Knowledge Extraction with Open-Source LLMs: An E.ON Data and Analytics Use Case

TUM School of Management, SS2023/2024

Francisco Monteiro
03771311

Ismet Buyar
03701344

Phillip Hauck
03662213

Abstract - This project studies the potential for open-source language models (LLMs) to be applied in question-answering use cases within E.ON's textual data processing framework. The present study focuses on optimizing BERT-family models, including German variants, and assesses their performance using the SQuAD and GermanQuAD datasets. The results of our experiments demonstrate that, through hyperparameter optimisation and parameter-efficient fine-tuning, larger models consistently outperform smaller ones. Furthermore, language-specific models were found to excel in their respective domains. We put forward a deployment architecture that is tailored for enterprise readiness, addressing issues of scalability, security and integration with E.ON's existing infrastructure. Our findings suggest that open-source LLMs offer viable, cost-effective alternatives to commercial solutions, providing E.ON with enhanced control over data and model customization while maintaining high performance in multilingual question-answering tasks.

I. Introduction

In the contemporary digital transformation landscape, large organizations deal with an overwhelming influx of textual data from both customers interactions and internal sources. This creates an operational and decision-making bottleneck for companies relying on manual processing and analysis of such data. E.ON, as a leading European energy and infrastructure network operator, epitomizes this challenge with its vast repositories of digitalized documentation. To face this data-intensive environment, companies, including E.ON, have increasingly turned to advanced natural language processing (NLP) tools for knowledge extraction tasks. There are innumerous commercial NLP solution providers. E.ON collaboration with Microsoft (one of these providers), leveraging OpenAI models through Azure, is currently presenting issues related to cost-effectiveness, quality regarding user experience, and customization (regarding fine-tuning and multimodality). Open-source large language models (LLM) constitute a potential alternative for these external commercial solutions, since they can be cheaper (depending on certain conditions discussed in section II) and can offer the company more control over model performance and customization.

Following the Cross-Industry Standard Process for Data Mining (CRISP-DM), we focus on the optimization of five open-source LLM's (Bert family models: Bert-base-cased, Roberta-base, Albert-large-v2, Roberta-large, Distilbert-base-uncased; German-finetuned models: gelectra-base, gbert-base, gbert-large), further evaluating their performance using metrics such as Exact Match, F1 and BLEU score. For that we use the datasets SQuAD1.1, SQuAD2.0, GermanQuAD. We focus on the Question Answering use case, which consists of posing questions about a given document and subsequent identification of an answer as spans of text within the document itself [1].

Section II presents the business understanding. Section III and IV illustrate the data understanding and preparation for our modeling and evaluation as discussed in sections V and VI, respectively. Section VI outlines deployment considerations, followed by a final section discussing the broader implications of this study.

II. Theoretical Background

A. Question Answering and Information retrieval

Question answering (Q&A) is the process of automatically providing responses to queries posed by humans in a natural language. The process entails the retrieval of relevant data (according to the question) from a vast repository of documents, subsequently presented in a succinct and accurate format. Q&A systems are designed to comprehend and process complex queries, frequently necessitating sophisticated natural language processing and information retrieval techniques to identify and extract the most relevant information from available data sources [2]. Information Retrieval (IR) is the process of obtaining relevant information from a substantial repository of data in response to a specific query. It includes the processes of indexing, searching, and ranking documents to align with the user's information needs. IR algorithms identify

documents chunks that are likely to contain the information sought by the user, thereby providing a ranked list of results based on relevance. These systems are foundational to search engines and other applications that require efficient access to large volumes of text data [2].

B. LLM/Closed/Open source

LLM's are advanced AI systems that can understand and generate different forms of content from text, code to images, video and audio [3]. LLMs leverage deep learning techniques and architectures, particularly transformers, to process vast amounts of textual data, allowing them to generate coherent and contextually relevant text, making them invaluable for a variety of NLP tasks. They are broadly categorized into closed-source and open-source models.

Open Source LLMs are language models whose source code is publicly available and can be freely accessed, used, modified, and distributed by anyone. "When you're doing research, you want access to the source code so you can fine-tune some of the pieces of the algorithm itself. With closed models, it's harder to do that" says Alireza Goudarzi, senior researcher of machine learning at GitHub [4]. While it offers benefits such as control, customization, community support, and transparency, it can require a significant financial and time investment in talent, customization, integration, maintenance, scaling, and security. It's also important to check the open-source software (OSS) licensing of these models before using them. These licenses provide a framework for how OSS can be shared, developed, and commercialized, being categorized as permissive licenses (free for commercial use), restricted licenses (restricted but not prohibited commercial use) or non-commercial licenses (prohibited commercial use) [4].

