments, and so on. Which of these should be proposed to the patient?

In the case of the third configuration, a GenAI tool can leverage the available information about the patient and the providers to recommend a few options to the human operator. The operator in turn has the responsibility to choose from these options. In other words, the “heavy lifting” is done by generative AI (narrowing down the choice set from thousands to a handful), but the final decision and responsibility rests with the human. Moreover, the human operators might have to add “a finishing touch” to the solution they recommend, such as explaining the choice to the patient.

This configuration plays to the key strengths of an LLM: it is good at generating options, but, due to hallucinations, it benefits from having a human be the last point in the customer support journey.

Configuration 4: Chatbot preparation with human operator responding

This configuration is similar to configuration 3, except that the LLM only recommends one solution. Its focus therefore is not on generating alternatives for the human operator to choose from, but rather to prepare the final answer to the customer as much as possible.

In our use case, imagine the LLM preparing a customized instruction message to the patient preparing for surgery. The message might include the arrival time, when to take (or not take) specific medications the patient is on, and post-surgical instructions. The role of the human operator in this case is to simply read and approve the message and potentially make minor edits.

This configuration is similar to the use case of medical providers using an LLM to summarize an encounter with a patient and then only reviewing the documentation for accuracy before storing it in the patient’s electronic health record rather than typing up the report from scratch. Or, think of a radiologist who uses GenAI to read an image and prepare a first draft of the report automatically but who makes the final sign-off on the report.

The key value proposition is to significantly lower the touch time of the human operator, thereby improving the efficiency of the customer support organization while leaving the “final word” with the human operator. Moreover, from the customer’s perspective, the experience is one of directly interacting with a human and receiving personalized support.

In configurations 3 and 4, the chatbots are doing a lot of the heavy lifting but ultimately the human operator is responsible for providing the customer support. In the final two configurations, we see the primary responsibility for providing customer support shifting from human operators to the chatbots, with the human role becoming more supervisory.

Configuration 5: Chatbot response with real-time auditing by human operator

Configuration 5 moves even more of the work to the chatbot. While in configuration 4 human operators still applied some “finishing touches” to the support request, in configuration 5 they only oversee the work of the chatbot. Such an oversight might involve a formal approval of a response back to the customer or handling an exception where the chatbot cannot help. This reduces the touch time dramatically, allowing one human operator to oversee multiple chatbots in parallel.

Ideally, the chatbots are capable of explicitly calling for human intervention when they are uncertain about a particular customer request. This could be achieved by having a second chatbot monitoring the interaction with the customer and, in real time, alert the human operator to intervene or provide explicit tools to the first chatbot that make it aware of its capabilities.

Configuration 5 has much larger efficiency potential than configuration 4. The challenge is to determine how many chatbots a single human operator can oversee at the same time. In our example, patients submit requests to the chatbot on the hospital’s website and the chatbots provide a response that, before being posted in the chat, needs to be approved by the human operator.

Configuration 6: Chatbot response with offline auditing by human operator

The highest level of autonomy is realized by providing the chatbot with the authority to handle a customer support request without a human in the loop. Returning to our example, patients approach the chatbots with questions and the chatbot provides the answers. This approach reduces the touch time to zero and thus has the highest efficiency potential.

However, the fact that there is no human in the loop does not imply that there should be no oversight. In configuration 6, management audits the automatically executed chats periodically, potentially with the help of an LLM to refine strategies and improve the customer support experience. During these audits, management can get a sense (though with a delay) about the quality of the support provided and what changes might be necessary.

Configuration 7: Hybrid configurations and implementation

A seventh configuration to consider is a combination of any of the previously discussed configurations. An organization might roll out a GenAI initiative by taking 100% of its calls using configuration 1 (human operators only) and use the accumulating data to fine-tune an LLM.

Then, as a second step, it might conduct a set of experiments that would confront chatbots with the situations encountered by the human operators and estimate a level of confidence with which the chatbots are handling requests correctly.

It is also conceivable that an organization might deploy different configurations based on the nature of the support request. For high-risk or high-compliance tasks, configurations 1 or 2 might be the best option, as they leave most of the autonomy with the human operator. For complex decision-making tasks, configurations 3 or 4 can remove cognitive burden from the human operator and allow them to quickly

arrive at the ideal solution. In scenarios with high volumes but simpler tasks, configurations 5 and 6 might work well.

Conclusion

GenAI, primarily in the form of large language models, has already begun transforming customer support. From travel and tourism to healthcare and education to financial services, chatbots based on LLM's have the potential to make customer support a higher-quality experience for the customer while also improving the efficiency for the organization providing it.

In our discussions and experience in several industries, we were surprised to see that the biggest challenge in the transition of GenAI tends not to be a legal or technical challenge but rather for management to develop a vision of how GenAI could be deployed. Before starting the technical development of a chatbot, executives need to ask themselves what type of bot they would like to get.

The goal of this paper is to help executives develop such a vision by reimagining customer support. We have presented five dimensions of chatbot designs and six configurations specifying workflows how humans and GenAI collaborate to provide customer support.

Our five dimensions of bot design and our six configurations together create a menu of design options. This menu can be used for ideating the needed vision. Should an organization work on a chatbot that has broad knowledge, recognizes a customer over a longer relationship, and is able to learn from past interactions and then deploy it by having a human operator interacting with the customer while getting real-time LLM-powered oversight? Or should the organization pursue a chatbot that is focused on particular support problems, has no memory of past interactions, but is able to autonomously interact with customers with only episodic quality auditing?

In our view, there is not one best design or workflow configuration. Instead, it is management's role to systematically explore the potential designs and configurations and use the business objectives described above to find the most promising customer support vision for their own organization.

Acknowledgements

We thank Vibhanshu Abhishek, Martin Bittner, Lilach Mollick and Hummy Song for their helpful comments.

Endnotes/References

Anthropic (2024) Introducing Contextual Retrieval. Retrieved (October 4, 2024),
<https://www.anthropic.com/news/contextual-retrieval>.

Ayers JW, Poliak A, Dredze M, Leas EC, Zhu Z, Kelley JB, Faix DJ, et al. (2023) Comparing Physician and Artificial Intelligence Chatbot Responses to Patient Questions Posted to a Public Social Media Forum. *JAMA Intern Med* 183(6):589–596.

Bastani H, Bastani O, Sungu A, Ge H, Kabakçı Ö, Mariman R (2024) Generative AI Can Harm Learning. (July 15) <https://papers.ssrn.com/abstract=4895486>.

Cevasco KE, Morrison Brown RE, Woldeselassie R, Kaplan S (2024) Patient Engagement with Conversational Agents in Health Applications 2016–2022: A Systematic Review and Meta-Analysis. *J Med Syst* 48(1):40.

Chen JT, Huang CM (2023) Forgetful Large Language Models: Lessons Learned from Using LLMs in Robot Programming. (October 10) <http://arxiv.org/abs/2310.06646>.

Dietvorst BJ, Simmons JP, Massey C (2015) Algorithm aversion: People erroneously avoid algorithms after seeing them err. *Journal of Experimental Psychology: General* 144(1):114–126.

Faysse M, Sibille H, Wu T, Omrani B, Viaud G, Hudelot C, Colombo P (2024) ColPali: Efficient Document Retrieval with Vision Language Models. (July 2) <http://arxiv.org/abs/2407.01449>.

Gemini Team, Anil R, Borgeaud S, Alayrac JB, Yu J, Soricut R, Schalkwyk J, et al. (2024) Gemini: A Family of Highly Capable Multimodal Models. (June 17) <http://arxiv.org/abs/2312.11805>.

Lekwijit S, Terwiesch C, Asch DA, Volpp KG (2024) Evaluating the Efficacy of Connected Healthcare: An Empirical Examination of Patient Engagement Approaches and Their Impact on Readmission. *Management Science* 70(6):3417–3446.

Meincke L, Nave G, Terwiesch C (2024) The AI Ethicist: Fact or Fiction? (October 11) <https://papers.ssrn.com/abstract=4609825>.

Meincke L, Carton A (2024) Beyond Multiple Choice: The Role of Large Language Models in Educational Simulations. (May 26) <https://papers.ssrn.com/abstract=4873537>.

Melnick K (2024) Air Canada chatbot promised a discount. now the airline has to pay it. *The Washington Post*. Retrieved (December 7, 2024), <https://www.washingtonpost.com/travel/2024/02/18/air-canada-airline-chatbot-ruling/>.

OpenAI, Achiam J, Adler S, Agarwal S, Ahmad L, Akkaya I, Aleman FL, et al. (2024) GPT-4 Technical Report. (March 4) <http://arxiv.org/abs/2303.08774>.

Roose K (2024) Can A.I. be blamed for a teen's suicide? *The New York Times*. Retrieved (December 7, 2024), <https://www.nytimes.com/2024/10/23/technology/characterai-lawsuit-teen-suicide.html>.

Sadasivan C, Cruz C, Dolgoy N, Hyde A, Campbell S, McNeely M, Stroulia E, Tandon P (2023) Examining Patient Engagement in Chatbot Development Approaches for Healthy Lifestyle and Mental Wellness Interventions: Scoping Review. *J Particip Med* 15:e45772.

Siggelkow N, Terwiesch C (2019) *Connected Strategy: Building Continuous Customer Relationships for Competitive Advantage* (Harvard Business Press).

Volpp KG, Troxel AB, Mehta SJ, Norton L, Zhu J, Lim R, Wang W, et al. (2017) Effect of Electronic Reminders, Financial Incentives, and Social Support on Outcomes After Myocardial Infarction: The HeartStrong Randomized Clinical Trial. *JAMA Internal Medicine* 177(8):1093–1101.

ⁱ There are also many different choices of algorithms for comparing user queries to chunks stored in a database. When multiple pieces of information are relevant, they need to be ordered (ranked) to ensure that the bot focuses on the most pertinent data. To improve chunking performance, chunks can be infused with additional information (Anthropic 2024). For instance, homework questions on a business case could repeat key information from the case in each chunk to improve retrieval performance. Alternatively, screenshots of documents can be used instead of text (Faysse et al. 2024). This approach reduces the reliance on efficient chunking and leverages existing formatting in documents.

ⁱⁱ Prompt-caching can be an effective solution to improve responsiveness and lower costs for information that remains static over multiple LLM calls.

MDSAA

Master Degree Program in
Data Science and Advanced Analytics

ENHANCING CUSTOMER SUPPORT WITH OPEN-SOURCE LLMS: DEVELOPMENT OF A QUESTION ANSWERING APPLICATION

Evaluating the potential of a LLM-based Question Answering application to improve customer support in Small and Medium-sized Enterprises as a cost-effective solution to lower the barrier of AI adoption

Jannik Kickler

Master Thesis

presented as partial requirement for obtaining a Master's Degree in Data Science and Advanced Analytics

**NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação**

Universidade Nova de Lisboa

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

ENHANCING CUSTOMER SUPPORT WITH OPEN-SOURCE LLMS:
DEVELOPMENT OF A QUESTION ANSWERING APPLICATION

Evaluating the potential of a LLM-based Question Answering application to improve customer support in Small and Medium-sized Enterprises as a cost-effective solution to lower the barrier of AI adoption

by

Jannik Kickler

Master Thesis presented as partial requirement for obtaining the Master's degree in Data Science and Advanced Analytics, with a specialization in Data Science

Supervised by

Professor Fernando Bação

July, 2024

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration. I further declare that I have fully acknowledged the Rules of Conduct and Code of Honor from the NOVA Information Management School.

Hamburg, 10th July 2024

Jannik Kickler

ABSTRACT

Today's customers expect 24/7 personalised service across multiple channels, putting immense pressure on organisations to deliver exceptional experiences while maintaining efficiency. Generative Artificial Intelligence (AI) is emerging as a potential solution, promising to automate tasks and personalise interactions. However, this technology remains largely out of reach for Small and Medium-sized Enterprises (SMEs) due to financial and technical constraints. This research investigates the potential of Large Language Models (LLMs) to enhance customer support capabilities in SMEs, with a particular focus on technical question answering. The research develops a Question Answering (QA) application that uses Retrieval-Augmented Generation (RAG) to optimise information retrieval from technical documents. The application developed exclusively using open-source tools and LLMs, aims to provide a cost-efficient solution for SMEs to lower the barrier of AI adoption. Guided by a Design Science Research Methodology (DSRM), the research details the development of the application and evaluates its performance and usability. The results show that while the application successfully returns answers with high factual accuracy and explainability scores, it suffers from high latency issues, with response times unsuitable for real-world use. The method used to incorporate human feedback when the LLM cannot answer a question demonstrated potential as a continuous learning mechanism for the application. However, this mechanism still needs further development to achieve consistent performance. In summary, while the open-source LLM-based QA application shows potential for improving SME customer support, significant improvements in computational resources and alternative judging approaches are required to fully realise its capabilities. Future work should focus on increasing processing speeds and exploring more powerful LLMs to reduce latency and improve answer relevance and correctness.

KEYWORDS

Generative Artificial Intelligence; Small and Medium-sized Enterprise; Artificial Intelligence Adoption; Large Language Models; Question Answering; Retrieval Augmented Generation; LLM-as-a-Judge, Natural Language Processing

Sustainable Development Goals (SDG):



TABLE OF CONTENTS

Statement of Integrity	i
Abstract	ii
List of Figures.....	v
List of Tables.....	vi
List of Abbreviations and Acronyms.....	vii
1. Introduction.....	1
1.1. Delimitations of the Research	2
1.2. Overview of the Research Structure	2
2. Background	4
2.1. Question Answering	4
2.2. Language Models.....	5
3. Related Work	7
3.1. Role of LLMs in Question Answering.....	7
3.2. Issues of LLMs in Question Answering	8
3.3. Mitigation Methods.....	9
3.3.1.Knowledge Injection via Fine-Tuning	9
3.3.2.Knowledge Injection via RAG	10
3.3.3.Comparison of Fine-Tuning and RAG	11
3.4. Trade-Offs between RAG and Fine-Tuning.....	12
4. Methodology	14
4.1. Problems and Motivation.....	14
4.2. Design Requirements	14
4.3. Research Objectives	16
4.4. Technological Stack	17
4.4.1.Computational Resources	17
4.4.2.Application Development Tools	17
4.4.3.Model Selection.....	18
4.5. Data Collection & Processing	21
4.6. Application Architecture	21
4.7. Evaluation	26
5. Discussion & Results	28
6. Conclusion	32
Bibliographical References.....	viii
Appendix A – Model Selection Prompts	xiv

Appendix B – Prompt Iteration Product Chain.....	xvi
Appendix C – Prompt Iteration Document Chain	xx
Appendix D – Prompt Iteration Judge	xxiii
Appendix E – Synthetic Dataset Prompt	xxvi
Appendix F – Bing Chat Prompt	xxvii

LIST OF FIGURES

Figure 1 – Internal Flow inside the Application following a User Query	22
Figure 2 - Total Frequency of Correctness Scores.....	28
Figure 3 - Mean Value per Metric	29

LIST OF TABLES

Table 1 - Comparison between RAG and FT.....	11
Table 2 - Definition of RAGAs Metrics.....	27

LIST OF ABBREVIATIONS AND ACRONYMS

AI	Artificial Intelligence
ANN	Artificial Neural Network
DSRM	Design Science Research Methodology
FT	Fine-Tuning
IRQA	Information-retrieval based Question Answering
KBQA	Knowledge-based Question Answering
LLM	Large Language Model
LM	Language Model
ML	Machine Learning
NLP	Natural Language Processing
QA	Question Answering
RAG	Retrieval Augmented Generation
RAGAs	Retrieval-Augmented Generation Assessment
RLHF	Reinforcement Learning from Human Feedback
RNN	Recurrent Neural Network
ROI	Return on Investment
SME	Small and Medium-sized Enterprise
SOTA	State-of-the-art

1. INTRODUCTION

Customer service has undergone significant transformation due to technological advancements, with Generative Artificial Intelligence (AI) emerging as a new major force. As highlighted in the 2023 Gartner Hype Cycle for Customer Service and Support Technologies, Generative AI is currently at the peak of expectations, with Gartner predicting that by 2025, 80% of customer service organizations will implement this technology for enhanced productivity and customer experience (Gartner, 2023b, 2023a). One of the ideas behind the adoption of AI in customer service is to be more engaged with customers, as their loyalty and increased interaction with brands contribute to greater lifetime value (Das et al., 2023). Today's customers expect personalized interactions and instant responses across multiple channels. With these growing demands, Generative AI presents a solution (Morgan, 2023). It can help to automate repetitive tasks, like searching and filtering through countless documents and resources, alleviating the burden of support staff. This enables human agents to focus on complex problems that demand a human touch, while simultaneously delivering personalized service to foster the relationship with the customer (Das et al., 2023; Podolsky, 2023; Wall Street Journal, 2023). Furthermore, the customer service industry is facing problems like employee burnout and inefficiency (Das et al., 2023; Drenik, 2023). Ranked among the world's top 10 most stressful jobs (Morgan, 2023), the industry witnesses significant talent shortages, leading to costly recruitment and training of new staff - impacting both customer and employee experiences. Providing support staff with access to Generative AI tools can contribute to the overall well-being and satisfaction of employees, increase the employee productivity and boost the performance of less skilled and less experienced workers (Brynjolfsson et al., 2023; Morgan, 2023; Podolsky, 2023).

While Generative AI offers significant benefits, its adoption is often concentrated in large enterprises. Small and Medium-sized Enterprises (SME), which make up the vast majority of businesses in the EU, often lack the financial resources and technical expertise to implement these solutions. The high cost of developing and integrating AI into existing workflows is a barrier, particularly for SMEs with limited budgets. In addition, the difficulty of quantifying the return on investment (ROI) discourages SMEs from pursuing AI technologies (European Economic and Social Committee. et al., 2021).

This research specifically examines the potential of Generative AI, particularly Large Language Models (LLMs), to enhance the capabilities of customer support teams in SMEs. Here, the focus is on tasks related to answering technical questions about the SMEs' products. The objective of this research is to develop an application that optimizes information retrieval from technical documents for SME customer support agents using LLMs or LLM-based methods. This will allow customer support agents to dedicate more time to complex customer interactions and complaint resolution, ultimately improving the customer experience. Furthermore, to address the financial limitations of SMEs, the application will be developed solely with entirely free and open-source tools and open-source LLMs, lowering the adoption

barrier for these businesses. The Design Science Research Methodology (DSRM) framework established by Peffers et al. (2007) will guide the development process, combining design science principles with LLM-based techniques to create this technical Question Answering (QA) application.

1.1. DELIMITATIONS OF THE RESEARCH

This research focuses on the development of an application to optimize information retrieval for customer support agents in SMEs. The application specifically targets technical questions about the SMEs' products, excluding broader customer service issues which require human empathy or complex problem-solving skills. Furthermore, the aim of the project was to demonstrate the potential of open-source LLMs for process optimization within an SME customer support team, rather than to develop a fully functional application ready for immediate use.

To ensure affordability for the SME audience, the research prioritizes free and open-source tools and LLMs. Consequently, it does not explore commercially available solutions. Additionally, the research focused primarily on utilizing LLMs to address the presented problem. Alternative information retrieval methods were not investigated.

The literature review was limited to publications in January 2024 and earlier and the selection of LLMs for the application was completed in February and March 2024, potentially excluding the most recent state-of-the-art (SOTA) models. In addition, the evaluation process focused on the application's accuracy and efficiency in retrieving relevant information and did not include metrics directly related to user experience or customer satisfaction.

Finally, for the development and testing of the application a text corpus provided by German lighting company was used. The text corpus consists of structured information about the SMEs lighting products such as product names, codes, light output and power consumption and more detailed information on specific products or the broader portfolio. This means that the application is not currently suitable for handling customer support enquiries in other languages.

1.2. OVERVIEW OF THE RESEARCH STRUCTURE

The development of the core application code was the largest part of this work and represents its central contribution. The application code, written in Python, is available on <https://github.com/jkickler/opensource-rag-qa-for-sme> to ensure transparency and allow others to run the application. The core code is in the "src" folder, which is organised for easy navigation. Within src, the application folder contains Python files for each major component of the application, such as the chatbot and knowledge base. The evaluation folder stores scripts used to evaluate the performance of the application. Finally, the "notebooks" folder contains Jupyter notebooks for various tasks. Here the notebooks for data collection and processing can be found, as well as for the evaluation process including the figures presented

in chapter 5. The application can be run directly in the Jupyter environment via the "run_application.ipynb". Alternatively, users can navigate to the src folder in their terminal and run "__main__.py". This will run the application directly in the terminal.

The research is structured as follows: First, chapter 2 gives a theoretical background to QA and Language Models (LM), which is essential for an understanding of the research. Afterwards chapter 3 discusses related literature with a particular focus on the role of LLMs in QA, as well as potential issues associated with their use. Chapter 4 describes the problem-centred methodology, focusing on the development, improvement, and the performance evaluation of the application, including the technological stack and the LLM selection process. Chapter 5 provides an analysis of the evaluation results and the overall design process to answer the research questions. It also includes a critical analysis of the findings. Finally, chapter 6 concludes the paper by summarizing the key takeaways from the entire research project and explores future research directions.

2. BACKGROUND

This chapter establishes the foundation for understanding the research by examining the concepts of QA and LMs. It begins with an exploration of the field of QA, focusing on its core concepts and historical development. Following this, it provides an overview of LMs and their evolution over time. This foundation is crucial for understanding how LMs contribute to the field of QA. Chapter 3, "Related Work", reviews additional research in this area.

2.1. QUESTION ANSWERING

QA is considered as one of the most important fields of Natural Language Processing (NLP). The primary goal of QA is to enable humans to ask questions to machines in natural language. This is achieved by either extracting information related to the input question from different knowledge sources or by generating answers without additional steps like searching through a ranked list of documents provided by a search engine (Farea et al., 2022; Ishwari et al., 2019). QA systems transform questions into dependency structures, which enables machines to identify relationships within a context or corpus. The relationships are detected based on matching words or embedded semantics, with techniques like word embeddings. Most QA systems focus on factoid questions, a subset of questions that can be answered with simple facts expressed in short texts, like "Which city is the capital of Portugal?" (Jurafsky & Martin, 2023) .

QA can be divided in two different paradigms (Jurafsky & Martin, 2023): Information-retrieval based QA (IRQA) and knowledge-based QA (KBQA). IRQA, also known as open-domain QA, relies on information retrieval from vast amounts of text on the web or in collections of documents. In this context, a document refers to any unit of text the system indexes or retrieves like web pages, news articles or shorter passages such as paragraphs and a collection represents the set of documents which contain the answer to the user's question. IRQA includes two stages: retrieval, which returns relevant documents from the collection, and reading, in which a neural reading comprehension algorithm extracts answer spans. In the retrieval stage, the user's query is processed, and most relevant documents are retrieved from a collection. This is done by converting the natural language query into vector embeddings, which are then mapped to documents with the potential answer based on a similarity score, like cosine similarity or dot product. The reader, a neural reading comprehension algorithm, takes the retrieved documents, passes over each passage in the documents and tries to find an answer from spans of text. KBQA, also called closed-domain QA, aims to answer natural language questions by mapping them to queries over structured databases. This paradigm can be further divided in two approaches: graph-based QA and QA by semantic parsing. In graph-based QA a knowledge graph is constructed from the knowledge base, with entities as nodes and relations as edges between the nodes. The other approach uses semantic parsers to map questions into logical forms like SQL queries or other executable programs (Farea et al., 2022; Jurafsky & Martin, 2023).

Looking back in history, QA has been a rapidly developing field in NLP research for over the last half century. Early text-based and knowledge-based systems were built on quite basic computers, had to follow specific query templates and focused on simply retrieving information in a small closed-domain setting. Generalization capabilities of these systems to other domains were limited. Two notable systems, which laid the foundation of KBQA systems in the 1960s, are BASEBALL and LUNAR, which delivered information about the US baseball league and soil samples from the Apollo lunar exploration (Jurafsky & Martin, 2023). In the 1990s two developments influenced QA substantially: First the emergence of the web demanded for better IRQA systems and secondly the foundation of the Text Retrieval Conference (TREC), which initiated new research fields about open-domain QA in the form of competitions, which are still held annually to solve complex QA problems (Jurafsky & Martin, 2023; Shen, 2023).

As natural language is always evolving, QA moved over the years to data driven methods, due to their capabilities and flexibility in handling natural language. Since the 2000s, feature-based systems, based on machine learning (ML) algorithms, have led to new innovative approaches in QA (Farea et al., 2022; Ishwari et al., 2019; Jurafsky & Martin, 2023; Shen, 2023). Compared to previous systems, machine learning algorithms are capable of learning from features extracted from documents, which enable them to infer answers. However, one prerequisite of these algorithms is that enough training data is provided. One notable example in feature-based systems is IBM's Watson DeepQA computer, which won the quiz show Jeopardy in 2010 (Shen, 2023).

The next advancement in QA systems came through the usage of Artificial Neural Networks (ANN), which have the ability to automatically learn underlying features in the data (Ishwari et al., 2019; Jurafsky & Martin, 2023; Shen, 2023). A special type of ANNs, the Recurrent Neural Networks (RNN), became very popular in QA, because of their ability to process sequential data, such as text, and therefore their capability to capture sequential relationships inside the data (Gillioz et al., 2020). The latest major impact experienced by the QA field was the invention of the Transformer architecture in 2017 (Vaswani et al., 2023). The new architecture removes the recurrent architecture to rely on a new self-attention mechanism. Furthermore, it solved the RNNs vanishing gradient problem and the parallelization issues (Gillioz et al., 2020; Shen, 2023).

2.2. LANGUAGE MODELS

In NLP, models based on the Transformer architecture invented by Vaswani et al. (2023) are also called LMs. According to Zhu et al. (2023) a LM is a model that determines the generative likelihood of sequences of words based on the context of the text. Through this, the model is able to predict the probability of subsequent words and can generate text. A new era of LMs, the pre-trained LMs, was initiated by BERT (Devlin et al., 2019), a transformer-based model pre-trained on an extensive unlabeled corpora, demonstrating promising results on various NLP tasks. Other influential pre-trained LMs emerged, notably GPT-2 (Radford et al., 2019),

which showed that zero-shot learning enables a model to transfer its knowledge to new NLP tasks without specific fine-tuning, and T5 (Raffel et al., 2023), demonstrating that a wide array of NLP problems and tasks can be transformed into a text-to-text format, which makes them solvable through text generation. Researchers found out that enlarging pre-trained LMs (e.g. increasing the parameter size or the training data) improves and enhances their abilities in downstream tasks like QA. New large-sized LMs like GPT-3 and PaLM surfaced (Brown et al., 2020; Chowdhery et al., 2022). These models, now categorized as LLMs, not only demonstrated enhanced performance but also exhibited higher capabilities in tackling complex tasks (e.g. multiple-choice QA), compared to their smaller predecessors. The newest generation of LLMs like GPT-4 (OpenAI, 2023) or ChatGPT (Kalyan, 2023) are now considered by research as general-purpose task-solvers instead of solely text generation models. Models are now able to parse multimodal signals (e.g. speech or images) next to text input (Zhao et al., 2023).

In order to gain an understanding of the capabilities of LLMs, it is important to be aware of the concept of “prompting”. It should be noted that, due to the wide range of research on prompting LLMs, only a brief explanation of key concepts will be given in the following.

Prompting is currently the primary method for utilizing LLMs for the purpose of solving various tasks and one of the most researched topics about LLMs (Liu et al., 2021; Zhao et al., 2023). As the quality of prompts has a significant impact on LLM performance, researchers are actively developing methods to generate effective prompts. A well-designed prompt is typically constructed from four key parts: a clear task description or specific instruction that the LLM is expected to follow, relevant input data to complete the task, any necessary contextual or background information, and a prompt style customized to both the specific LLM and the task at hand (Zhao et al., 2023). The further success of LLMs is often attributed to in-context learning, a widely used prompting technique which was first proposed along with GPT-3 (Brown et al., 2020). By in-context learning the model is conditioned to solve more complex tasks, where the prompt contains a small number of examples of the desired input-output pairs (few-shot) or instructions describing the task (zero-shot) (Dong et al., 2023; Kojima et al., 2023). While LLMs excel at intuitive and single-step tasks, which can be solved with in-context learning, even the largest models (with 100B parameters or more) struggle with problems that require complex, multi-step reasoning (Kojima et al., 2023). To address this, a new technique called chain-of-thought prompting was invented (Wei et al., 2023). Unlike a few-shot prompt with standard question and answer examples, a chain-of-thought prompt provides LLMs the standard question with step-by-step answers of reasoning. These examples act as a reasoning path, helping models to break down complex problems into smaller, more manageable steps (Kojima et al., 2023; Zhao et al., 2023).

3. RELATED WORK

This chapter examines existing research on the application of LLMs to QA. It explores the potential of LLMs in this domain, as well as the challenges that arise. To address these challenges, researchers have proposed various mitigation methods. One of these, knowledge injection, is explored in depth, focusing on two prominent approaches: Fine-Tuning and Retrieval Augmented Generation. A comparison of these methods is provided to determine their relative effectiveness and examining the potential trade-offs between them.

3.1. ROLE OF LLMs IN QUESTION ANSWERING

To understand how LLMs perform in QA tasks, an essential ability of LLMs is knowledge utilization. Knowledge utilization is defined as the ability to access the factual knowledge from pre-trained corpus or retrieve external data if required (Zhao et al., 2023). The evaluation of this ability involves assessing LLMs by using different benchmarks or datasets, such as TriviaQA or SQuAD (Guo et al., 2023) and is divided into three approaches: closed-book QA, open-book QA and knowledge completion.

Closed-book QA tasks test the LLMs' ability to answer questions solely based on their pre-trained acquired knowledge and given context, without external sources. Furthermore, studies showed that the performance and capacity of LLMs in closed-book QA tasks rises as the model and data size increases, because more knowledge is learned or memorized from the pre-training data. In open-book QA tasks, a LLM can extract information from external knowledge bases or document collections to answer questions. Text retrievers are often paired with LLMs, which allows the LLM to better understand the question and provide a more accurate answer (Zhao et al., 2023).

Open-book QA tasks also evaluate the recency of knowledge information, which is critical for avoiding incorrect answers to time-sensitive questions. Studies here showed that text retrievers can largely improve the accuracy of generated answers of LLMs (Zhao et al., 2023).

In the last evaluation approach, knowledge completion, studies explore whether LLMs can serve as an independent knowledge base (Petroni et al., 2019; Roberts et al., 2020), focusing only on how much and what kind of knowledge the LLM learned during pre-training. Further works showed challenges in tasks related to specific domains, if these were not included in the training (Zhao et al., 2023).

Numerous works have been dedicated to the evaluation of LLMs' knowledge utilization, with studies covering various domains such as education, news, healthcare and social media (Kalyan, 2023; Zhao et al., 2023). Although LLMs generally perform satisfactorily in QA, as shown by the majority of these studies, they have not yet reached everywhere SOTA results when compared to specialized QA systems in relevant domains (Bahak et al., 2023; Chang et al., 2023; Kalyan, 2023; Liang et al., 2023; Tan et al., 2023). However, in closed-book QA LLMs

already perform well and are on par with the performance of SOTA IRQA systems, but they still struggle with QA tasks which require fine-grained knowledge, even when present in the pre-training data (Zhao et al., 2023).

3.2. ISSUES OF LLMs IN QUESTION ANSWERING

As seen in chapter 3.1, LLMs already show promising results in QA tasks. However, when LLMs are used for QA, there are challenges related to the general issues of LLMs. One of the major issues of LLMs is hallucination, which occurs in almost every existing LLM (Zhao et al., 2023). Hallucination is currently one of the most researched topics in the LLM domain. It is a phenomenon in which a LLM generates content that is not based on factual or accurate information, potentially leading to the creation of fictional, misleading or entirely fabricated details. This is due to the model's ability to produce plausible-sounding text based on learned patterns, even when the content doesn't match reality (Huang et al., 2023; Y. Zhang et al., 2023). In addition, studies show that LLMs have difficulty recognising and dealing with hallucinated content that they have created themselves. Hallucination can have several different causes and many mitigation and detection techniques are continuously developed (Rawte et al., 2023).

Another major issue is the knowledge boundary, which can be divided into outdated knowledge and a lack of specific domain knowledge (Zhao et al., 2023). Outdated knowledge describes the static nature of LLMs' knowledge. The knowledge acquired by LLMs during their training has clear temporal boundaries and can become outdated over time. Therefore, it is difficult for them to provide information about recent events if their parameters are not updated (Kaddour et al., 2023). Outdated knowledge also influences hallucination, because if the LLM doesn't know the answer, it might fabricate facts or provide answers which might have been correct in the past, but are now outdated (Huang et al., 2023). Another knowledge boundary is the domain knowledge deficit observed in some LLMs. Although they are primarily trained on large publicly available datasets, their expertise in specialized domains is inherently limited due to the lack or small size of domain-specific training data (Ovadia et al., 2023; Zhao et al., 2023). When faced with queries that require domain-specific knowledge, such as medical questions, these models may start to hallucinate and generate misleading or false information (Huang et al., 2023).

In addition to knowledge boundaries, LLMs also face knowledge shortcuts and challenges in acquiring rare, long-tail knowledge (Ovadia et al., 2023). This means that a model is exposed to information during its training process but struggles to learn and reproduce this knowledge. This is especially true for rare facts that are sparsely represented in the training dataset (Huang et al., 2023).

Despite their effectiveness, language models (LLMs) are also commonly referred to as "black box" systems. This label stems from the current lack of straightforward mechanisms to provide insight into their predictions, which poses challenges in terms of explainability and

transparency. Explainability and transparency of LLM output is particularly important to end-users, as it helps to explain model reasoning in a non-technical way, thereby improving understanding of the capabilities of LLMs and the potential generation of false output (Luo & Specia, 2024).

3.3. MITIGATION METHODS

There are many mitigating techniques in the literature to address the issues highlighted in chapter 3.2. One obvious solution is to retrain the model, but this is both costly and time consuming, and there is a risk that LLMs will forget previously acquired knowledge when they are retrained. Another approach, known as knowledge editing, explores adjusting the parameters or adding new parameter layers to update existing knowledge. However, these methods are complex and have not performed well on LLMs in studies, compared to other methods (Huang et al., 2023; Kaddour et al., 2023; N. Zhang et al., 2024; Zhao et al., 2023). A third approach, knowledge injection, involves the addition of external information or data to augment the knowledge of a LLM to improve its performance (Ovadia et al., 2023). This chapter focuses on knowledge injection, specifically exploring two widely used approaches: Fine-Tuning (FT) and Retrieval Augmented Generation (RAG).

Before going into the details and for a better understanding, Gao et al. (2024) outline the two approaches as follows: “*RAG is like giving a model a textbook for tailored information retrieval, perfect for specific queries. On the other hand, FT is like a student internalizing knowledge over time, better for replicating specific structures, styles, or formats.*”

3.3.1. Knowledge Injection via Fine-Tuning

One way of injecting knowledge into a model is through FT, where the model's training process continues, updating parameters using a task-specific, often narrower, dataset. By exposing the model to specific knowledge bases, it is expected that the model parameters will adapt, optimizing it for targeted applications and improving performance in specialized domains.

FT includes several approaches, one of which is supervised FT, which requires sets of labeled input-output pairs (Ovadia et al., 2023). One method of supervised FT is instruction tuning, introduced by Wei et al. (2022). This process involves annotating or collecting massive task instruction-following data consisting of (instruction, response) pairs, where instruction is defined as human instructions and output as the desired model response that follows the instruction. The LLM is then fine-tuned using this data, thereby increasing its capabilities and controllability (Huang et al., 2023; S. Zhang et al., 2023; Y. Zhang et al., 2023; Zhao et al., 2023). The advantage of instruction tuning is that it provides more controllable and predictable model behavior compared to standard LLMs. The instructions serve to constrain the LLMs' output and to match with the desired response, which provides a method for humans to intervene with the LLMs' behavior. Furthermore, it is computationally efficient and can help to adapt LLMs to a specific domain compared to retraining a model or to change the models' architecture. Despite its advantages, instruction tuning presents challenges, including the

difficulty of producing high-quality instruction datasets and concerns about capturing only surface-level patterns rather than understanding and learning the task (S. Zhang et al., 2023).

Another notable FT method is Reinforcement Learning from Human Feedback (RLHF), which uses RL or RL-inspired optimisation strategies and collected human feedback data (Zhao et al., 2023). While instruction tuning enables LLMs to follow user instructions, RLHF aims to better align LLMs with human preferences and values after training (Huang et al., 2023). RLHF works by first training a reward model as a proxy for human preference, assigning appropriate reward values to LLM responses, and then optimizing the LLM with feedback from the reward model using RL algorithms. Incorporating human feedback into the training loop has been shown to be effective in improving the alignment of LLMs and guiding them to produce high-quality, safe responses (Lambert et al., 2022; Y. Zhang et al., 2023). Despite its benefits, implementing RLHF poses practical challenges. Firstly, collecting human feedback data is challenging because the selection and instruction of human representatives can introduce biases into the model. In addition, humans can make simple mistakes due to constraints such as limited time, interest or attention. Secondly, it is difficult to represent individual human values with a reward function. The complex and context-dependent nature of human preferences makes accurate modelling difficult, and incorrect assumptions about human decision-making can adversely affect the outcome (Casper et al., 2023).

3.3.2. Knowledge Injection via RAG

An alternative approach to enhancing a model's knowledge base is in-context learning, which aims to improve the performance of pre-trained LLMs on new tasks by modifying the input query without directly changing the model's parameters (Ovadia et al., 2023). One form of in-context learning is RAG, which was introduced by Meta (formerly Facebook) in the mid-2020s (Lewis et al., 2021). Since its introduction, RAG has evolved significantly and stands out as one of the most popular techniques for enriching LLM knowledge. RAG extends the capabilities of the LLM by using external sources of knowledge and follows a retrieve-then-read pipeline (Zhao et al., 2023). In the initial step, a retriever component acts like a search engine, matching the user's query against external data sources to find documents or text spans most relevant to the user's query. These retrieved documents are then passed to the generator component, which is itself a LLM. This LLM then uses the original query along with the retrieved context to generate a more complete answer. This ensures that answers are based on retrieved evidence, significantly improving accuracy and relevance (Gao et al., 2024). The main advantage of RAG is that it eliminates the need to retrain or change the parameters of LLMs for task or domain-specific applications. Instead, an external knowledge base can be added to enrich the query, thereby improving the model's responses. In addition, the RAG architecture is highly practical and has a low barrier to entry for integration into existing systems (Gao et al., 2024; Ovadia et al., 2023).