On the other hand, closed-source LLM's source code is not publicly available. They are often developed by large corporations and offered by API providers such as OpenAI, Microsoft, Google, and Amazon. Examples of closed-source LLMs are the GPT series from OpenAI. As these commercial solutions are usually off-the-shelf high-quality products, they're often far more accessible for companies that don't have in-house AI expertise and want to rely on vendor expertise for integration and maintenance, according to Eddie Aftandilian, a principal researcher at GitHub [4]. Although these models often deliver superior performance due to substantial resource investment in their development, they lack transparency and customization options, limiting user insight into their operation and preventing modifications to suit specific needs [4].

Regarding costs, while open-source models have low or no acquisition costs, the total cost of ownership can quickly skyrocket due to unexpected scaling, or simply because of the need for specialized talent [3]. Closed source models involve initial purchasing or licensing fees, but offer predictable costs including support and scaling packages, easing budgeting [3].

III. Business Understanding

E.ON Group is one of Europe's largest operators of energy networks, infrastructure, and customer solutions. With a workforce of 74 thousand employees and serving approximately 47 million customers across more than 15 countries, E.ON is a key player in the energy sector [5]. The contemporary energy market is increasingly interconnected, volatile, and complex, necessitating comprehensive digitalization to manage these challenges effectively. As part of its growth strategy, E.ON is digitizing and standardizing its entire system (networks, products, customer interfaces, and internal processes), with the ambitious goal of becoming the first "All Digital" energy company. E.ON's 2023 Annual Report highlights significant milestones in this digital transformation, including 40% of customers being already served via digital sales platforms, and the installation of 13.8 million smart energy meters, with over 15,000 secondary smart substations [5].

As a large, digitalized enterprise, E.ON manages an extensive volume of digital textual data from both customer interactions and internal sources. Advanced Natural Language Processing (NLP) tools are one solution for automatizing/streamlining this data processing, to further perform knowledge extraction tasks. Major commercial solution providers for these services (such as OpenAI, Microsoft, Amazon, Google, etc), offer a different set of Large Language Models (LLMs) and associated services that facilitate this automation [6]. By reading customer success stories [7] - [9], the most usual use cases include smart internal chatbots for information retrieval, performing sentiment analysis of customer feedback, and enhancing products and services with LLM capabilities, as demonstrated by companies like Iveco Group, Adidas, Clariant, PwC, and Lonely Planet.

In 2022, E.ON decided to integrate Microsoft Azure commercial solution for Large Language Model (LLM) services into its operations, incorporating Azure OpenAI models into its E.ON GPT platform. The focus was to perform tasks such as document summarization, question-answering, and citation generation. Companies typically prioritize several key requirements for these solutions: data privacy and security, ease of integration, cost-effectiveness, quality (regarding both model performance and user experience), and customization (possibility for fine-tuning, multimodality). After conversations with the company, we learned that E.ON is regrettably facing significant challenges with its current commercial solution, which fails to meet three of these critical requirements:

Quality regarding user experience (performance and deployable capacity): There are limitations in the performance and deployable capacity of models in Europe, resulting in suboptimal user experiences due to insufficient bandwidth (tokens-per-minute). This issue is exacerbated in Germany, where data center infrastructure is limited, which led E.ON to relocate its models to data centers in Sweden and Paris.

Cost-Effectiveness: Their current commercial solution is costly, with expenses escalating with the number of users. In addition to the model costs, E.ON incurs expenses from an embedding model service, also paid per API request. For instance, assuming each E.ON employee generates 10,000 (input and output) tokens daily over 22 workdays per month, with an average LLM service cost of €12.88 per million tokens [6], we project that the monthly costs can rise to more than €200,000 if all employees use this service under the previous assumptions. To calculate the price per million tokens, we only consider OpenAI models provided by Microsoft Azure (GPT-4Turbo, GPT-3.5 Turbo instruct, GPT-3.5 Turbo) and calculate their average blended price (blend of Input & Output token prices (3:1 ratio)) [6].

Customization: Fine-tuning restrictions in Europe present a significant challenge, since E.ON aims to adapt models to perform specific tasks related to pricing and grid management topics, which is hindered by these limitations.

Given these challenges, there is an opportunity to explore open-source LLM solutions as a viable alternative to commercial offerings. Open-source solutions can potentially meet all the necessary pre-requisites, provided E.ON invests in key resources. These resources include significant technical expertise in areas such as security protocols, integration with the company's IT infrastructure, model tuning, maintenance and scalability [6]. Additionally, it is imperative to consider the switching costs associated with discontinuing outsourcing to pursue in-house development. These costs include sunk investment costs, lost performance costs, management system upgrade costs, uncertainty costs, induction-retraining-performance costs, candidate search costs, information transfer/setup costs, and cognitive and behavioral learning costs [10].