3.3.3. Comparison of Fine-Tuning and RAG

The decision between RAG and FT should be based on the specific requirements of the scenario and the characteristics of each approach. A comprehensive comparison of RAG and FT has already been made in the study by Gao et al. (2024), the results of which are presented in Table 1.

Table 1 - Comparison between RAG and FT (Gao et al., 2024)

Metric	RAG	FT
Knowledge Updates	Directly updating the retrieval knowledge base ensures that the information remains current without the need for frequent retraining, making it well-suited for dynamic data environments.	Stores static data, requiring retraining for knowledge and data updates.
External Knowledge	Proficient in leveraging external resources, particularly suitable for accessing documents or other structured/unstructured databases.	Can be utilized to align the externally acquired knowledge from pretraining with large language models, but may be less practical for frequently changing data sources.
Data Processing	Involves minimal data processing and handling.	Depends on the creation of high-quality datasets, and limited datasets may not result in significant performance improvements.
Model Customization	Focuses on information retrieval and integrating external knowledge but may not fully customize model behavior or writing style.	Allows adjustments of LLM behavior, writing style, or specific domain knowledge based on specific tones or terms.
Interpretability	Responses can be traced back to specific data sources, providing higher interpretability and traceability.	Similar to a black box, it is not always clear why the model reacts a certain way, resulting in relatively lower interpretability.
Computational Resources	Depends on computational resources to support retrieval strategies and technologies related to databases. Additionally, it requires the maintenance of external data source integration and updates.	The preparation and curation of high-quality training datasets, defining fine-tuning objectives, and providing corresponding computational resources are necessary.
Latency Requirements	Involves data retrieval, which may lead to higher latency.	LLM after fine-tuning can respond without retrieval, resulting in lower latency.
Reducing Hallucinations	Inherently less prone to hallucinations as each answer is grounded in retrieved evidence.	Can help reduce hallucinations by training the model based on specific domain data but may still exhibit hallucinations when faced with unfamiliar input.

3.4. TRADE-OFFS BETWEEN RAG AND FINE-TUNING

This chapter examines the potential of RAG and FT to address the issues highlighted in chapter 3.2, with a focus on usage in the QA context. Each of the issues will be examined in detail.

Regarding hallucination, RAG demonstrates clear benefits, as each answer to a query is based on evidence from retrieved documents or text spans of the knowledge base. In addition, different retrieval algorithms can be tested to improve retrieval performance if necessary. On the other hand, FT can mitigate hallucination by training on a specific domain, but it may still occur with queries involving unfamiliar input.

In terms of knowledge boundaries, in a RAG architecture, the knowledge base can be updated directly and frequently, which addresses the problem of outdated knowledge. Moreover, if questions related to a specific domain cannot be correctly answered, the knowledge base can be updated accordingly. In FT, outdated knowledge is still an issue because the FT process updates parameters at a specific point in time, which again represents temporal boundaries already existing after the pretraining of the LLM. For the domain knowledge deficit, FT allows the adaptation of the LLM to the specific domain, including behavior and writing style, a feature not provided by the RAG approach.

Concerning knowledge shortcuts, in RAG, the knowledge base can be updated with the knowledge that the LLM struggled to learn and for FT, the LLM can be fine-tuned to specific missing knowledge that was not captured during the pre-training phase. However, determining which knowledge was learned by the LLM and which was not is a separate research field and beyond the scope of this paper. As a result, neither framework provides a clear solution for this issue.

Regarding explainability and transparency RAG offers a method for tracing the response back to specific documents of the knowledge base, thereby improving explainability and giving higher transparency on which knowledge the LLM created the response. In contrast, FT adjusts only the parameters of the LLM without adding a mechanism to improve explainability or transparency, therefore FT presents no real solution to this issue.

In addition to the issues outlined, critical factors such as computational resources, data processing and latency requirements must be considered. RAG requires an external knowledge base, and the maintenance of this knowledge base is critical. While the data processing and handling for RAG is comparatively less than for FT, each query to the RAG architecture involves retrieving information from the knowledge base, potentially resulting in higher latency. A LLM after FT tends to be more performant as there is no retrieval step involved. However, FT requires the creation of a high-quality training dataset, which can increase the need for computational resources and intensify the data processing process.

In summary, RAG and FT offer different advantages and trade-offs, and the choice between them should be based on the specific requirements and considerations of the scenario or application in question.

4. METHODOLOGY

The methodology of this research starts with an analysis of the challenges and a definition of the design requirements. It then outlines the research objectives and presents the technological stack, which was used to create the application, covering available computational resources, development tools and LLM selection. Afterwards, the data collection and processing are described, and the components of the application are presented. Finally, the chapter details how the application's performance is evaluated.

4.1. PROBLEMS AND MOTIVATION

The customer service industry faces issues such as employee burnout and inefficiency, leading to a talent shortage in the industry. As a result, costly recruitment and training is required - impacting both the customer and employee experience. Generative AI tools such as LLMs offer a promising solution to reduce workload and improve employee well-being. A study conducted by Brynjolfsson et al. (2023) has shown that AI assistance can significantly increase employee productivity, allowing human agents to successfully handle more chats per hour and resolve them faster. In addition, AI can boost the performance of less skilled and less experienced workers, helping them to reach proficiency more quickly.

However, cost is a major factor for SMEs considering AI adoption. The high cost of implementing and adapting business processes for AI solutions is a significant barrier for companies, with SMEs being more affected than large companies due to their comparatively limited financial resources. In addition, a lack of clarity around the return on investment (ROI) and overall business case for AI often discourages SMEs from pursuing these technologies (European Economic and Social Committee. et al., 2021). Therefore, the development of this application prioritizes cost-effectiveness within the resource-constrained environment of an SME. This is achieved by relying solely on free, open-source libraries or tools for application design, and by using open-source LLMs to answer the technical questions. In addition, the application is designed to run on consumer-grade hardware, minimizing the cost of acquiring computational resources.

4.2. DESIGN REQUIREMENTS

Beyond addressing the challenges presented, several other factors need to be considered when designing a technical QA application. Firstly, factual accuracy of the answers returned by the application is paramount. If the application would return false information and the human agent is unable to detect it, this could potentially cause significant problems. In addition, the application must contain the most up-to-date product information to ensure that its responses are relevant and accurate. This is particularly important as products can change or new products can be added to the portfolio, such as the introduction of new technical guides covering newly developed products. The application must also deliver fast and responsive performance to optimize day-to-day operations and minimize customer waiting

times. Furthermore, the design of the application should prioritize ease of use, matching the skills of the SME's customer support team and minimizing the need for extensive technical expertise. Finally, to promote trust and transparency, the application should be explainable, allowing users to show the source of the information used to answer their questions.

With the application's requirements established, it is evident that a normal LLM cannot be used for the creation of the application. As previously discussed in chapter 3.2, LLMs are prone to issues such as hallucination or outdated information, which could result in generating factually incorrect information. This presents a risk for the application. Therefore, it is crucial to determine which of the knowledge injection methods discussed in chapter 3.3 is the most appropriate to implement in the application, ensuring it best matches the identified requirements.

The method of FT appears to be a promising option. However, despite its ability to enhance performance by customizing the LLM for a specific domain and task, it has notable drawbacks for our SME-focused application. The main challenge lies in its reliance on large, pre-labeled datasets. These datasets are expensive and time-consuming to create, which is a significant barrier for resource-constrained SMEs. Additionally, the FT process itself can be computationally expensive, requiring significant hardware resources that may not be readily available to SMEs. Furthermore, even if these hurdles are overcome, the fine-tuned LLM may still struggle with factual accuracy, especially as the target domain evolves. As FT focuses on adapting the internal parameters of the LLM to the data provided, it may not explicitly learn the underlying factual relationships between concepts. This can result in the LLM generating responses that are consistent with the training data, but factually outdated or incorrect in the real world, as the specific data it was trained on might not reflect the latest information.

In contrast, RAG presents an alternative that addresses the limitations of FT and aligns well with the needs of the SME application. By leveraging an external knowledge base, RAG avoids the need for training the LLM on a domain-specific pre-labeled dataset. This significantly reduces the cost and complexity of building the application. Additionally, RAG mitigates the risk of factual errors by grounding responses in retrieved documents from the knowledge base, which ensures a higher degree of accuracy. RAG offers several other advantages for SMEs. The knowledge base can be easily edited and updated, allowing SMEs to keep pace with changes and incorporate new information as needed. This is particularly beneficial for SMEs operating in fast-moving sectors. Finally, the transparency advantage of RAG is significant. By tracing the LLM's generated answer back to specific documents of the knowledge base, any potential issues that may arise can be identified and resolved. It is important to consider that RAG might have higher inference time compared to fine-tuned models, because of the retrieval step involved. However, for many SME applications, this might be a minor trade-off compared to benefits presented by RAG.

Overall, considering cost constraints and the focus on factual accuracy and explainability, RAG appears to be the better mitigation method for creating the application. It offers a balance between performance and cost-effectiveness while allowing SMEs to maintain control over the knowledge base and ensure reliable outputs.

While RAG offers significant advantages, it is important to consider scenarios where the user's query cannot be effectively answered by documents stored in the knowledge base. Retrievers work by embedding the user's query in a vector representation and calculating its similarity score against the vectorised documents in the knowledge base (Gao et al., 2024). This matching process ensures that an answer is always returned, however, a high similarity score does not guarantee that the retrieved context is truly relevant to the asked question. In such cases, the LLM may not provide an answer at all, or start to hallucinate.

Therefore, a fallback scenario for unanswerable questions needs to be explored. One possible solution could be to integrate a method of contacting a human expert into the application when the LLM reaches its limits. This human expert, be it someone from technical support, product management or R&D, can address the user's query and provide the accurate answer. In addition, the application should continuously learn from such interactions. By storing the user's question and its accurate answer in the knowledge base, the application can enhance its ability to handle similar queries in the future. When a similar question is asked again, the retriever can return the previously stored question and provide the LLM with the corresponding answer as a retrieved document.

4.3. RESEARCH OBJECTIVES

Limited resources often hinder SMEs from implementing Generative AI customer support solutions. Existing applications can be costly, and technical expertise for development or maintenance may be scarce. This research explores this gap by building a cost- efficient customer support application for SMEs using solely free and open-source tools and LLMs. Through a comprehensive experimental design, the main artifact of this research will be the identification of an application that can effectively handle technical questions encountered by the SME's customer support team. Additionally, the research will determine the most suitable open-source LLM for this specific application, evaluate the effectiveness of the RAG mitigation method and explore how feedback from human experts can be integrated to implement a continuous learning approach within the application.

To achieve these goals, this research will be guided by the following two primary research questions:

Research Question 1 (RQ1): To what extent can a cost-efficient customer support application, built with open-source software and consumer-grade hardware, leverage LLMs to effectively answer technical questions?

Research Question 2 (RQ2): How can human expert feedback in a customer support LLM application continuously enhance its performance?

4.4. TECHNOLOGICAL STACK

This chapter explores the key technologies that enable the application to run on consumer hardware. It begins with an outline of the computational resources available to the project, followed by an explanation of the development tools used to build the application, and concludes with a discussion of the LLM selection process.

4.4.1. Computational Resources

Before looking at the development tools used for the application, it is necessary to consider an equally important aspect: the computational resources available for the project. As mentioned earlier, the goal is to build and run the application on inexpensive consumer hardware, which can easily be bought by every SME on the market. The experiments conducted were done with a computer with 16 GB of RAM, an Intel® Core™ i7-11800H CPU, and an NVIDIA® GeForce RTX™ 3060 laptop GPU with 6 GB of VRAM.

4.4.2. Application Development Tools

Several development tools or Python libraries were essential in the development of the application. The first key component of the application is the knowledge base, which serves as the external data source or information storage in the application. This knowledge base is constructed of two parts: a SQL database and a vector database. The SQL database stores from all products the technical specifications, like power and energy consumption, while the vector database handles textual data from PDFs and other text documents. Both databases are made persistent on the machine and are therefore managed locally. To build the SQL database, SQLite is used - a lightweight, disk-based database that does not require a separate server process. It allows access using an easy SQL query language and serves as a fast and user-friendly alternative to other SQL-based databases (Hipp, 2020; Python Software Foundation, 2024). For the vector database, Chroma is used - an open-source embedding database created to simplify the development of LLM applications. Chroma offers an user-friendly interface for storing and managing embeddings plus to perform efficient searches on these embeddings. Similar to the SQLite database, the vector database can also be stored on disk (Chroma, 2024).

Crucial in the construction of the application is a library for running a LLM on a local machine. One such library is LLaMa.CPP. Designed by Georgi Gerganov, it plays a crucial role in providing inference capabilities to a wide range of LLMs, which can be run on consumer hardware (Gerganov, 2024). Its CPU-first C++ library design reduces complexity and ensures an easy integration of LLMs into various programming environments, helping it achieve widespread adoption across multiple platforms. Originally designed as a CPU-only library, LLaMa.CPP also supports GPU frameworks such as NVIDIA's CUDA toolkit, which further reduces the inference time of locally running LLMs. An important aspect of using LLaMa.CPP is the requirement to

use models saved in the GGML/GGUF file format, a custom format created for quantized models (Gerganov, 2024; Hugging Face, 2024a). Quantization is a technique used to reduce computational and memory requirements by representing weights and activations with low-precision data types, such as 8-bit integers, instead of the standard 32-bit floating point (Hugging Face, 2024b). Reducing the number of bits means a decrease in memory storage, less energy consumption and faster operations for the model. While quantized LLMs may experience some loss in performance due to compressing the weights, the impact is softened by the LLM benefitting from in-context learning, by providing the model some degree of context during inference time, therefore the impact of the reduced performance should be minimal (Capelle, 2023). Lastly, to run the models in a Python environment, the library llama-cpp-python is utilized, which offers a high-level API to simplify the integration of LLMs run with LLaMa.CPP with other Python libraries (Andrei, 2024).

The next tool, LangChain, acts as the orchestrator for the components of the application, by integrating the other tools used in this work to construct the final workflow. As one of the most famous toolkits for orchestrating applications based on LLMs, LangChain offers several functionalities that are critical for LLM integration, like formatting and managing LLM input and output and specialized prompt templates. In addition, LangChain allows to connect and retrieve data from external databases such as vector stores and to combine different components to a sequence of different calls. These sequences are called “chains”, which allow the construction of complex workflows and help in a better coordination of LLM interactions (Briggs & Ingham, 2024; Chase, 2022).

Finally, for evaluating the performance of RAG applications, RAGAs (Retrieval-Augmented Generation Assessment) is used (Es et al., 2023). Evaluating RAG applications can be particularly challenging. Taking into consideration the different aspects such as the ability to retrieve relevant context passages from the documents, the LLMs ability to understand them, and to generate an accurate answer to the question requires a thorough approach. RAGAs was designed to specifically address these challenges and allows for evaluation at component level or of the complete application. It provides a set of LLM-assisted evaluation metrics, without the need for human-annotated ground truth data when measuring the performance of RAG applications. Ground truth means here the correct or expected answer to a question (Es et al., 2024).

4.4.3. Model Selection

Primary source for all models selected in this work is Hugging Face, a widely recognised open-source platform within the AI community. Its extensive model hub provides access to a wide collection of open-source models applicable to various NLP tasks (Hugging Face, 2024c).

The selection of the optimal embedding model was a critical step in the development of the application. Two key requirements were important. Firstly, the SME data provided is primarily in German, with occasional English words mixed in, therefore a multilingual embedding model

is needed. Secondly, the execution of the embedding model must be possible on the local machine. Hugging Face provides on their website the Massive Text Embedding Benchmark (MTEB) Leaderboard, which ranks the best multilingual embedding models by an average score (Muennighoff et al., 2023). After evaluating the top five models, the choice fell on the "multilingual-e5-large" (mE5-large) model due to its good performance on the German and English embeddings and the ability to run the model with the computational resources available (Wang et al., 2024).

For identifying the best suitable LLM, several selection criteria were defined to ensure that the LLM meets the requirements of the application:

- **Parameter size** - The limited computational resources require a model with a manageable parameter size (Aráñuez, 2024).
- **Licensing** - The model has a license which permits commercial usage.
- **Availability in GGUF** - The model must be available in GGUF file format to run with LLaMa.CPP.
- **Multilingual capability** - The model should be pre-trained on a English and German dataset or fine-tuned on a German dataset, in order to handle the provided SME data.

Given the abundance of open-source LLMs, the search was narrowed down to focus on the two most prominent models on the market: LLaMa 2 and Mistral. Both models experience high popularity in the open-source community, because of their good performance on several benchmarks. Furthermore both models are available in GGUF file format on the Hugging Face hub.

LLaMa-2, developed by Meta and released in July 2023, is a collection of pre-trained models, which is available in different parameter sizes of 7B, 13B and 70B (Touvron et al., 2023). The LLaMa family is constantly growing and widely adapted by research groups to develop better open-source LLMs, either to compete with closed-source models or to create task or domain-specific LLMs (Minaee et al., 2024). The model is licensed under his own LLaMa 2 license, which allows commercial usage. However, access to the model needs to be requested directly at Meta.

Mistral was launched in September 2023 by French startup Mistral AI, which is a 7B-parameter LLM engineered for superior performance and efficiency (Jiang et al., 2023). Evaluations show that Mistral-7B outperforms LLaMa-2-13B on all benchmarks tested by the Mistral team, and even beats the larger model LLaMa-34B of the first LLaMa generation on reasoning, mathematics and code generation tasks (Minaee et al., 2024). In addition, Mistral is licensed under the Apache 2.0 license, which allows commercial use.

After comparing both models, the choice fell on Mistral-7B for two reasons. Firstly, Mistral-7B outperforms the larger LLaMa-2-13B on all benchmarks and secondly, the available computational resources only allow models with 7B parameters or less to be run, making it

impossible to use LLaMa-2-70B. Mistral AI released two different models, the base pre-trained and an instruction fine-tuned model, both of which were trained exclusively on English data (Jiang et al., 2023). To assess their ability to handle German prompts, tests were conducted with both models. In one test the models were asked a single question in German and the model should return a satisfactory and relevant answer. In the other test, the model is given a RAG-prompt, which includes an instruction, a question and the retrieved context relevant to the question. Here the LLM should generate the correct answer, based on the context provided. The respective prompts for the tests can be found in Appendix A. Unfortunately, in all test runs for both models the responses were either in English or in incorrect German, which was predictable given that the pre-training of both models was in English only. Therefore, further research was carried out to see if German open-source models exist, which are based on the Mistral architecture, specifically which were fine-tuned on a German corpus. The research found three promising models.

LeoLM-Mistral-7B is one of the first open and commercially available German Foundation LMs, which extends Mistral-7B's capabilities by continuing pre-training on a large corpus of German-language and mostly locality-specific text (Plüster & Schuhmann, 2024). Dedicated to German-language tasks, DiscoLM-German-7B is the second chosen model. DiscoLM succeeds another German LLM family, which utilized LeoLM-Mistral-7B as its base model and furthermore the training process involved supervised FT followed by RLHF. According to its creators, DiscoResearch, the model keeps its English fluency while excelling in understanding, generating, and interacting with German language content, resulting in an answer quality that often surpasses GPT-4 (DiscoResearch, 2024). The last model, SauerkrautLM-7b-HerO, was created by merging two other popular Mistral-7B based models, followed by FT using the Sauerkraut dataset, a mixture of an original German corpus and translated data from English to German (VAGO Solutions, 2024).

After the comparison of the three models, it was decided to exclude LeoLM from further testing. This decision was based on two factors. Firstly, DiscoLM's base model is LeoLM, which suggests its superior performance (DiscoResearch, 2024) . Secondly, VAGO Solution provided a detailed evaluation of SauerkrautLM, showing that the model outperformed LeoLM on various benchmarks (VAGO Solutions, 2024). Afterwards, DiscoLM and SauerkrautLM were tested with the same prompts previously used with the Mistral models (Appendix A). The results showed promising results, where both LLMs produced accurate and grammatically correct German responses to single question and RAG prompts. However, there was a clear difference in inference time. DiscoLM showed a noticeable delay in generating responses compared to SauerkrautLM. Moreover, DiscoLM created many trailing blank spaces at each generated response, which made it difficult for the application to further process the LLM output. Attempts to correct this problem by adjusting prompts or DiscoLLM' parameters were unsuccessful. Therefore, based on the test results, SauerkrautLM was chosen as the final LLM for the application.

4.5. DATA COLLECTION & PROCESSING

The initial data for building the knowledge base of the application was provided by a German SME in the lighting sector. The data consists entirely of information on lighting products, such as LED lamps and luminaires and is primarily in German, with English words mixed in. It can be divided in two parts: a spreadsheet that includes structured information about the products such as product names, codes, light output and power consumption and a collection of PDF brochures and installation guides offering more detailed information on specific products or the broader portfolio.

After the exploration of the structured product information, several processing steps are required, which are carried out using the Pandas library (The pandas development team, 2024). First, column names are reviewed and adjusted. Special characters like the German umlauts (Collins English Dictionary, 2024) are removed, and some company-specific terms are replaced with more generic descriptions to improve the LLM's understanding, for example, "PK I" is renamed to "Produktkategorie_PK I", which translates to product category one. Afterwards, missing values are handled: columns with strings indicating True/False values are converted to binary encoding and missing values or other inconsistencies in columns are corrected through value imputation. Finally, columns containing numerical information stored as strings are converted to integer or float for better usability. The preprocessed data is then uploaded to a SQLite database.

The text extraction of the PDF files is done with LangChain. The long documents are loaded and further processed by breaking them down into smaller text chunks. Chunking the documents into smaller, more manageable sections is important, because each LLM has a specific context window, a limit on the amount of text the model can effectively analyze at one time (Briggs & Ingham, 2024). For the chunking LangChain's recursive text splitter is used, which aims to keep related pieces of text next to each other (Chase, 2022). The chunk size is determined using the German Bert Tokenizer from Hugging Face, with a maximum of 300 tokens and an overlap of 20 tokens to ensure contextual continuity (Deepset, 2019). These processed chunks, along with extracted metadata such as author, keywords and source are then embedded with mE5-large, indexed and stored in a Chroma vector store.

4.6. APPLICATION ARCHITECTURE

This chapter describes the constructed application, which is designed to answer technical questions about the SMEs products with the help of a LLM and human experts. The LLM attempts to answer user queries using the context retrieved from the knowledge base, which stores information from various sources. However, if the answer of the LLM is inaccurate, the query is sent to a human expert for review and correction. Crucially, these previous communications with human experts are stored in the knowledge base, allowing the application to learn and improve its performance over time. For a deeper understanding of the application, Figure 1 illustrates the architecture and internal operations that take place

following a user query. Afterwards, each component within the application is explained in detail.

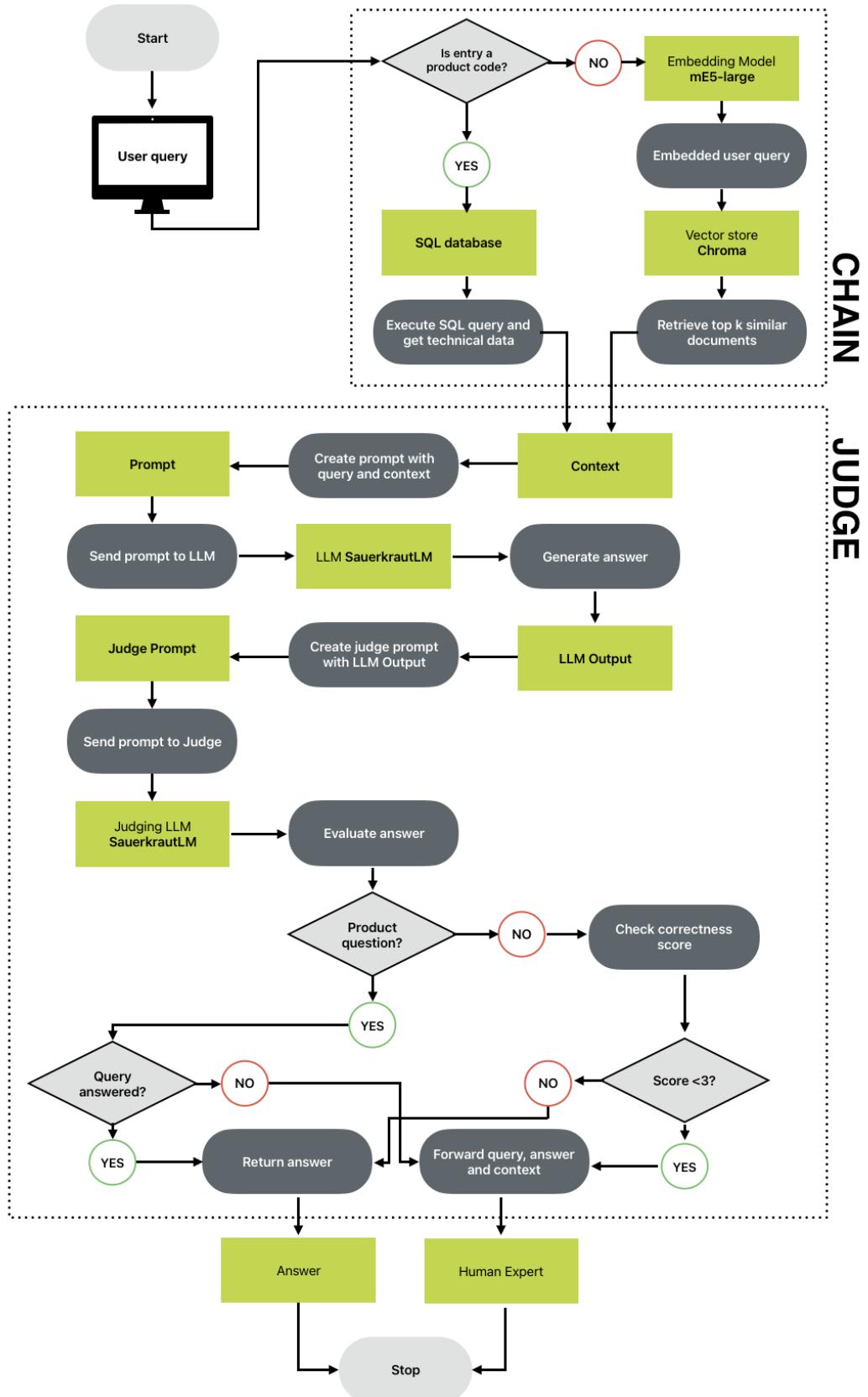


Figure 1 – Internal Flow inside the Application following a User Query

It is important to note that Figure 1 shows a simplified architecture, as showing all the internal operations would be too complex for a single diagram. Furthermore, the product chain and document chain are combined into the box “CHAIN” for clarity. Within the diagram, green boxes represent objects like the LLM or vector store which interact with each other. Dark grey boxes represent these interactions and light grey boxes represent conditional statements. Since the figure focuses on the user's query processing within the application, the step where the human expert's answer is stored in the knowledge base is not shown, as this happens separately from the operations illustrated.

With a foundational understanding of the internal flow of the application from Figure 1, let's look at specific components in more detail. The chatbot component in the application acts as the user interface, accessible through a terminal emulator. It functions as a central communication hub, receiving user requests and directing them to the appropriate components within the application based on the content of the request.

The knowledge base, the applications information storage, resembles the SQL database and vector store created in chapter 4.5. Additionally, a separate message repository is hosted here, which archives all communication between the application and the human experts.

If a user query cannot be answered by the LLM, another application component steps in. The so-called communication handler forwards the user query, the proposed answer from the LLM and the retrieved context to a human expert for clarification. This routing process involves storing all information in a text and JSON file within the knowledge base's message repository. The human expert can then access these text files and provide the correct answer to the question. Afterwards, the handler reads the answer and sends it back to the knowledge base. Here, the expert's answer and the corresponding user query are transformed into a text document, which is embedded with mE5-large and added to the vector store. The newly uploaded text is tagged with the keyword “expert_answer”. By tagging the document, the expert answers can quickly be identified and updated, for example if the correct answer changed or the question-answer pair needs to be deleted. This process is very crucial for the whole application workflow. By storing past interactions, the application builds a richer context for future queries. When a user asks the same or a similar question that the LLM could not answer before, the context retrieved now includes the previous question and the human expert's answer. This enables the LLM to potentially generate the correct answer itself, eliminating the need to involve a human expert again. As a result, this allows the application to learn and improve its performance over time. On a side note, it is important to mention that in a real business environment, the communication handler would possibly be replaced by a standard API and the message repository by an email server. However, due to resource and time constraints, a simplified representation was chosen to speed up the testing and creation of the entire application.

Drawing on LangChain's concept of chains, the application components responsible for the execution of an LLM are also called chains, as this allows different processing stages to be

isolated and coordinated. Three different chains are used within the application: the product chain, the document chain and the judge. Each chain consists of a chain-specific prompt and a LLM for the execution of this prompt. The chains were designed in such a way that the current LLM can be seamlessly replaced by any future LLM with superior performance. The chain-specific prompts were developed through iterative testing. Initial prompts were created and tested with the LLM, and then refined based on the results. This process involved refining the wording, instructions and layout for each prompt, with all tested variations documented in the Appendix.

The product chain is responsible for generating answers to all queries related to a specific product code entered by the user. The prompt for this chain is constructed from chain-specific instructions, the user's product-related question and the context relevant to the entered product code. The retrieval process within this chain involves retrieving all technical information relating to the specified product from the SQL database using a predefined SQL query. This retrieved information forms the context provided to the LLM. Finally, a response schema is included in the prompt that instructs the LLM to generate the response in JSON format. This way further operations in the application can be easier executed. Specific for this chain is one instruction in the response schema: the LLM should include a "solved" name/value pair within the JSON object. This field serves as a self-assessment for the LLM during response generation, showing whether it could answer the question using the provided context (returning "true" for success and "false" for failure). This field is important for the later checking, if the user query should be forwarded to a human expert. The final prompt of the product chain prompt is given in Appendix B.

The document chain addresses more general user queries about the SME's products and therefore needs access to the documents in the vector store. The prompt of the document chain also consists of chain-specific instructions, the user's query and the context related to the query. Here, the context is generated by retrieving the top-k documents from the vector store, which have the highest similarity to the embedded user query. Again, instructing the LLM to generate the response in JSON format is done using a response schema. The final prompt of the document chain prompt is given in Appendix C. Due to the success of the "solved" field in the product chain, the same approach was tested for the document chain, which unfortunately proved to be unreliable. The testing revealed inconsistencies: in some cases, "solved" returned true even though the question could not be answered with the context retrieved, and vice versa, returned false even though the question could be answered with the retrieved context. Therefore, another way had to be found, how to trigger the communication handler to forward the user query if the LLM is unable to generate an accurate answer.

The last chain in the application is the judge. The judge's role is to evaluate the correctness of the answers generated by both the product and document chain and decides, if the user's question needs to be sent to the human expert for revision. For the product chain, a

straightforward evaluation is done by simply checking the “solved” field within the LLM output. If the value is true, the LLM answer is returned to the user and a false value triggers the communication handler to forward the query to a human expert. As mentioned above, this method did not work for the document chain, therefore a new evaluation technique was needed.

One promising solution presented the concept of LLM-as-a-Judge. The main idea of LLM-as-a-Judge is that a separate SOTA LLM is used to judge the results generated by the original LLM (Zheng et al., 2023). One implementation of the LLM-as-a-Judge is presented in a case study conducted by Databricks about best practices for LLM evaluation in RAG applications (Leng et al., 2023). The case study compared different grading scales and prompting techniques to identify the most effective approach for the judge technique. As a start, Leng et al (2023) reused the original prompt from Zheng et al. (2023) and modified it, by adding metrics for correctness, comprehensiveness and readability. In addition, the instructions in the prompt tell the judge LLM to provide a one-line justification before giving each score, with the aim of benefiting from chain-of-thought reasoning. Furthermore, two versions of the prompt are created: a zero-shot version of the prompt which does not provide any example, and the few-shot version of the prompt which provides one example question and correct answer for each score. Ultimately, and based on the findings of their research, Leng et al (2023) recommend a 1-5 grading scale with a zero-shot prompt setup, which shows the best results.

While the approach presented in the case study showed a promising solution, certain adaptations were necessary for the application. The case study uses OpenAI’s models as judges, which is not feasible for this application, because using OpenAI’s APIs would add additional costs and raise concerns about sharing confidential information. In addition, the limited computational resources would not support running a second open-source LLM as a judge. As a consequence, the only viable solution is using SauerkrautLM as the judging LLM in the application, alongside its answer generation tasks in the other chains.

The first setup of the judge chain used SauerkrautLM as the judging LLM with a 1-5 grading scale and a zero-shot prompt that included metrics for correctness, comprehensiveness and readability. Unfortunately, this setup produced no results, because the model could not provide scores for all metrics and no justification. One reason for this failure may be that the initial prompt is too complex for the LLM and therefore, the number of metrics was reduced. The simplified prompt focuses only on the correctness score, as this metric is the most critical one in determining whether the generated response correctly answers the user’s query or whether a human expert is required. Unfortunately, the LLM did not produce satisfactory results for this prompt either.

The next idea in optimizing the prompt was to switch from a zero-shot prompt to a few-shot prompt, hoping that in-context learning would improve performance. The modified prompt provided examples of questions and answers for each score on the 1-5 grading scale. However, even with the examples, the model still struggled to consistently generate correct scores with

justifications for multiple questions in a row. As a last option, it was decided to experiment with a slightly different grading scale from 1-5 to 0-4, to see if a change in scale would improve the performance of the model. Final tests showed that SauerkrautLM performed better with the new scaling as with the previous one. Therefore, the final prompt for the judge chain includes a 0-4 grading scale with a few-shot prompt focused solely on the correctness metric. For reference, Appendix D presents the development process of the judge prompt, including all tested variations and the final prompt.

This prompt setup of the judge allows for reliable scoring of the responses generated by the document chain. Finally, to implement a mechanism for when the communication handler should send a question to a human expert, a correctness score threshold of three was set. Generated answers scoring three or higher are returned to the user, while questions scoring below three are sent to a human expert for further review.

4.7. EVALUATION

Evaluating the effectiveness of the RAG application involved assessing its ability to accurately retrieve and judge responses to technical questions. It should be noted that the evaluation is limited to general technical questions, meaning all questions that are passed to the document chain component. In addition, it was decided to assess the performance of the application against a strong external model. The objective of this comparison is to evaluate how the output of the application compares to the output generated by a leading model on the market. For this comparison, Bing Chat is selected, which is powered by the SOTA model GPT-4 (Microsoft, 2024).

Two requirements must be met for the evaluation of a RAG application: First, the availability of a test dataset with question-answer pairs and their context, and second, an evaluator to measure the accuracy of the application on this dataset (Roucher, 2024).

Creating a high-quality test dataset for RAG applications is often a time-consuming and labour-intensive process. Typically, this involves the manual generation of the dataset by a human agent (Roucher, 2024). However, such a dataset was not provided by the SME for this project. As a solution, a LLM is used to generate a synthetic dataset from the documents provided in the vector store. Hereby, SauerkrautLM was tasked to create 200 factual question-answer pairs based on the documents (the specific prompt can be found in Appendix E). Afterwards 50 questions were selected for further evaluation. It is important to note that all questions and contexts within the created test dataset refer to publicly available technical information related to the SME's products. Therefore, Bing Chat can be used as a competing model in the comparison, with no concern about the leakage of sensitive internal company information to the model.

The evaluator for a RAG application is an LLM, more specifically an LLM-as-a-Judge agent is used to measure the accuracy on the test dataset. Again, SauerkrautLM will act as the judging LLM. To measure the performance of the application, four key metrics provided by the RAGAs

library were used. Two of the metrics, Faithfulness and Answer Relevancy, focus on the generation of the answer given by the LLM itself, while the other two, Answer Correctness and Answer Semantic Similarity, focus on the end-to-end evaluation of the entire RAG application. A detailed explanation of the calculations and further information about the metrics can be found in the RAGAs documentation (Es et al., 2024). RAGAs offers more metrics for evaluating the retrieval component of the RAG application. However, as the Bing Chat is provided with exactly the same context as that created in the application run, these two are not taken into account. Table 2 summarizes all the metrics used in the evaluation with brief definitions. All metrics return scores ranging from 0 to 1, where a higher score indicates a better result.

Table 2 - Definition of RAGAs Metrics (Es et al., 2024)

Metric	Definition
Answer Correctness	Accuracy of an answer compared to the ground truth. A higher score means the generated answer is closer to the ground truth.
Answer Semantic Similarity	Evaluation of how similar the meaning of the generated answer is compared to the ground truth.
Answer Relevancy	Return how relevant the generated answer is to the question. Lower scores are given for incomplete or redundant answers.
Faithfulness	Measures how well the generated answer matches the retrieved context.

The evaluation process itself consisted of several steps. First, the 50 questions from the test dataset were processed through the application, generating answers and retrieving contexts in each run. In addition, the judge chain assigned correctness scores and recorded the judges' decision as to whether the answer generated by the application was correct or whether a human expert needed to be contacted. Second, a list of prompts was created and later sent to Bing Chat. Each prompt is constructed from one of the 50 questions and the corresponding context previously retrieved by the application run. This is necessary because Bing Chat does not have direct access to the knowledge base and therefore cannot retrieve the context on its own. The prompt for Bing Chat can be found in Appendix F. This resulted in two datasets: one for the application results, with the columns question, ground truth, retrieved context, SauerkrautLM answer, correctness score, forwarded to human expert and another for the Bing Chat results with the columns question, ground truth, context, Bing Chat answer. Lastly, both datasets were evaluated, and the metrics scores calculated.

5. DISCUSSION & RESULTS

The evaluation of the application was divided into two parts: the analysis of the human expert contact rate and the model comparison on the RAGAs metrics.