IV. Data Understanding

In this chapter we will introduce the datasets used for training the different models.

The Stanford Question Answering Dataset (SQuAD) is a benchmark for machine reading comprehension tasks. It consists of question-answer pairs, where the answer is a segment of text from a corresponding Wikipedia article, the context. Version 1.1 was first introduced in a paper in 2016 by Pranav Rajpukar. With over 100.000 question-answer pairs distributed over 20.000 contexts from more than 500 articles, it provides a foundation for assessing basic reading comprehension [11][12].

The dataset consists of a development and a train set. The train set makes up most of the data and is used to train the model. The development set serves as a validation tool during training. The model's performance on the dev set is evaluated to identify areas for improvement. This helps prevent overfitting, a phenomenon where the model performs well on the training data but fails to generalize to unseen data [13][14].

SQuAD 2.0 was published in 2018 and introduces a more realistic scenario where the system might encounter unanswerable questions, making it a more robust benchmark. It incorporates the questions from its predecessor and adds more than 50.000 unanswerable questions [15]. This also becomes apparent when looking at figure 1, which shows the ratios between possible/impossible questions.

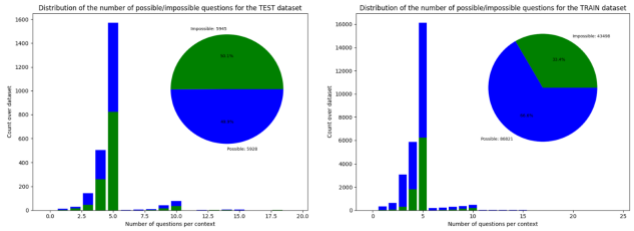

Figure 1: Distribution of possible/impossible questions per context with ratio for the test and train dataset of SQuAD 2.0

SQuAD remains a cornerstone for training and evaluating a wide range of machine reading comprehension models. Its large size and diverse question types allow researchers to assess a model's ability to understand complex factual passages and identify the correct answer span [14].

SQuAD's design has inspired the creation of similar datasets in other languages like GermanQuAD, a high-quality, human-labeled german QA dataset consisting of 13 722 questions. GermanQuAD was developed by Deepset AI and released in 2021. It aims to address the gap in high-quality German QA datasets, fostering research in non-English question answering. While following SQuAD's structure, it also incorporates "self-sufficient questions" that contain all relevant information for answering, without relying solely on the context [16]. Additionally, it introduces a distinction between long and short answers in its labeling.

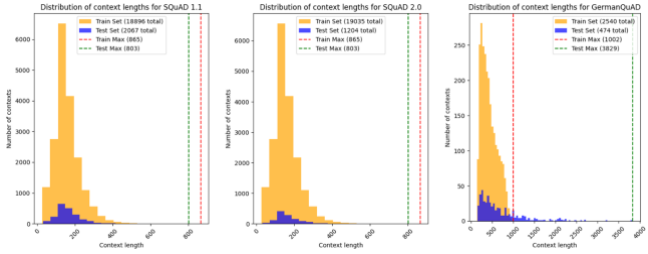

Figure 2: Distribution of context lengths per dataset for SQuAD 1.1, SQuAD 2 .0 and GermanQuAD

This is also reflected in the distribution of context lengths, as shown in figure 2. While the SQuAD datasets show a significantly higher number of contexts, the contexts themselves are longer.

V. Data Preparation

This section details the data preparation steps performed for the analysis, training and evaluation of a machine learning model for question answering (QA) tasks.

A . Data Cleaning for GermanQuAD

The GermanQuAD dataset required preliminary cleaning due to the Python library used for the JSON data loading encountering issues with special characters. Since the pandas data frames, the primary data structure for the analysis and training, were initially built using said JSON data loader, it was required to address these character encoding issues beforehand. However, during the development of the training algorithm, the decision was made to use the load_data function provided by the hugging face framework. This loads the dataset from the hugging face repository instead of a JSON file, thus avoiding the encoding issues.

B. Creating a Custom Data Split

While both SQuAD and GermanQuAD come with pre-defined train and development datasets, for our specific needs, we opted to create a custom data split. This process involves taking a part of the training dataset for the validation dataset. In general, we aim for all three datasets for a 70/20/10 train/validation/test. Figure 3 displays the data split with the ratios for the different datasets. Once the model's training is complete, the final evaluation of its performance is conducted on the unseen validation set. This provides an unbiased estimate of the model's generalizability to real-world data beyond the training set. Splitting the data into these subsets allows for more robust model evaluation and further prevents avoids overfitting [14].