The analysis of the human expert contact rate focused on determining how many of the 50 questions would have been forwarded to human experts because the LLM was unable to answer them with the retrieved context. Figure 2 shows the frequency of correctness scores assigned by the LLM-as-judge agent over the 50 questions. The results show that 7 out of 50 questions would have been sent to human experts because their correctness scores fell below the predefined threshold of 3. However, closer examination of the seven questions with the LLM generated answers and ground truths revealed some errors in the application. For one question, the answer was incomplete, suggesting that the LLM generation process had been interrupted mid-sentence for no known reasons. In another case, the generated answer was matching with the ground truth, yet the judge assigned a correctness score of 1. In addition, for some questions where the judge assigned a score of 0, a human evaluator might have assigned a higher score. These results suggest that the application still assigns incorrect scores or encounters malfunctions.

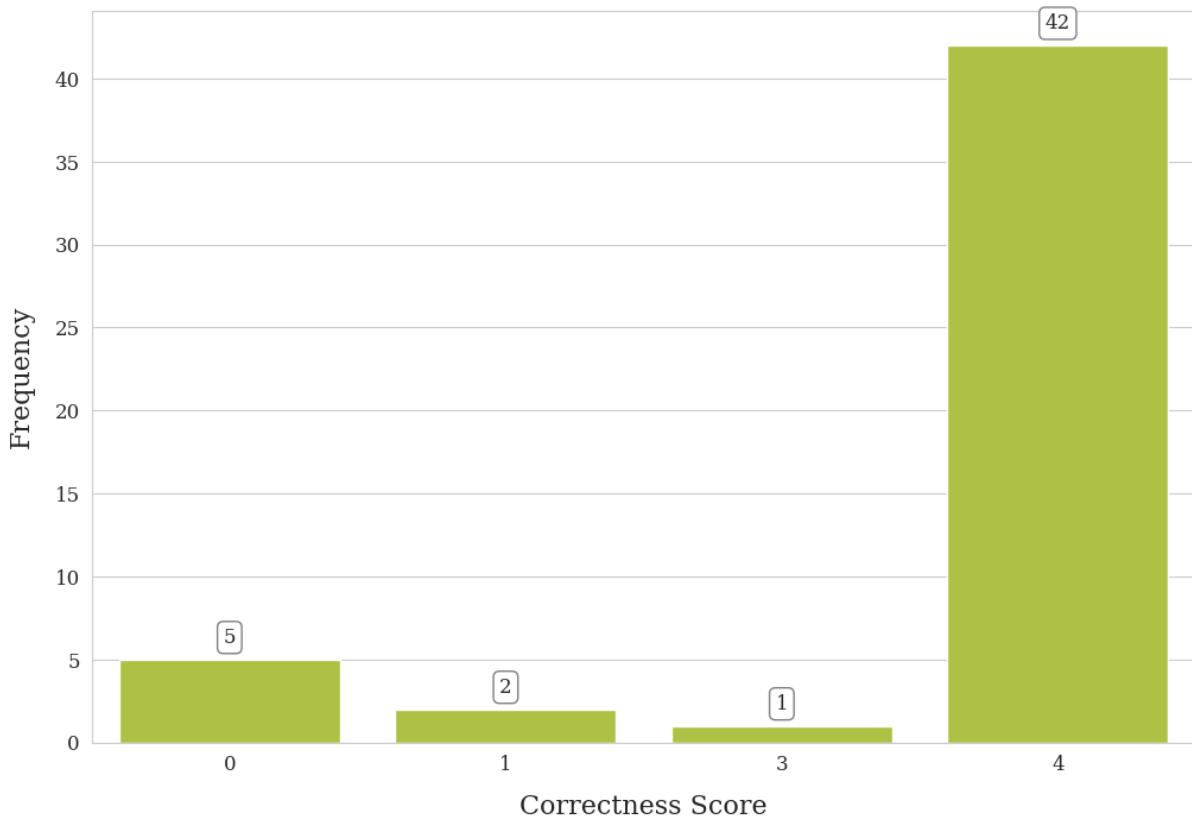


Figure 2 - Total Frequency of Correctness Scores

To ensure a fair comparison, the seven questions identified as problematic within the application were excluded from the mean calculations for the next part of the evaluation.

Figure 3 shows the mean values of all metrics for all 43 remaining questions across both models.

The evaluation showed that both models performed relatively similarly on most metrics, however there are differences. For Answer Correctness, both models showed lower than expected performance, with the application achieving slightly higher scores. Answer Relevance showed the most significant difference between the models, where Bing Chat outperformed the application by 0.11 points, indicating a greater ability to generate answers that directly address user queries. For the remaining metrics, Answer Semantic Similarity and Faithfulness, both models scored above 0.9 points with relatively close results. This suggests that both the application and Bing Chat were generally successful in understanding the relevant context and ensuring that the answers generated by the LLM were faithful to that context.

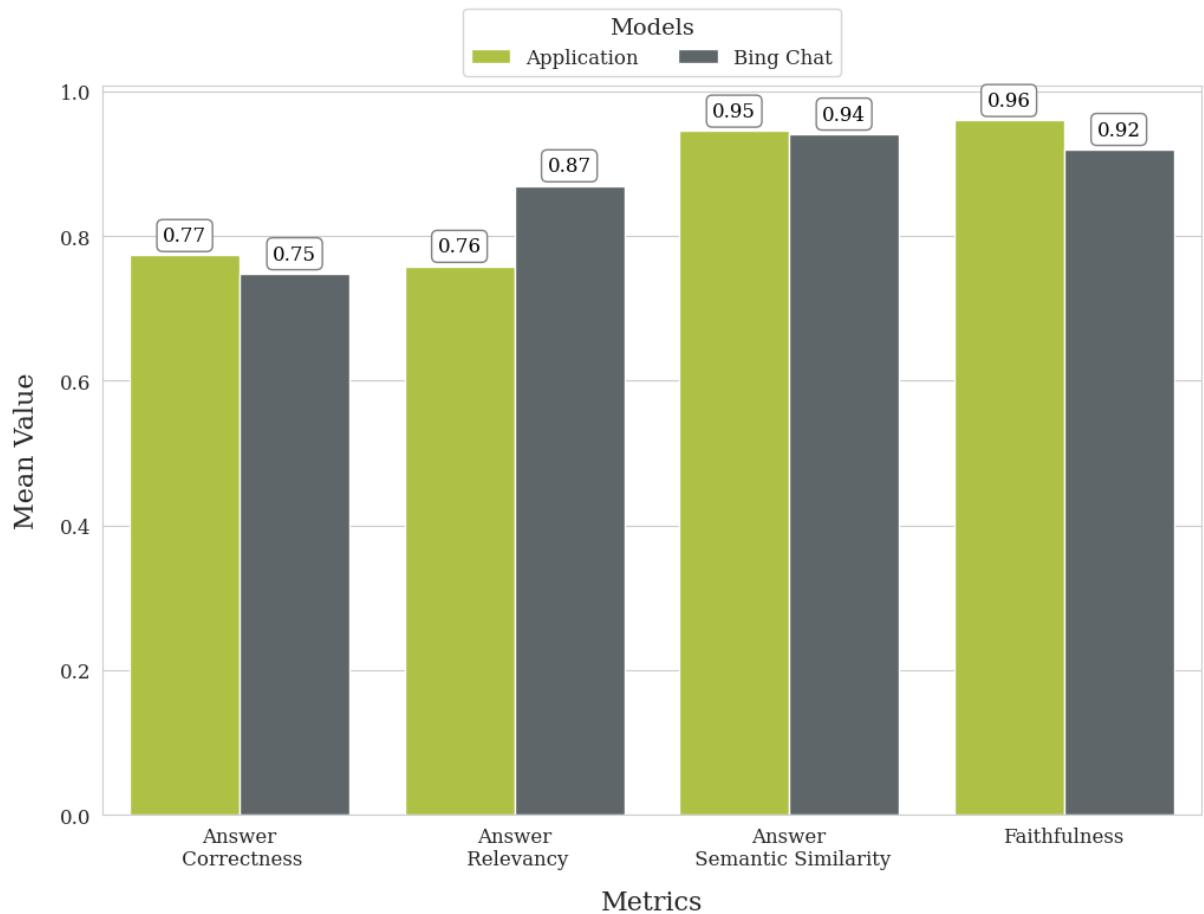


Figure 3 - Mean Value per Metric

The decision to leverage RAG proved to be a suitable approach for building the technical QA application. As expected, RAG aligns well with the key requirements identified in chapter 4.2. First and foremost, the application prioritizes factual accuracy by grounding answers in retrieved documents from the knowledge base. This directly addresses the requirement for highest factual accuracy from the generated answers. In addition, SMEs can easily update the knowledge base, ensuring that the application stays up to date with the changing product

portfolio of the SME. Furthermore, RAG promotes explainability, as the source documents used to answer questions can be traced in the application logs, fulfilling the transparency requirement. The design itself prioritizes ease of use, making it accessible to customer support teams with varying levels of technical expertise. However, a more user-friendly interface that replaces the terminal emulator could further improve usability.

Unfortunately, one key requirement remained unmet: low latency. While RAG offers many benefits, the retrieval step can result in longer processing times than other methods. Testing confirmed this, with product-related questions routed to the product chain component taking an average of 45 seconds to be answered. The document chain showed even higher inference times, with an average of 2 minutes waiting for a response. This big difference in inference time is due to the judge component. When a user query is directed to the document chain, the judging LLM afterwards analyzes the generated answer and assigns a correctness score. Therefore, the entire process involves two times prompting a LLM. In contrast, the product chain only requires verification of the “solved” field in the answer, resulting in a single LLM prompt and faster processing.

The customized implementation of the LLM-as-a-Judge approach has proven effective for handling questions that the LLM cannot answer. These questions are automatically forwarded to human experts for revision. Furthermore, by incorporating the message repository into the knowledge base, a continuous learning method has been integrated into the application, allowing it to access previously reviewed questions and accurate answers. Looking at the evaluation results, it can be seen that the application could reduce the workload of human experts by answering most questions without the need for intervention. However, some problems were identified. The LLM-as-judge agent assigned incorrect correctness scores in some cases, and the LLM answer generation was interrupted mid-sentence at least once. In addition, the judging LLM could wrongly forward acceptable answers to human experts for revision.

When compared to another model, Bing Chat, the application performed similarly on metrics such as understanding context and answer faithfulness, but Bing Chat was significantly better at generating answers that directly addressed the user's query. These findings highlight the need for further refinement within the application, in particular to improve the applications ability to generate answers that are directly relevant to user queries. In addition, addressing potential errors that lead to incomplete answers or incorrect scoring is crucial.

In summary, and addressing the research questions that guided this research, the findings demonstrate that the LLM-based QA application, built solely on open-source software and consumer-grade hardware, is usable for effectively answering technical questions (RQ1). However, the increased inference time due to the document retrieval step makes the application unsuitable for real-world use in its current state, hindering its ability to provide

fast and responsive answers that optimize day-to-day operations and minimize customer wait times.

Furthermore, human expert feedback can continuously enhance the application's performance (RQ2). The implemented method for incorporating this feedback through answer revision and knowledge base updates shows promise for improvement. However, there are limitations in the ability of the judging LLM to consistently assess answer quality.

6. CONCLUSION

This research explored using LLMs to empower SME customer support through a cost-efficient, open-source QA application. Through a comprehensive experimental design, an application was created that can effectively handle technical questions encountered by the SME's customer support team. The application leveraged the RAG approach, prioritizing factual accuracy and explainability of the sources used for answering questions. However, the application's latency is too high for generating answers, which hinders its real-world usability due to slow response times. The implemented method for incorporating human feedback through answer revision showed promise as a continuous learning mechanism, but the LLM-as-judge component requires further development for consistent performance.

Overall, the application has the potential to be a valuable tool for SMEs struggling with AI adoption in their customer support processes. However, further improvements are necessary to unlock its full potential. One potential improvement step could involve increasing the computational resources dedicated to the application, particularly VRAM, to reduce LLM inference time and make it more viable in real-world scenarios. Additionally, by increasing computational resources, a larger and more powerful open-source LLM could be implemented in the application, which could improve answer relevance and correctness —another key area for improvement. Exploring alternative judging approaches, such as the jury-based approach proposed by Verga et al. (2024) where a panel of LLMs evaluates outputs, could further enhance the effectiveness of the judge component.

Lastly, several recommendations can be put forward for SMEs considering the implementation or development of a similar application. Firstly, SMEs should consider deploying the application in the cloud, which provides scalable computing resources tailored to current demands. This allows SMEs to manage changing workloads without the need for investment in hardware. Furthermore, it could be beneficial to adopt a hybrid approach for the application, using a combination of FT and RAG. By finetuning a LLM to company-specific data, the LLM would learn the company's unique writing style and domain-specific knowledge. This could lead to an improved understanding of the documents retrieved from the knowledge base and would result in more accurate answers from the LLM. Finally, it is important to consider the development of a user-friendly interface for the application. Creating an intuitive and easy-to-use interface for customer service agents of all technical skill levels will ensure smooth adoption and maximise the benefits of the application.

BIBLIOGRAPHICAL REFERENCES

- Andrei. (2024). *Abetlen/llama-cpp-python* [Python]. <https://github.com/abetlen/llama-cpp-python>
- Aránguiz, V. N. (2024). *Vokturz/can-it-run-llm*. Hugging Face. <https://huggingface.co/spaces/Vokturz/can-it-run-llm>
- Bahak, H., Taheri, F., Zojaji, Z., & Kazemi, A. (2023). *Evaluating ChatGPT as a Question Answering System: A Comprehensive Analysis and Comparison with Existing Models* (arXiv:2312.07592). arXiv. <http://arxiv.org/abs/2312.07592>
- Briggs, J., & Ingham, F. (2024). Langchain AI Handbook. *Pinecone LangChain AI Handbook*. <https://www.pinecone.io/learn/series/langchain/>
- Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., & Henighan, T. (2020). *Language Models are Few-Shot Learners*.
- Brynjolfsson, E., Li, D., & Raymond, L. (2023). *Generative AI at Work* (arXiv:2304.11771). arXiv. <https://doi.org/10.48550/arXiv.2304.11771>
- Capelle, T. (2023, December 3). How to Run LLMs Locally With llama.cpp and GGML. *W&B Fully Connected*. <https://wandb.ai/capecape/LLMs/reports/How-to-Run-LLMs-Locally-With-llama-cpp-and-GGML--Vmlldzo0Njg5NzMx>
- Casper, S., Davies, X., Shi, C., Gilbert, T. K., Scheurer, J., Rando, J., Freedman, R., Korbak, T., Lindner, D., Freire, P., Wang, T., Marks, S., Segerie, C.-R., Carroll, M., Peng, A., Christoffersen, P., Damani, M., Slocum, S., Anwar, U., ... Hadfield-Menell, D. (2023). *Open Problems and Fundamental Limitations of Reinforcement Learning from Human Feedback* (arXiv:2307.15217). arXiv. <https://doi.org/10.48550/arXiv.2307.15217>
- Chang, Y., Wang, X., Wang, J., Wu, Y., Yang, L., Zhu, K., Chen, H., Yi, X., Wang, C., Wang, Y., Ye, W., Zhang, Y., Chang, Y., Yu, P. S., Yang, Q., & Xie, X. (2023). *A Survey on Evaluation of Large Language Models*. <https://doi.org/10.48550/ARXIV.2307.03109>
- Chase, H. (2022). *LangChain* [Python]. <https://github.com/langchain-ai/langchain>
- Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., Schuh, P., Shi, K., Tsvyashchenko, S., Maynez, J., Rao, A., Barnes, P., Tay, Y., Shazeer, N., Prabhakaran, V., ... Fiedel, N. (2022). *PaLM: Scaling Language Modeling with Pathways* (arXiv:2204.02311). arXiv. <http://arxiv.org/abs/2204.02311>
- Chroma. (2024). *Chroma-core/chroma* [Python]. Chroma. <https://github.com/chroma-core/chroma>
- Collins English Dictionary. (2024, June 6). *UMLAUT definition and meaning* [Collins English Dictionary]. <https://www.collinsdictionary.com/dictionary/english/umlaut>
- Das, A. C., Gomes, M., Patidar, I. L., Phalin, G., Sawhney, R., & Thomas, R. (2023, March 27). *AI customer service for higher customer engagement / McKinsey*. McKinsey & Company. <https://www.mckinsey.com/capabilities/operations/our-insights/the-next-frontier-of-customer-engagement-ai-enabled-customer-service>

- Deepset. (2019, June 14). *Google-bert/bert-base-german-cased*. Hugging Face.
<https://huggingface.co/google-bert/bert-base-german-cased>
- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding* (arXiv:1810.04805). arXiv.
<http://arxiv.org/abs/1810.04805>
- DiscoResearch. (2024). *DiscoResearch/DiscoLM_German_7b_v1* (Version v1) [Computer software]. DiscoResearch.
https://huggingface.co/DiscoResearch/DiscoLM_German_7b_v1
- Dong, Q., Li, L., Dai, D., Zheng, C., Wu, Z., Chang, B., Sun, X., Xu, J., Li, L., & Sui, Z. (2023). *A Survey on In-context Learning* (arXiv:2301.00234). arXiv.
<http://arxiv.org/abs/2301.00234>
- Drenik, G. (2023, December 6). *How AI Is Revolutionizing Customer Service With Human-Like Responses*. Forbes. <https://www.forbes.com/sites/garydrenik/2023/12/06/how-ai-is-revolutionizing-customer-service-with-human-like-responses/>
- Es, S., James, J., Espinosa-Anke, L., & Schockaert, S. (2023). *RAGAS: Automated Evaluation of Retrieval Augmented Generation* (arXiv:2309.15217). arXiv.
<https://doi.org/10.48550/arXiv.2309.15217>
- Es, S., James, J., Espinosa-Anke, L., & Schockaert, S. (2024). *Explodinggradients/ragas* [Python]. Exploding Gradients. <https://github.com/explodinggradients/ragas>
- European Economic and Social Committee., SpaceTec Partners., & UnternehmerTUM. (2021). *Boosting the use of artificial intelligence in Europe's micro, small and medium-sized enterprises*. Publications Office. <https://data.europa.eu/doi/10.2864/08775>
- Farea, A., Yang, Z., Duong, K., Perera, N., & Emmert-Streib, F. (2022). *Evaluation of Question Answering Systems: Complexity of judging a natural language* (arXiv:2209.12617). arXiv. <https://doi.org/10.48550/arXiv.2209.12617>
- Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Guo, Q., Wang, M., & Wang, H. (2024). *Retrieval-Augmented Generation for Large Language Models: A Survey* (arXiv:2312.10997). arXiv. <https://doi.org/10.48550/arXiv.2312.10997>
- Gartner. (2023a, August 30). *Gartner Reveals Three Technologies That Will Transform Customer Service and Support By 2028*. Gartner.
<https://www.gartner.com/en/newsroom/press-releases/2023-08-30-gartner-reveals-three-technologies-that-will-transform-customer-service-and-support-by-2028>
- Gartner. (2023b). *The Future of Customer Service*. Gartner.
- Gerganov, G. (2024). *Ggerganov/llama.cpp* [C++].
<https://github.com/ggerganov/llama.cpp?tab=readme-ov-file>
- Gillioz, A., Casas, J., Mugellini, E., & Khaled, O. A. (2020). *Overview of the Transformer-based Models for NLP Tasks*. 179–183. <https://doi.org/10.15439/2020F20>
- Guo, Z., Jin, R., Liu, C., Huang, Y., Shi, D., Supryadi, Yu, L., Liu, Y., Li, J., Xiong, B., & Xiong, D. (2023). *Evaluating Large Language Models: A Comprehensive Survey* (arXiv:2310.19736). arXiv. <https://doi.org/10.48550/arXiv.2310.19736>
- Hipp, R. D. (2020). *SQLite*. SQLite. <https://www.sqlite.org/about.html>

- Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., Chen, Q., Peng, W., Feng, X., Qin, B., & Liu, T. (2023). *A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions* (arXiv:2311.05232). arXiv. <https://doi.org/10.48550/arXiv.2311.05232>
- Hugging Face. (2024a). *GGUF - Hugging Face Documentation*. Hugging Face. <https://huggingface.co/docs/hub/en/gguf>
- Hugging Face. (2024b). *Quantization—Hugging Face Documentation*. Hugging Face. https://huggingface.co/docs/optimum/en/concept_guides/quantization
- Hugging Face. (2024c, April 24). *Hugging Face – The AI community building the future*. <https://huggingface.co/>
- Ishwari, K. S. D., Aneeze, A. K. R. R., Sudheesan, S., Karunaratne, H. J. D. A., Nugaliyadde, A., & Mallawarrachchi, Y. (2019). *Advances in Natural Language Question Answering: A Review* (arXiv:1904.05276). arXiv. <https://doi.org/10.48550/arXiv.1904.05276>
- Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., Casas, D. de las, Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L. R., Lachaux, M.-A., Stock, P., Scao, T. L., Lavril, T., Wang, T., Lacroix, T., & Sayed, W. E. (2023). *Mistral 7B*. <https://arxiv.org/abs/2310.06825v1>
- Jurafsky, D., & Martin, J. H. (2023). *Speech and Language Processing* (3rd ed.). <https://web.stanford.edu/~jurafsky/slp3/>
- Kaddour, J., Harris, J., Mozes, M., Bradley, H., Raileanu, R., & McHardy, R. (2023). *Challenges and Applications of Large Language Models* (arXiv:2307.10169). arXiv. <https://doi.org/10.48550/arXiv.2307.10169>
- Kalyan, K. S. (2023). *A Survey of GPT-3 Family Large Language Models Including ChatGPT and GPT-4* (arXiv:2310.12321). arXiv. <http://arxiv.org/abs/2310.12321>
- Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., & Iwasawa, Y. (2023). *Large Language Models are Zero-Shot Reasoners* (arXiv:2205.11916). arXiv. <http://arxiv.org/abs/2205.11916>
- Lambert, N., von Werra, L., Havrilla, A., & Castricato, L. (2022, December 9). Illustrating Reinforcement Learning from Human Feedback (RLHF). *Hugging Face Blog*. <https://huggingface.co/blog/rhf>
- Leng, Q., Uhlenhuth, K., & Polyzotis, A. (2023, September 12). Best Practices for LLM Evaluation of RAG Applications. *Databricks Engineering Blog*. <https://www.databricks.com/blog/LLM-auto-eval-best-practices-RAG>
- Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., & Kiela, D. (2021). *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks* (arXiv:2005.11401). arXiv. <http://arxiv.org/abs/2005.11401>
- Liang, P., Bommasani, R., Lee, T., Tsipras, D., Soylu, D., Yasunaga, M., Zhang, Y., Narayanan, D., Wu, Y., Kumar, A., Newman, B., Yuan, B., Yan, B., Zhang, C., Cosgrove, C., Manning, C. D., Ré, C., Acosta-Navas, D., Hudson, D. A., ... Koreeda, Y. (2023). *Holistic Evaluation of Language Models* (arXiv:2211.09110). arXiv. <https://doi.org/10.48550/arXiv.2211.09110>

- Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., & Neubig, G. (2021). *Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing* (arXiv:2107.13586). arXiv. <http://arxiv.org/abs/2107.13586>
- Luo, H., & Specia, L. (2024). *From Understanding to Utilization: A Survey on Explainability for Large Language Models* (arXiv:2401.12874). arXiv. <http://arxiv.org/abs/2401.12874>
- Microsoft. (2024). *Bing Chat [GPT-4 language model]*. <https://www.bing.com/chat>
- Minaee, S., Mikolov, T., Nikzad, N., Chenaghlou, M., Socher, R., Amatriain, X., & Gao, J. (2024). *Large Language Models: A Survey* (arXiv:2402.06196). arXiv. <http://arxiv.org/abs/2402.06196>
- Morgan, B. (2023, August 16). *What Impact Will AI Have On Customer Service?* Forbes. <https://www.forbes.com/sites/blakemorgan/2023/08/16/what-impact-will-ai-have-on-customer-service/>
- Muennighoff, N., Tazi, N., Magne, L., & Reimers, N. (2023). *MTEB: Massive Text Embedding Benchmark* (arXiv:2210.07316). arXiv. <http://arxiv.org/abs/2210.07316>
- OpenAI. (2023). *GPT-4 Technical Report* (arXiv:2303.08774). arXiv. <http://arxiv.org/abs/2303.08774>
- Ovadia, O., Brief, M., Mishaeli, M., & Elisha, O. (2023). *Fine-Tuning or Retrieval? Comparing Knowledge Injection in LLMs* (arXiv:2312.05934). arXiv. <https://doi.org/10.48550/arXiv.2312.05934>
- Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, 24(3), 45–77. <https://doi.org/10.2753/MIS0742-1222240302>
- Petroni, F., Rocktäschel, T., Lewis, P., Bakhtin, A., Wu, Y., Miller, A. H., & Riedel, S. (2019). *Language Models as Knowledge Bases?* (arXiv:1909.01066). arXiv. <https://doi.org/10.48550/arXiv.1909.01066>
- Plüster, B., & Schuhmann, C. (2024). *LeoLM/leo-mistral-hessianai-7b* [Computer software]. LeoLM. <https://huggingface.co/LeoLM/leo-mistral-hessianai-7b>
- Podolsky, M. (2023, May 12). *Council Post: The AI Revolution In Customer Service: What Do We Do Next?* Forbes. <https://www.forbes.com/sites/forbesbusinesscouncil/2023/05/12/the-ai-revolution-in-customer-service-what-do-we-do-next/>
- Python Software Foundation. (2024). *sqlite3—Python documentation*. Python Documentation. <https://docs.python.org/3/library/sqlite3.html>
- Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). *Language Models are Unsupervised Multitask Learners*.
- Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., & Liu, P. J. (2023). *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer* (arXiv:1910.10683). arXiv. <https://doi.org/10.48550/arXiv.1910.10683>
- Rawte, V., Sheth, A., & Das, A. (2023). *A Survey of Hallucination in Large Foundation Models* (arXiv:2309.05922). arXiv. <https://doi.org/10.48550/arXiv.2309.05922>

- Roberts, A., Raffel, C., & Shazeer, N. (2020). *How Much Knowledge Can You Pack Into the Parameters of a Language Model?* (arXiv:2002.08910). arXiv.
<http://arxiv.org/abs/2002.08910>
- Roucher, A. (2024). RAG Evaluation. *Hugging Face Open-Source AI Cookbook*.
https://huggingface.co/learn/cookbook/en/rag_evaluation
- Shen, J. (2023, October 2). Evolution of Question Answering Technology: A six-decade Journey. *Ambit Blog*. <https://www.ambit.ai/resources/evolution-of-question-answering-technology-a-six-decade-journey>
- Tan, Y., Min, D., Li, Y., Li, W., Hu, N., Chen, Y., & Qi, G. (2023). *Can ChatGPT Replace Traditional KBQA Models? An In-depth Analysis of the Question Answering Performance of the GPT LLM Family* (arXiv:2303.07992). arXiv.
<http://arxiv.org/abs/2303.07992>
- The pandas development team. (2024). *pandas-dev/pandas: Pandas* (v2.2.2) [Computer software]. Zenodo. <https://doi.org/10.5281/ZENODO.3509134>
- Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., & Lample, G. (2023). *LLaMA: Open and Efficient Foundation Language Models* (arXiv:2302.13971). arXiv.
<http://arxiv.org/abs/2302.13971>
- VAGO Solutions. (2024). *VAGOsolutions/SauerkrautLM-7b-HerO* [Computer software]. VAGO Solutions. <https://huggingface.co/VAGOsolutions/SauerkrautLM-7b-HerO>
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2023). *Attention Is All You Need* (arXiv:1706.03762). arXiv.
<http://arxiv.org/abs/1706.03762>
- Verga, P., Hofstatter, S., Althammer, S., Su, Y., Piktus, A., Arkhangorodsky, A., Xu, M., White, N., & Lewis, P. (2024). *Replacing Judges with Juries: Evaluating LLM Generations with a Panel of Diverse Models* (arXiv:2404.18796). arXiv.
<https://doi.org/10.48550/arXiv.2404.18796>
- Wall Street Journal. (2023, July 13). How AI Is Reshaping Customer Service. *Q&A*.
<https://partners.wsj.com/salesforce/how-generative-ai-will-revolutionize-customer-service/>
- Wang, L., Yang, N., Huang, X., Yang, L., Majumder, R., & Wei, F. (2024). *Multilingual E5 Text Embeddings: A Technical Report* (arXiv:2402.05672). arXiv.
<http://arxiv.org/abs/2402.05672>
- Wei, J., Bosma, M., Zhao, V. Y., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., & Le, Q. V. (2022). *Finetuned Language Models Are Zero-Shot Learners* (arXiv:2109.01652). arXiv.
<http://arxiv.org/abs/2109.01652>
- Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., & Zhou, D. (2023). *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models* (arXiv:2201.11903). arXiv. <http://arxiv.org/abs/2201.11903>
- Zhang, N., Yao, Y., Tian, B., Wang, P., Deng, S., Wang, M., Xi, Z., Mao, S., Zhang, J., Ni, Y., Cheng, S., Xu, Z., Xu, X., Gu, J.-C., Jiang, Y., Xie, P., Huang, F., Liang, L., Zhang, Z., ...

- Chen, H. (2024). *A Comprehensive Study of Knowledge Editing for Large Language Models* (arXiv:2401.01286). arXiv. <http://arxiv.org/abs/2401.01286>
- Zhang, S., Dong, L., Li, X., Zhang, S., Sun, X., Wang, S., Li, J., Hu, R., Zhang, T., Wu, F., & Wang, G. (2023). *Instruction Tuning for Large Language Models: A Survey* (arXiv:2308.10792). arXiv. <http://arxiv.org/abs/2308.10792>
- Zhang, Y., Li, Y., Cui, L., Cai, D., Liu, L., Fu, T., Huang, X., Zhao, E., Zhang, Y., Chen, Y., Wang, L., Luu, A. T., Bi, W., Shi, F., & Shi, S. (2023). *Siren's Song in the AI Ocean: A Survey on Hallucination in Large Language Models* (arXiv:2309.01219). arXiv. <http://arxiv.org/abs/2309.01219>
- Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., Du, Y., Yang, C., Chen, Y., Chen, Z., Jiang, J., Ren, R., Li, Y., Tang, X., Liu, Z., ... Wen, J.-R. (2023). *A Survey of Large Language Models* (arXiv:2303.18223). arXiv. <https://doi.org/10.48550/arXiv.2303.18223>
- Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E. P., Zhang, H., Gonzalez, J. E., & Stoica, I. (2023). *Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena* (arXiv:2306.05685). arXiv. <http://arxiv.org/abs/2306.05685>

APPENDIX A – MODEL SELECTION PROMPTS

Prompts for testing capabilities of model Mistral-7B

Single Question Prompts

[INST]

You are a friendly assistant that helps users with their questions.

Question: Erkläre mir das Prinzip von LED-Lampen.

[/INST]

[INST]

You are a friendly assistant that helps users with their questions.

Question: Wer ist der Bundeskanzler von Deutschland und was sind seine Aufgaben?

[/INST]

[INST]

You are a friendly assistant that helps users with their questions.

Question: Erzähle mir etwas über Deutschland.

[/INST]

RAG Prompt

[INST]

You are an assistant for question-answering tasks. Use the following pieces of retrieved context to answer the question.

Question: Wie funktioniert eine LED?

Context: Wie bereits aufgezeigt, befinden sich auf dem Halbleiterkristall zwei Schichten von Halbleitermaterialien. Eine n-dotierte Schicht besitzt einen Überschuss an Elektronen, eine zweite, dünne p-dotierte Schicht verfügt über zu viele Defektelelektronen, auch Elektronenlöcher genannt. Wird über die beiden Schichten eine Spannung in Flussrichtung der LED angelegt, wandern die überschüssigen Elektronen in Richtung p-Schicht. In der sogenannten Sperrsicht treffen sie auf die Elektronenlöcher. Dort beginnen sich die ausgewanderten Elektronen und die Elektronenlöcher zu rekombinieren. Das bedeutet: Beide Ladungsträger vereinigen sich. Somit stellt die Rekombination den Umkehrprozess der Ionisation dar. Im Gegensatz zur Ionisation, die Energie benötigt, wird bei der Rekombination Energie frei, die in Form von Lichtblitzen (Photonen) über die dünne p-Schicht abgegeben wird. Der LED-Chip gibt diese Lichtblitze nach außen weiter, wobei die Innenseiten der Reflektorwanne das austretende Licht verstärken. Dieser Vorgang läuft kontinuierlich weiter, solange über die Spannungsquelle freie Elektronen zugeführt werden. Neben den Lichtblitzen entsteht während der Rekombination außerdem eine kleine Menge an Abwärme.

[/INST]

Prompts for testing capabilities of DiscoLM and SauerkrautLM

Single Question Prompts

```
<|im_start|>system  
You are a friendly assistant that helps users with their questions.  
<|im_end|>  
<|im_start|>user  
Erkläre mir das Prinzip von LED-Lampen.<|im_end|>  
<|im_start|>assistant
```

```
<|im_start|>system  
You are a friendly assistant that helps users with their questions.  
<|im_end|>  
<|im_start|>user  
Wer ist der Bundeskanzler von Deutschland und was sind seine  
Aufgaben?<|im_end|>  
<|im_start|>assistant
```

```
<|im_start|>system  
You are a friendly assistant that helps users with their questions.  
<|im_end|>  
<|im_start|>user  
Erzähle mir etwas über Deutschland.<|im_end|>  
<|im_start|>assistant
```

RAG Prompt

```
<|im_start|>system  
You are an assistant for question-answering tasks. Use the following pieces  
of retrieved context to answer the question.
```

Context: Wie bereits aufgezeigt, befinden sich auf dem Halbleiterkristall zwei Schichten von Halbleitermaterialien. Eine n-dotierte Schicht besitzt einen Überschuss an Elektronen, eine zweite, dünnerne p-dotierte Schicht verfügt über zu viele Defektelektronen, auch Elektronenlöcher genannt. Wird über die beiden Schichten eine Spannung in Flussrichtung der LED angelegt, wandern die überschüssigen Elektronen in Richtung p-Schicht. In der sogenannten Sperrsicht treffen sie auf die Elektronenlöcher. Dort beginnen sich die ausgewanderten Elektronen und die Elektronenlöcher zu rekombinieren. Das bedeutet: Beide Ladungsträger vereinigen sich. Somit stellt die Rekombination den Umkehrprozess der Ionisation dar. Im Gegensatz zur Ionisation, die Energie benötigt, wird bei der Rekombination Energie frei, die in Form von Lichtblitzen (Photonen) über die dünne p-Schicht abgegeben wird. Der LED-Chip gibt diese Lichtblitze nach außen weiter, wobei die Innenseiten der Reflektorwanne das austretende Licht verstärken. Dieser Vorgang läuft kontinuierlich weiter, solange über die Spannungsquelle freie Elektronen zugeführt werden. Neben den Lichtblitzen entsteht während der Rekombination außerdem eine kleine Menge an Abwärme.

```
<|im_end|>  
<|im_start|>user  
Question: Wie funktioniert eine LED?<|im_end|>  
<|im_start|>assistant
```

APPENDIX B – PROMPT ITERATION PRODUCT CHAIN

In the following all iterations of the product chain prompt are presented. Words enclosed in curly brackets {} represent placeholders for the user's query and the retrieved context, that will be inserted into the prompt at runtime of the application.

V1 – Prompt in German

```
<|im_start|>system  
Als Experte für Lampenprodukte wirst du Fragen zu bestimmten Produkten  
beantworten. Nutze die unten stehenden Produktinformationen, um Fragen zu  
beantworten. Achte auf die Spaltennamen in der Produktinformation.  
Wenn du die Frage nicht beantworten kannst, gebe dies an und erfinde keine  
Antworten. Gib deine Antwort als JSON Objekt zurück.  
  
Produktinformationen:  
{context}  
  
<|im_end|>  
<|im_start|>user  
Frage: {question} <|im_end|>  
<|im_start|>assistant
```

V2 – Prompt in English

```
<|im_start|>system  
As a lamp product expert, you will answer questions about specific  
products. Utilize the product information provided below to answer  
questions. Pay attention to the column names in the product information.  
If you cannot answer the question, indicate this and do not invent answers.  
Return your answer as a JSON object.  
  
Product information:  
{context}  
  
<|im_end|>  
<|im_start|>user  
Question: {question} <|im_end|>  
<|im_start|>assistant
```

V3 – Prompt with response schema

```
<|im_start|>system  
As a lamp product expert, you will answer questions about specific  
products. Utilize the product information provided below to answer  
questions. Pay attention to the column names in the product information.  
If you cannot answer the question, indicate this and do not invent answers.
```

Use the following format for the response:

The output should be a markdown code snippet formatted in the following
schema, including the leading and trailing "```json" and "```":

```
```json  
{
 "question": string // Question of the user
 "answer": string // Answer to the question
}
``
```

Product information:

{*context*}

```
<|im_end|>
<|im_start|>user
Question: {question} <|im_end|>
<|im_start|>assistant
```

## V4 – Prompt with additional instructions

```
<|im_start|>system
As a lamp product expert, you will answer questions about specific
products. Utilize the product information provided below to answer
questions. Pay attention to the column names in the product information.
If the user requests a datasheet and there are uncertainties, provide
either the "eCAT Datenblatt" or the "EU-Verordnung-Produktdatenblatt". When
the user asks for an energy efficiency label, they are referring to the EEL
label. If the user asks for the designation or the name, always provide the
long designation (Bezeichnung_lang)
The product information is stored in a Python dictionary, where the 'key'
represents the column and the 'value' of the dictionary object represents
the product information. For example, in the pair key = 'Leistung' and
value = '12W', 'Leistung' represents the column and '12W' represents the
information.
Before providing an answer, review the product information to ensure you
have a column related to the question. If you cannot answer the question,
indicate this and do not invent answers.
```

Use the following format for the response:

The output should be a markdown code snippet formatted in the following
schema, including the leading and trailing ````json" and `````:

```
```json
{
    "question": string // Question of the user
    "answer": string // Answer to the question
}..
````
```

Product information:

{**context**}

```
<|im_end|>
<|im_start|>user
Question: {question} <|im_end|>
<|im_start|>assistant
```

## V5 – Final prompt with ‘solved’ field

```
<|im_start|>system
As a lamp product expert, you will answer questions about specific products. Utilize the product information provided below to answer questions. Pay attention to the column names in the product information. If the user requests a datasheet and there are uncertainties, provide either the "eCAT Datenblatt" or the "EU-Verordnung-Produktdatenblatt". When the user asks for an energy efficiency label, they are referring to the EEL label. If the user asks for the designation or the name, always provide the long designation (Bezeichnung_lang)
```

The product information is stored in a Python dictionary, where the 'key' represents the column and the 'value' of the dictionary object represents the product information. For example, in the pair key = 'Leistung' and value = '12W', 'Leistung' represents the column and '12W' represents the information.

Before providing an answer, review the product information to ensure you have a column related to the question. If you cannot answer the question, indicate this and do not invent answers.

Product information:

{context}

Use the following format for the response:

The output should be a markdown code snippet formatted in the following schema, including the leading and trailing ````json" and `````:

```
```json  
{  
    "question": string // Question of the user  
    "answer": string // Answer to the question  
    "solved": string // Could the model answer the question with given  
product information? True if yes, otherwise False  
}  
..  
  
<|im_end|>  
<|im_start|>user  
Question: {question} <|im_end|>  
<|im_start|>assistant
```

APPENDIX C – PROMPT ITERATION DOCUMENT CHAIN

In the following all iterations of the document chain prompt are presented. Words enclosed in curly brackets {} represent placeholders for the user's question and the retrieved context, that will be inserted into the prompt at runtime of the application.

V1 – Prompt with response schema

```
<|im_start|>system  
You are an expert assistant specializing in answering questions about  
lamps. Your goal is to provide accurate, concise, and helpful responses  
based on the given context.
```

Format your response according to the provided schema: The output should be a markdown code snippet formatted in the following schema, including the leading and trailing `` `json` and `` `:

```
```json  
{
 "answer": string // Answer to the question
}
...
```

Context:  
**{context}**

```
<|im_end|>
<|im_start|>user
Question: {question} <|im_end|>
<|im_start|>assistant
```

## V2 – Prompt with additional instructions

```
<|im_start|>system
You are an expert assistant specializing in answering questions about
lamps. Your goal is to provide accurate, concise, and helpful responses
based on the given context.
Keep your answer brief and to the point, limiting it to a maximum of 3
sentences.
If you cannot answer the question based on the given context, inform the
user rather than inventing an answer.
If the context is incomplete or irrelevant, politely inform the user that
the provided information is insufficient to answer their question.

Format your response according to the provided schema: The output should be
a markdown code snippet formatted in the following schema, including the
leading and trailing ````json" and `````:

```json
{
    "answer": string // Answer to the question
}..
Context:
{context}

<|im_end|>
<|im_start|>user
Question: {question} <|im_end|>
<|im_start|>assistant
```

V3 – Final prompt with improved instructions

```
|im_start|>system
You are an expert assistant specializing in answering questions about
lamps. Your goal is to provide accurate, concise, and helpful responses
based on the given context. Follow these guidelines when answering the
user's question:

    - Review the provided context carefully to ensure you have information
      related to the question.
    - Keep your answer brief and to the point, limiting it to a maximum of
      3 sentences.
    - If you cannot answer the question based on the given context, inform
      the user rather than inventing an answer.
    - If the context is incomplete or irrelevant, politely inform the user
      that the provided information is insufficient to answer their question.
```

Format your response according to the provided schema: The output should be a markdown code snippet formatted in the following schema, including the leading and trailing ````json" and `````:

```
```json
{
 "answer": string // Answer to the question
}..
````
```

Context:
{context}

```
<|im_end|>
<|im_start|>user
Question: {question} <|im_end|>
<|im_start|>assistant
```

APPENDIX D – PROMPT ITERATION JUDGE

In the following all iterations of the judge prompt are presented. Words enclosed in curly brackets {} represent placeholders for the user's question, the retrieved context and the generated answer from the LLM, that will be inserted into the prompt at runtime of the application.

V1 – Zero-shot prompt with 1-5 grading scale

```
<|im_start|>system
Please act as an impartial judge and evaluate the quality of the provided answer which
attempts to answer the provided question based on a provided context.
You'll be given a function grading_function which you'll call for each provided context,
question and answer to submit your reasoning and score for the correctness of the
answer.
Please make sure you always call the function to submit result.

- Correctness: If the answer correctly answer the question, below are the details for
different scores:
  - Score 1: the answer is completely incorrect, doesn't mention anything about the
question or is completely contrary to the correct answer.
  - Score 2: the answer provides some relevance to the question and answer one aspect of
the question correctly.
  - Score 4: the answer mostly answer the question but is missing or hallucinating on
one critical aspect.
  - Score 5: the answer correctly answer the question and not missing any major aspect

Use the following function for the response:
{
  "name": "grading_function",
  "description": "Call this function to submit the grading for the answer",
  "parameters": {
    "type": "object",
    "properties": {
      "reasoning_for_correctness": {
        "type": "string",
        "description": "Your reasoning for giving the grading for the
correctness of the answer. Provide explanation.",},
      "correctness": {
        "type": "integer",
        "description": "Your integer grading between 0 to 4 for the correctness
of the answer.",},
    },
    "required": [
      "reasoning_for_correctness",
      "correctness",],
  },
}

<|im_start|>user
Provided question:
{question}

Provided answer:
{answer}

Provided context:
{context} <|im_end|>
<|im_start|>assistant
```

V2 – Few-shot prompt with 1-5 grading scale

```
<|im_start|>system
Please act as an impartial judge and evaluate the quality of the provided answer which attempts to
answer the provided question based on a provided context.
You'll be given a function grading_function which you'll call for each provided context, question
and answer to submit your reasoning and score for the correctness of the answer.
Please make sure you always call the function to submit result.

- Correctness: If the answer correctly answer the question, below are the details for different
scores:
  - Score 1: the answer is completely incorrect, doesn't mention anything about the question or is
completely contrary to the correct answer.
    - For example, when asked "Welches sind die Vorteile der SME LED-Beleuchtung?", the answer is
empty string, or content that's completely irrelevant, or sorry I don't know the answer.
  - Score 2: the answer provides some relevance to the question and answer one aspect of the
question correctly.
    - Example:
      - Question: Welches sind die Vorteile der SME LED-Beleuchtung?
      - Answer: SME LED-Beleuchtung ist eine gute Wahl.
      - Or answer: Vorteile der SME LED-Beleuchtung sind, dass sie energieeffizient ist und
lange hält.
  - Score 4: the answer mostly answer the question but is missing or hallucinating on one critical
aspect.
    - Example:
      - Question: Welches sind die Vorteile der SME LED-Beleuchtung?
      - Answer: "SME LED-Beleuchtung bietet viele Vorteile, wie Energieeinsparung, lange
Lebensdauer und einfache Installation.
      Sie ist außerdem in verschiedenen Farben und Helligkeiten erhältlich, sodass sie für
jeden Bedarf die richtige Wahl ist."
  - Score 5: the answer correctly answer the question and not missing any major aspect
    - Example:
      - Question: Welches sind die Vorteile der SME LED-Beleuchtung?
      - Answer: SME LED-Beleuchtung punktet mit Energieeinsparung (bis zu 90!), langer
Lebensdauer, Umweltfreundlichkeit und besserer Lichtqualität.
      Die große Auswahl an Formen und smarten Funktionen machen sie ideal für jeden Bedarf.
      Sparen Sie Kosten, schützen Sie die Umwelt und genießen Sie angenehmes Licht mit SME
LED."]

Use the following function for the response:
{
  "name": "grading_function",
  "description": "Call this function to submit the grading for the answer",
  "parameters": {
    "type": "object",
    "properties": {
      "reasoning_for_correctness": {
        "type": "string",
        "description": "Your reasoning for giving the grading for the correctness of the
answer. Provide explanation." },
      "correctness": {
        "type": "integer",
        "description": "Your integer grading between 0 to 4 for the correctness of the
answer." },
      "required": [
        "reasoning_for_correctness",
        "correctness" ],
    }
  }
}

<|im_start|>user
Provided question:
{question}

Provided answer:
{answer}

Provided context:
{context} <|im_end|>
<|im_start|>assistant
```

V3 – Final few-shot prompt with 0-4 grading scale

```
<|im_start|>system
Please act as an impartial judge and evaluate the quality of the provided answer which attempts to
answer the provided question based on a provided context.
You'll be given a function grading_function which you'll call for each provided context, question
and answer to submit your reasoning and score for the correctness of the answer.
Please make sure you always call the function to submit result.

- Correctness: If the answer correctly answer the question, below are the details for different
scores:
  - Score 0: the answer is completely incorrect, doesn't mention anything about the question or is
completely contrary to the correct answer.
    - For example, when asked "Welches sind die Vorteile der SME LED-Beleuchtung?", the answer is
empty string, or content that's completely irrelevant, or sorry I don't know the answer.
  - Score 1: the answer provides some relevance to the question and answer one aspect of the
question correctly.
    - Example:
      - Question: Welches sind die Vorteile der SME LED-Beleuchtung?
      - Answer: SME LED-Beleuchtung ist eine gute Wahl.
      - Or answer: Vorteile der SME LED-Beleuchtung sind, dass sie energieeffizient ist und
lange hält.
  - Score 3: the answer mostly answer the question but is missing or hallucinating on one critical
aspect.
    - Example:
      - Question: Welches sind die Vorteile der SME LED-Beleuchtung?
      - Answer: "SME LED-Beleuchtung bietet viele Vorteile, wie Energieeinsparung, lange
Lebensdauer und einfache Installation.
      Sie ist außerdem in verschiedenen Farben und Helligkeiten erhältlich, sodass sie für
jeden Bedarf die richtige Wahl ist."
  - Score 4: the answer correctly answer the question and not missing any major aspect
    - Example:
      - Question: Welches sind die Vorteile der SME LED-Beleuchtung?
      - Answer: SME LED-Beleuchtung punktet mit Energieeinsparung (bis zu 90!), langer
Lebensdauer, Umweltfreundlichkeit und besserer Lichtqualität.
      Die große Auswahl an Formen und smarten Funktionen machen sie ideal für jeden Bedarf.
      Sparen Sie Kosten, schützen Sie die Umwelt und genießen Sie angenehmes Licht mit SME
LED."]

Use the following function for the response:
{
  "name": "grading_function",
  "description": "Call this function to submit the grading for the answer",
  "parameters": {
    "type": "object",
    "properties": {
      "reasoning_for_correctness": {
        "type": "string",
        "description": "Your reasoning for giving the grading for the correctness of the
answer. Provide explanation."},
      "correctness": {
        "type": "integer",
        "description": "Your integer grading between 0 to 4 for the correctness of the
answer."},
      },
      "required": [
        "reasoning_for_correctness",
        "correctness",],
    },
  }
<|im_start|>user
Provided question:
{question}

Provided answer:
{answer}

Provided context:
{context} <|im_end|>
<|im_start|>assistant
```

APPENDIX E – SYNTHETIC DATASET PROMPT

The word enclosed in curly brackets {} represents the placeholder for the context, that will be inserted into the prompt at runtime of the application.

Prompt for generating question-answer pairs

```
<|im_start|>system  
Your task is to write a factoid question and an answer given a context.  
Your factoid question should be answerable with a specific, concise piece  
of factual information from the context.  
Your factoid question should be formulated in the same style as questions  
users could ask in a search engine.  
This means that your factoid question MUST NOT mention something like  
"according to the passage" or "context".
```

Provide your answer as follows:

The output should be a markdown code snippet formatted in the following schema, including the leading and trailing `` `json` and `` ` ` ` ` :

```
`` `json  
{  
    "question": string // your factoid question in German  
    "answer": string // your answer to the factoid question in German  
}  
`` ` ` ` `
```



```
<|im_end|>  
<|im_start|>user  
Context: {context} <|im_end|>  
<|im_start|>assistant
```

APPENDIX F – BING CHAT PROMPT

Curly brackets {} act as placeholders within the Bing Chat prompt. These placeholders are filled with the user's question and retrieved context during prompt generation.

Prompt for inserting into Bing Chat

You are an expert assistant specializing in answering questions about lamps. Your goal is to provide accurate, concise, and helpful responses based on the given context. Follow these guidelines when answering the user's question:

- Review the provided context carefully to ensure you have information related to the question.
- Keep your answer brief and to the point, limiting it to a maximum of 3 sentences.
- If you cannot answer the question based on the given context, inform the user rather than inventing an answer.
- If the context is incomplete or irrelevant, politely inform the user that the provided information is insufficient to answer their question.

Format your response according to the provided schema: The output should be a markdown code snippet formatted in the following schema, including the leading and trailing ```json and ```:

```
```json
{
 "answer": string // Answer to the question
}
```

```

Context:
{context}

Question:
{question}



NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa

Towards Robust Knowledge Representations in Multilingual LLMs for Equivalence and Inheritance based Consistent Reasoning

Gaurav Arora

Amazon

gaurvar@amazon.com

Srujana Merugu

Amazon

smerugu@amazon.com

Shreya Jain

IIT Jammu*

2020uee0135@iitjammu.ac.in

Vaibhav Saxena

Amazon

saxenvai@amazon.com

Abstract

Reasoning and linguistic skills form the cornerstone of human intelligence, facilitating problem-solving and decision-making. Recent advances in Large Language Models (LLMs) have led to impressive linguistic capabilities and emergent reasoning behaviors, fueling widespread adoption across application domains. However, LLMs still struggle with complex reasoning tasks, highlighting their systemic limitations. In this work, we focus on evaluating whether LLMs have the requisite representations to reason using two foundational relationships: "equivalence" and "inheritance". We introduce novel tasks and benchmarks spanning six languages and observe that current SOTA LLMs often produce conflicting answers to the same questions across languages in 17.3-57.5% of cases and violate inheritance constraints in up to 37.2% cases. To enhance consistency across languages, we propose novel "Compositional Representations" where tokens are represented as composition of equivalent tokens across languages, with resulting conflict reduction (up to -4.7%) indicating benefits of shared LLM representations.

1 Introduction

Reasoning is the capacity to employ logic and analyze relationships among entities to extrapolate from known evidence to derive new insights. Language significantly bolsters this process by supplying the necessary structure and vocabulary for encoding complex ideas, thus facilitating hypothesis generation and evaluation. The intricate connection between linguistic and reasoning capabilities is a hallmark of human intelligence, enabling abstract thinking, problem-solving, and decision-making.

Recent advancements in LLMs such as ChatGPT (OpenAI, 2022) and Claude (Anthropic, 2023c) showcase their exceptional language generation capabilities and their potential to boost performance

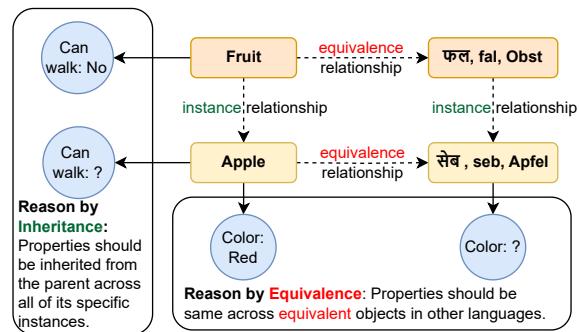


Figure 1: An example of Reasoning by "Equivalence" and Reasoning by "Inheritance" based on *existence* of equivalence/inheritance relationship between concepts.

across diverse Natural Language Processing (NLP) tasks (Ahuja et al., 2023), with multiple studies also pointing to emergent reasoning abilities at scale (Wei et al., 2022). However, LLMs continue to face challenges with complex reasoning tasks such as planning and problem-solving, indicating that their expansive modeling capacity and extensive training regime might enable them to mask deeper systemic shortcomings through superficial reasoning. As LLMs increasingly permeate applications catering to multilingual users with complex needs, gaining a deeper understanding of their functioning becomes imperative, as it could uncover systemic gaps and pave the way for superior models.

Robust reasoning hinges on the availability of powerful constructs such as entity-relation (ER) graphs and rules to interpret relationships. For instance, an ER graph with entities A, B, and C, where "A is the father of B" and "C is the wife of A" allows us to deduce that C is likely the mother of B, based on interpretation of the relationships "father" and "wife". While there are myriad relationships underpinning reasoning such as "cause and effect" and "comparison", in our current work, we focus on "equivalence" and "inheritance" due to their predominance in enhancing the efficiency of

*Contributed to this work during her internship at Amazon

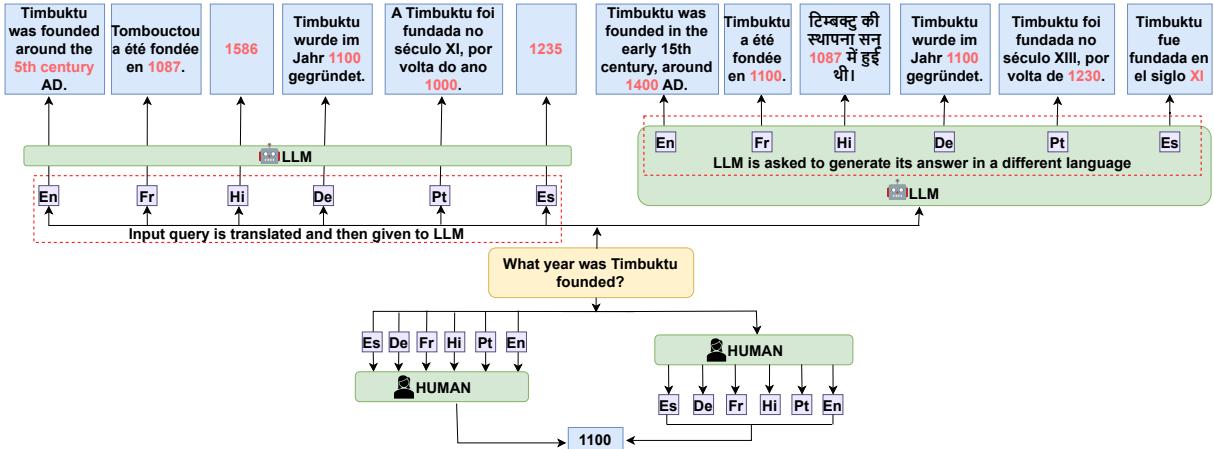


Figure 2: An example of LLM (Claude-v1 Instant) lacking "equivalence relationship between equivalent concepts across languages" due to tight coupling of knowledge representation and language expression unlike in Humans.

logical inference through property transfer, which is also reflected in their adoption as core constructs of knowledge representation and programming languages (Minsky et al., 1974). Fig 1 illustrates these relationships showing how humans create the necessary representations of "Apple" to reason across equivalent objects ("seb" : "Apple" in Hindi (transliterated), "Apfel" : "Apple" in German, etc) independent of the language/script of expression (Reasoning by Equivalence), and inherit properties from the abstract concept "Fruit" across all of its specific instances ("Apple", "Orange" etc) (Reasoning by Inheritance). Here, we expect representations of equivalent objects to be similar, while that of inherited objects satisfy transitivity. These representations are crucial for efficient learning, knowledge sharing, and updation of beliefs.

Typically, humans create abstractions using denotational semantics, i.e., a word's meaning is defined by objects it describes, which is the favored approach in logical theory. In contrast, LLMs use distributional semantics, i.e., a word's meaning stems from the training data context, which can be problematic when the data has gaps, such as infrequent connections between equivalent words across languages. Hence, despite impressive performance on NLP tasks(Ahuja et al., 2023), it is unclear if LLMs create the necessary representations within and across languages to support consistent reasoning across equivalent and inherited objects.

Contributions. In this work, we focus on whether LLMs have the requisite representations to reason by equivalence and inheritance across languages and make the below contributions.

1. We introduce a novel task and parallel bench-

mark datasets of factoid QA to evaluate "Reasoning by Equivalence" in LLMs and assess the performance of multiple SOTA LLMs on this task across 6 languages (English, French, Spanish, German, Portuguese and Hindi). On our benchmarks, LLMs generate conflicting answers across languages in 17.3-57.5% of cases indicating a significant gap. We also perform a controlled experiment to identify factors promoting consistency across languages and find a strong positive correlation with similarity in script and typology.

2. We present another task and associated new benchmark to evaluate "Reasoning by Inheritance" in LLMs and study the proficiency of multiple SOTA LLMs across six languages. Our results indicate that LLM answers violate inheritance constraints in up to 37.2% cases across these languages with most violations observed in Hindi.

3. We also propose a novel method for constructing "Compositional Representations" in LLMs by representing tokens as composition of other equivalent tokens in vocabulary, which grants the model access to (otherwise) distant representations of equivalent objects across languages, thereby facilitating improved knowledge sharing and reduction in conflicts with gains up to 4.7% compared to baselines.

To the best of our knowledge, this is the first quantitative study of LLM reasoning via equivalence and inheritance across languages. We will share the benchmarks as a community resource and to ensure reproducibility after organization approval. Note that even when the desired equivalence and inheritance relationships hold and properties transfer correctly (our current focus), there may be gaps in LLM's multi-step reasoning pro-

cess due to other factors as we discuss in detail in Appendix A.1.

2 Related Work

Multilingual NLP. LLMs like ChatGPT (OpenAI, 2022), GPT-4 (OpenAI et al., 2024), Claude (Anthropic, 2023c), BLOOMZ (Muennighoff et al., 2023), XGLM (Lin et al., 2022) have shown impressive performance on standard multilingual NLP tasks and benchmarks (Ahuja et al., 2023; Zhao et al., 2023; Enis and Hopkins, 2024; Ahuja et al., 2024). Despite extensive evaluations and the existence of parallel multilingual datasets such as MLQA and XQUAD (Ahuja et al., 2024), to the best of our knowledge, there is no prior work or tailored benchmarks for assessing LLMs’ ability to reason by equivalence and inheritance across multiple languages. Further, there is a chance of public benchmarks with duplicated knowledge across languages being included in LLM training data, rendering reasoning related assessments unreliable. Our study is the first to create controlled benchmarks and evaluate LLMs on these reasoning tasks to identify gaps and potential contributing factors.

Reasoning in LLMs. Reasoning abilities of LLMs have been studied for problem solving, decision making, and critical thinking (Huang and Chang, 2023; Wei et al., 2022; Bubeck et al., 2023). Prior work has also looked at evaluating ability of encoder only models like BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019) to understand ontological knowledge (Wu et al., 2023). In this work, we focus on reasoning based on two foundational relationships: equivalence and inheritance and evaluate popular LLMs on these dimensions across multiple languages. In recent years, there have been advances (Aspis et al., 2022; Lazzari et al., 2024; Marconato et al., 2023) in neuro-symbolic architectures that combine symbolic and sub-symbolic components to enable efficient computation of symbolic representations and deductive reasoning. However, these works do not present a detailed analysis of representations of equivalent or related entities and these methods also entail much higher computational costs, limiting their adoption.

Representation Learning in NLP. Improving distributed representations led to significant performance improvements in past (Liu et al., 2021; Devlin et al., 2018; Mikolov et al., 2013). Prior work on adapting attention mechanisms to bridge gaps across disparate but related inputs such as

translated/transliterated data has led to improved multilingual representations (Conneau et al., 2020; Khanuja et al., 2021; Arora et al., 2023). In our current work, we bridge the gap between distant representation spaces of various languages by adapting the attention mechanism to better utilize the token-language mapping.

3 Reasoning by Equivalence

Reasoning by equivalence is a core building block that enables efficient and scalable reasoning across contexts, with the efficiency being determined by the size of the equivalence classes (sets of equivalent objects). Construction of these “equivalence classes”, i.e., “abstract concepts” from specific contexts and reusing these abstract concepts flexibly beyond the specific contexts (An et al., 2023; Mitchell, 2021; Kumar et al., 2023; Giunchiglia and Walsh, 1992; Hull, 1920), is a natural human skill. The human ability to acquire knowledge from one language (e.g., “apple is red”) and construct representations shared across languages as in Fig 1 is a prime example. Similar to multilingual humans, LLMs also see large amount of multilingual data during pre-training (Blevins and Zettlemoyer, 2022). For instance, pretraining data of GPT-3 and BLOOM spanned 119 and 46 languages (Brown et al., 2020; Scao et al., 2023) respectively. In this section, we evaluate if SOTA LLMs also have the ability to reason by equivalence across languages given their impressive multilingual capabilities, and if this ability is due to shared representations (i.e. existence of equivalence relationship) or duplication of knowledge across languages. Note that the existence of “equivalence relationships encoded in LLM representations” is a fundamental prerequisite for complex reasoning even though it does not guarantee that LLMs can effectively leverage it for multi-step logical reasoning due to gaps in LLM’s inference mechanism based on associative attention (see Appendix A.1).

3.1 How good are LLMs at exhibiting “Reasoning by Equivalence”?

We evaluate if LLMs exhibit “Reasoning by Equivalence” specifically across languages by estimating how dependent LLM’s answers are on the language of input/output expression. A high dependency indicates tight coupling of the knowledge representation with the language and points to lack of shared abstractions and limited ability to reason

| | En-Fr | En-Es | En-De | En-Pt | En-Hi | Avg |
|--------------------------|--------------|--------------|--------------|--------------|--------------|--------------|
| Claude v1 Instant | 31.41 | 32.66 | 30.81 | 31.99 | 56.31 | 36.64 |
| Claude v2 | 22.20 | 22.54 | 21.75 | 22.89 | 47.02 | 27.28 |
| Claude v3 Sonnet | 20.99 | 21.23 | 20.60 | 21.49 | 43.51 | 25.56 |
| BLOOMZ-7B | 37.23 | 34.67 | 50.88 | 33.95 | 52.69 | 41.88 |
| XGLM-7.5B | 42.99 | 42.49 | 38.74 | 41.74 | 57.53 | 44.70 |

Table 1: Conflict rate between LLM answers to parallel En-XX questions, XX = [Fr, Es, De, Pt, Hi].

| | En-Fr | En-Es | En-De | En-Pt | En-Hi | Avg |
|--------------------------|--------------|--------------|--------------|--------------|--------------|--------------|
| Claude v1 Instant | 29.91 | 30.44 | 29.72 | 30.28 | 42.92 | 32.65 |
| Claude v2 | 20.45 | 21.13 | 20.52 | 21.38 | 28.14 | 22.32 |
| Claude v3 Sonnet | 17.89 | 18.06 | 17.34 | 17.99 | 28.61 | 19.97 |

Table 2: Conflict rate between LLM responses in [En, Fr, Es, De, Pt, Hi] for input questions in En.

by equivalence with knowledge duplication. Fig 2 shows an example where Claude-v1 instant’s answers are significantly dependent on the language of input/output expression. Below, we outline our methodology, dataset, metrics and results.

3.1.1 Methodology

Let $R = \{r_1, \dots, r_L\}$ be a set of languages with $[X^{r_1} X^{r_2}, \dots, X^{r_L}]$ denoting parallel questions across $r_i \in R$. We perform two assessments.

Dependency on Input Language (DIL). Given L parallel questions $[X^{r_1}, X^{r_2}, \dots, X^{r_L}]$; we generate LLM answers¹ Y^{r_i} for each X^{r_i} independently. Choosing an anchor language $r_{\text{anchor}} \in R$, we check for each $r \in R \setminus \{r_{\text{anchor}}\}$ if $Y^{r_{\text{anchor}}}$ and Y^r conflict with each other.

Dependency on Output Language (DOL). Given r_{anchor} , we input $X^{r_{\text{anchor}}}$ into the LLM and generate answers Y^r for $X^{r_{\text{anchor}}}$ for all $r \in R$ independently. Then, we check for each $r \in R \setminus \{r_{\text{anchor}}\}$ if $Y^{r_{\text{anchor}}}$ and Y^r conflict with each other.

Here, two answers are called conflicting if they contain contradictory information and not merely if there are different or one of them is non-informative. For our experiments, we consider English (En), French (Fr), Spanish (Es), German (De), Portuguese (Pt) and Hindi (Hi) languages, i.e. $R = \{\text{En}, \text{Fr}, \text{Es}, \text{De}, \text{Pt}, \text{Hi}\}$ and $r_{\text{anchor}} = \text{En}$. Since authors in (Lin et al., 2022; Ahuja et al., 2023) show that English instructions in the prompt perform better than instructions written in the native language for non-English languages, we tune the English instructions in prompt separately for each LLM and then keep these consistent for that LLM across all DIL and DOL experiments.

¹Temperature=0 across the paper for deterministic outputs.

3.1.2 Dataset and Metrics

Dataset. To ensure feasibility of automated evaluation via LLM-based judges and reduce variations due to subjective interpretation and cultural variations, our evaluation focused primarily on objective factual/attribute-based questions on entities. We prepare En factual questions dataset consisting of 88,334 questions on well known named entities and translate the dataset to Fr, Es, De, Pt and Hi using AWS Translate (AWS, 2017b). See Appendix A.5.1 for more details on the dataset and Fig 15 for a few sample questions.

Metrics. We compute the conflicts among answers generated by LLM for different input/output expression languages.² We define conflict rate between (r_i, r_j) language pair as fraction of total answer pairs which are conflicting, i.e. $\text{ConflictRate}(r_i, r_j) = \frac{\sum_{k=1}^{|D|} J(Y_k^{r_i}, Y_k^{r_j})}{|D|}$, where $|D|$ is dataset size and J returns 1 if $(Y_k^{r_i}, Y_k^{r_j})$ are conflicting, else returns 0. We use Claude v3 Sonnet as the judge J with prompt shown in Fig 13 in Appendix A.4. Table 5 in Appendix A.4 shows that the average precision of our judge is >95%.

3.1.3 Analysis and Results

Table 1 shows conflict rate for various LLMs for DIL task.³ We can see that conflict rate reduces with increase in model strength. Open source models lag behind closed source models by a significant margin with 25-44% average conflict rate across

²Since we care about consistent and common knowledge representation in LLMs for equivalent concepts, we only assess conflicting LLM responses to equivalent questions and not worry about factual accuracy of responses.

³GPT-3.5 had similar results as Claude v3 Sonnet but we could not add those results due to organization policy.

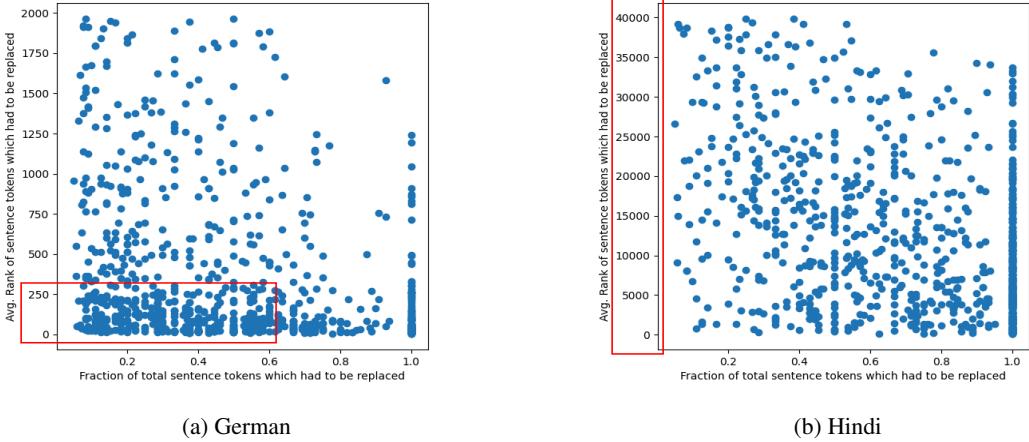


Figure 3: Avg. Rank and Fraction of tokens which had to be replaced in German and Hindi with parallel English token to achieve consistent answer as English.

different LLMs and languages. Since we establish from results in Table 1 that LLMs are highly dependent upon input expression language, and that this dependency is consistent across varied open-source and closed-source LLMs of varied size, we evaluate only Claude family models for DOL task as shown in Table 2. We observe similar trends as average conflict rate of 19-32%. Both these results show that knowledge representation is tightly coupled with expression/language in LLMs, indicating a lack of right abstractions and limited knowledge sharing across languages in LLMs. Fig 16 and Fig 17 in Appendix A.5.2 show sample conflicting answers from various LLMs to equivalent questions from DIL and DOL tasks respectively.

3.2 Factors affecting LLMs ability to exhibit “Reasoning by Equivalence”

To better understand knowledge transfer and source of conflicts across languages in LLMs, we perform a controlled experiment wherein we create synthetic QnA data with non-existent named entities that LLM does not have any prior knowledge on and train it on synthetic data in one language and test for its transfer in other languages.

3.2.1 Controlled Experiment

Dataset. We create synthetic data of non-existent named entities and hallucinated articles about those entities using Claude v1-instant. We also generate factual questions about synthetic named entities which can be answered only from hallucinated articles. We only keep those questions for which Claude’s answers with and without the article conflict with each other to ensure any LLM is un-

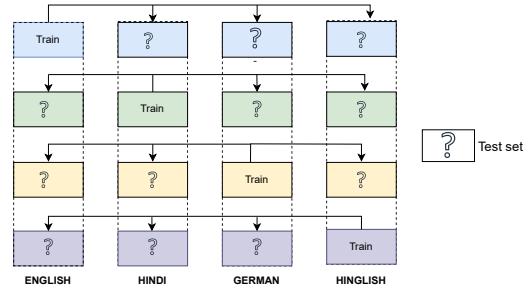


Figure 4: Setup for the controlled experiment. We train on unique 25% of the data for each language and test on its parallel data in other three languages.

likely to have any prior knowledge of our synthetic data. Our synthetic QnA dataset has 2063 synthetic named entities and has 32016 QnA pairs. As we perform the controlled experiment with En, De, Hi and HiEn (transliterated Hindi), and our synthetic question set is in En, we also translate it to De and Hi using AWS Translate, and transliterate to HiEn using IndicTrans (Bhat et al., 2015).

Experiment Setup. We train XGLM-4.5B on unique 25% of the synthetic data for each language and test on its parallel data in other three languages. Specifically, we train on concatenated $D_1^{\text{en}}, D_2^{\text{hi}}, D_3^{\text{de}}, D_4^{\text{hien}}$ data where D_s^r denotes synthetic data in r^{th} language from s^{th} quarter of D^r . For $s = 1$, En is the anchor language and we evaluate knowledge transfer to Hi, De, HiEn by looking at conflicts between LLM answer to En question from D_1^{en} and LLM answer to parallel Hi, De, HiEn questions from $D_1^{\text{hi}}, D_1^{\text{de}}, D_1^{\text{hien}}$ respectively. As shown in Fig 4, same procedure is followed for $s \in \{2, 3, 4\}$ where Hi, De, HiEn are anchor languages respectively.

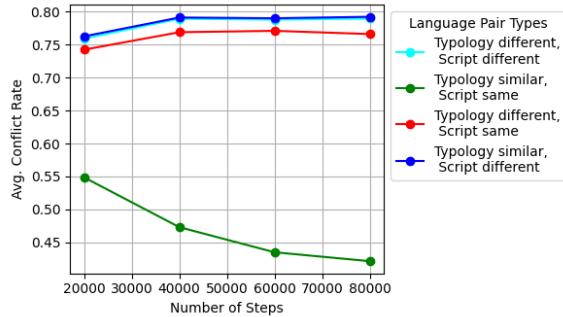


Figure 5: Conflict rate for different language pair types.

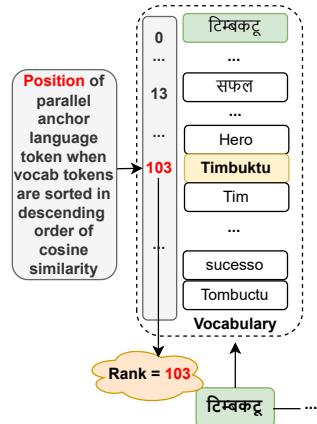
3.2.2 Results and Analysis

Fig 5 shows there is significantly higher knowledge transfer between languages with similar typology and same script, as compared to the pairs where either typology or script is different (see Fig 18 in Appendix A.5.3 for conflict rate of all language pairs individually). For instance, conflict rate for En-De is much lower than that of En-HiEn, which in turn is lower than that of En-Hi.

We define **Rank** of a token x in a non-anchor question as the *position* at which its parallel token from anchor question occurs if we sort all tokens in the vocabulary by cosine similarity with x in descending order. Rank of a token captures the relative proximity to the anchor language's parallel token in the embedding space.

Figure 6: Illustration of Rank.

To assess what it would take to get a non-conflicting answer, we replace tokens in the non-anchor question with their parallel anchor question tokens in the descending order of rank, i.e., farthest (non-anchor,anchor) parallel tokens are replaced first, till we obtain a non-conflicting answer. Fig 3a shows dense region of De tokens with small ranks which had to be replaced by their parallel anchor question tokens to reach a non-conflicting consistent answer. This shows that having close enough representations for equivalent tokens in different languages might also not be enough, and they have to be same for LLM to learn consistent knowledge.



This limitation stems from the representation space LLMs operate in since we project discrete symbols in the continuous embedding space. On the other hand, for Hi in Fig 3b, average rank of replaced tokens is significantly higher relative to De, likely due to the typology and script differences in case of En – Hi, which points to the need for near similar representations to share knowledge.

These results show that consistency of LLMs for distant languages is likely to stem from duplication of knowledge across languages in the training data, whereas for languages with similar typology and script, knowledge sharing occurs due to similar representation of equivalent tokens and information propagation. The effect of "duplicate knowledge in LLM training" is also reflected in conflict rate numbers of Tables 1 and 2 which are lower than those in Table 4. Results in Tables 1 and 2 are based on factual questions on well-known real entities, while the Table 4 results are from the controlled experiment with questions about synthetically created non-existent named entities. For the real entities in Tables 1 and 2, LLMs may have been pre-trained on duplicate information about the same entity expressed in multiple languages leading to consistent responses even when the LLM representations of the corresponding entities are significantly different (i.e., no equivalence relationship). However, for the synthetic entities in Table 4, LLMs are unlikely to have seen duplicate knowledge across languages during pre-training and has to learn about them (in one language) during the controlled training process resulting in significantly higher "conflict rates". See Appendix A.2 for more discussion on the effect of duplication of knowledge across languages in LLM's training data.

4 Reasoning by Inheritance

Reasoning by Inheritance is also a key building block of common sense and logical reasoning in humans enabled by concept abstractions. Humans identify common patterns amongst instances of the same type and create abstract concepts to reason consistently across all specific instances by inheriting properties from the abstract concept as shown in Fig 1. In this section, we investigate if SOTA LLMs can use their ontological knowledge and inherit properties from abstract concepts consistently across various specific instances of the abstract concept within multiple languages.