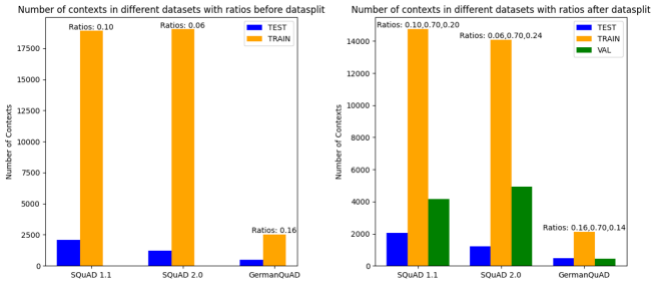

Figure 3: Number of contexts for each dataset before and after datasplit

For the SQuAD 2.0 data split, we invested efforts to also maintain the ratio of impossible/ possible questions to further increase the robustness of the validation process. The same goes for GermanQuAD and the ratios between short and long answers. While the questions in this dataset are labeled with an is_impossible attribute, as QuAD 2.0, it finds no utilisation since its set to false for all questions.

C. Context Truncation for Model Compatibility

The contexts within the SQuAD and GermanQuAD datasets may vary significantly in length. To ensure compatibility with the respective models' architectures, we implemented context truncation. This process involves limiting the maximum length of each context to a predefined value based on the model's maximum token count. Transformer-based models used for question answering, like those available through Hugging Face, have a predefined maximum token limit, which restricts the total number of tokens the model can process at once [1].

VI. Modelling

A. Experimental Framework

Our experimental framework was designed to comprehensively evaluate the performance of various BERT-family models on question-answering tasks across different languages and datasets. We conducted a series of experiments that included: (1) assessing the raw performance of five BERT-family models on SQuAD 1.1 and 2.0; (2) evaluating three German-specific models on GermanQuAD; (3) applying hyperparameter optimization techniques (Random Search, Grid Search, and TPE) to BERT-base-cased on both SQuAD versions; and (4) implementing PEFT techniques (LoRA and QLoRA) on BERT-base-cased for SQuAD 1.1 and 2.0, both with raw hyperparameters and optimized hyperparameters. Each experiment was run with three trials to ensure reliability. We compared models based on Exact Match, F1, and BLEU scores, allowing us to analyze the trade-offs between model size, language specificity, and fine-tuning efficiency. This comprehensive approach enabled us to explore the interaction between hyperparameter optimization and parameter-efficient fine-tuning,

providing E.ON with insights into both performance improvements and potential efficiency gains for their multilingual question-answering needs.

B. BERT Family Models

In our study, we evaluated a diverse set of BERT-family open-source language models to assess their performance on question-answering tasks. The selected models represent a range of architectures and sizes within the BERT family. BERT-base-cased, the original BERT model with case-sensitive inputs, served as our baseline. RoBERTa-base, an optimized version of BERT, was chosen for its improved training methodology. We included ALBERT-large-v2 to explore the benefits of its parameter-efficient design, which allows for a larger model with fewer parameters. RoBERTa-large was selected to investigate the performance gains of a larger model size. Finally, DistilBERT-base-uncased was included to assess the viability of a compressed, more efficient model. This selection allowed us to compare models of varying sizes, architectures, and training approaches within the BERT family, providing a comprehensive view of their capabilities in question-answering tasks.

C. German-finetuned Models

To address the specific requirements of German language tasks, we expanded our evaluation to include three German-specific models. The gelectra-base model, a German adaptation of Google's ELECTRA architecture, was chosen for its efficient pre-training approach. gBERT-base and gBERT-large, both variants of the BERT architecture specifically pre-trained on German text corpora, were included to assess the impact of model size and language-specific pre-training on performance. These models were selected to provide a comprehensive evaluation of German language capabilities, allowing us to compare their performance against the multilingual models and to assess the benefits of language-specific pre-training for E.ON's German language needs. The inclusion of these models enables us to provide insights into the trade-offs between general multilingual models and language-specific models for German text processing tasks.

D. Hyperparameter Optimization Algorithms

To optimize the performance of our models, we employed three distinct hyperparameter optimization algorithms: Random Search, Grid Search, and Tree-structured Parzen Estimator (TPE). These algorithms were chosen to explore different approaches to navigating the hyperparameter space. Random Search and TPE were configured with continuous ranges for learning rate (1e-5 to 5e-5) and weight decay (0.005 to 0.02), allowing for finer granularity in these parameters. In contrast, Grid Search used discrete values for all parameters to systematically explore predefined combinations. The number of epochs (2, 3, 4, 5) and batch size (8, 16, 32) were kept as discrete options across all methods. We utilized the Optuna framework to implement these optimization techniques, conducting three trials for each algorithm to balance thoroughness with computational constraints. This approach allowed us to compare the effectiveness of different optimization strategies in finding optimal hyperparameters for our models, potentially uncovering performance improvements that manual tuning might miss.