We evaluate if an LLM exhibits "Reasoning by

| | En | Fr | Es | De | Pt | Hi | Avg |
|--------------------------|-------------|-------------|-------------|-------------|-------------|--------------|-------------|
| Claude v1 Instant | 3.36 | 18.33 | 17.16 | 9.12 | 15.05 | 36.5 | 16.59 |
| Claude v2 | 3.37 | 4.55 | 11.69 | 4.96 | 6.78 | 18.53 | 8.31 |
| Claude v3 Sonnet | 0.13 | 4.43 | 7.27 | 1.86 | 5.41 | 14.69 | 5.63 |
| BLOOMZ-7B | 4.78 | 8.76 | 7.49 | 8.08 | 10.4 | 23.27 | 10.46 |
| XGLM-7.5B | 36.61 | 30.87 | 35.65 | 36.5 | 37.27 | 33.59 | 35.08 |

Table 3: Conflict rate of LLM answers on inheritance-based questions on common concepts in En, Fr, Es, De, Pt, Hi.

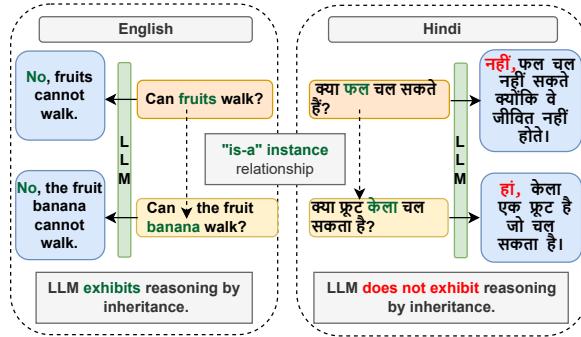


Figure 7: An example of LLM (Claude v1 Instant) exhibiting reasoning by inheritance in En but not in Hi.

Inheritance” within a language by checking if specific instances of an abstract parent concept inherit properties of the parent without conflicts. Fig 7 shows an example wherein Claude-v1 instant does not exhibit reasoning by inheritance in Hi due to the lack of the right abstractions.

Dataset. We prepare a set of 35 abstract concepts and 2396 well known named entities which are specific instances of those abstract concepts. For each one of the abstract concept, we hand-curate set of properties that all of its specific instances should inherit. We create templatized questions from those properties to prepare En dataset and translate it to Fr, Es, De, Pt and Hi using AWS Translate (AWS, 2017b). Fig 19 in Appendix A.6.1 shows a sample of our dataset.

Methodology. We evaluate the consistency of LLMs in inheriting and applying ontological knowledge consistently across specific instances of abstract concepts by directly asking in native language if the specific instance has the property of abstract concept and checking if the answer for abstract concept and specific instance are conflicting.

Metrics. We compute conflict rate as in the Section 3 but focus on comparing LLM answers on an inheritable property for an abstract parent concept and that of its specific instances.

Results. Table 3 shows conflict rate for questions requiring reasoning by inheritance across six

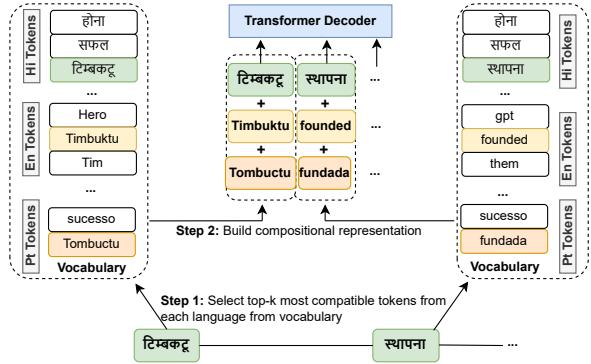


Figure 8: CoRe Illustration: LLM can access distant equivalent representations to permit knowledge sharing.

languages.³ We notice that conflict rate is low for En for most models but quite high for Hi, which likely has low representation in the model training corpus. This indicates that the ability to reason by inheritance likely depends on the amount of specific language data during training LLMs, which in turn points to gaps in inductive biases in LLMs. Fig 20 in Appendix A.6.2 shows few LLM responses that violate inheritance constraints with conflicting answers for parent and child concepts.

5 Compositional Representation (CoRe)

We now consider mechanisms to mitigate lack of knowledge consistency across languages which emerged as a problem in the prior sections. Our analysis points to two key observations:

1. “Identical” representations for equivalent concepts ensures perfect knowledge transfer while “distant” representations lead to separate copies of knowledge being learned.
2. Languages from different families (such as En and Hi) have distant LLM representations for equivalent concepts resulting in low knowledge sharing between them and high inconsistency unless duplicate information is fed in both languages.

Based on these observations, we propose CoRe with the aim of bridging distant representation spaces to enable greater knowledge sharing. It

hinges on the key idea that representing a concept via composition of representations of all equivalent concepts across languages would enable LLM to maintain consistent knowledge for that concept across languages as shown in Fig. 8. This is in contrast to the current LLM models (Fig. 9) where a sentence input into the transformer decoder in LLMs does not have access to representations of equivalent tokens from distant languages and can only utilize the localised knowledge in that language’s representation space.

5.1 Methodology

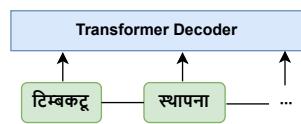


Figure 9: Default mechanism in Transformers.

Attention (Vaswani et al., 2017) is an essential mechanism of transformer architecture that converts an input sequence into a latent encoding using representational vectors formed from the input, i.e., queries, keys and values to determine the importance of each portion of input while decoding. Typically, in transformers, each token pays attention to other nearby tokens in the input sequence. Since this process can miss out on equivalent tokens in other languages, especially in the absence of a parallel multi-lingual corpus, we modify the learning approach to consider all tokens in the vocabulary as candidates for attention. Specifically our methodology consists of two steps: (a) proximal token selection, and (b) construction of compositional representation, which we describe below.

Step 1: Proximal token selection. For a given token, we first select the top- n proximal tokens across each language based on compatibility of the existing representations. Formally, let $X = [x_i]_{i=1}^N$ and $Z = [z_i]_{i=1}^N$ denote the sequence of input tokens and the associated embedding representations of size d_o . Let $U = [u_j]_{j=1}^M$ be all the vocabulary tokens and $B = [b_j]_{j=1}^M$ be the associated embedding representations of size d_o . Further, let U^r be the vocabulary tokens associated with the language $r \in R$, where R denotes the entire set of languages being considered. Let $Q = [q_i]_{i=1}^N$, $K = [k_j]_{j=1}^M$ and $V = [v_j]_{j=1}^M$ be the sequences of query, key and value vectors of dimensions d_k , d_k and d_v respectively, given by $q_i = z_i^T W_Q$, $k_j = b_j^T W_K$ and $v_j = b_j^T W_V$ where $W_Q \in R^{d_o \times d_k}$, $W_K \in R^{d_o \times d_k}$ and $W_V \in R^{d_o \times d_v}$ are the learned projection matrices.

ces. Let $C = \frac{QK^T}{\sqrt{d_k}} \in R^{N \times M}$ be the compatibility matrix between Q and K .

For each input token x_i , we identify the top n proximal or most-compatible tokens from each language $r \in R$ as per the compatibility values:

$$U_{sel}^r(i) = \{u_j | C_{ij} \in \text{top-}n(\{C_{ij} | u_j \in U^r\}),$$

where $\text{top-}n(\cdot)$ denotes the largest n values of the input set. Note that in addition to using the organic representations for estimating compatibility, we could also use additional cues from domain ontologies or dictionaries to construct this proximal token set. Further, to ensure computational efficiency, instead of considering the entire set of vocabulary, a smaller candidate pool of tokens per language can be chosen using K-NN based on existing token embeddings at each stage.

Step 2: Construction of compositional representation. The next step is to build a compositional representation from all the selected proximal tokens similar to regular attention mechanism. To ensure only the selected proximal tokens contribute, we define $f(C) = [f_{ij}]$ where,

$$f_{ij} = \begin{cases} 0 & \text{if } u_j \in U_{sel}^r(i) \quad \forall r \in R \\ -\infty & \text{otherwise.} \end{cases} \quad (1)$$

We augment representation of Z being input to the decoder layer as $Z' = \text{softmax}(C + f(C))V$. Since R includes all the languages including ones distant from that of the input sequence X , Z' becomes a composition of equivalent tokens from all the languages yielding more consistent responses.

5.2 Experiments and Results

CoRe augments the transformer architecture and can be used while pretraining LLMs. However, since pretraining entails additional compute cost, it is preferable to use CoRe with existing pre-trained models. For our experiments, we augment pre-trained XGLM-4.5B model with CoRe.

Dataset, Downstream Task and Implementation Details. Since we want to examine if CoRe helps improve consistency among distant languages, we use the same dataset and setup as the controlled experiment in Sec 3.2.1. We augment XGLM architecture and add CoRe to it. We initialize learnable projection matrices W_Q , W_K , W_V by identity matrix to ensure stable continual training and choose $n \in \{5, 10, 15\}$ for selecting the top- n tokens for our experiment. In our experiments, for

| | n | en-hi | en-de | en-hien | hi-en | hi-de | hi-hien | de-en | de-hi | de-hien | hien-en | hien-hi | hien-de |
|---------------------|----------|--------------|--------------|----------------|--------------|--------------|----------------|--------------|--------------|----------------|----------------|----------------|----------------|
| XGLM-4.5B | - | 74.4 | 58.6 | 73.2 | 72.6 | 73.6 | 72.4 | 57.1 | 75 | 74.3 | 70.5 | 72.1 | 71.8 |
| XGLM-4.5B
+ CoRe | 5 | 73.5 | 55.6 | 71.9 | 71.9 | 72.7 | 71.9 | 55.7 | 74 | 73.3 | 68.4 | 70.4 | 69.5 |
| | 10 | 72.8 | 53.9 | 73.1 | 71 | 72.2 | 72.2 | 53.8 | 73.9 | 73.9 | 69.8 | 70 | 69.1 |
| | 15 | 73.2 | 56.8 | 72.3 | 71.3 | 73.3 | 71.8 | 56.8 | 75.5 | 73.5 | 68.7 | 71 | 69.2 |

Table 4: Effect of CoRe on conflict rate for different language pairs.

each one of the languages we are working with (En, Hi, De, HiEn), we consider the top- n most compatible tokens from all vocabulary tokens of that language and En. This considers En as the anchor language and helps build a “bridge” between distant representation spaces of other languages and En. We did this for more efficient experimentation but there is no constraint in CoRe on the set of languages from which we can choose top- n proximal tokens. We identify the language(s) a vocabulary token can belong to beforehand using language detector from AWS Comprehend ([AWS, 2017a](#)) and store the asset for repeated use during forward pass. We use Pytorch ([Paszke et al., 2019](#)) and Huggingface Transformers library ([Wolf et al., 2020](#)) for implementation. We train baseline XGLM-4.5B and XGLM-4.5B+CoRe for 20k steps on p3dn.24xlarge machine with 8 GPUs with learning rate of 1e-05 and linear learning rate scheduler with 80040 max steps, gradient accumulation steps of 2 and per device training batch size of 1.

Results. Table 4 shows that CoRe consistently reduces conflict rate across 12 language pairs, with gains up to 4.7%. Variation in conflict rate with different values of n suggests that keeping n static is not ideal as it might add noise to representations in some cases. Distribution of cosine similarity of random 1K parallel En-De words in Fig 10 shows that representations from CoRe for equivalent words are closer with similar behavior observed for other language pairs. Fig 21 in Appendix A.7.1 shows sample questions where XGLM-4.5B+CoRe provides more consistent answers compared to XGLM-4.5B (baseline). To evaluate CoRe’s impact on a downstream task we did a small scale experiment for NLI task on XNLI ([Conneau et al., 2018](#)) dataset and observed 14% reduction in inconsistency across predictions for parallel En-De NLI data points without hurting NLI performance, see Appendix A.7.2 for more details.

Efficiency of CoRe. In our experiments, adding CoRe increased the training time by \sim 50% but that is without using FlashAttention ([Dao et al., 2022](#)) for CoRe which is expected to be \sim 2x faster.

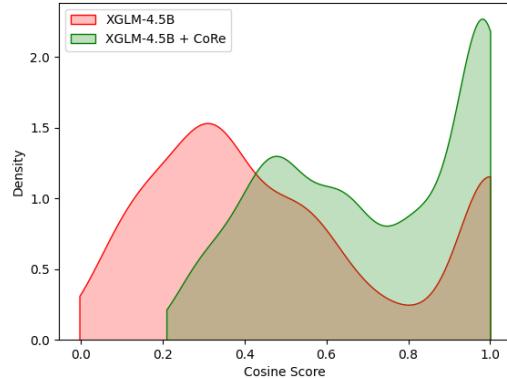


Figure 10: Distribution of cosine score of random 1K parallel En-De words.

We were unable to experiment using FlashAttention because we only had access to V100 machine. Since compatibility matrix is constructed by dot product of Q and K matrices as described in Sec 5.1, this is a computationally intensive operation which can be made more efficient by considering a smaller candidate pool of tokens per language of interest using a more efficient method like K-NN before constructing compatibility matrix. We will be experimenting with these alternatives which can improve computational efficiency of CoRe as part of our future work.

6 Conclusion

We introduce “Reasoning by Equivalence” and “Reasoning by Inheritance” tasks and evaluate popular LLMs to highlight the lack of consistent representation across languages. This systemic gap manifests in inefficient learning, limited knowledge sharing, and over-reliance on extensive data and computational resources, pointing to the need for better representations. We also perform controlled experiments to identify the influencing factors and propose CoRe to bridge the gap between distant language representations which leads to 4.7% boost in performance. We hope our work spurs further research on gaining richer understanding of LLM-based reasoning across languages.

Limitations

Our current work has a few limitations, which we discuss below.

Scope of Relationships: Our study focuses on LLMs' ability to reason based on two foundational relationships: "equivalence" and "inheritance." Future research could broaden this scope to include other key relationships such as "cause and effect," "comparison," and "mereological" relationships. Our proposed CoRe approach is also applicable only to symmetric relationships such as "equivalence," but there is a possibility of extending to asymmetric and transitive relationships using hyperbolic representations.

Evaluation Focus: Current experiments primarily targeted objective, factoid-based questions, chosen for their clarity and the feasibility of automated evaluation via LLM-based judges. This approach facilitated a less ambiguous assessment of LLM reasoning capabilities and the benefits of our CoRe-based mitigation. However, reasoning tasks do encompass subjective, long-form generation tasks such as summarization and problem solving, which can be explored in future, since that could entail access to expensive human-in-the-loop evaluations.

LLM Architectures: Our current work focuses exclusively on transformer-based autoregressive generative LLMs, which include widely used models such as Claude, XGLM, GPT-4, and LLama. Recent advances in neuro-symbolic methods offer alternative architectures and training methods that enhance reasoning abilities albeit at a higher computational costs that limits their adoption in real-world application. Our research identifies specific gaps in the popular LLMs, highlighting the need to integrate ideas from neuro-symbolic research.

Ethics Statement

Our research aims to identify and address gaps in the reasoning abilities of widely used LLMs, particularly for low-resource languages used by a large population of the world. To ensure the validity of our findings, we created a new parallel factoid QA datasets and conducted controlled experiments to prevent data duplication across languages in LLM training data from influencing LLM reasoning performance. The datasets used have no associated privacy or intellectual property concerns, and we plan to open-source them post-review to adhere to double-blind protocol and ensure reproducibility. Evaluation was performed using automated

LLMs, with prompts detailed in the appendix for transparency. Beyond the common ethical considerations of using generative language models, our work did not involve any additional ethical issues.

References

- Kabir Ahuja, Harshita Diddee, Rishav Hada, Millilcent Ochieng, Krithika Ramesh, Prachi Jain, Akashay Nambi, Tanuja Ganu, Sameer Segal, Maxamed Axmed, Kalika Bali, and Sunayana Sitaram. 2023. [Mega: Multilingual evaluation of generative ai](#). *Preprint*, arXiv:2303.12528.
- Sanchit Ahuja, Divyanshu Aggarwal, Varun Gumma, Ishaan Watts, Ashutosh Sathe, Millicent Ochieng, Rishav Hada, Prachi Jain, Maxamed Axmed, Kalika Bali, and Sunayana Sitaram. 2024. [Megaverse: Benchmarking large language models across languages, modalities, models and tasks](#). *Preprint*, arXiv:2311.07463.
- Shengnan An, Zeqi Lin, Bei Chen, Qiang Fu, Nanning Zheng, and Jian-Guang Lou. 2023. [Does deep learning learn to abstract? a systematic probing framework](#). *Preprint*, arXiv:2302.11978.
- Anthropic. 2023a. Anthropic support. <https://support.anthropic.com/en/articles/7996845-what-are-some-things-i-can-use-claude-for>. Accessed: 14-10-2024.
- Anthropic. 2023b. Claude 2 model card. <https://www-cdn.anthropic.com/5c49cc247484cecf107c699baf29250302e5da70/claude-2-model-card.pdf>. Accessed: 14-10-2024.
- Anthropic. 2023c. Welcome to claude. <https://docs.anthropic.com/claude/docs/intro-to-claude>. Accessed: 30-04-2024.
- Anthropic. 2024. Claude 3 model card. https://www-cdn.anthropic.com/de8ba9b01c9ab7cbabf5c33b80b7bbc618857627/Model_Card_Claude_3.pdf. Accessed: 14-10-2024.
- Gaurav Arora, Srujana Merugu, and Vivek Sembium. 2023. [CoMix: Guide transformers to code-mix using POS structure and phonetics](#). In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7985–8002, Toronto, Canada. Association for Computational Linguistics.
- Yaniv Aspis, Krysia Broda, Jorge Lobo, and Alessandra Russo. 2022. [Embed2Sym - Scalable Neuro-Symbolic Reasoning via Clustered Embeddings](#). In *Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning*, pages 421–431.
- AWS. 2017a. Aws comprehend. <https://aws.amazon.com/comprehend/>. Accessed: 14-10-2024.

- AWS. 2017b. Aws translate. <https://docs.aws.amazon.com/translate/>. Accessed: 30-04-2024.
- AWS. 2023. Aws bedrock claudie. <https://aws.amazon.com/bedrock/claudie/>. Accessed: 14-10-2024.
- Irshad Ahmad Bhat, Vandana Mujadia, Aniruddha Tamewar, Riyaz Ahmad Bhat, and Manish Shrivastava. 2015. Iiit-h system submission for fire2014 shared task on transliterated search. In *Proceedings of the Forum for Information Retrieval Evaluation, FIRE '14*, pages 48–53, New York, NY, USA. ACM.
- Terra Blevins and Luke Zettlemoyer. 2022. Language contamination helps explains the cross-lingual capabilities of English pretrained models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3563–3574, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.
- Sébastien Bubeck, Varun Chandrasekaran, Ronen Elidan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. *Preprint*, arXiv:2303.12712.
- Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.
- Alexis Conneau, Guillaume Lample, Ruty Rinott, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: evaluating cross-lingual sentence representations. *CoRR*, abs/1809.05053.
- Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.
- Maxim Enis and Mark Hopkins. 2024. From llm to nmt: Advancing low-resource machine translation with claudie. *Preprint*, arXiv:2404.13813.
- Fausto Giunchiglia and Toby Walsh. 1992. A theory of abstraction. *Artificial Intelligence*, 57(2):323–389.
- Jie Huang and Kevin Chen-Chuan Chang. 2023. Towards reasoning in large language models: A survey. *Preprint*, arXiv:2212.10403.
- Clark L Hull. 1920. Quantitative aspects of the evolution of concepts: An experimental study. *Psychological Monographs*, 28(1):1–85.
- Nora Kassner, Philipp Dufter, and Hinrich Schütze. 2021. Multilingual LAMA: Investigating knowledge in multilingual pretrained language models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3250–3258, Online. Association for Computational Linguistics.
- Simran Khanuja, Diksha Bansal, Sarvesh Mehtani, Savya Khosla, Atreyee Dey, Balaji Gopalan, Dilip Kumar Margam, Pooja Aggarwal, Rajiv Teja Nagipogu, Shachi Dave, Shruti Gupta, Subhash Chandra Bose Gali, Vish Subramanian, and Partha Talukdar. 2021. Muril: Multilingual representations for indian languages. *Preprint*, arXiv:2103.10730.
- Sreejan Kumar, Ishita Dasgupta, Nathaniel D. Daw, Jonathan D. Cohen, and Thomas L. Griffiths. 2023. Disentangling abstraction from statistical pattern matching in human and machine learning. *Preprint*, arXiv:2204.01437.
- Nicolas Lazzari, Stefano De Giorgis, Aldo Gangemi, and Valentina Presutti. 2024. Sandra – a neuro-symbolic reasoner based on descriptions and situations. *Preprint*, arXiv:2402.00591.
- Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, Tianlu Wang, Shuhui Chen, Daniel Simig, Myle Ott, Naman Goyal, Shruti Bhosale, Jingfei Du, Ramakanth Pasunuru, Sam Shleifer, Punit Singh Koura, Vishrav Chaudhary, Brian O’Horo, Jeff Wang, Luke Zettlemoyer, Zornitsa Kozareva, Mona Diab, Veselin Stoyanov, and Xian Li. 2022. Few-shot learning with multilingual generative language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9019–9052, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

- Zhiyuan Liu, Yankai Lin, and Maosong Sun. 2021. [Representation learning for natural language processing](#). *CoRR*, abs/2102.03732.
- Emanuele Marconato, Gianpaolo Bontempo, Elisa Ficarra, Simone Calderara, Andrea Passerini, and Stefano Teso. 2023. [Neuro-symbolic continual learning: Knowledge, reasoning shortcuts and concept rehearsal](#). *Preprint*, arXiv:2302.01242.
- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. [Efficient estimation of word representations in vector space](#). *Preprint*, arXiv:1301.3781.
- Marvin Minsky et al. 1974. A framework for representing knowledge.
- Melanie Mitchell. 2021. [Abstraction and analogy-making in artificial intelligence](#). *Annals of the New York Academy of Sciences*, 1505(1):79–101.
- Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, Xiangru Tang, Dragomir Radev, Alham Fikri Aji, Khalid Almubarak, Samuel Albanie, Zaid Alyafeai, Albert Webson, Edward Raff, and Colin Raffel. 2023. [Crosslingual generalization through multitask finetuning](#). *Preprint*, arXiv:2211.01786.
- OpenAI. 2022. Introducing chatgpt. <https://openai.com/index/chatgpt>. Accessed: 30-04-2024.
- OpenAI, Josh Achiam, and Steven Adler et al. 2024. [Gpt-4 technical report](#). *Preprint*, arXiv:2303.08774.
- Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. [Pytorch: An imperative style, high-performance deep learning library](#). In H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.
- Greg Rushing. 2020. Aws machine learning blog. <https://aws.amazon.com/blogs/machine-learning/amazon-translate-ranked-as-1-machine-translation-provider-by-intento>. Accessed: 11-10-2024.
- Teven Le Scao, Angela Fan, and Christopher Akiki et al. 2023. [Bloom: A 176b-parameter open-access multilingual language model](#). *Preprint*, arXiv:2211.05100.
- Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *ArXiv*, abs/1706.03762.
- Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. [Emergent abilities of large language models](#). *Preprint*, arXiv:2206.07682.
- Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pieric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. [Transformers: State-of-the-art natural language processing](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.
- Weiqi Wu, Chengyue Jiang, Yong Jiang, Pengjun Xie, and Kewei Tu. 2023. [Do PLMs know and understand ontological knowledge?](#) In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3080–3101, Toronto, Canada. Association for Computational Linguistics.
- Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. [A survey of large language models](#). *Preprint*, arXiv:2303.18223.

A Appendix

A.1 Existence of Equivalence/Inheritance Relationships vs Using them in Multi Step Reasoning

It is important to distinguish between **(a)** the existence of "equivalence" and "inheritance" relationships among concepts in LLM representations, and **(b)** the LLM's ability to actually utilize these relationships (encoded in the representations) for multi step reasoning. The former can be considered a prerequisite for the latter.

By the "existence of an equivalence relationship" between A and B , we mean that the LLM's representations and their direct use during inference allows properties of A and B to be transferred to each other or reconciled for conflicts. Similarly, an "inheritance relationship" between A and B would imply that A inherits properties of B .

Consider the question "Does the river Kaveri flow in the same continent as the river Seine?". We could answer this correctly as

1. The Kaveri river flows in India
2. The Seine river flows in France
3. India is part of the continent Asia
4. France is part of the continent Europe
5. **Kaveri = कावेरी**
6. **India = भारत**
7. **flows in = में बहती**
8. **Seine = सीन**
9. **France = फ्रांस**
10. **Continent = महाद्वीप**

Figure 11: An example of prior information which can be combined to answer the question "Does the river Kaveri flow in the same continent as the river Seine?" in both English and Hindi. Equivalence relationships are in bold.

"No" using different sets of prior information with varying levels of reasoning applied. For example, in **Scenario 1**, we might directly utilise the information "Kaveri and Seine flow in different continents". Alternatively, in **Scenario 2**, we might need to combine multiple pieces of information (expressed as entity-attribute or entity relationship predicates) as shown in Fig 11 to arrive at the answer. Scenario 2 demonstrates the ability to generalize from a small set of training data with limited compute effort to address a broader range of questions. To arrive at the correct answer, we need both the basic building block relationships as well as the capability to effectively combine them as part of LLM inference.

In our current study, we are primarily focused on evaluating whether the basic equivalence/inheritance relationships exist in LLMs by assessing simple property transfer and conflict resolution on questions related to equivalent entities or parent-child entities. It is possible that even when the desired relationships hold and properties transfer correctly, there may be gaps in the overall multi-step reasoning process, resulting in an inaccurate LLM response. This is because the existence of "abstraction" or "equivalence/inheritance relationships encoded in LLM representations" does not necessarily mean that the LLM would always effectively leverage this information for its reasoning since the LLM's inference mechanism, which relies on associative attention, differs from logical operations. We do not yet evaluate this larger capability (item b) because there are multiple factors involved, and there is likely a gap in item (a) itself.

- | |
|--|
| EN: Which continent is Kaveri located in?
HI: कावेरी किस महाद्वीप में स्थित है? |
|--|

Figure 12: An example parallel question from MLAMA (Kassner et al., 2021) dataset.

A.2 Impact of Prior Information seen during Training

Consider a pair of parallel questions from the MLAMA (Kassner et al., 2021) dataset ("Which continent is Kaveri located in ?") as shown in Fig 12. When an LLM provides the same response "Asia" without conflicts for the two parallel questions in En and Hi, it could be due to one of the following two reasons:

1. **Existence of Equivalence relationship/Common Abstraction:** The LLM is aware of equivalence between the En and Hi versions of entities "Kaveri" and "India" in Fig 11 beyond just the basic language constructs.
2. **Duplication of Information:** The LLM could have seen two parallel, aligned pieces of information that "Kaveri is located in Asia" in both En and Hi even though their LLM's representations of the corresponding equivalent entities are highly divergent. In the latter case, while the LLM can answer this specific question well without conflicts, that behavior might not generalize well to other questions about "Kaveri" in Hi that were not part of the training data.

Hence, conflict rate estimates could lead to misleading conclusions about the existence of equivalence and inheritance relationships for well-known real entities due to the bias introduced by prior duplicate (or even contradictory) information in the LLM training data across languages. This **effect of "duplicate knowledge in LLM training"** is also reflected in our results, where the conflict rate numbers in Table 4 are higher than those in Tables 1 and 2. We expect the effect observed in Tables 1 and 2 to be more pronounced in public datasets like MLAMA, as they may be part of the LLM's training data, either directly or indirectly. For instance, we generated responses from Claude v1 instant for parallel En – De 100 randomly sampled questions from MLAMA dataset and observed ~15% conflict

| En-Fr | En-Es | En-De | En-Pt | En-Hi | Avg |
|-------|-------|-------|-------|-------|--------|
| 96% | 97% | 94% | 95% | 94% | 95.20% |

Table 5: Precision of Claude v3 Sonnet as the judge in identifying conflicting answers across language pairs

rate which is much lower than the average results in Tables 1 and 2 (~30%).

A.3 Accounting for LLMs’ proficiency in different languages

We evaluate LLMs only on the languages for which there is official documentation of support for that language or there is prior work demonstrating good performance on NLP tasks of that language. For example, Anthropic claims support for English, Spanish, Portuguese, French, German and multiple other languages (Anthropic, 2023a; AWS, 2023), and also showcases Claude’s multilingual capabilities on these languages. Hindi is also mentioned in Claude model cards (Anthropic, 2023b, 2024). In our study, the LLMs we evaluate on a language are proficient with respect to the linguistic patterns and the common vocabulary of that language, which is different from the knowledge (factual information) aspects. Further in our evaluation, we focused on well-formed objective factual questions to avoid variations due to subjective interpretations and cultural nuances. This ensures that the knowledge consistency or alternatively the conflict rate in responses across languages primarily depends on (a) the knowledge duplication in the training data and (b) the effectiveness of knowledge transfer, i.e., equivalent entity representations across the languages. The point we wish to highlight in our work is that LLM training and representations should be designed so as to enable efficient knowledge transfer within and across languages.

A.4 Judge Precision

Table 5 shows precision of Claude v3 Sonnet as the judge in identifying conflicting answers across language pairs obtained by annotating random sample of 100 answer pairs for 5 language pairs each. Prompt for Claude v3 Sonnet judge is shown in Fig 13.

A.5 Reasoning by Equivalence

A.5.1 Data Preparation

We hand-curated 51 (parent) abstract concepts manually (e.g., monuments, actors, cities, etc.) that primarily correspond to common nouns. See Fig

14 for full list of the 51 abstract concepts. Then, we created 3641 named entities that are specific instances of these (parent) abstract concepts, e.g., Taj Mahal is a specific instance of monument) using Claude with human review to weed out non-existent ones. For each one of the 3641 named entities, we prepared a set of questions which have objective or factual answers using Claude with the following prompt.

Give different unambiguous complete questions about “{entity}” which have specific factual answers.

We built this prompt after multiple iterations of analysis of generated questions for a small sample of entities. The dimensions of evaluation were: (i) The generated question should be complete and unambiguous, i.e. it should be clear which entity the question is about and what attribute/fact is being asked, (ii) The answer to the generated question should be an unambiguous factual response. From the final set of generated 88,334 questions, we randomly sampled and manually annotated 500 generated questions of which 96.2% were complete and unambiguous, and 99.4% had an unambiguous factual answer. The annotations were done by a professional English speaker.

Our original question set is in En which we also translate to Fr, Es, De, Pt and Hi using AWS Translate (AWS, 2017b). We translated using AWS translate which is one of the best commercial translation services (Rushing, 2020). AWS translate is expected to translate questions while preserving their meaning. We rely on AWS translate to translate the concept into most natural variant in case multiple variants are possible. To get an estimate of lower bound of AWS translate’s performance on our datasets, we consider the En-Hi translation task since Hi being a non-Latin language with different lexical representation is much more divergent from En. We enlisted a professional bilingual Hindi and English speaker to annotate a random sample of 200 En-Hi question pairs on the translation accuracy and observe 97.5% accuracy. This high translation accuracy is likely due to the nature of our English questions, which are well-formed and unambiguous. Fig 15 shows few sample questions from our dataset.

A.5.2 Analysis

Fig 16 and Fig 17 show sample conflicting answers from various LLMs to equivalent questions from DIL and DOL tasks respectively.

```

def create_prompt(question, response1, response2):
    prompt_data = f"""Human: Below given are two answers to the question '{question}'.

    First Answer: {response1}

    Second Answer: {response2}

    Are above answers contradictory to each other. First share your reasoning in one line briefly within <thinking></thinking> tags. Be precise.
    Finally, within <response></response> tags provide final answer with a Yes or No only.

    Assistant:"""
    return prompt_data

```

Figure 13: Prompt for Claude v3 Sonnet to use it as the judge.

```

{'Monument', 'Australian Open', 'Thomas Cup', 'US Open', 'Vegetables', 'Bikes', 'Organizations', 'Football Asian Cup', 'Songs', 'Beaches', 'Actress', 'All England Open', 'Fruits', 'Sudirman Cup', 'Football African Cup of Nations', 'Museum', 'French Open', 'National Parks', 'Singer', 'Books', 'City', 'Movie', 'State', 'Poets', 'Uber Cup', 'Cars', 'Country', 'Badminton Player', 'Animals', 'ICC Cricket World Cup years', 'Movie Directors', 'Cricket Player', 'Football Copa America', 'World Championships', 'Politicians', 'Actor', 'Football Player', 'Lakes', 'Football CONCACAF Gold Cup', 'World Junior Championships', 'UEFA European Championship', 'ICC Champions Trophy', 'Asia Cup', 'ICC Test Championship', 'ICC World Twenty20', 'Flowers', 'Tennis Player', 'Authors', 'Fibers', 'FIFA World Cup', 'Wimbledon'}

```

Figure 14: List of 51 (parent) abstract concepts.

| Parent Concept | Child Concept | Question (En) | Question (Fr) | Question (Hi) | Question (De) | Question (Pt) | Question (Es) |
|----------------|---------------------|--|---|---|--|---|---|
| Monument | Great Wall of China | How long is the Great Wall of China? | Quelle est la longueur de la Grande Muraille de Chine ? | चीन की महान दीवार कितनी लंबी है? | Wie lang ist die Chinesische Mauer? | Quanto tempo dura a Grande Muralha da China? | ¿Qué longitud tiene la Gran Muralla China? |
| Movie | Swades | Who starred as the lead actor in the movie Swades? | Qui a joué le rôle principal dans le film Swades? | स्वदेस फिल्म में मुख्य अभिनेता कौन था जिसने अभिनय किया था? | Wer spielte die Hauptrolle im Film Swades? | Quem estrelou como ator principal no filme Swades? | ¿Quién interpretó al actor principal en la película Swades? |
| Cricket Player | Kieron Pollard | What is the highest individual score by Kieron Pollard in a T20 international match? | Quel est le meilleur score individuel de Kieron Pollard lors d'un match international du T20 ? | एक T20 अंतर्राष्ट्रीय मैच में कोइरन पॉलर का बल्लचार क्या? | Was ist die höchste Einzelpunktzahl von Kieron Pollard in einem T20-Länderspiel? | Qual é a maior pontuação individual de Kieron Pollard em uma partida internacional T20? | ¿Cuál es la puntuación individual más alta de Kieron Pollard en un partido internacional del T20? |
| French Open | 2006 | Who did Justine Henin beat in the 2006 French Open women's singles final? | Qui a battu Justine Henin lors de la finale du simple féminin des Internationaux de France 2006 ? | 2006 के फ्रेंच ओपन महिला एकल फाइनल में जस्टिन हेनिन ने किसे हराया था? | Wen hat Justine Henin 2006 im Finale der French Open im Dameneinzel besiegt? | Quem Justine Henin venceu na final individual feminina do Aberto da França de 2006? | ¿A quién derrotó Justine Henin en la final individual femenina del Abierto de Francia 2006? |
| Country | Germany | What is the capital of Germany? | Quelle est la capitale de l'Allemagne ? | जर्मनी की राजधानी क्या है? | Was ist die Hauptstadt von Deutschland? | Qual é a capital da Alemanha? | ¿Cuál es la capital de Alemania? |

Figure 15: Sample questions from our question bank which we use to evaluate LLMs for reasoning by equivalence.

A.5.3 Additional Details on Controlled Experiment

For computing the rank of a multi-token word, we compute the rank of each one of its tokens and consider the minimum rank amongst them as the rank of multi-token word. Figure 18 shows conflict rate of all (anchor, non-anchor) language pairs with number of training steps.

A.6 Reasoning by Inheritance

A.6.1 Dataset

Fig 19 shows sample questions about abstract concepts and their specific instances from our dataset.

A.6.2 Qualitative Analysis

Fig 20 shows a few sample errors from various LLMs wherein they violate inheritance constraints by giving conflicting answers for parent and child concept, and across children of the same type.

A.7 Compositional Representation (CoRe)

A.7.1 Qualitative Analysis

Fig 21 shows few sample anchor and non-anchor language questions with their answers from XGLM-4.5B (baseline) and XGLM-4.5B+CoRe.

A.7.2 Effect of CoRe on a Downstream Task

The main focus of our work in “Reasoning by Equivalence” was to assess if LLMs exhibit consistent knowledge across languages, and we proposed CoRe to improve consistency. However, to evaluate CoRe’s impact on a downstream task we performed a small experiment for NLI task on XNLI (Conneau et al., 2018) dataset (on first 100 examples from validation set) with the same zero shot setup as XGLM paper (Lin et al., 2022) and observed comparable performance with and without CoRe (46-47 for En, 39-40 for De). However, we observed a significant reduction in inconsistency across predictions for parallel *En – De* NLI data points (28% inconsistency in predictions without CoRe vs 14% inconsistency with CoRe, wherein model’s prediction is said to be inconsistent if it predicts different labels for parallel datapoints across languages (*En* and *De* in this case)). We anticipate that incorporating CoRe during pre-training or continual training with larger dataset could yield further improvement in downstream task performance as well which we could not do because of compute constraints. We consider those experiments and further evaluation of CoRe across various different tasks

| Parent Concept | Child Concept | Lang. | Question | Sample answers from different LLMs | | | |
|----------------|---------------------|-------|---|------------------------------------|---|-------------------|--|
| Monument | Great Wall of China | En | How long is the Great Wall of China? | BLOOMZ -7B | 2400 km | Claude v2 | The Great Wall of China is 21,196 km (13,171 miles) long. |
| | | Fr | Quelle est la longueur de la Grande Muraille de Chine ? | | 2400 km | | La Grande Muraille de Chine fait environ 21 196 km de long. |
| | | Hi | चीन की महान दीवार कितनी लंबी है? | | 3,000 km | | चीन की महान दीवार की कुल लंबाई लगभग 21,196 किलोमीटर है। |
| | | De | Wie lang ist die Chinesische Mauer? | | 1949 | | Die Chinesische Mauer ist etwa 21.196 Kilometer lang. |
| | | Pt | Quanto tempo dura a Grande Muralha da China? | | over 2400 years | | A Grande Muralha da China tem uma extensão de aproximadamente 21.196 km. |
| | | Es | ¿Qué longitud tiene la Gran Muralla China? | | 18,000 km | | La Gran Muralla China tiene una longitud de aproximadamente 21.196 kilómetros. |
| Movie | Swades | En | Who starred as the lead actor in the movie Swades? | XGLM-7.5 B | Amitabh Bachchan. | Claude v1 Instant | Shah Rukh Khan |
| | | Fr | Qui a joué le rôle principal dans le film Swades ? | | Amitabh Bachchan. | | Shah Rukh Khan |
| | | Hi | स्वदेस फिल्म में मुख्य अभिनेता के रूप में किसने अभिनय किया था? | | दिलीप कुमार. | | अमिताभ बच्चन |
| | | De | Wer spielte die Hauptrolle im Film Swades? | | Swades. | | Shah Rukh Khan |
| | | Pt | Quem estrelou como ator principal no filme Swades? | | Amitabh Bachchan. | | Shah Rukh Khan |
| | | Es | ¿Quién interpretó al actor principal en la película Swades? | | Amitabh Bachchan. | | Shah Rukh Khan |
| Cricket Player | Kieron Pollard | En | What is the highest individual score by Kieron Pollard in a T20 international match? | Claude v3 Sonnet | 83 not out against Sri Lanka in 2021 | Claude v2 | 75 runs |
| | | Fr | Quel est le meilleur score individuel de Kieron Pollard lors d'un match international du T20 ? | | 83 non out contre le Sri Lanka en 2021 | | 119 |
| | | Hi | एक T20 अंतर्राष्ट्रीय मैच में कोरोन पोलर्ड का सर्वोच्च व्यक्तिगत स्कोर क्या है? | | 109 न (नावाद) | | 56 |
| | | De | Was ist die höchste Einzelpunktzahl von Kieron Pollard in einem T20-Länderspiel? | | 83 Runs gegen Sri Lanka im Jahr 2010 | | 75 |
| | | Pt | Qual é a maior pontuação individual de Kieron Pollard em uma partida internacional T20? | | 89 não eliminado contra Sri Lanka em 2010 | | 104 não fora |
| | | Es | ¿Cuál es la puntuación individual más alta de Kieron Pollard en un partido internacional del T20? | | 83 no out contra Argentina en 2022 | | 104 |
| French Open | 2006 | En | Who did Justine Henin beat in the 2006 French Open women's singles final? | Claude v1 Instant | Kim Clijsters | Claude v3 Sonnet | Justine Henin beat Svetlana Kuznetsova in the 2006 French Open women's singles final. |
| | | Hi | 2006 के फ्रेंच ओपन महिला एकल फाइनल में जस्टिन हेनिन ने किसे हराया था? | | 2006 के फ्रेंच ओपन महिला एकल फाइनल में जस्टिन हेनिन ने निकोल गाविडिसागा को हराया था। | | जस्टिन हेनिन ने 2006 के फ्रेंच ओपन महिला एकल फाइनल में रूसी निकोल गाविडिसागा को हराया था। |
| | | De | Wen hat Justine Henin 2006 im Finale der French Open im Dameneinzel besiegt? | | Amélie Mauresmo | | Justine Henin besiegte Svetlana Kusnetsova im Finale der French Open 2006 im Dameneinzel. |
| | | Pt | Quem Justine Henin venceu na final individual feminina do Aberto da França de 2006? | | Justine Henin venceu a italiana Amélie Mauresmo na final individual feminina do Aberto da França de 2006. | | Justine Henin venceu Svetlana Kusnetsova na final individual feminina do Aberto da França de 2006. |
| | | Es | ¿A quién derrotó Justine Henin en la final individual femenina del Abierto de Francia 2006? | | A Amélie Mauresmo | | Justine Henin derrotó a Svetlana Kusnetsova en la final individual femenina del Abierto de Francia 2006. |
| | | | | | | | |
| Country | Germany | En | What is the capital of Germany? | XGLM-7.5 B | Berlin. | BLOOMZ -7B | Berlin |
| | | Fr | Quelle est la capitale de l'Allemagne ? | | Berlin. | | Berlin |
| | | Hi | जर्मनी की राजधानी क्या है? | | बर्लिन. | | Berlin |
| | | De | Was ist die Hauptstadt von Deutschland? | | Berlin. | | Berlin |
| | | Pt | Qual é a capital da Alemanha? | | Berlin. | | Berlin |
| | | Es | ¿Cuál es la capital de Alemania? | | Berlin. | | Berlin |

Figure 16: Sample errors from different LLMs on DIL task. Red and Green colored cells highlight conflicting and non-conflicting answers with the anchor language (En) answer, respectively.

| Parent Concept | Child Concept | Question | Lang. | Sample answers from different LLMs | | | | | |
|----------------|------------------------|--|-------|------------------------------------|--|------------------|---|--|--|
| Monument | Great Wall of China | How long is the Great Wall of China? | En | Claude v1 Instant | The Great Wall of China is approximately 13,170 miles (21,196 kilometers) long. | Claude v2 | The Great Wall of China is 21,196 km (13,171 miles) long. | La Grande Muraille de Chine fait environ 21 000 kilomètres de long. | Die Große Mauer ist über 21.000 Kilometer lang. |
| | | | Fr | | La Grande Muraille de Chine mesure environ 21 000 kilomètres de long. | | | | |
| | | | Hi | | चीन की महान दीवार की लंबाई लगभग 21,196 किलोमीटर है। | | | | |
| | | | De | | Die Chinesische Mauer ist etwa 21.196 Kilometer lang. | | | | |
| | | | Pt | | A Grande Muralha da China mede aproximadamente 21 mil quilômetros de comprimento. | | | | |
| | | | Es | | La Gran Muralla China mide aproximadamente 21,196 kilómetros de largo. | | | | |
| Movie | Jo Jeeta Wohi Sikandar | Who directed the music for the movie Jo Jeeta Wohi Sikandar? | En | Claude v1 Instant | A.R. Rahman | Claude v3 Sonnet | Jatin-Lalit composed the music for the 1992 Hindi film Jo Jeeta Wohi Sikandar. | La musique du film Jo Jeeta Wohi Sikandar a été composée par Uttam Singh. | Die Musik für den Film Jo Jeeta Wohi Sikandar wurde von Bappi Lahiri komponiert. |
| | | | Fr | | A. R. Rahman | | | | |
| | | | Hi | | फिल्म 'जो जीता वोही सिंकंदर' के लिए संगीत का निर्देशन अनु मलिक ने किया था। | | | | |
| | | | De | | Der indische Filmmusikkomponist Anu Malik komponierte die Musik für den Film Jo Jeeta Wohi Sikandar. | | | | |
| | | | Pt | | Anees Bazmee | | | | |
| | | | Es | | A. R. Rahman dirigió la música para la película Jo Jeeta Wohi Sikandar. | | | | |
| Cricket Player | Kieron Pollard | What is the highest individual score by Kieron Pollard in a T20 international match? | En | Claude v1 Instant | 75 runs against Sri Lanka in 2012 | Claude v3 Sonnet | 83 not out against Sri Lanka in 2021 | Le score individuel le plus élevé de Kieron Pollard dans un match international T20 est 83 non out contre l'Argentine en 2022. | Die Musik für den Film Jo Jeeta Wohi Sikandar wurde von Bappi Lahiri komponiert. |
| | | | Fr | | 127 | | | | |
| | | | Hi | | कियरोन पोलार्ड ने टी20 इंटरनेशनल मैच में सबसे ज्यादा अंकों ले रन बनाए हैं 75 रन। | | | | |
| | | | De | | 127 gegen Sri Lanka im Jahr 2019 | | | | |
| | | | Pt | | 127 contra o Sri Lanka em 2012 | | | | |
| | | | Es | | 127 | | | | |
| French Open | 2006 | Who did Justine Henin beat in the 2006 French Open women's singles final? | En | Claude v1 Instant | Svetlana Kuznetsova | Claude v3 Sonnet | Justine Henin beat Svetlana Kuznetsova in the 2006 French Open women's singles final. | Justine Henin besiegte Swetlana Kusnezowa im Finale der French Open 2006 im Dameneinzel. | Justine Henin venceu Svetlana Kuznetsova na final de simples feminino do Aberto da França de 2006. |
| | | | Hi | | जस्टिन हेनिन ने 2006 फ्रेंच ओपन महिला एकल फाइनल में मारिया शारपेवा को हराया था। | | | | |
| | | | De | | Justine Henin besiegte Katarina Srebotnik im Damen-Einzel-Finale der French Open 2006. | | | | |
| | | | Pt | | Maria Sharapova | | | | |
| | | | Es | | A Kim Clijsters en la final femenina individual del Abierto de Francia 2006. | | | | |

Figure 17: Sample errors from different LLMs on DOL task. Red and Green colored cells highlight conflicting and non-conflicting answers with the anchor language (En) answer, respectively.

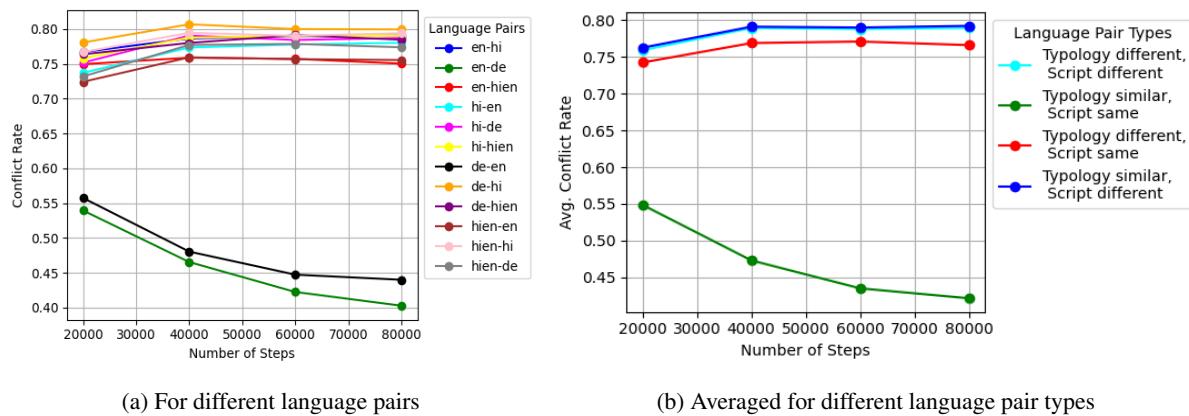


Figure 18: Figure showing how conflict rate changed during training in controlled experiment.

| Parent Concept | Child Concept | Language | Question (Parent Concept) | Question (Child Concept) |
|-----------------|-----------------------------------|----------|---|---|
| Fruit | Banana | De | Kann die Frucht laufen? | Kann die Fruchtbane laufen? |
| Fruit | Banana | Hi | क्या फूट चल सकता है? | क्या फूट केला चल सकता है? |
| Fruit | Lychee | Hi | क्या फूट चल सकता है? | क्या फूट लीची चल सकती है? |
| Books | Dhalgren | En | Can the Book eat? | Can the Book Dhalgren eat? |
| Books | Dhalgren | Hi | क्या किताब खा सकती है? | क्या बुक धालग्रेन खा सकता है? |
| Books | The Great Gatsby | Hi | क्या किताब खा सकती है? | क्या किताब द गेट गैटसबी खा सकती है? |
| Monument | Airavatesvara Temple (Tamil Nadu) | En | Can the Monument smell other things? | Can the Monument Airavatesvara Temple (Tamil Nadu) smell other things? |
| Monument | Airavatesvara Temple (Tamil Nadu) | De | Kann das Monument andere Dinge riechen? | Kann das Monument Airavatesvara-Tempel (Tamil Nadu) andere Dinge riechen? |
| Monument | Great Wall of China (China) | De | Kann das Monument andere Dinge riechen? | Kann das Monument Chinesische Mauer (China) andere Dinge riechen? |
| Mountain Ranges | Shivaliks | Pt | A Cordilheira pode voar? | Os Shivaliks da Cordilheira podem voar? |
| Mountain Ranges | Shivaliks | Es | ¿Puede volar la Cordillera? | ¿Pueden volar los Shivaliks de la Cordillera? |
| Mountain Ranges | Himalaya | Es | ¿Puede volar la Cordillera? | ¿Puede volar la Cordillera del Himalaya? |

Figure 19: Dataset sample of questions about abstract concepts and their specific instances which we use to evaluate LLMs for reasoning by inheritance.

| Parent Concept | Child Concept | Language | Question (Parent Concept) | Question (Child Concept) | Sample answers from different LLMs | | | Comment |
|-----------------|-----------------------------------|----------|---|---|------------------------------------|---|--|--|
| | | | | | Model | To Parent Concept Question | To Child Concept Question | |
| Fruit | Banana | De | Kann die Frucht laufen? | Kann die Fruchtbane laufen? | Claude v2 | Nein, eine Frucht kann nicht laufen, da sie eine Frucht ist und keine Beine zum Laufen hat. | Nein, die Fruchtbane kann nicht laufen, da sie eine Frucht ist und keine Beine zum Laufen hat. | LLM is able to exhibit reasoning by inheritance for "banana" in De, but not in Hi. In Hi though, it is able to reason correctly for "Lychee", a different specific instance of "Fruit", but not for "banana". |
| Fruit | Banana | Hi | क्या फूट चल सकता है? | क्या फूट केला चल सकता है? | | नहीं, फूट चल नहीं सकता। | हाँ, फूट केला चल सकता है। | |
| Fruit | Lychee | Hi | क्या फूट चल सकता है? | क्या फूट लीची चल सकती है? | Claude v2 | नहीं, फूट लीची चल नहीं सकती। | नहीं, फूट लीची चल नहीं सकती। | LLM is able to exhibit reasoning by inheritance for "Dhalgren" in En, but not in Hi. In Hi though, it is able to reason correctly for "The Great Gatsby", a different specific instance of "Book", but not for "Dhalgren". |
| Books | Dhalgren | En | Can the Book eat? | Can the Book Dhalgren eat? | | No, the Book cannot eat because it is an inanimate object. | No, the Book Dhalgren cannot eat since it is an inanimate object. | |
| Books | Dhalgren | Hi | क्या किताब खा सकती है? | क्या बुक धालग्रेन खा सकता है? | BLOOMZ-7B | नहीं, किताब खाना नहीं खा सकती। | हाँ, बुक धालग्रेन खा सकता है। | LLM is able to exhibit reasoning by inheritance for "Airavatesvara Temple (Tamil Nadu)" in En, but not in De. In De though, it is able to reason correctly for "Great Wall of China (China)", a different specific instance of "Monument", but not for "Airavatesvara Temple (Tamil Nadu)". Answers are also not consistent across children of the same type, within and across languages. |
| Books | The Great Gatsby | Hi | क्या किताब खा सकती है? | क्या किताब द गेट गैटसबी खा सकती है? | | नहीं, किताब नहीं खा सकती। | नहीं, किताब नहीं खा सकती। | |
| Monument | Airavatesvara Temple (Tamil Nadu) | En | Can the Monument smell other things? | Can the Monument Airavatesvara Temple (Tamil Nadu) smell other things? | XGLM-7.5B | Yes | Yes | LLM is able to exhibit reasoning by inheritance for "Shivaliks" in Pt, but not in Es. In Es though, it is able to reason correctly for "Himalaya", a different specific instance of "Mountain Ranges", but not for "Shivaliks". |
| Monument | Airavatesvara Temple (Tamil Nadu) | De | Kann das Monument andere Dinge riechen? | Kann das Monument Airavatesvara-Tempel (Tamil Nadu) andere Dinge riechen? | | no | Yes | |
| Monument | Great Wall of China (China) | De | Kann das Monument andere Dinge riechen? | Kann das Monument Chinesische Mauer (China) andere Dinge riechen? | | no | no | |
| Mountain Ranges | Shivaliks | Pt | A Cordilheira pode voar? | Os Shivaliks da Cordilheira podem voar? | XGLM-7.5B | A Cordilheira pode voar. | Os Shivaliks da Cordilheira podem voar. | LLM is able to exhibit reasoning by inheritance for "Shivaliks" in Pt, but not in Es. In Es though, it is able to reason correctly for "Himalaya", a different specific instance of "Mountain Ranges", but not for "Shivaliks". |
| Mountain Ranges | Shivaliks | Es | ¿Puede volar la Cordillera? | ¿Pueden volar los Shivaliks de la Cordillera? | | Yes, it can fly. | No. | |
| Mountain Ranges | Himalaya | Es | ¿Puede volar la Cordillera? | ¿Puede volar la Cordillera del Himalaya? | | Yes, it can fly. | Yes, it can. | |

Figure 20: Sample errors from various LLMs wherein they violate inheritance constraints. Red and Green colored cells highlight conflicting and non-conflicting answers with the parent concept question's answer, respectively.

on NLP benchmarks as part of future work.

| Anchor | | Non-Anchor | | XGLM-4.5B (Baseline) | | XGLM-4.5B + CoRe | |
|--------|---|------------|--|----------------------|---|------------------|--|
| Lang | Anchor Question | Lang | Non-Anchor Question | Anchor Answer | Non-Anchor Answer | Anchor Answer | Non-Anchor Answer |
| en | When was Mukesh Singh born? | de | Wann wurde Mukesh Singh geboren? | March 15, 1990. | Mukesh Singh wurde am 15. März 1985 in Pokhara, Nepal, geboren. | March 15, 1990. | March 15, 1990. |
| en | How tall is Mohammad Zaheer? | de | Wie groß ist Mohammad Zaheer? | 6 feet 2 inches | 6 Fuß 7 Zoll | 6 feet 5 inches | 6 Fuß 5 Zoll |
| en | Who did Isla Kirkland take over from as CEO of Zetabyte Corporation in 2020? | de | Von wem hat Isla Kirkland 2020 die Position des CEO der Zetabyte Corporation übernommen? | Daniel Jones | Amanda Zhou | Daniel Jones | Daniel Jones |
| en | In which Indian city was Madhuri Deshpande born? | de | In welcher indischen Stadt wurde Madhuri Deshpande geboren? | Hyderabad | Vijayawada | Hyderabad | Hyderabad |
| en | What is the headquarters location of WattGrid Technologies? | hi | WattGrid Technologies का मुख्यालय क्या है? | Shanghai | शीजिंग, शीन | Shanghai | शंचाई, चीन |
| en | What is the height in meters of the tallest building called Maxima Tower in Ultima? | hi | अल्टिमा में मैक्सिमा टावर नामक सबसे ऊची इमारत की मीटर में ऊचाई कितनी है? | 250 meters | 450 मीटर | 250 meters | 250 मीटर |
| en | In what city and country was Sanjay Kapoor born? | hien | sanjay kapur kaa janm kis shahar or desh main hua tha? | Mumbai, India | sanjay kapur kaa janm 15 march 1985 ko kapurkapur, bharat main hua tha. | Mumbai, India | sanjay kapur kaa janm 15 march 1985 ko mumbai, maharashtra main hua tha. |
| en | Which team did Nepal, led by Bikram Khatri, defeat in the 2020 ACC Eastern Region T20 tournament final? | hien | 2020 ACC purvi kshetra T20 tournament ke final main bikram khatri ke netrity vaali ne kis team ko haraaya? | Malaysia | australia | Malaysia | malaysia |
| hi | संदीप सिंह कितने साल के थे जब उन्होंने नेपाली राष्ट्रीय टीम के लिए पदार्पण किया था? | en | How old was Sandeep Singh when he made his debut for the Nepal national team? | 16 | 19 years old | 16 | 16 |
| hi | जॉन डेविडसन किस धरेन फ्रिकेट टीम का प्रतिनिधित्व करते हैं? | en | Which domestic cricket team does John Davidson represent? | न्यू साउथ वेल्स | Queensland | न्यू साउथ वेल्स | New South Wales |
| hien | srianka ke ubharte hue ballebaajhi khiladi ramesh tendulkar ne apane karivar main ab tak kitne list e match khеле ha? | hi | श्रीलंका के उभरते हुए बल्लेबाजी क्षितिजी रमेश ठेंडुलकर ने अपने करियर में अब तक कितने लिस्ट ए मैच खेले हैं? | 30 | 250 | 25 | 25 |
| hien | jhang vei ne apane test padaaran par koun sa jersey number pahana tha? | hi | आंग वेँ ने अपने टेस्ट पदार्पण पर कौन सा जर्सी नवर पहना था? | 10 | 19 | 10 | 10 |

Figure 21: Sample anchor and non-anchor language questions wherein we get consistent answers after training with CoRe.

UNLOCKING EFFICIENT, SCALABLE, AND CONTINUAL KNOWLEDGE EDITING WITH BASIS-LEVEL REPRESENTATION FINE-TUNING

Tianci Liu¹, Ruirui Li², Yunzhe Qi³, Hui Liu², Xianfeng Tang², Tianqi Zheng², Qingyu Yin²,

Monica Cheng², Jun Huan⁴, Haoyu Wang⁵, Jing Gao¹

¹Purdue University ²Amazon ³UIUC ⁴AWS AI Lab ⁵SUNY Albany

¹{liu3351, jinggao}@purdue.edu ²ruirul@amazon.com ⁵hwang28@albany.edu

ABSTRACT

Large language models (LLMs) have achieved remarkable performance on various natural language tasks. However, they are trained on static corpora and their knowledge can become outdated quickly in the fast-changing world. This motivates the development of knowledge editing methods designed to update certain knowledge in LLMs without changing unrelated others. To make selective edits, previous efforts often sought to update a small amount of parameters in some specific layer(s) of a LLM. Nonetheless, in challenging scenarios, they still fall short in making successful edits while preserving knowledge irrelevant to the updates simultaneously, resulting in a notable *editing-locality* trade-off. In this work, we question if the trade-offs are caused by the fact that parameter-based updates have a global effect, i.e., edited parameters affect all inputs indiscriminately. In light of this, we explore the feasibility of representation fine-tuning, which applied some linear update to a few representations in a learned subspace, for knowledge editing. While being effective to enhance an LLM’s general ability as demonstrated in the previous work, we theoretically show that this linear update imposes a tension in editing-locality trade-off. Subsequently, BaFT is proposed to break the linearity. BaFT computes a weight for each basis that spans a dimension of the subspace based on the input representation. This input-dependent weighting mechanism allows BaFT to manage different types of knowledge in an adaptive way, thereby achieving a better editing-locality trade-off. Experiments on three LLMs with five editing benchmarks in diverse scenarios show the superiority of our method.

1 INTRODUCTION

Language models (LMs) parameterized by deep neural networks (Vaswani et al., 2017; Lewis et al., 2019; Radford et al., 2019; Brown et al., 2020) have thrived in producing fluent and meaningful texts on diverse natural language generation and classification tasks (See et al., 2019; Raffel et al., 2020; Ji et al., 2023). These successes underscore the versatility of LMs, establishing them as the foundations for different natural language processing applications (Bommasani et al., 2021; Zhou et al., 2023). Additionally, with model sizes continually increasing, large language models (LLMs) have demonstrated unprecedented abilities to follow natural language instructions (Dong et al., 2022b; Ouyang et al., 2022), empowering zero-shot adaptations to unseen tasks (Kojima et al., 2022), and paving the way towards artificial general intelligence (Bubeck et al., 2023).

Despite their remarkable performance, the real-world deployment of LLMs remains largely unresolved. While LLMs can understand a wide range of contexts, they can only provide feedback based on the *static* knowledge from the data on which they were trained. In a fast-changing world, most knowledge quickly becomes outdated. This could amplify critical issues such as making factual fallacy (De Cao et al., 2021) or producing harmful generations (Hartvigsen et al., 2022).

As a remedy, *knowledge editing*, whose goal is to update an LLM with some specific new knowledge without hurting irrelevant knowledge, has been proposed (Wang et al., 2023; Zhang et al., 2024b). Early effort of full fine-tuning proved ineffective as it also disrupted irrelevant knowledge (Wang

et al., 2023), leading to an *editing-locality* trade-off. Here *locality* refers to the ability to maintain the knowledge that is irrelevant to the updates. To achieve a good locality, the model update needs to be *selective* and should rely on a small number of parameters (Wang et al., 2023), and thus parameter-efficient fine-tuning (PEFT) methods like AdaLoRA (Zhang et al., 2023) have shown good performance (Wu et al., 2023). On the other hand, Huang et al. (2023); Dong et al. (2022a) restricted updates to specific feed-forward network (FFN) layer that served for knowledge storing (Dai et al., 2021). Meng et al. (2022a;b) refined the process through a *locate-and-edit* paradigm which involves an additional *locating* stage to identify which layer the target knowledge is stored. Nonetheless, these methods still exhibit a certain editing-locality trade-off, regardless of whether locating is performed. We note that these methods are parameter-based and have a global effect, i.e., the edited parameters affect *all* inputs indiscriminately. This observation challenges to what extent an editing can truly benefit from the targeted effort to identify “better” parameters that “memorize” certain knowledge (Hase et al., 2024). In other words, it is an open question *if such trade-offs are due to the coarse control of global parameter-based updates*.

This paper, following Hernandez et al. (2023) that modifies LLM knowledge by updating representations, explores *selective* representation-based knowledge editing, and paves a way for an affirmative answer to the above question. Our work is built upon ReFT (Wu et al., 2024) that fine-tunes a few representations in a low-rank linear subspace, and performs on par with PEFT methods such as LoRA family (Hu et al., 2021; Zhang et al., 2023; Ding et al., 2023) and others (Houlsby et al., 2019; Chen et al., 2024). Unlike parameter-based updates that apply to all inputs, ReFT only alters representations at some locations. Consequently, ReFT can achieve a better editing-locality trade-off than parameter-based updates. Notwithstanding, in spite of this promising result, the *subspace-level linearity* still restricts ReFT from providing precise enough updates for knowledge editing.

Specifically, ReFT applies the linear update in the subspace for *all* selected representations. While being effective to enhance an LLM’s general ability such as commonsense reasoning (Wu et al., 2024), this subspace-level control can be too coarse for knowledge editing. As a consequence, when ReFT achieves high editing performance, certain unrelated knowledge may be modified incorrectly, provably jeopardizing its locality. This insight is formalized in Sec 2, where a theoretical analysis on this inherent tension is derived, based on two reasonable assumptions on how representations convey different knowledge. Notably, *our analysis reveals an intrinsic limitation of linear representation fine-tuning. It not only holds for knowledge editing, but also applies to other tasks that require selective updates such as continual learning and machine unlearning, and can be of independent interest to these communities*. This theoretical result is one of the main contributions of this paper.

In light of this insight, we derive BaFT, a more precise representation fine-tuning method for knowledge editing. Noting that the subspace is spanned by a group of bases vectors, BaFT instead learns a *basis-level* update. This involves computing a weight for each basis for a given representation, then learning a linear update along this basis. Since each basis spans a rank-1 subspace, BaFT is a generalization of ReFT, in the sense that if all bases use the same constant weight 1, BaFT reduces to ReFT. By using different weights combinations on distinct types of knowledge, BaFT can manage them in a more adaptive way. When auxiliary locality information (e.g., what knowledge should *not* be updated) is available, BaFT can freely restrict the impact of unimportant bases only, while ReFT needs to regulate the whole subspace rigidly. This flexibility makes BaFT highly suitable for knowledge editing and performs on par with the strongest baseline that relies on external memories to memorize new knowledge and requires 10-20 times more parameters. In conclusion, *BaFT, as a new representation fine-tuning method, successfully reaches a better editing-locality trade-off while maintaining the parameter efficiency of ReFT*. This is another main contribution of this work.

Our paper is organized as follows. Sec 2 details the proposed BaFT. Extensive experimental results in Sec 3 demonstrate the superiority of our method for conducting knowledge editing at much better parameter efficiency than existing methods. In the remaining part of this paper, we review related works in Sec 4, and conclude the paper in Sec 5.

2 PROPOSED METHOD

Grounded in a theoretical analysis, we show that the linearity nature of existing representation fine-tuning method induces an inherent limitation on its editing-locality trade-off. We then propose BaFT towards a fine-grained controlled representation fine-tuning in accordance with knowledge editing.

2.1 PRELIMINARIES

Given input $\mathbf{x} = (x_1, \dots, x_n)$, where each $x_i \in \mathcal{V}$ is a token from vocabulary \mathcal{V} , a language model (LM) parameterized by θ assigns probability $p_\theta(\mathbf{x})$ using the chain rule (Bengio et al., 2000):

$$p_\theta(\mathbf{x}) = \prod_{i=1}^n p_\theta(x_i | x_1, \dots, x_{i-1}) \triangleq \prod_{i=1}^n p_\theta(x_i | \mathbf{x}_{<i}),$$

where $p_\theta(x_i | \mathbf{x}_{<i})$ is the predicted distribution of the next token x_i over \mathcal{V} given previous $\mathbf{x}_{<i}$. In specific, for an L -layer LM, let $\mathbf{h}_i^{(l)}$ denote the intermediate *representation* of the i -th token at the l -th layer. The predicted distribution is given by softmax regression parameterized by \mathbf{W} at layer L :

$$p_\theta(x_i | \mathbf{x}_{<i}) = \text{softmax}(\mathbf{W}\mathbf{h}_i^{(L)}).$$

To generate a sentence \mathbf{x} , the LM repeatedly computes $p_\theta(x_i | \mathbf{x}_{<i})$ and draws x_i from it; then x_i is fed back into the LM as part of the inputs for future steps. The generation process completes if a special token that marks the end of the sentence is returned, or the maximum length is reached.

Knowledge Editing aims to incorporate new provided knowledge into a pre-trained LM while preserving other existing knowledge that shouldn't be modified. Formally speaking, any knowledge can be represented in natural language with a textual pair (\mathbf{x}, \mathbf{y}) , where \mathbf{x} entails some *subject* and *relation*, and \mathbf{y} refers to the corresponding *object*. For instance, given \mathbf{x} being *The current president of United States is*, \mathbf{y} can be *Joe Biden*. Knowledge editing seeks to maximize the chance of an LLM responding with \mathbf{y} given \mathbf{x} , while satisfying the following additional criteria at the same time (Zhang et al., 2024b; Liu et al., 2025a): (1) **Generality**: there are different ways to express *US president*, wherefore the edited model should generalize. (2) **Portability**: relevant knowledge such as *the first lady of United States* should be updated as well. (3) **Locality**: irrelevant knowledge such as *the prime minister of United Kingdom* should not be affected. Notably, such requirements of modifying only specific internal knowledge in a LM has been proved challenging. As revealed in previous works (Zhang et al., 2024b), this process should update only a minimal amount of parameters.

Representation Fine-tuning (ReFT), proposed by Wu et al. (2024), is a recent parameter-efficient fine-tuning (PEFT) method that outperformed other approaches such as LoRA in updating pre-trained LM on several tasks with much less parameters. Building upon the so-called *linear representation hypothesis* (Park et al., 2023) which presumes that *concepts are encoded in linear subspace of representations*, ReFT learns *low-rank linear updates* on representations. In particular, to update the d -dimensional representation $\mathbf{h}_i^{(l)}$ at layer l for the i -th token, ReFT learns

$$\Phi_l(\mathbf{h}_i^{(l)}; \phi_l) = \mathbf{h}_i^{(l)} + \mathbf{R}_l^\top (\mathbf{A}_l \mathbf{h}_i^{(l)} + \mathbf{b}_l - \mathbf{R}_l \mathbf{h}_i^{(l)}), \quad (1)$$

where $\phi_l = (\mathbf{R}_l, \mathbf{A}_l, \mathbf{b}_l)$ are learnable parameters added to layer l . Here $\mathbf{R}_l \in \mathbb{R}^{r \times d}$ is a low-rank matrix (i.e., $r \ll d$) containing mutually orthogonal rows that specifies a subspace to make the update, and $(\mathbf{A}_l, \mathbf{b}_l)$ predicts the *updated* representation in this subspace. Finally, ReFT requires hyper-parameter $I \subset [n]$ to specify which locations need updates. Put together, ReFT intervenes the layer l 's output by

$$\mathbf{h}_i^{(l)} \leftarrow \left(\Phi_l(\mathbf{h}_i^{(l)}) \text{ if } i \in I \text{ else } \mathbf{h}_i^{(l)} \right)_{i \in 1, \dots, n}.$$

From now on, we omit indices i, l when discussing how a representation \mathbf{h} is intervened for brevity.

2.2 EDITING KNOWLEDGE BY FINE-TUNING REPRESENTATIONS

ReFT has demonstrated impressive performance on tasks such as commonsense reasoning that largely rely on an LLM's ability to understand and generate text by updating *just a few* (i.e., those in I) representations. However, it is unknown whether this lightweight approach can benefit knowledge editing, which requires modifying some selective internal knowledge. Here, we show that the linearity nature of ReFT limits its editing and locality performance. In specific, for all inputs, ReFT applies the *same* linear update without distinction:

$$\Phi(\mathbf{h}) = \mathbf{h} + \mathbf{R}^\top (\mathbf{A}\mathbf{h} + \mathbf{b} - \mathbf{R}\mathbf{h}) = \underbrace{(\mathbf{I} + \mathbf{R}^\top (\mathbf{A} - \mathbf{R}))}_{\text{weight}} \mathbf{h} + \underbrace{\mathbf{R}^\top \mathbf{b}}_{\text{bias}}.$$

The coarse control from the linear ReFT makes it less suitable for knowledge editing for two reasons.

First, ReFT uses its learned subspace for editing in a predetermined manner, regardless of varying levels of learning difficulty for different types of knowledge. This can lead to sub-optimal performance. As an evidence, we fit a rank-12 subspace for ReFT and checked how many dimensions (bases) contribute negligible updates, as a measure of *dimension redundancy*. To this end, we count for each dimension, if its update magnitude is less than M times of the maximal dimension. Fig 1 shows these results. We noted that the dimension redundancy indeed varies on different types of knowledge.

Second, the linearity of ReFT leads to an inherent editing-locality trade-off: it is challenging to maintain good *generality* and *locality* at the same time. Formally, given some knowledge involves subject s , relation r , and object o that can be updated by ReFT, we make the following assumptions.

Assumption 2.1. Let text x encodes s, r . Since the knowledge can be edited by ReFT, text y generated by the LM will convey o if its intermediate representation takes some targeted value t .

Assumption 2.2. (Hartvigsen et al., 2024) For any \mathbf{h} carrying some knowledge, there exists a positive $\varepsilon(\mathbf{h})$ -radius ℓ_2 ball $B(\mathbf{h}, \varepsilon(\mathbf{h}))$ around \mathbf{h} such that any $\mathbf{h}' \in B(\mathbf{h}, \varepsilon(\mathbf{h}))$ conveys the same knowledge, we refer to $B(\mathbf{h}, \varepsilon(\mathbf{h}))$ as a *stable-ball* of \mathbf{h} .

We provide a few clarifications on the two assumptions. The first assumes that a piece of knowledge can be generated (retrieved) from some associated representation. The second, as in Hartvigsen et al. (2024), assumes that the knowledge is locally stable around its representation, so that a small perturbation won't change the carried knowledge. Under these two assumptions, The following Thm 2.3 reveals a tension between maintaining good *generality* and *locality* simultaneously, with its proof deferred to App B.1.

Theorem 2.3. When fine-tuning an LM, ReFT learns to update the old representation \mathbf{h}_0 to targeted $t = \Phi(\mathbf{h}_0)$. If ReFT maintains good generality such that $\forall \mathbf{h} \in B(\mathbf{h}_0, \varepsilon(\mathbf{h}_0))$,

$$\|\Phi(\mathbf{h}) - \Phi(\mathbf{h}_0)\| = \|\Phi(\mathbf{h}) - t\| < \varepsilon(t),$$

where $\|\cdot\|$ denote the ℓ_2 norm. Then for any irrelevant input \mathbf{h}_{ir} with a small stable-ball radius

$$\varepsilon(\mathbf{h}_{ir}) < \frac{\|t - \mathbf{h}_0\| - (\varepsilon(t) + \varepsilon(\mathbf{h}_0))}{\varepsilon(t) + 2\varepsilon(\mathbf{h}_0)} \varepsilon(\mathbf{h}_0),$$

and is not too far from \mathbf{h}_0 such that

$$\|\mathbf{h}_{ir} - \mathbf{h}_0\| = \varepsilon(\mathbf{h}_{ir}) + \varepsilon(\mathbf{h}_0),$$

ReFT will output $\Phi(\mathbf{h}_{ir}) \notin B(\mathbf{h}_{ir}, \varepsilon(\mathbf{h}_{ir}))$ and break its locality guarantee.

Intuitively speaking, Thm 2.3 formalizes that ReFT update has to be *large* enough to make successful edit; and *smooth* enough to achieve good generality. Then, due to its linearity, it will inevitably hurt the locality of some irrelevant knowledge. This limitation does not rely on the specific r (i.e., subspace rank) being used. In summary, ReFT is less suitable for knowledge editing because of the two limitations, which motivates BaFT as presented in the next section.

2.3 BAFT: BASIS-LEVEL REPRESENTATION FINE-TUNING

Given the two limitations from *linearity*, i.e. using the whole linear *subspace* to update all representation without distinction, and the finding of dimension (basis) redundancy, we propose to take the importance of each dimension into account. Since in ReFT, the subspace is parameterized by a set of orthogonal bases vectors, we assign each *basis* a *learnable weight* to determine how much it contributes to the current editing. This input-dependent weighting mechanism makes our method applies a non-linear update. We dub our method *basis-level representation fine-tuning* (BaFT).

To be more specific, at a layer where ReFT takes place, we learn an r -dimensional update by

$$\Phi(\mathbf{h}) = \mathbf{h} + \sum_{k=1}^r w_k(\mathbf{h}) \mathbf{r}_k (\mathbf{a}_k^\top \mathbf{h} + b_k - \mathbf{r}_k^\top \mathbf{h}), \quad (2)$$

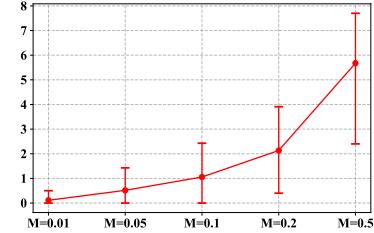


Figure 1: Averaged (w/ max-min range) number of *redundant* dimensions (which have update M times smaller than maximal values), in a rank-12 ReFT update.

where $\mathbf{r}_1, \dots, \mathbf{r}_r$ are r d -dimensional orthogonal bases, $\mathbf{a}_1, \dots, \mathbf{a}_r$ and b_1, \dots, b_r are r arbitrary vectors and scalars, respectively. Finally, $w_k(\mathbf{h}) \in [0, 1]$ are r learnable weights. Put together, $w_k(\mathbf{h})(\mathbf{a}_k^\top \mathbf{h} + b_k - \mathbf{r}_k^\top \mathbf{h})$ predicts the magnitude of update along direction of basis \mathbf{r}_k , and BaFT combines r total updates to form the final intervention. Fig 4 illustrates the overall flow of BaFT.