The choice of these three algorithms was motivated by their complementary strengths:

Random Search provides a baseline approach, efficiently sampling from the hyperparameter space without making assumptions about the underlying structure.

Grid Search offers a systematic exploration of predefined hyperparameter combinations, ensuring coverage of specific points of interest in the parameter space.

TPE, a more advanced Bayesian optimization method, learns from previous trials to focus on promising areas of the hyperparameter space, potentially leading to more efficient optimization [17].

By comparing these methods, we aimed to identify the most effective approach for tuning our models, while also gaining insights into the sensitivity of model performance to different hyperparameters.

E. Parameter-Efficient Fine-Tuning

In our study, we implemented two Parameter-Efficient Fine-Tuning (PEFT) techniques: Low-Rank Adaptation (LoRA) and Quantized LoRA (QLoRA). LoRA was configured using the PEFT library, targeting attention modules with a rank of 16 and a scaling factor of 32. QLoRA extended this approach by incorporating 4-bit quantization using BitsAndBytes, enabling even greater memory efficiency. These techniques allow for efficient adaptation of large language models with minimal parameter updates, potentially enabling the use of more powerful models within existing hardware constraints [18]. By implementing these PEFT methods, we aimed to explore the balance between model performance and resource efficiency, addressing E.ON's needs for cost-effective and powerful language models.

F. Sampling and Hyperparameters

For SQuAD 1.1 and 2.0, we utilized a 5% sample to manage computational resources, while for GermanQuAD, we employed the full dataset due to its smaller size. Baseline models were trained using standard hyperparameters, including a learning rate of 3e-5, 3 epochs, and a batch size of 16, with mixed precision training enabled. For larger models like ALBERT-large-

v2, RoBERTa-large and gBERT-large, we adjusted the batch size and implemented gradient accumulation to address memory limitations. Our implementation leveraged the Transformers library from Hugging Face and PyTorch, with training conducted on GPU-accelerated hardware, specifically T4 GPUs through Google Colab+. This setup allowed us to efficiently evaluate various models and techniques while balancing computational constraints and performance requirements.

VII. Evaluation

A. SQuAD 1.1 Performance

The raw performance of BERT family models on SQuAD 1.1 (Appendix B, Table 1) shows that larger models generally outperform smaller ones. RoBERTa-large achieved the best results with an EM of 0.816 and F1 score of 0.904, significantly outperforming the smaller DistilBERT-base-uncased.

For BERT-base-cased (Appendix B, Table 2), hyperparameter optimization yielded modest improvements. The TPE method performed best, increasing the F1 score from 0.714 to 0.742.

The application of PEFT techniques showed mixed results. LoRA (Appendix B, Table 3) initially performed poorly but showed significant improvement with hyperparameter optimization, reaching an F1 score of 0.473 with TPE. QLoRA (Appendix B, Table 4) struggled to match the baseline performance, with the best F1 score of 0.166 achieved through Grid Search.

B. SQuAD 2.0 Performance

On SQuAD 2.0, which includes unanswerable questions, we observed generally lower performance across all models (Appendix B, Table 5). RoBERTa-large again led with an EM of 0.786 and F1 score of 0.827.

BERT-base-cased showed some improvement with hyperparameter optimization (Appendix B, Table 6), with Grid Search achieving the best F1 score of 0.573, up from the raw 0.534.

Interestingly, both LoRA and QLoRA implementations on SQuAD 2.0 (Appendix B, Tables 7 and 8) resulted in consistent scores across all optimization techniques, suggesting potential implementation issues or dataset-specific challenges.

C. GermanQuAD Performance

For German language tasks (Appendix B, Table 9), gBERT-large demonstrated superior performance with an EM of 0.624 and F1 score of 0.821. This highlights the importance of language-specific pre-training for non-English tasks.

D. Key Findings

Model Size Impact: Larger models consistently outperformed smaller ones across datasets, indicating a strong correlation between model capacity and question-answering performance.

Hyperparameter Optimization: The effectiveness of hyperparameter optimization varied across models and datasets. TPE and Grid Search often yielded the best results, but improvements were generally modest.

PEFT Techniques: LoRA and QLoRA showed potential on SQuAD 1.1 but struggled with SQuAD 2.0. This suggests that these techniques may require further refinement for more complex question-answering tasks.

Dataset Complexity: Performance on SQuAD 2.0 was generally lower than on SQuAD 1.1, reflecting the increased difficulty of handling unanswerable questions.

Language-Specific Models: The strong performance of gBERT-large on GermanQuAD underscores the value of language-specific models for non-English tasks.

These findings provide valuable insights for E.ON in selecting and optimizing open-source language models for their multilingual question-answering needs. While larger models offer superior performance, the potential of PEFT techniques to balance performance and efficiency requires further investigation.

VIII. Deployment

While our current implementation is not enterprise-ready, we propose a potential deployment architecture for E.ON's future development. The architecture consists of Streamlit for the frontend, Django for the backend, Docker Compose for orchestration, and a private HuggingFace Model Registry as the model hub.

This setup would provide a user-friendly interface, robust backend operations, easy system management, and secure model storage. Key considerations for enterprise-readiness include scalability, security, model versioning, monitoring, compliance with data protection regulations, latency optimization, and multi-language support.

Significant development and testing would be required to fully integrate this system with E.ON's existing infrastructure and meet enterprise-level requirements. However, this architecture provides a solid foundation for transitioning from experimental results to a production-ready question-answering system.

IX. Conclusion

This study has provided a comprehensive evaluation of open-source language models for question-answering

tasks, with a focus on their potential application in E.ON's data processing pipeline. Our experiments spanned multiple BERT-family models, including language-specific variants for German, and explored the effectiveness of hyperparameter optimization and parameter-efficient fine-tuning techniques.

Key findings from our study include the superior performance of larger models like RoBERTa-large and ALBERT-large-v2 across both SQuAD 1.1 and SQuAD 2.0 datasets, suggesting that model capacity plays a crucial role in question-answering tasks. For German-specific tasks, gBERT-large demonstrated exceptional performance, highlighting the importance of language-specific pre-training.

Hyperparameter optimization techniques, particularly TPE and Grid Search, showed potential for improving model performance, though the gains were often modest. The exploration of parameter-efficient fine-tuning methods (LoRA and QLoRA) yielded mixed results, with some improvements on SQuAD 1.1 but challenges on the more complex SQuAD 2.0 dataset.

These findings offer E.ON valuable insights into the current state of open-source language models for question-answering tasks. The strong performance of larger models suggests they could be viable alternatives to commercial solutions, potentially offering significant cost savings and greater flexibility. However, the mixed results from PEFT techniques indicate that more research is needed to fully leverage these methods for E.ON's specific use cases.

The proposed deployment architecture, leveraging Streamlit, Django, Docker Compose, and a private HuggingFace Model Registry, provides a roadmap for transitioning from experimental results to a production-ready system. However, significant work remains to address enterprise-level requirements such as scalability, security, and compliance.

X. Outlook

Looking ahead, several avenues for future work emerge to build upon the findings of this study and address the challenges identified. Further optimization of PEFT techniques for question-answering tasks, particularly for more complex datasets like SQuAD 2.0, could yield significant improvements in efficiency without sacrificing performance.

Exploration of multi-lingual models presents an opportunity to address E.ON's diverse language needs more efficiently. This could potentially reduce the need for maintaining separate models for each language, streamlining the deployment process.

Investigation of domain-specific pre-training offers a promising direction to better align models with E.ON's specific content areas, such as energy and infrastructure. This tailored approach could enhance performance on E.ON's proprietary data and unique use cases.

As the system moves closer to production, evaluation of inference speed and resource requirements will become crucial to ensure real-world applicability. This is particularly important in the context of the proposed deployment architecture, where balancing performance with computational efficiency will be key.

Development of a robust model versioning and A/B testing framework will facilitate continuous improvement of deployed models. This will allow E.ON to iteratively refine their question-answering system based on real-world performance and user feedback.

As the field of natural language processing continues to evolve rapidly, E.ON should maintain a flexible approach, continuously evaluating new models and techniques as they emerge. By building on the insights from this study and addressing the identified challenges, E.ON can work towards a robust, efficient, and cost-effective question-answering system that meets its diverse multilingual needs while maintaining control over its data and infrastructure.

The journey from this experimental study to a fully deployed, enterprise-ready system will require careful planning, further research, and iterative development. However, the potential benefits in terms of improved data processing, cost savings, and enhanced flexibility make this a worthwhile endeavor for E.ON's digital transformation strategy.

References

[1] n.d. “NLP Course, Question Answering.” Hugging Face. Accessed: Jul. 21, 2024. [Online.] Available: https://huggingface.co/learn/nlp-course/chapter7/7

[2] D. Jurafsky and J. H. Martin, Speech and Language Processing, 3rd ed. Accessed: Jul. 21, 2024. [Online.] Available: https://web.stanford.edu/~jurafsky/slp3/14.pdf [Page: 14]

[3] M. Malec. "Open Source vs. Closed LLMs Guide." Hatchworks. Accessed: Jul. 21, 2024. [Online.] Available: https://hatchworks.com/blog/gen-ai/open-source-vs-closed-llms-guide/

[4] A. D. Adimi. "Comparison & Cost Analysis: Should we invest in open-source or closed-source LLMs?." Medium. Accessed:

Jul. 21, 2024. [Online.] Available: https://medium.com/@ja_adimi/comparison-cost-analysis-should-we-invest-in-open-source-or-closed-source-llms-bfd646ae1f74

[5] E.ON, “Integrated annual report,” 2023.