While appears distinct, Lem 2.4 shows that BaFT generalizes ReFT. See its proof in App B.3.

Lemma 2.4. *Let $\mathbf{R} = [\mathbf{r}_1; \dots; \mathbf{r}_r] \in \mathbb{R}^{r \times d}$, $\mathbf{A} = [\mathbf{a}_1, \dots, \mathbf{a}_r] \in \mathbb{R}^{r \times d}$, $\mathbf{b}^\top = (b_1, \dots, b_r)$, and $\mathbf{W}(\mathbf{h}) = \text{diag}(w_1(\mathbf{h}), \dots, w_r(\mathbf{h}))$ be a diagonal matrix. BaFT in Eqn (2) can be expressed as*

$$\Phi(\mathbf{h}) = \mathbf{h} + \mathbf{R}^\top \mathbf{W}(\mathbf{h}) (\mathbf{A}\mathbf{h} + \mathbf{b} - \mathbf{R}\mathbf{h}). \quad (3)$$

When using constant weighting $\mathbf{W}(\mathbf{h}) = \mathbf{I}$, BaFT reduces to ReFT.

2.4 TRAINING OBJECTIVE OF BAFT

We end this section by detailing the training of BaFT. For consistency we use ϕ_l to denote the collection of learnable parameters at layer l : \mathbf{R} , \mathbf{A} , \mathbf{b} , and newly introduced parameters in \mathbf{W} . Given a set of pre-specified layers C_l that need interventions, we optimize the collection of all learnable parameters $\phi = \{\phi_l\}_{l \in C_l}$ using the following losses.

Teacher-forcing Loss. Following Wu et al. (2024), we train BaFT with a language modeling objective, and minimize the cross-entropy loss with teacher-forcing (Lamb et al., 2016) at output positions

$$L_1(\phi) \triangleq - \sum_{i=1}^m \log p_\theta(y_i | \mathbf{x}\mathbf{y}_{<i}; \phi),$$

where the intervention is applied to the last P positions in \mathbf{x} , together with all entries in \mathbf{y} .

Incremental Load Balancing Loss. When editing multiple pieces of knowledge, different bases need, on average, balanced weights. Otherwise, using a few fixed bases for *all* edits is equivalent to using a *fixed* subspace spanned by these bases, and BaFT will reduce to a smaller ReFT. To avoid this reduction, inspired by the sparse mixture of expert (Shazeer et al., 2017; Fedus et al., 2022), we regularize the *squared coefficient of variation* of $(w_1(\mathbf{h}), \dots, w_r(\mathbf{h}))$. However, as new knowledge may emerges one by one, making direct average over multiple samples infeasible, we compute the metric in an incremental way. Namely, when editing the t -th knowledge, we minimize

$$\mathcal{R}_{\text{bal}}(\phi) \triangleq \sum_{k=1}^r \frac{(\bar{w}_k(t) - \bar{w}(t))^2}{(r-1)\bar{w}(t)}, \quad \text{where } \bar{w}(t) = \frac{1}{r} \sum_{k=1}^r \bar{w}_k(t),$$

and $\bar{w}_k(t)$ averages weights w_k over the *current* and *past* training samples at selected positions. For incremental optimization, we only minimize $\mathcal{R}_{\text{bal}}(\phi)$ with respect to the current weight on the t -th knowledge, as highlighted by expressing $\bar{w}_k(t)$ as a function of current step t .

Locality Regularization. In some scenarios, it is feasible to obtain examples of irrelevant knowledge during training (Wang et al., 2024d; Yu et al., 2024). Such information can benefit the training of BaFT as well. Following Wang et al. (2024d), we incorporate the margin loss as a regularizer. Let \mathbf{h} and \mathbf{h}_{ir} denote the representations of *editing* and *irrelevant* knowledge, respectively, we minimize

$$\mathcal{R}_{\text{loc}}(\phi) = \underbrace{\max(0, \mathbf{W}(\mathbf{h}_{\text{ir}}) - \alpha)}_{\text{irr. weight } w(\mathbf{h}_{\text{ir}}) \leq \alpha} + \underbrace{\max(0, \beta - \mathbf{W}(\mathbf{h}))}_{\text{edit. weight } w(\mathbf{h}) \geq \beta} + \underbrace{\max(0, \gamma - (\mathbf{W}(\mathbf{h})_{\text{max}} - \mathbf{W}(\mathbf{h}_{\text{ir}})_{\text{max}}))}_{\text{edit weight } \geq \text{loc weight}}.$$

At a colloquial level, $\mathcal{R}_{\text{loc}}(\phi)$ encourages that weights for irrelevant knowledge should be as small as α , editing knowledge's weight should be no less as β , and at the same time, the most important weights from the two groups should have a gap that is as large as γ .

In execution, we rescale the three terms to the same magnitude and solve the following objective

$$\min_{\phi} L(\phi) \triangleq \min_{\phi} L_1(\phi) + \mathcal{R}_{\text{bal}}(\phi) + \mathcal{R}_{\text{loc}}(\phi). \quad (4)$$

ReFT, as a special case of BaFT, only minimizes $L_1(\phi)$.

3 EXPERIMENT

We test the proposed BaFT for knowledge editing on three 7B-level autoregressive language models (LMs) over five public benchmarks. Ablation studies are also conducted. Experiment results show that BaFT can achieve excellent performance at much better parameter efficiency.

3.1 EXPERIMENT SETUP

Base Models. We conduct experiments on three representative LLMs from different model families. **LLaMA 2-7b** (and **LLaMA 2-7b-Chat**) (Touvron et al., 2023) have been widely studied in the literature (Zhang et al., 2024b; Wang et al., 2024d) and we follow this convention. Trending **LLaMA 3-8b-Instruct** (Dubey et al., 2024) and **Gemma 1.1-7b-Instruct** (Team et al., 2024) are also studied. From now on, we refer to the three LLMs as **LLaMA 2(-chat)**, **LLaMA 3**, and **Gemma** for brevity.

Tasks. Following previous works (Wang et al., 2023; Zhang et al., 2024b), we edit different kinds of knowledge: WikiData_{recent}, WikiData_{counterfact} (Cohen et al., 2024), WikiBio (Hartvigsen et al., 2024), ConvSent (Mitchell et al., 2022), and ZsRE (Yao et al., 2023). Due to page limitation, we refer readers to Zhang et al. (2024b) for more benchmark details. When editing an LLM, three scenarios are considered. **Single Editing** updates one piece of knowledge at a time. **Continual Editing** and **Batched Editing**, on the other hand, update multiple pieces of knowledge in a sequential or batched way. The two latter are more challenging due to potential forgetting and knowledge conflicting problems, as observed in the literature (Hartvigsen et al., 2024; Wang et al., 2024d).

Baselines. We follow Zhang et al. (2024b); Wang et al. (2024e) and choose AdaLoRA (Zhang et al., 2023), ROME and FT-L (Meng et al., 2022a), and MEMIT (Meng et al., 2022b) as baselines. In continual editing scenarios, we further include representative memory-based methods GRACE (Hartvigsen et al., 2024), MELO (Yu et al., 2024), and WISE (Wang et al., 2024d). All these baselines, same as ours, *do not require a large-scale hard-to-access training data, or training additional models*: AdaLoRA learns a low-rank update for model parameters on the new knowledge while keeping less important parameters unchanged, thereby achieving a highly efficient and precise PEFT. ROME applies a causal-tracing analysis to identify the layer wherein the knowledge is stored and then solves an analytic rank-one update. FT-L, on the other hand, directly finetunes the layer identified by ROME with an additional KL divergence loss. MEMIT extends ROME to a batched editing setting by identifying a series of layers to edit and finding the updates as least squares solutions. GRACE, MELO, and WISE are specialized for continual editing. They leverage side parameters to save new knowledge and learn gating mechanism to determine whether pre-trained or new knowledge should be used during inference. Finally, we include ReFT as a baseline that uses a subspace of the same rank as BaFT.

Evaluation Criteria. We evaluate the performance from multiple aspects (Zhang et al., 2024b; Wang et al., 2024d). Given an edited model, **reliability (Rel.)** evaluates whether it successfully learns the new knowledge; **generality (Gen.)** measures to what extent it can generalize to rephrased knowledge inquiries; **locality (Loc.)** quantifies how much the model can retain its original output on irrelevant knowledge inquiries; **portability (Por.)** checks if the model is able to transfer new knowledge to related content. We report the average of different metrics¹ for more complete comparisons.

Implementation Details. Our experiments are conducted with EasyEdit (Wang et al., 2024e). More implementation details and hyper-parameters can be found in App C.

3.2 SINGLE EDITING PERFORMANCE

We evaluate the effectiveness of the proposed BaFT for conducting Single Editing on WikiData_{recent}, WikiData_{counterfact}, WikiBio, and ConvSent (only supports LLaMA family). The four benchmarks do not contain irrelevant data. Consequently, *BaFT training does not involve the locality regularization*.

Single Editing results are reported in Tab 1. The proposed BaFT performs highly competitively in all cases. BaFT and ReFT use a subspace of the same rank to edit representations, so an ideal BaFT should achieve reliability comparable to ReFT that can edit representations freely. Indeed, BaFT maintains a better editing-locality trade-off: it consistently achieves better locality and portability

¹Not all benchmarks support all metrics.

than ReFT with no degradation of reliability. In comparison, other baselines suffer from notable editing-locality trade-off, i.e., achieve high reliability at a price of low locality. These methods also exhibit significant performance gaps when editing different LLMs. These results demonstrate BaFT as a new promising editing solution.

Table 1: Single Editing performance on four benchmark datasets. Results marked with “ \heartsuit ” are taken from [Zhang et al. \(2024b\)](#). Unsupported experiments are marked with “ \times ”. Best Avg. results are in **bold** and second best are underlined.

| | Wiki _{recent} | | | | Wiki _{counterfact} | | | | WikiBio | | | | ConvSent | |
|-----------------------|------------------------|------|------|-------------|-----------------------------|------|------|-------------|-----------------------|------|-------------|----------|----------|--|
| | LLaMA 2-7b-chat | | | | LLaMA 3-8b-Instruct | | | | Gemma 1.1-7b-Instruct | | | | | |
| | Rel. | Por. | Loc. | Avg. | Rel. | Por. | Loc. | Avg. | Rel. | Loc. | Avg. | Rel. | Rel. | |
| AdaLoRA [✓] | 1.00 | 0.65 | 0.56 | 0.74 | 1.00 | 0.70 | 0.70 | 0.80 | 1.00 | 0.81 | 0.91 | 0.45 | | |
| FT-L [✓] | 0.56 | 0.41 | 0.44 | 0.47 | 0.45 | 0.34 | 0.50 | 0.51 | 0.66 | 0.80 | 0.73 | 0.50 | | |
| ROME [✓] | 0.97 | 0.55 | 0.55 | 0.69 | 0.99 | 0.56 | 0.52 | 0.69 | 0.96 | 0.63 | 0.80 | 0.46 | | |
| MEMIT [✓] | 0.97 | 0.56 | 0.52 | 0.68 | 0.98 | 0.59 | 0.47 | 0.68 | 0.94 | 0.62 | 0.78 | 0.45 | | |
| ReFT | 1.00 | 0.60 | 0.71 | <u>0.77</u> | 1.00 | 0.72 | 0.78 | <u>0.83</u> | 1.00 | 0.91 | <u>0.96</u> | 1.00 | | |
| BaFT (Ours) | 1.00 | 0.61 | 0.73 | 0.78 | 1.00 | 0.72 | 0.81 | 0.84 | 1.00 | 0.94 | 0.97 | 1.00 | | |
| <hr/> | | | | | | | | | | | | | | |
| LLaMA 3-8b-Instruct | | | | | | | | | | | | | | |
| | | | | Rel. | Por. | Loc. | Avg. | Rel. | Loc. | Avg. | Rel. | Rel. | | |
| AdaLoRA | 1.00 | 0.61 | 0.45 | 0.69 | 1.00 | 0.74 | 0.51 | 0.75 | 1.00 | 0.79 | 0.90 | 1.00 | | |
| FT-L | 0.47 | 0.27 | 0.22 | 0.32 | 0.43 | 0.32 | 0.22 | 0.32 | 0.56 | 0.71 | 0.64 | 0.52 | | |
| ROME | 0.99 | 0.58 | 0.49 | 0.69 | 0.99 | 0.58 | 0.41 | 0.66 | 0.92 | 0.68 | 0.80 | 0.98 | | |
| MEMIT | 0.99 | 0.54 | 0.48 | 0.67 | 0.99 | 0.58 | 0.43 | 0.67 | 0.96 | 0.71 | 0.84 | 0.32 | | |
| ReFT | 1.00 | 0.62 | 0.62 | 0.75 | 1.00 | 0.72 | 0.74 | 0.82 | 1.00 | 0.87 | <u>0.94</u> | 0.98 | | |
| BaFT (Ours) | 1.00 | 0.62 | 0.64 | 0.75 | 1.00 | 0.72 | 0.75 | 0.82 | 1.00 | 0.91 | 0.96 | 0.96 | | |
| <hr/> | | | | | | | | | | | | | | |
| Gemma 1.1-7b-Instruct | | | | | | | | | | | | | | |
| | | | | Rel. | Por. | Loc. | Avg. | Rel. | Loc. | Avg. | Rel. | Rel. | | |
| AdaLoRA | 1.00 | 0.58 | 0.28 | 0.62 | 1.00 | 0.70 | 0.35 | 0.68 | 1.00 | 0.70 | 0.85 | \times | | |
| FT-L | 0.35 | 0.20 | 0.03 | 0.26 | 0.20 | 0.18 | 0.01 | 0.13 | 0.24 | 0.14 | 0.19 | \times | | |
| ROME | 0.79 | 0.38 | 0.27 | 0.48 | 0.82 | 0.47 | 0.27 | 0.52 | 0.47 | 0.31 | 0.39 | \times | | |
| ReFT | 1.00 | 0.54 | 0.55 | <u>0.70</u> | 1.00 | 0.63 | 0.72 | <u>0.78</u> | 1.00 | 0.82 | <u>0.91</u> | \times | | |
| BaFT (Ours) | 1.00 | 0.54 | 0.58 | 0.71 | 1.00 | 0.62 | 0.77 | 0.80 | 1.00 | 0.85 | 0.93 | \times | | |

3.3 CONTINUAL AND BATCHED EDITING PERFORMANCE.

Next, we study the two challenging scenarios, where massive editings are conducted in a sequential (continual) or batched way. We follow [Wang et al. \(2024d\)](#) and experiment with LLaMA 2 (non-chat version), LLaMA 3, and Gemma on ZsRE. We note that the state-of-the-art continual editing method WISE contains substantially larger parameter size and is much more computationally expensive. For fair comparison, we include WISE_{light}, a lightweight version of WISE that contains 1/8 learnable parameters of the original WISE to make its training affordable. We want to highlight that WISE_{light} does not change editing mechanism², and still contains more learnable parameters than BaFT and ReFT (10 and 20 times respectively). Learnable parameters used in different methods, along with their time consumptions, are reported in Tab 3.

Continual Editing Performance. Tab 2 presents the main results of continually editing 1000 pieces of ZsRE knowledge. BaFT again achieves remarkable editing performance while maintaining excellent locality on LLMs from different families, reaching the best two in nearly all scenarios. In comparison, standard methods AdaLoRA, FT-L, ROME, and MEMIT encounter considerable performance gaps over different LLMs. Meanwhile, they fall short in editing multiple pieces of knowledge that emerge sequentially. WISE performs slightly better but its parameter efficiency is much lower, as we will show soon. GRACE is designed for continual editing but still suffers from failure on editing Gemma. These methods might benefit from a more extensive hyper-parameter tuning for each LLM. Nonetheless, their prolonged running time makes this process expensive, if not unaffordable.

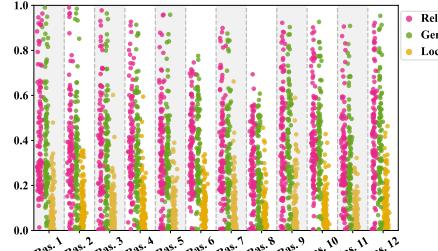


Figure 2: Bases weights used for editing and irrelevant knowledge (averaged over different positions).

²WISE finds an important FFN layer to conduct knowledge editing. For each new knowledge, it finetunes a small portion of *randomly* selected parameters in this layer. WISE_{light} uses a smaller randomly chosen pool.

When comparing BaFT and ReFT with each other, we note that as in Single Editing, BaFT maintains, if not surpasses, the editing ability of ReFT. In addition, when the editing number T increases, BaFT shows excellent robustness against forgetting, as indicated by its capability of preserving high locality in all scenarios. We further visualize bases weights in Fig 2, where a one-layer BaFT is used to edit LLaMA 2 on 100 ZsRE knowledge with $T = 10$ (achieved reliability, generality, and locality are 0.75, 0.71, and 0.98 respectively). Rel., Gen., and Loc. refers to *new*, *rephrased*, and *unrelated* knowledge, respectively. We note that BaFT evenly distributes the editing over all bases, and unrelated knowledge receives significantly lower weights. These results confirm that BaFT leverages the fine-grained basis-level control as designed in Sec 2, thereby excelling at Continual Editing.

Table 2: Continual Editing performance on ZsRE dataset, evaluated after conducting T times of editing sequentially. Results marked with “ \heartsuit ” are taken from Wang et al. (2024d). Best Avg. results are in **bold** and second best are underlined.

| | $T = 1$ | | | | $T = 10$ | | | | $T = 100$ | | | | $T = 1000$ | | | |
|-----------------------------------|-----------------------|------|------|-------------|---------------------|------|------|-------------|-----------|------|------|-------------|------------|------|------|-------------|
| | | | | | LLaMA 2-7b | | | | | | | | | | | |
| | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. |
| AdaLoRA | 1.00 | 0.90 | 0.92 | 0.94 | 0.39 | 0.38 | 0.50 | 0.42 | 0.06 | 0.06 | 0.06 | 0.06 | 0.00 | 0.00 | 0.00 | 0.00 |
| FT-L | 0.57 | 0.53 | 0.96 | 0.69 | 0.35 | 0.31 | 0.12 | 0.26 | 0.29 | 0.26 | 0.09 | 0.21 | 0.24 | 0.20 | 0.25 | 0.23 |
| ROME | 0.96 | 0.91 | 0.98 | 0.95 | 0.80 | 0.76 | 0.77 | 0.78 | 0.18 | 0.18 | 0.07 | 0.12 | 0.00 | 0.01 | 0.00 | 0.00 |
| MEMIT | 0.95 | 0.90 | 0.99 | 0.95 | 0.77 | 0.74 | 0.90 | 0.80 | 0.25 | 0.24 | 0.19 | 0.02 | 0.04 | 0.04 | 0.02 | 0.03 |
| MELO | 1.00 | 0.40 | 0.99 | 0.80 | 0.95 | 0.40 | 0.99 | 0.78 | 0.61 | 0.40 | 0.99 | 0.67 | 0.40 | 0.40 | 0.99 | 0.60 |
| GRACE \heartsuit | 0.98 | 0.08 | 1.00 | 0.69 | 0.96 | 0.00 | 1.00 | 0.65 | 0.96 | 0.00 | 1.00 | 0.65 | 0.97 | 0.08 | 1.00 | 0.68 |
| WISE _{full} \heartsuit | 0.98 | 0.92 | 1.00 | 0.97 | 0.94 | 0.88 | 1.00 | 0.94 | 0.90 | 0.81 | 1.00 | 0.90 | 0.77 | 0.72 | 1.00 | 0.83 |
| WISE _{light} | 0.95 | 0.83 | 1.00 | 0.93 | 0.93 | 0.74 | 1.00 | 0.89 | 0.83 | 0.73 | 0.99 | 0.85 | 0.49 | 0.47 | 1.00 | 0.65 |
| ReFT | 1.00 | 0.95 | 0.94 | 0.96 | 0.90 | 0.85 | 0.88 | 0.87 | 0.78 | 0.74 | 0.83 | 0.78 | 0.58 | 0.56 | 0.73 | 0.62 |
| BaFT (Ours) | 1.00 | 0.94 | 0.97 | 0.97 | 0.89 | 0.84 | 0.97 | <u>0.90</u> | 0.75 | 0.70 | 0.98 | 0.81 | 0.63 | 0.60 | 0.98 | <u>0.74</u> |
| | | | | | LLaMA 3-8b-Instruct | | | | | | | | | | | |
| | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. |
| AdaLoRA | 1.00 | 0.99 | 0.85 | 0.95 | 0.27 | 0.26 | 0.26 | 0.26 | 0.03 | 0.03 | 0.01 | 0.02 | 0.00 | 0.00 | 0.00 | 0.00 |
| FT-L | 0.51 | 0.52 | 0.68 | 0.57 | 0.25 | 0.20 | 0.03 | 0.16 | 0.19 | 0.16 | 0.02 | 0.12 | 0.16 | 0.14 | 0.01 | 0.10 |
| ROME | 0.99 | 0.96 | 0.96 | <u>0.97</u> | 0.62 | 0.63 | 0.42 | 0.56 | 0.07 | 0.07 | 0.01 | 0.05 | 0.03 | 0.03 | 0.00 | 0.02 |
| MEMIT | 0.99 | 0.96 | 0.98 | 0.98 | 0.68 | 0.66 | 0.71 | 0.68 | 0.03 | 0.03 | 0.02 | 0.03 | 0.00 | 0.00 | 0.00 | 0.00 |
| MELO | 1.00 | 0.29 | 1.00 | 0.76 | 0.97 | 0.30 | 1.00 | 0.76 | 0.55 | 0.31 | 0.99 | 0.62 | 0.31 | 0.30 | 0.99 | 0.53 |
| GRACE | 0.33 | 0.00 | 0.54 | 0.29 | 0.33 | 0.02 | 0.56 | 0.30 | 0.33 | 0.02 | 0.57 | 0.31 | 0.33 | 0.02 | 0.55 | 0.30 |
| WISE _{light} | 0.95 | 0.91 | 0.99 | 0.95 | 0.82 | 0.76 | 1.00 | 0.86 | 0.63 | 0.57 | 1.00 | <u>0.73</u> | 0.39 | 0.37 | 1.00 | <u>0.59</u> |
| ReFT | 1.00 | 0.97 | 0.93 | <u>0.97</u> | 0.90 | 0.84 | 0.87 | <u>0.87</u> | 0.68 | 0.61 | 0.74 | <u>0.68</u> | 0.48 | 0.45 | 0.64 | <u>0.52</u> |
| BaFT (Ours) | 1.00 | 0.95 | 0.96 | 0.97 | 0.89 | 0.82 | 0.95 | 0.89 | 0.70 | 0.64 | 0.93 | 0.76 | 0.50 | 0.49 | 0.93 | 0.64 |
| | Gemma 1.1-7b-Instruct | | | | | | | | | | | | | | | |
| | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. |
| AdaLoRA | 1.00 | 0.97 | 0.67 | 0.88 | 0.19 | 0.20 | 0.18 | 0.19 | 0.03 | 0.03 | 0.01 | 0.02 | 0.00 | 0.00 | 0.00 | 0.00 |
| FT-L | 0.28 | 0.33 | 0.09 | 0.23 | 0.14 | 0.06 | 0.00 | 0.07 | 0.07 | 0.04 | 0.00 | 0.04 | 0.05 | 0.04 | 0.00 | 0.03 |
| ROME | 0.75 | 0.71 | 0.88 | 0.78 | 0.18 | 0.18 | 0.05 | 0.14 | 0.01 | 0.01 | 0.01 | 0.01 | 0.00 | 0.00 | 0.00 | 0.00 |
| MEMIT | 1.00 | 0.20 | 1.00 | 0.73 | 0.96 | 0.23 | 1.00 | 0.73 | 0.52 | 0.26 | 0.95 | 0.58 | 0.26 | 0.25 | 0.95 | 0.49 |
| GRACE | 0.39 | 0.00 | 1.00 | 0.46 | 0.39 | 0.01 | 1.00 | 0.47 | 0.39 | 0.01 | 1.00 | 0.47 | 0.39 | 0.01 | 1.00 | 0.47 |
| WISE _{light} | 0.99 | 0.96 | 1.00 | 0.98 | 0.90 | 0.84 | 0.99 | 0.91 | 0.79 | 0.71 | 0.95 | 0.82 | 0.48 | 0.42 | 0.98 | 0.63 |
| ReFT | 1.00 | 0.86 | 0.91 | <u>0.92</u> | 0.92 | 0.81 | 0.81 | <u>0.85</u> | 0.66 | 0.58 | 0.69 | <u>0.64</u> | 0.50 | 0.46 | 0.65 | <u>0.54</u> |
| BaFT (Ours) | 1.00 | 0.84 | 0.94 | <u>0.93</u> | 0.92 | 0.80 | 0.92 | <u>0.88</u> | 0.70 | 0.62 | 0.92 | <u>0.75</u> | 0.48 | 0.45 | 0.92 | <u>0.62</u> |

Table 3: Parameter size and editing time with an NVIDIA V100 32-GB GPU (averaged over 100 samples). ROME, MEMIT, and GRACE do not contain pre-specified learnable parameters.

| | LLaMA 2-7b(-chat) | | LLaMA 3-8b-Instruct | | Gemma 1.1-7b-Instruct | |
|-----------------------|-------------------|------------------|---------------------|------------------|-----------------------|------------------|
| | # Params. | Time (sec./edit) | # Params. | Time (sec./edit) | # Params. | Time (sec./edit) |
| AdaLoRA | 6,292,224 | 26.24 | 5,112,576 | 28.71 | 4,817,568 | 44.24 |
| FT-L | 45,088,768 | 9.73 | 58,720,256 | 10.84 | 75,497,472 | 11.95 |
| ROME | / | 27.27 | / | 25.01 | / | 52.07 |
| MEMIT | / | 20.01 | / | 25.35 | / | 43.45 |
| GRACE | / | 34.38 | / | 87.08 | / | 20.20 |
| WISE _{light} | 5,636,096 | 58.00 | 7,340,032 | 65.77 | 9,437,184 | 7.79 |
| ReFT | 393,264 | 10.99 | 393,264 | 9.33 | 294,960 | 10.13 |
| BaFT (Ours) | 606,256 | 13.46 | 606,256 | 12.69 | 454,704 | 454,704 |

Batched Editing Performance. We further compare BaFT and ReFT against baselines that *admit* batched data for editing, namely, AdaLoRA, FT-L, and MEMIT. LMs were edited on ZsRE dataset, and batch sizes were set to 10 and 50 respectively.

We visualize the average of reliability, generality, and locality in Fig 3, and defer the complete results to App D. The proposed BaFT again achieved a great balance between good edit success and high locality, outperforming all baselines in 5 out of 6 scenarios. Surprisingly, when $T = 10$, LoRA

and MEMIT were capable of benefiting from editing multiple samples in a batch than one by one. We conjecture that learning multiple pieces of knowledge in a batch helps mitigate their overfitting on any single knowledge, thereby weakened the forgetting problem to some extent. This finding suggests that caching more knowledge and editing them in a batch can be beneficial in some cases.

Parameter Efficiency. Continual and Batched Editing involve learning more knowledge than in Single Editing. As a result, achieving good editing performance while maintaining high parameter efficiency is non-trivial, as using fewer parameters increases the workload of each parameter to learn more knowledge. We note that while WISE_{light} achieved comparable performance to BaFT, its parameter efficiency was much lower: on LLaMA 2-7b, the edit success dropped from 0.77 (of WISE) to 0.49 when editing 1000 pieces of knowledge, around 22% lower than BaFT which uses 10 times less parameters, as per Table 3. Similar trends can be found when making comparison with LoRA in Batched Editing scenarios. In conclusion, BaFT is capable of achieving much better parameter efficiency than existing methods.

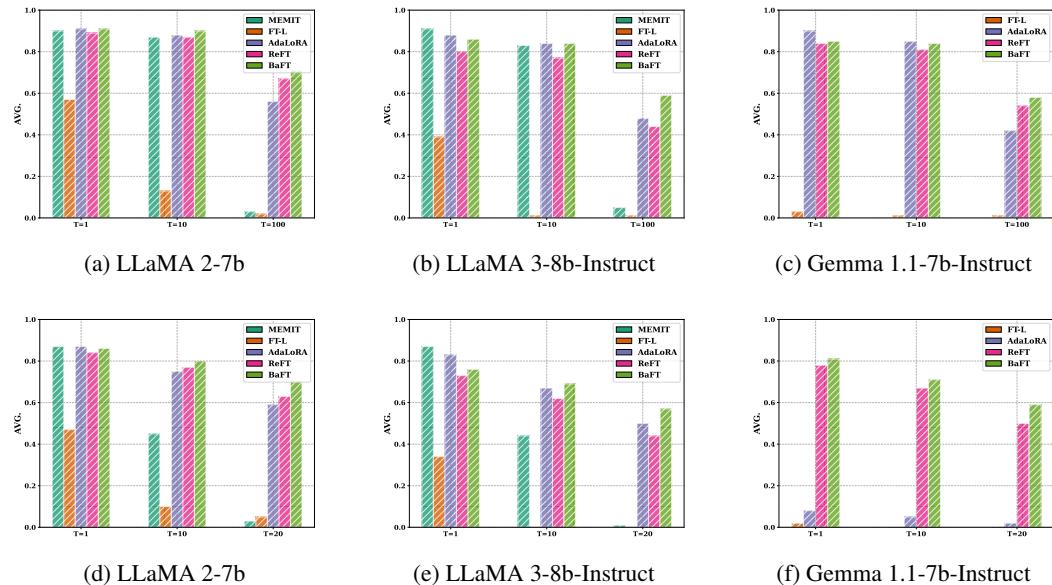


Figure 3: Batched Editing Performance under sequence length. The first row uses batch size 10 and the second row uses batch size 50.

3.4 ABLATION STUDY

We end this section with an ablation study on BaFT to showcase how each component contributes to the final performance. Results from continually editing LLaMA 2-7b with 100 ZsRE knowledge are presented in Tab 4. We note that introducing a coarse-grained *subspace-level* weighting (*ss-w*) which assigns all bases with the same weight along did not benefit ReFT. Moreover, both locality regularization (*lr*) and fine-grained basis-level weighting (*ba-w*) helped improve locality. Remarkably, the basis-level weighting, as observed in all Single Editing scenarios, did not lead to edit performance degradation. Locality regularization, while greatly improved the locality, induced a trade-off with editing performance at the same time. Notably, the degradation is amplified when the subspace-level weighting was used, echoing well with our theoretical analysis.

In conclusion, the proposed BaFT makes two improvements over ReFT. First, the weighting offers a fine-grained level learning, leading to better locality without hurting editing performance. Second, a fine-grained basis-level control allows one to regularize locality by altering only the important parts, leading to a better empirical editing-locality trade-off.

4 RELATED WORKS

Existing editing methods mainly fall into two classes.

Internal Storage updates model parameters for the adaptation. Early efforts involved fine-tuning a LLM directly but suffered from severe forgetting of original knowledge (Wang et al., 2023). For more precise editing, Zhu et al. (2020) imposed a relaxed ℓ_2 norm constraint on the parameter updates, and Huang et al. (2023); Dong et al. (2022a) limited the updates to some specific feed-forward network (FFN) layers, based on findings that knowledge is often stored therein (Dai et al., 2021). For further refinement, the *locate-and-edit* paradigm (Meng et al., 2022a;b) first identifies the layer storing a specific knowledge, and then modifies its parameters in an analytic form or via least squared solution. On the other hand, PEFT methods such as AdaLoRA (Zhang et al., 2023) also provided performance on par with locating-based solutions (Wu et al., 2023; Wang et al., 2024b). However, these methods are *parameter-based* and offer a similar level of control, in the sense that all inputs are altered in the same way. As a result, they suffer from an equal level of *editing-locality* trade-off (Wang et al., 2023; 2024d). These findings raised a question as *to what extent knowledge can be accurately attributed to some specific parameters* (Hase et al., 2024). Inspired by the recent advance of improving a LLM’s general ability such as commonsense reasoning by fine-tuning its representations (Wu et al., 2024), in this work we show that updating representations at only a few locations can provide strong editing performance. By pushing the fine-tuning towards a new basis-level, our BaFT achieved better fine-grained control and superior editing-locality trade-off.

External Storage resorts to external memories without changing original parameters. Methods include meta-learning based MEND (Mitchell et al., 2021) and its multi-task version InstructEdit (Zhang et al., 2024a), IKE (Zheng et al., 2023) and LTE (Jiang et al., 2024) that bear the similarity to Retrieval-Augmented Generation (Gao et al., 2023; Wang et al., 2024a; Xu et al., 2024; Yu et al., 2025; Liu et al., 2025b), augmentation based StableKE (Wei et al., 2024), and proxy model based SERAC (Mitchell et al., 2022). Notwithstanding, these methods need large-scaled *hard-to-access* data to retrieve from (e.g., IKE, LTE), or to train extra model on (e.g., MEND, InstructEdit, SERAC). As a consequence, they have limited practicality and fall short on Continual Editing that requires frequent updates (Wang et al., 2024d). Recently, methods specialized for Continual Editing were proposed (Hartvigsen et al., 2024; Yu et al., 2024; Wang et al., 2024d). These approaches injected lightweight adapters (Hartvigsen et al., 2024) or weight copies (Wang et al., 2024d) to memorize new knowledge, and learned some gating mechanism to determine whether original or new knowledge to use. Specifically, GRACE (Hartvigsen et al., 2024) maintained a code-book to determine which adapter will be used based on representation similarity, and MELO (Yu et al., 2024) used dynamic LoRA. WISE (Wang et al., 2024d) learned activation threshold to trigger new learned weights. However, these methods have several limitations. First, they often show unsatisfactory generalizability, as observed in Wang et al. (2024d) and confirmed in our experiments. Second, they require prolonged training (and inference) time, due to the need of maintaining non-constant numbers of external memories. Finally, existing gating mechanisms cannot be learned when multiple pieces of knowledge appear, making them incompatible for Batched Editing. In comparison, BaFT learns a pre-specified set of parameters and lets bases weights play the role of gating. This design makes BaFT suitable for both Continual and Batched Editing. Moreover, as *editing* and *activation* are conducted in a representation subspace, BaFT is able to achieve good generalizability at better parameter efficiency.

5 CONCLUSION AND FUTURE WORKS

In this work, we propose a new representation based method towards more efficient knowledge editing. Grounded in a theoretical analysis, we show that updating all selected representations with one linear subspace in a predetermined manner imposes a tension in editing-locality trade-off. Subsequently, BaFT as a better solution is proposed. Given a representation, BaFT first computes a weight for each basis that spans the linear subspace, then conducts a linear update along this basis direction. Because bases weights are determined from the current representation with non-linear functions, BaFT fine-tunes the representation in a non-linear way. This fine-grained control leads to better performance on editing three representative LLMs in various scenarios, on par with or outperforming the strongest baselines at much better parameter efficiency. As detailed in App A, there are some limitations in this work, and we plan to work on in our future work.

Table 4: Component effects in BaFT.

| | Rel. | Gen. | Loc. | Avg. |
|----------|------|------|------|------|
| ReFT | 0.76 | 0.71 | 0.84 | 0.77 |
| +ss-w. | 0.74 | 0.68 | 0.81 | 0.74 |
| +ba-w | 0.77 | 0.71 | 0.86 | 0.78 |
| +ss-w&lr | 0.67 | 0.61 | 0.99 | 0.76 |
| BaFT | 0.73 | 0.67 | 0.98 | 0.79 |

ACKNOWLEDGEMENT

This work is supported in part by the US National Science Foundation under grant NSF IIS-2141037. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

- Genrich Belitskii et al. *Matrix norms and their applications*, volume 36. Birkhäuser, 2013.
- Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. A neural probabilistic language model. *Advances in neural information processing systems*, 13, 2000.
- Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Pixa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.
- Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.
- Wei Chen, Zichen Miao, and Qiang Qiu. Inner product-based neural network similarity. *Advances in Neural Information Processing Systems*, 36:73995–74020, 2023.
- Wei Chen, Zichen Miao, and Qiang Qiu. Parameter-efficient tuning of large convolutional models. *arXiv preprint arXiv:2403.00269*, 2024.
- Roi Cohen, Eden Biran, Ori Yoran, Amir Globerson, and Mor Geva. Evaluating the ripple effects of knowledge editing in language models. *Transactions of the Association for Computational Linguistics*, 12:283–298, 2024.
- Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. Knowledge neurons in pretrained transformers. *arXiv preprint arXiv:2104.08696*, 2021.
- Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. *arXiv preprint arXiv:2104.08164*, 2021.
- Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235, 2023.
- Qingxiu Dong, Damai Dai, Yifan Song, Jingjing Xu, Zhifang Sui, and Lei Li. Calibrating factual knowledge in pretrained language models. *arXiv preprint arXiv:2210.03329*, 2022a.
- Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. A survey for in-context learning. *arXiv preprint arXiv:2301.00234*, 2022b.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models, 2024.
- William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity, 2022. URL <https://arxiv.org/abs/2101.03961>.

- Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Haofen Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2, 2023.
- Thomas Hartvigesen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. Toxigen: A large-scale machine-generated dataset for adversarial and implicit hate speech detection. *arXiv preprint arXiv:2203.09509*, 2022.
- Tom Hartvigesen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. Aging with grace: Lifelong model editing with discrete key-value adaptors. *Advances in Neural Information Processing Systems*, 36, 2024.
- Peter Hase, Mohit Bansal, Been Kim, and Asma Ghandeharioun. Does localization inform editing? surprising differences in causality-based localization vs. knowledge editing in language models. *Advances in Neural Information Processing Systems*, 36, 2024.
- Evan Hernandez, Belinda Z Li, and Jacob Andreas. Inspecting and editing knowledge representations in language models. *arXiv preprint arXiv:2304.00740*, 2023.
- Neil Houlsby, Andrei Giurjiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pp. 2790–2799, 2019.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021.
- Zeyu Huang, Yikang Shen, Xiaofeng Zhang, Jie Zhou, Wenge Rong, and Zhang Xiong. Transformer-patcher: One mistake worth one neuron. *arXiv preprint arXiv:2301.09785*, 2023.
- Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.
- Xuyin Jiang, Yufei Wang, Chuhan Wu, Wanjun Zhong, Xingshan Zeng, Jiahui Gao, Liangyou Li, Xin Jiang, Lifeng Shang, Ruiming Tang, et al. Learning to edit: Aligning llms with knowledge editing. *arXiv preprint arXiv:2402.11905*, 2024.
- Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022.
- Alex Lamb, Anirudh Goyal, Ying Zhang, Saizheng Zhang, Aaron Courville, and Yoshua Bengio. Professor forcing: A new algorithm for training recurrent networks, 2016.
- Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.
- Tianci Liu, Zihan Dong, Linjun Zhang, Haoyu Wang, and Jing Gao. Mitigating heterogeneous token overfitting in llm knowledge editing. *arXiv preprint arXiv:2502.00602*, 2025a.
- Tianci Liu, Haoxiang Jiang, Tianze Wang, Ran Xu, Yue Yu, Linjun Zhang, Tuo Zhao, and Haoyu Wang. Roserag: Robust retrieval-augmented generation with small-scale llms via margin-aware preference optimization. *arXiv preprint arXiv:2502.10993*, 2025b.
- Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019.
- Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372, 2022a.
- Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*, 2022b.

- Zichen Miao, Ze Wang, Wei Chen, and Qiang Qiu. Continual learning with filter atom swapping. In *International Conference on Learning Representations*, 2021.
- Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. Fast model editing at scale. *arXiv preprint arXiv:2110.11309*, 2021.
- Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D Manning, and Chelsea Finn. Memory-based model editing at scale. In *International Conference on Machine Learning*, pp. 15817–15831, 2022.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022.
- Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. *arXiv preprint arXiv:2311.03658*, 2023.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.
- Abigail See, Aneesh Pappu, Rohun Saxena, Akhila Yerukola, and Christopher D Manning. Do massively pretrained language models make better storytellers? *arXiv preprint arXiv:1909.10705*, 2019.
- Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer, 2017.
- Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, et al. Gemma: Open models based on gemini research and technology, 2024.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikołay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- Haoyu Wang, Ruirui Li, Haoming Jiang, Jinjin Tian, Zhengyang Wang, Chen Luo, Xianfeng Tang, Monica Cheng, Tuo Zhao, and Jing Gao. Blendfilter: Advancing retrieval-augmented large language models via query generation blending and knowledge filtering. *arXiv preprint arXiv:2402.11129*, 2024a.
- Haoyu Wang, Tianci Liu, Ruirui Li, Monica Cheng, Tuo Zhao, and Jing Gao. Roselora: Row and column-wise sparse low-rank adaptation of pre-trained language model for knowledge editing and fine-tuning. *arXiv preprint arXiv:2406.10777*, 2024b.
- Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024c.
- Peng Wang, Zexi Li, Ningyu Zhang, Ziwen Xu, Yunzhi Yao, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. Wise: Rethinking the knowledge memory for lifelong model editing of large language models. *arXiv preprint arXiv:2405.14768*, 2024d.

- Peng Wang, Ningyu Zhang, Bozhong Tian, Zekun Xi, Yunzhi Yao, Ziwen Xu, Mengru Wang, Shengyu Mao, Xiaohan Wang, Siyuan Cheng, Kangwei Liu, Yuansheng Ni, Guozhou Zheng, and Huajun Chen. Easynet: An easy-to-use knowledge editing framework for large language models, 2024e.
- Song Wang, Yaochen Zhu, Haochen Liu, Zaiyi Zheng, Chen Chen, et al. Knowledge editing for large language models: A survey. *arXiv preprint arXiv:2310.16218*, 2023.
- Zihao Wei, Liang Pang, Hanxing Ding, Jingcheng Deng, Huawei Shen, and Xueqi Cheng. Stable knowledge editing in large language models. *arXiv preprint arXiv:2402.13048*, 2024.
- Suhang Wu, Minlong Peng, Yue Chen, Jinsong Su, and Mingming Sun. Eva-kellm: A new benchmark for evaluating knowledge editing of llms. *arXiv preprint arXiv:2308.09954*, 2023.
- Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D Manning, and Christopher Potts. Reft: Representation finetuning for language models. *arXiv preprint arXiv:2404.03592*, 2024.
- Ran Xu, Hui Liu, Sreyashi Nag, Zhenwei Dai, Yaochen Xie, Xianfeng Tang, Chen Luo, Yang Li, Joyce C Ho, Carl Yang, et al. Simrag: Self-improving retrieval-augmented generation for adapting large language models to specialized domains. *arXiv preprint arXiv:2410.17952*, 2024.
- Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. Editing large language models: Problems, methods, and opportunities. *arXiv preprint arXiv:2305.13172*, 2023.
- Lang Yu, Qin Chen, Jie Zhou, and Liang He. Melo: Enhancing model editing with neuron-indexed dynamic lora. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19449–19457, 2024.
- Yue Yu, Wei Ping, Zihan Liu, Boxin Wang, Jiaxuan You, Chao Zhang, Mohammad Shoeybi, and Bryan Catanzaro. Rankrag: Unifying context ranking with retrieval-augmented generation in llms. *Advances in Neural Information Processing Systems*, 37:121156–121184, 2025.
- Ningyu Zhang, Bozhong Tian, Siyuan Cheng, Xiaozhuan Liang, Yi Hu, Kouying Xue, Yanjie Gou, Xi Chen, and Huajun Chen. Instructedit: Instruction-based knowledge editing for large language models. *arXiv preprint arXiv:2402.16123*, 2024a.
- Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, et al. A comprehensive study of knowledge editing for large language models. *arXiv preprint arXiv:2401.01286*, 2024b.
- Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. In *International Conference on Learning Representations*, 2023.
- Ce Zheng, Lei Li, Qingxiu Dong, Yuxuan Fan, Zhiyong Wu, Jingjing Xu, and Baobao Chang. Can we edit factual knowledge by in-context learning? *arXiv preprint arXiv:2305.12740*, 2023.
- Ce Zhou, Qian Li, Chen Li, Jun Yu, Yixin Liu, Guangjing Wang, Kai Zhang, Cheng Ji, Qiben Yan, Lifang He, et al. A comprehensive survey on pretrained foundation models: A history from bert to chatgpt. *arXiv preprint arXiv:2302.09419*, 2023.
- Chen Zhu, Ankit Singh Rawat, Manzil Zaheer, Srinadh Bhojanapalli, Daliang Li, Felix Yu, and Sanjiv Kumar. Modifying memories in transformer models. *arXiv preprint arXiv:2012.00363*, 2020.

A MORE DISCUSSIONS AND LIMITATIONS ON BaFT

In this section we provide more discussions on the proposed BaFT. Fig 4 demonstrates the workflow of our method. There are also some limitations in this work, and we plan to explore in the future.

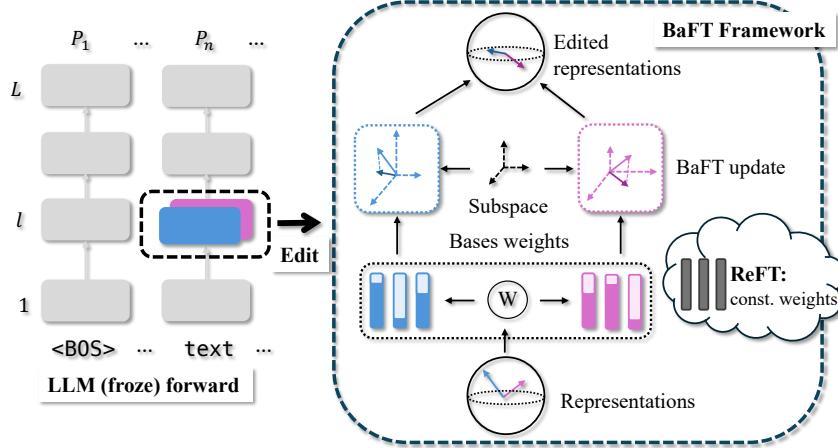


Figure 4: BaFT learns basis-level weights to edit different representations (highlighted in different colors). When using constant weights, BaFT reduces to ReFT.

First, The empirical success of BaFT was mainly established on standard benchmarks EasyEdit (Wang et al., 2024e), which may not be sufficient to reflect the diverse real-world applications. Second, BaFT as a generalization of ReFT requires hyper-parameter tuning to determine proper positions and layers to add interventions. Our choice was selected based on recommended values from ReFT (Wu et al., 2024). We plan to explore automating this process by imposing proper sparsity constraints on weights in our future work. Third, the promising performance of BaFT demonstrates its potential for efficient knowledge editing. However, it is still an open question if representation-based method is capable of fitting any editing (or updates) learnable for parameter-based methods. In other words, it is unknown if there is some knowledge that can be learned by parameter-based method but is unlearnable by updating representations. We plan to explore this direction in our future work.

B OMITTED PROOF

We include omitted proof here.

B.1 PROOF OF THM 2.3

We start with restating the two assumptions and the theorem.

Assumption B.1. Let text x encodes s, r , text y generated by the LM will convey o if its intermediate representation takes some targeted value t .

Assumption B.2. For any \mathbf{h} carrying some high-level knowledge, there exists a positive $\varepsilon(\mathbf{h})$ -radius ℓ_2 ball $B(\mathbf{h}, \varepsilon(\mathbf{h}))$ around \mathbf{h} such that any $\mathbf{h}' \in B(\mathbf{h}, \varepsilon(\mathbf{h}))$ conveys the same knowledge, we refer to $B(\mathbf{h}, \varepsilon(\mathbf{h}))$ as a *stable-ball* of \mathbf{h} .

Theorem B.3. *When finetuning a LM, ReFT learns to update the old representation \mathbf{h}_0 to targeted $t = \Phi(\mathbf{h}_0)$. If ReFT maintains good generality such that $\forall \mathbf{h} \in B(\mathbf{h}_0, \varepsilon(\mathbf{h}_0))$,*

$$\|\Phi(\mathbf{h}) - \Phi(\mathbf{h}_0)\| = \|\Phi(\mathbf{h}) - t\| < \varepsilon(t),$$

where $\|\cdot\|$ denote the ℓ_2 norm. Then for any irrelevant input \mathbf{h}_{ir} with a small stable-ball radius

$$\varepsilon(\mathbf{h}_{ir}) < \frac{\|t - \mathbf{h}_0\| - (\varepsilon(t) + \varepsilon(\mathbf{h}_0))}{\varepsilon(t) + 2\varepsilon(\mathbf{h}_0)} \varepsilon(\mathbf{h}_0),$$

and is not too far from \mathbf{h}_0 such that

$$\|\mathbf{h}_{ir} - \mathbf{h}_0\| = \varepsilon(\mathbf{h}_{ir}) + \varepsilon(\mathbf{h}_0),$$

ReFT will output $\Phi(\mathbf{h}_{ir}) \notin B(\mathbf{h}_{ir}, \varepsilon(\mathbf{h}_{ir}))$ and break its locality guarantee.

Proof. First, since old and new knowledge associates with different objects o , by Asmp 2.2, \mathbf{h}_0 and \mathbf{t} must have non-overlapped stable-ball. Otherwise, we can find

$$\mathbf{h} \in B(\mathbf{h}_0, \varepsilon(\mathbf{h}_0)) \cap B(\mathbf{t}, \varepsilon(\mathbf{t})),$$

that preserves the knowledge of both \mathbf{h}_0 and \mathbf{t} that are different, which is impossible. This implies

$$\|\mathbf{t} - \mathbf{h}_0\| \geq \varepsilon(\mathbf{t}) + \varepsilon(\mathbf{h}_0).$$

In addition, by definition of ReFT, we have

$$\begin{aligned} \mathbf{t} - \mathbf{h}_0 &= \Phi(\mathbf{h}_0) - \mathbf{h}_0 \\ &= \mathbf{h}_0 + \mathbf{R}^\top(\mathbf{A}\mathbf{h}_0 + \mathbf{b}) - \mathbf{h}_0 \\ &= \mathbf{R}^\top(\mathbf{A} - \mathbf{R})\mathbf{h}_0 + \mathbf{R}^\top\mathbf{b} \\ &\stackrel{(a)}{=} \mathbf{H}\mathbf{h}_0 + \mathbf{R}^\top\mathbf{b}, \end{aligned}$$

where in the last step (a), we defined $\mathbf{H} \triangleq \mathbf{R}^\top(\mathbf{A} - \mathbf{R})$ for simplicity.

Next, according to the *generality* condition, for any $\mathbf{h} \in B(\mathbf{h}_0, \varepsilon(\mathbf{h}_0))$, we have

$$\begin{aligned} &\|\Phi(\mathbf{h}) - \Phi(\mathbf{h}_0)\| \\ &= \|(\mathbf{I} + \mathbf{R}^\top(\mathbf{A} - \mathbf{R}))(\mathbf{h} - \mathbf{h}_0)\| \\ &= \|(\mathbf{I} + \mathbf{H})(\mathbf{h} - \mathbf{h}_0)\| \\ &< \varepsilon(\mathbf{t}). \end{aligned}$$

Since \mathbf{h} can take any direction, we know $\mathbf{h} - \mathbf{h}_0$ can be an arbitrary vector that has norm no greater than $\varepsilon(\mathbf{h}_0)$. Let $\mathbf{h} - \mathbf{h}_0$ takes the direction of the first right singular vector, then

$$\|(\mathbf{I} + \mathbf{H})(\mathbf{h} - \mathbf{h}_0)\| = \sigma_{\max}(\mathbf{I} + \mathbf{H})\|\mathbf{h} - \mathbf{h}_0\| < \varepsilon(\mathbf{t}).$$

This implies that the operator norm of $\mathbf{I} + \mathbf{H}$, denoted by σ_{\max} , is upper bounded by

$$\sigma_{\max}(\mathbf{I} + \mathbf{H}) \leq \frac{\varepsilon(\mathbf{t})}{\varepsilon(\mathbf{h}_0)}.$$

By triangle inequality of the operator norm (Belitskii et al., 2013), we further know

$$\sigma_{\max}(\mathbf{H}) = \sigma_{\max}(\mathbf{I} + \mathbf{H} - \mathbf{I}) \leq \sigma_{\max}(\mathbf{I} + \mathbf{H}) + \sigma_{\max}(\mathbf{I}) \leq \frac{\varepsilon(\mathbf{t})}{\varepsilon(\mathbf{h}_0)} + 1.$$

Now, for any irrelevant \mathbf{h}_{ir} , we have

$$\begin{aligned} &\|\Phi(\mathbf{h}_{\text{ir}}) - \mathbf{h}_{\text{ir}}\| \\ &= \|\mathbf{H}\mathbf{h}_{\text{ir}} + \mathbf{R}^\top\mathbf{b}\| \\ &\stackrel{(a)}{=} \|\mathbf{H}\mathbf{h}_{\text{ir}} + (\mathbf{t} - \mathbf{h}_0) - \mathbf{H}\mathbf{h}_0\| \\ &= \|\mathbf{H}(\mathbf{h}_{\text{ir}} - \mathbf{h}_0) + (\mathbf{t} - \mathbf{h}_0)\| \\ &\stackrel{(b)}{\geq} \left| \|\mathbf{t} - \mathbf{h}_0\| - \|\mathbf{H}(\mathbf{h}_{\text{ir}} - \mathbf{h}_0)\| \right|, \end{aligned} \tag{†}$$

where (a) substitutes

$$\mathbf{t} - \mathbf{h}_0 = \mathbf{H}\mathbf{h}_0 + \mathbf{R}^\top\mathbf{b},$$

and (b) holds from the reverse triangle inequality.

When the irrelevant \mathbf{h}_{ir} has a small stable-ball radius,

$$\varepsilon(\mathbf{h}_{\text{ir}}) < \frac{\|\mathbf{t} - \mathbf{h}_0\| - (\varepsilon(\mathbf{t}) + \varepsilon(\mathbf{h}_0))}{2\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t})} \varepsilon(\mathbf{h}_0),$$

and is close to \mathbf{h}_0 such that

$$\|\mathbf{h}_{\text{ir}} - \mathbf{h}_0\| = \varepsilon(\mathbf{h}_{\text{ir}}) + \varepsilon(\mathbf{h}_0),$$

we have

$$\begin{aligned} \|\mathbf{H}(\mathbf{h}_{\text{ir}} - \mathbf{h}_0)\| &\leq \sigma_{\max}(\mathbf{H})\|\mathbf{h}_{\text{ir}} - \mathbf{h}_0\| \\ &\leq \left(\frac{\varepsilon(\mathbf{t})}{\varepsilon(\mathbf{h}_0)} + 1 \right) (\varepsilon(\mathbf{h}_{\text{ir}}) + \varepsilon(\mathbf{h}_0)) \\ &\leq \left(\frac{\varepsilon(\mathbf{t}) + \varepsilon(\mathbf{h}_0)}{\varepsilon(\mathbf{h}_0)} \right) \\ &\quad \times \left(\varepsilon(\mathbf{h}_0) \frac{\|\mathbf{t} - \mathbf{h}_0\| - (\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t}))}{2\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t})} + \varepsilon(\mathbf{h}_0) \right) \\ &\stackrel{(a)}{<} \left(\frac{\varepsilon(\mathbf{t}) + \varepsilon(\mathbf{h}_0)}{\varepsilon(\mathbf{h}_0)} \right) \\ &\quad \times \left(\varepsilon(\mathbf{h}_0) \frac{\|\mathbf{t} - \mathbf{h}_0\| - (\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t}))}{\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t})} + \varepsilon(\mathbf{h}_0) \right) \\ &= \|\mathbf{t} - \mathbf{h}_0\| - (\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t})) \\ &\quad + (\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t})) \\ &= \|\mathbf{t} - \mathbf{h}_0\| \end{aligned}$$

where (a) holds from the fact that $\varepsilon(\mathbf{h}_0) > 0$, so dropping one $\varepsilon(\mathbf{h}_0)$ in the denominator provides a valid upper bound. Therefore, we can safely remove the absolute value function in Eqn (†) and get

$$\begin{aligned} \|\Phi(\mathbf{h}_{\text{ir}}) - \mathbf{h}_{\text{ir}}\| &= \|\mathbf{H}\mathbf{h}_{\text{ir}} + \mathbf{R}^\top \mathbf{b}\| \\ &\geq \|\mathbf{t} - \mathbf{h}_0\| - \|\mathbf{H}(\mathbf{h}_{\text{ir}} - \mathbf{h}_0)\| \\ &= \|\mathbf{t} - \mathbf{h}_0\| - \|\mathbf{H}(\mathbf{h}_{\text{ir}} - \mathbf{h}_0)\| \\ &\geq \|\mathbf{t} - \mathbf{h}_0\| - \sigma_{\max}(\mathbf{H})\|\mathbf{h}_{\text{ir}} - \mathbf{h}_0\| \\ &\geq \|\mathbf{t} - \mathbf{h}_0\| - \left(\frac{\varepsilon(\mathbf{t})}{\varepsilon(\mathbf{h}_0)} + 1 \right) (\varepsilon(\mathbf{h}_{\text{ir}}) + \varepsilon(\mathbf{h}_0)) \\ &\geq \|\mathbf{t} - \mathbf{h}_0\| - \left(\frac{\varepsilon(\mathbf{t}) + \varepsilon(\mathbf{h}_0)}{\varepsilon(\mathbf{h}_0)} \right) \\ &\quad \times \left(\frac{\|\mathbf{t} - \mathbf{h}_0\| - (\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t}))}{2\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t})} \varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{h}_0) \right) \\ &= \|\mathbf{t} - \mathbf{h}_0\| - \left(\frac{\varepsilon(\mathbf{t}) + \varepsilon(\mathbf{h}_0)}{\varepsilon(\mathbf{h}_0)} \right) \\ &\quad \times \left(\frac{\|\mathbf{t} - \mathbf{h}_0\| - (\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t})) + 2\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t})}{2\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t})} \varepsilon(\mathbf{h}_0) \right) \\ &= \|\mathbf{t} - \mathbf{h}_0\| - (\varepsilon(\mathbf{t}) + \varepsilon(\mathbf{h}_0)) \left(\frac{\|\mathbf{t} - \mathbf{h}_0\| + \varepsilon(\mathbf{h}_0)}{2\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t})} \right). \end{aligned}$$

Finally, it is easy to verify that this term is an upper bound of $\varepsilon(\mathbf{h}_{\text{ir}})$, since

$$\begin{aligned}
\|\Phi(\mathbf{h}_{\text{ir}}) - \mathbf{h}_{\text{ir}}\| - \varepsilon(\mathbf{h}_{\text{ir}}) &= \|\mathbf{H}\mathbf{h}_{\text{ir}} + \mathbf{R}^\top \mathbf{b}\| - \varepsilon(\mathbf{h}_{\text{ir}}) \\
&\stackrel{(a)}{\geq} \left(\|\mathbf{t} - \mathbf{h}_0\| - (\varepsilon(\mathbf{t}) + \varepsilon(\mathbf{h}_0)) \left(\frac{\|\mathbf{t} - \mathbf{h}_0\| + \varepsilon(\mathbf{h}_0)}{2\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t})} \right) \right) \\
&\quad - \left(\frac{\|\mathbf{t} - \mathbf{h}_0\| - (\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t}))}{2\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t})} \varepsilon(\mathbf{h}_0) \right) \\
&= \frac{1}{2\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t})} \left(\|\mathbf{t} - \mathbf{h}_0\| (2\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t})) \right. \\
&\quad \left. - (\|\mathbf{t} - \mathbf{h}_0\| + \varepsilon(\mathbf{h}_0))(\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t})) \right. \\
&\quad \left. - (\|\mathbf{t} - \mathbf{h}_0\| - (\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t}))) \varepsilon(\mathbf{h}_0) \right) \\
&= \frac{1}{2\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t})} \left(\|\mathbf{t} - \mathbf{h}_0\| (2\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t}) - \varepsilon(\mathbf{h}_0) - \varepsilon(\mathbf{t})) \right. \\
&\quad \left. - (\varepsilon(\mathbf{h}_0)(\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t})) - \varepsilon(\mathbf{h}_0)(\varepsilon(\mathbf{h}_0) + \varepsilon(\mathbf{t}))) \right) \\
&= 0,
\end{aligned}$$

where (a) applies the lower bound to the first term, and the upper bound to the second term. In conclusion, we have

$$\|\Phi(\mathbf{h}_{\text{ir}}) - \mathbf{h}_{\text{ir}}\| \geq \varepsilon(\mathbf{h}_{\text{ir}}),$$

i.e., $\Phi(\mathbf{h}_{\text{ir}}) \notin B(\mathbf{h}_{\text{ir}}, \varepsilon(\mathbf{h}_{\text{ir}}))$. This completes our proof. \square

B.2 MORE DISCUSSIONS ON ASMP 2.1.

Our Thm 2.3 is built upon Asmp 2.1. Informally, It assumes that the knowledge can be generated if representation takes some specific value. While this assumption may not hold especially in challenging scenarios (see App A for more discussions), it is reasonable for Thm 2.3.

Particularly, The goal of Thm 2.3 is to reveal how *linearity* in ReFT can *inevitably* hurt locality, *even if it appears successful in editing*. Therefore, our focus is on cases where *ReFT is capable of conducting the edits*. The existence of such cases are confirmed by our experiments, and by its effectiveness in diverse post-training tasks as demonstrated in Wu et al. (2024). Presuming such a success, given that ReFT can only update representations, Asmp 2.1 assumes that by updating representations to some targeted (possibly unknown) value, ReFT steers output y to convey the desired knowledge.

B.3 PROOF OF LEM 2.4

Lemma B.4. Let $\mathbf{R} = [\mathbf{r}_1; \dots; \mathbf{r}_r]$, $\mathbf{A} = [\mathbf{a}_1, \dots, \mathbf{a}_r]$, $\mathbf{b}^\top = (b_1, \dots, b_k)$, and $\mathbf{W}(\mathbf{h}) = \text{diag}(w_1(\mathbf{h}), \dots, w_r(\mathbf{h}))$. Then BaFT

$$\Phi(\mathbf{h}) = \mathbf{h} + \sum_{k=1}^r w_k(\mathbf{h}) \mathbf{r}_k (\mathbf{a}_k^\top \mathbf{h} + b_k - \mathbf{r}_k^\top \mathbf{h}),$$

can be expressed in a matrix form

$$\Phi(\mathbf{h}) = \mathbf{h} + \mathbf{R}^\top \mathbf{W}(\mathbf{h}) (\mathbf{A}\mathbf{h} + \mathbf{b} - \mathbf{R}).$$

When using constant weighting $\mathbf{W}(\mathbf{h}) = \mathbf{I}$, BaFT becomes to ReFT. Otherwise, rows of \mathbf{WR} are not orthonormal, making BaFT and ReFT nonequivalent.

Table 5: Hyper-parameters of different methods. For baselines, we only provided settings that were different from Wang et al. (2024e).

| | HParams. | LLaMA 2-7b-(chat) | LLaMA 3-8b-Instruct | Gemma 1.1-7b-Instruct |
|-----------------------|----------------------|---|---|---|
| | | Value | Value | Value |
| FT-L | / | Following Wang et al. (2024e)'s recommendation for LLaMA 2. | | |
| ROME | / | Following Wang et al. (2024e)'s recommendation for LLaMA 2. | | |
| MEMIT | / | Following Wang et al. (2024e)'s recommendation for LLaMA 2. | | / |
| AdaLoRA | Maximum Steps | 70 for Single and Continual Editing; 200 for Batched Editing | | |
| GRACE | Maximum Steps | 100 | 250 | 100 |
| | Lay. to Interven | 27 | 27 | 24 |
| WISE _{light} | Param. Updates | Restrict the original WISE logic to a randomly selected 1/8 area. | | |
| | Subspace Rank | 12 | | |
| | Pos. to Intervene | Last 3 of Input + Output | | |
| | Lay. to Intervene | 9;18;24;28 | 9;18;24;28 | 18;20;22;24 |
| BaFT & ReFT | Learning Rate | 3e-4 for Single and Continual Editing; 1e-4 for Batched Editing | | |
| | Maximum Steps | 40 for Single and Continual Editing; 70 for Batched Editing | | |
| | Locality Reg. (BaFT) | $\alpha = 0.01, \beta = 0.05, \gamma = 0.02$ | $\alpha = 0.01, \beta = 0.1, \gamma = 0.05$ | $\alpha = 0.01, \beta = 0.1, \gamma = 0.05$ |
| | Maximum Steps | 40 for Single and Continual Editing; 70 for Batched Editing | | |

Proof. The derivations essentially come from the fact that matrix product can be expressed by summation of outer products. In particular, we have

$$\begin{aligned}
\Phi(\mathbf{h}) &= \mathbf{h} + \sum_{k=1}^r w_k(\mathbf{h}) \mathbf{r}_k (\mathbf{a}_k^\top \mathbf{h} + b_k - \mathbf{r}_k^\top \mathbf{h}) \\
&= \mathbf{h} + \left(\sum_{k=1}^r w_k(\mathbf{h}) \mathbf{r}_k \mathbf{a}_k^\top - \sum_{k=1}^r w_k(\mathbf{h}) \mathbf{r}_k \mathbf{r}_k^\top \right) \mathbf{h} + \sum_{k=1}^r w_k(\mathbf{h}) \mathbf{r}_k b_k \\
&= \mathbf{h} + (\mathbf{R}^\top \mathbf{W}(\mathbf{h}) \mathbf{A} - \mathbf{R}^\top \mathbf{W}(\mathbf{h}) \mathbf{R}) \mathbf{h} + \mathbf{R}^\top \mathbf{W}(\mathbf{h}) \mathbf{b} \\
&= \mathbf{h} + \mathbf{R}^\top \mathbf{W}(\mathbf{h}) ((\mathbf{A} - \mathbf{R}) \mathbf{h} + \mathbf{b}) \\
&= \mathbf{h} + \mathbf{R}^\top \mathbf{W}(\mathbf{h}) (\mathbf{A}\mathbf{h} + \mathbf{b} - \mathbf{R}),
\end{aligned}$$

when $\mathbf{W}(\mathbf{h}) = \mathbf{I}$ takes the identity matrix, BaFT reduces to ReFT. This completes the proof. \square

C IMPLEMENTATION DETAILS

We provide more implementation details about different methods.

Throughout all experiments, BaFT used a logistic regression for $w_k(\mathbf{h})$ for all $k \in [r]$. ReFT was implemented as a special case of BaFT with constant weight $\mathbf{W} = \mathbf{I}$. Load balancing loss and the optional Locality regularization were removed as they were defined for weights. In addition, BaFT and ReFT used the same optimizer AdamW (Loshchilov & Hutter, 2019) and learning rate. An early stopping was performed if the training loss is smaller than a pre-specified threshold 0.01. We also added this early stopping to AdaLoRA after observing a similar improvement. We kept encountering numeric issues when running MEMIT on Gemma, so we omitted these results.. Other hyper-parameters are reported in Tab 5.

D MORE EXPERIMENT RESULTS

D.1 COMPLETE BATCHED CONTINUAL EDITING RESULTS

Here we report the complete Batched Editing results in Tab 6 and Tab 7 using batch size 10 and 50 respectively. The averaged result are shown in Fig 3. We noted that such a batched setting makes knowledge editing resemble more conventional continual learning (Miao et al., 2021; Chen et al., 2023; Wang et al., 2024c).

Table 6: Batched Editing performance on ZsRE dataset, evaluated after conducting T times of editing with batch size 10 sequentially. Best Avg. results are in **bold** and second best are underlined.

| | $T = 1$ | | | | $T = 10$ | | | | $T = 100$ | | | | | | | |
|-------------|-----------------------|------|------|-------------|----------|------|------|-------------|---------------------|------|------|-------------|------|------|------|-------------|
| | LLaMA 2-7b | | | | | | | | LLaMA 3-8b-Instruct | | | | | | | |
| | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. |
| MEMIT | 0.89 | 0.84 | 0.97 | 0.90 | 0.87 | 0.82 | 0.93 | 0.87 | 0.04 | 0.04 | 0.02 | 0.03 | 0.06 | 0.06 | 0.03 | 0.05 |
| FT-L | 0.43 | 0.42 | 0.87 | 0.57 | 0.12 | 0.10 | 0.17 | 0.13 | 0.03 | 0.03 | 0.00 | 0.02 | 0.01 | 0.01 | 0.00 | 0.01 |
| AdaLoRA | 1.00 | 0.85 | 0.88 | 0.91 | 0.95 | 0.82 | 0.87 | 0.88 | 0.46 | 0.45 | 0.77 | 0.56 | 0.34 | 0.34 | 0.75 | 0.48 |
| ReFT | 0.94 | 0.86 | 0.86 | 0.89 | 0.92 | 0.83 | 0.86 | 0.87 | 0.64 | 0.60 | 0.76 | 0.67 | 0.46 | 0.43 | 0.44 | 0.44 |
| BaFT (Ours) | 0.93 | 0.84 | 0.95 | 0.91 | 0.92 | 0.83 | 0.95 | 0.90 | 0.59 | 0.55 | 0.98 | 0.71 | 0.43 | 0.40 | 0.95 | 0.59 |
| | Gemma 1.1-7b-Instruct | | | | | | | | | | | | | | | |
| | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. |
| FT-L | 0.04 | 0.04 | 0.02 | 0.03 | 0.01 | 0.01 | 0.00 | 0.01 | 0.01 | 0.01 | 0.00 | 0.01 | 0.01 | 0.01 | 0.00 | 0.01 |
| AdaLoRA | 1.00 | 0.87 | 0.83 | 0.90 | 0.93 | 0.81 | 0.82 | 0.85 | 0.34 | 0.34 | 0.59 | 0.42 | 0.48 | 0.44 | 0.69 | 0.54 |
| ReFT | 0.90 | 0.75 | 0.86 | 0.84 | 0.88 | 0.72 | 0.84 | 0.81 | 0.60 | 0.56 | 0.74 | 0.63 | 0.48 | 0.42 | 0.43 | 0.44 |
| BaFT (Ours) | 0.90 | 0.74 | 0.91 | 0.85 | 0.89 | 0.73 | 0.90 | 0.84 | 0.45 | 0.41 | 0.87 | 0.58 | 0.49 | 0.43 | 0.78 | 0.57 |

Table 7: Batched Editing performance on ZsRE dataset, evaluated after conducting T times of editing with batch size 50 sequentially. Best Avg. results are in **bold**.

| | $T = 1$ | | | | $T = 10$ | | | | $T = 20$ | | | | | | | |
|-------------|-----------------------|------|------|-------------|----------|------|------|-------------|---------------------|------|------|-------------|------|------|------|-------------|
| | LLaMA 2-7b | | | | | | | | LLaMA 3-8b-Instruct | | | | | | | |
| | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. |
| MEMIT | 0.87 | 0.82 | 0.91 | 0.87 | 0.45 | 0.43 | 0.46 | 0.45 | 0.03 | 0.03 | 0.02 | 0.03 | 0.00 | 0.00 | 0.04 | 0.01 |
| FT-L | 0.39 | 0.39 | 0.63 | 0.47 | 0.13 | 0.10 | 0.07 | 0.10 | 0.07 | 0.05 | 0.02 | 0.05 | 0.01 | 0.00 | 0.00 | 0.00 |
| AdaLoRA | 1.00 | 0.86 | 0.76 | 0.87 | 0.78 | 0.69 | 0.79 | 0.75 | 0.51 | 0.51 | 0.76 | 0.59 | 0.42 | 0.40 | 0.69 | 0.50 |
| ReFT | 0.90 | 0.77 | 0.85 | 0.84 | 0.80 | 0.69 | 0.82 | 0.77 | 0.60 | 0.56 | 0.74 | 0.63 | 0.48 | 0.42 | 0.43 | 0.44 |
| BaFT (Ours) | 0.92 | 0.78 | 0.89 | 0.86 | 0.80 | 0.69 | 0.90 | 0.80 | 0.62 | 0.57 | 0.92 | 0.70 | 0.49 | 0.43 | 0.78 | 0.57 |
| | Gemma 1.1-7b-Instruct | | | | | | | | | | | | | | | |
| | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. | Rel. | Gen. | Loc. | Avg. |
| FT-L | 0.02 | 0.02 | 0.01 | 0.02 | 0.00 | 0.00 | 0.00 | 0.00 | 0.01 | 0.00 | 0.00 | 0.00 | 0.01 | 0.00 | 0.04 | 0.00 |
| AdaLoRA | 0.13 | 0.08 | 0.02 | 0.08 | 0.08 | 0.06 | 0.02 | 0.05 | 0.03 | 0.03 | 0.00 | 0.02 | 0.01 | 0.00 | 0.00 | 0.00 |
| ReFT | 0.88 | 0.68 | 0.79 | 0.78 | 0.71 | 0.57 | 0.72 | 0.67 | 0.48 | 0.42 | 0.61 | 0.50 | 0.49 | 0.43 | 0.78 | 0.57 |
| BaFT (Ours) | 0.87 | 0.68 | 0.87 | 0.81 | 0.72 | 0.57 | 0.83 | 0.71 | 0.50 | 0.45 | 0.81 | 0.59 | 0.49 | 0.43 | 0.78 | 0.57 |

D.2 DOWNSTREAM LOCALITY PERFORMANCE

In this section we study how different editing methods affect the LLM’s performance on unrelated downstream task, as an additional measure of locality. To this end, we follow [Yao et al. \(2023\)](#) and evaluate how the LLM’s ability of answering PIQA questions from [Bisk et al. \(2020\)](#) that are unrelated to the editing. The correctness is measured by whether the LLM chooses the correct answer according to its perplexity. For more details we refer the readers to [Yao et al. \(2023\)](#).

Table 8: Downstream task (PIQA) performance after being edited with 100 ZsRE knowledge. LLM uses LLaMA-2.

| | Base | AdaLoRA | FT-L | ROME | MEMIT | MELO | WISE _{light} | ReFT | BaFT |
|------------|------|---------|------|------|-------|------|-----------------------|------|------|
| PIQA Accu. | 0.77 | 0.48 | 0.75 | 0.5 | 0.52 | 0.77 | 0.77 | 0.77 | 0.77 |

D.3 MORE DISCUSSION ON BAFT VS WISE

In our experiment, we note that BaFT achieves better parameter efficiency and speeds, at a cost of slightly lower performance, resulting in an *efficiency-effectiveness trade-off*. Notably, this efficiency of BaFT can be valuable in applications that require frequent knowledge updates.

In order to improve the effectiveness of BaFT, one possible solution is to use more parameters, given that BaFT parameter efficiency is already much higher than state-of-the-art baseline WISE. As discussed in Sec 3.3, when WISE’s parameters number is reduced from WISE_{full} WISE_{light}, its performance degrades drastically. In comparison, BaFT uses even much less parameters, but maintains a highly comparable performance. Given this, we expect that using better training hyperparameters such as learning rate to make mild performance improvement, and more parameters are needed.

To validate this, we tried to add intervention to all layers (a common practice in ReFT ([Wu et al., 2024](#))) and increase the subspace rank to 16. This made BaFT performance on editing LLaMA-2 with 100 ZsRE knowledge increased from 0.80 (Rel: 0.73, Gen: 0.68, Loc: 0.98) to 0.82 (Rel: 0.77, Gen: 0.73, Loc: 0.95). However, we noted that going higher subspace rank didn’t help.

Therefore, we conjecture that to build larger BaFT (and ReFT), we need to incorporate sparsity on basis activation as well. This can help alleviate unintentional parameter updates as in GRACE ([Hartvigsen et al., 2024](#)) and WISE ([Wang et al., 2024d](#)). In addition, such a sparsity opens the door of automating position selections: as when all bases are inactivated, BaFT makes no updates on the representation, which is equivalent to dropping the position from the fine-tuning process. We plan to explore this direction in our future work.