[6] n.d., "Artificial Intelligence and Analysis" Artificial Analysis. Accessed: June 9, 2024. [Online.] Available: https://artificialanalysis.ai/

[7] n.d. "Stories." OpenAI. Accessed: June 9, 2024. [Online.] Available: https://openai.com/news/stories/

[8] n.d. "AI Customer Stories." Microsoft. Accessed: June 9, 2024. [Online.] Available: https://www.microsoft.com/en-us/ai/ai-customer-stories

[9] n.d. "Bedrock Testimonials.” Amazon Web Services. Accessed: June 9, 2024. [Online.] Available: https://aws.amazon.com/bedrock/testimonials/.

[10] D. Whitten, S. Chakrabarty, R. Wakefield. “The strategic choice to continue outsourcing, switch vendors, or backsource: Do switching costs matter?” D. Whitten et al. / Information & Management, vol 47, issue 3, pp. 167–17, April 2010. doi: https://doi.org/10.1016/j.im.2010.01.006

[11] P. Rajpurkar, J Zhang, K. Lopyrev, P. Liang. “SQuAD: 100,000+ Questions for Machine Comprehension of Text” [Online.] Available: arXiv preprint arXiv:1606.06901, 2016

[12] n.d. “Dataset card for SQuAD” Hugging Face. Accessed: Jul. 21, 2024. [Online.] Available: https://huggingface.co/datasets/rajpurkar/squad

[13] n.d. “NLP Course” Hugging Face. Accessed: Jul. 21, 2024. [Online.] Available: https://huggingface.co/learn/nlp-course/

[14] G. James, D. Witten, T. Hastie, R. Tibshirani. “An Introduction to Statistical Learning: with applications in R (2nd ed.)”. Springer, 2013.

[15] Rajpurkar, P., Jia, R., & Liang, P. “Know what you don't know: Unanswerable questions for SQuAD” [Online.] Available: arXiv preprint arXiv:1806.03822, 2018

[16] n.d. “Dataset card for GermanQuAD” Hugging Face. Accessed: Jul. 21, 2024. [Online.] Available: https://huggingface.co/datasets/deepset/germanquad

[17] Khoei, T. T, Ismail, S & Kaabouch, N., "Boosting-based Models with Tree-structured Parzen Estimator Optimization to Detect Intrusion Attacks on Smart Grid," *2021 IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, New York, NY, USA, 2021, pp. 0165-0170, doi: 10.1109/UEMCON53757.2021.9666607.

[18] Hu, T., Shen, Y., Wallace, E., & Tian, Y. (2023). QLoRA: Efficient Finetuning of Quantized LLMs. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*.

Appendix A

Table containing different LLM's, respective API providers, and metrics such as model's quality, price, output speed, and total response time. All data was extracted from the website Artificial Analysis [6].

Model	API Providers	Quality	Price (USD per 1M tokens)	Output Speed	Total Response Time
GPT-4o	OpenAI	100	7.5	87	1.72
GPT-4 Turbo	OpenAI, Microsoft Azure	94	15	28.27 (OpenAI) to 32.71 (Microsoft Azure)	3.58 (Microsoft Azure) to 4.25 (OpenAI)
GPT-4	OpenAI, Microsoft Azure	84	37.5	28.01 (OpenAI) to 22.26 (Microsoft Azure)	4.57 (OpenAI) to 5.11 (Microsoft Azure)
GPT-3.5 Turbo Instruct	OpenAI, Microsoft Azure	60	1.63	67.55 (Open AI) to 133.84 (Microsoft Azure)	1.37 (Microsoft Azure) to 1.89 (OpenAI)
GPT-3.5 Turbo	OpenAI, Microsoft Azure	59	0.77	71.49 (Microsoft) to 76.03 (OpenAI)	1.73 (Microsoft Azure) to 1.87 (OpenAI)
Gemini 1.5 Pro	Google	95	5.25	61.64	2.65
Gemini 1.5 Flash	Google	84	0.53	164.99	1.61
Gemini 1.0 Pro	Google	62	0.75	88.75	3.04
Llama 3 (70B)	Microsoft Azure, Amazon Bedrock, Groq, Together.ai, Perplexity, Fireworks, Lepton AI, Deepinfra, Replicate, Databricks (only Llama 3 (70B)), and OctoAI	83	0.61 (Deepinfra) to 5.67 (Microsoft Azure)	16.51 (Microsoft Azure) to 348 (Groq)	0.64 (Groq) to 9.31 (Microsoft Azure)
Llama 3 (8B)		64	0.6 (Deepinfra) to 0.55 (Microsoft Azure)	1214 (Groq) to 77 (Microsoft Azure)	0.42 (Groq) to 3.19 (Replicate)
Mistral Large	Mistral, Microsoft Azure, Amazon Bedrock	76	6	24 (Microsoft Azure) to 39.77 (Mistral)	2.96 (Mistral) to 6.49 (Microsoft Azure)
Mixtral 8x22B	Mistral, Together.ai, Perplexity, Fireworks, Deepinfra, and OctoAI	71	0.65 (Deepinfra) to 3.67 (Mistral)	39.58 (Together.ai) to 91.86 (OctoAI)	1.35 (OctoAI) and 3.1 (Together.ai)
Mistral Small	Mistral, Microsoft Azure	71	1.5	53.91 (Mistral) to 61.29 (Microsoft azure)	2.17 (Mistral) to 2.79 (Microsoft Azure)
Mistral Medium	Mistral	70	4.05	36.67	3.39
Mixtral 8x7B	Same as Mixtral 8x22B plus: Amazon Bedrock, Groq, Lepton AI, Replicate, Databricks	61	0.24 (Groq and Deepinfra) to 0.7 (Mistral)	52.05 (Amazon Bedrock) to 555.21 (Groq)	0.56 (Groq) to 2.71 (Replicate)
Claude 3.5 Sonnet	Anthropic	98	6	79	2.18
Claude 3 Opus	Anthropic, Amazon Bedrock	93	30	23.16 (Anthropic) to 27.4 (Amazon Bedrock)	5.72 (Anthropic) to 6.15 (Amazon Bedrock)
Claude 3 Haiku	Anthropic, Amazon Bedrock	74	0.5	103.65 (Amazon Bedrock) 146.55 (Anthropic)	1.26 (Anthropic) to 1.42 (Amazon Bedrock)
Command-R+	Microsoft Azure, Cohere	75	6	56.73 (Cohere) to 59.67 (Microsoft Azure)	2.13 (Cohere) to 2.15 (Microsoft Azure)

Notes:
Price: blend of input and output token prices (3:1 ratio).
Output Speed: tokens per second received while the model is generating tokens (ie. after first chunk has been received from the API).
Total Response Time: Time to receive a 100 token response. Estimated based on Latency (time to receive first chunk) and Output Speed (output tokens per second).

Appendix B

Tables containing the study's experiment results.

Table 1: Raw performance of BERT family models on SQuAD 1.1 (no PEFT)

Model	Exact Match	F1 Score	BLEU Score
RoBERTa-base	0.750	0.851	0.1459
ALBERT-large-v2	0.775	0.873	0.1638
RoBERTa-large	0.816	0.904	0.1705
DistilBERT-base-uncased	0.540	0.663	0.0922

Table 2: BERT-base-cased performance on SQuAD 1.1 (baseline, no PEFT)

Optimization	Exact Match	F1 Score	BLEU Score
Raw	0.604	0.714	0.0264
Best Random	0.602	0.602	0.0257
Best Grid	0.602	0.727	0.0262
Best TPE	0.623	0.742	0.0272

Table 3: BERT-base-cased with LoRA on SQuAD 1.1

Optimization	Exact Match	F1 Score	BLEU Score
Raw	0.042	0.102	0.0352
Best Random	0.038	0.137	0.0469
Best Grid	0.172	0.315	0.0382
Best TPE	0.333	0.473	0.0328

Table 4: BERT-base-cased with QLoRA on SQuAD 1.1

Optimization	Exact Match	F1 Score	BLEU Score
Raw	0.008	0.080	0.0402
Best Random	0.044	0.140	0.0244
Best Grid	0.034	0.166	0.0356
Best TPE	0.011	0.080	0.0310

Table 5: Raw performance of BERT family models on SQuAD 2.0 (no PEFT)

Model	Exact Match	F1 Score	BLEU Score
RoBERTa-base	0.590	0.637	0.1455
ALBERT-large-v2	0.737	0.775	0.1920
RoBERTa-large	0.786	0.827	0.1705
DistilBERT-base-uncased	0.460	0.479	0.0417

Table 6: BERT-base-cased performance on SQuAD 2.0 (baseline, no PEFT)

Optimization	Exact Match	F1 Score	BLEU Score
Raw	0. 504	0. 534	0. 0937
Best Random	0. 504	0. 535	0. 0943
Best Grid	0. 543	0. 573	0. 1102
Best TPE	0. 524	0. 552	0. 1136

Table 7: BERT-base-cased with LoRA on SQuAD 2.0

Optimization	Exact Match	F1 Score	BLEU Score
Raw	0.521	0. 521	0.0000
Best Random	0. 521	0. 521	0.0000
Best Grid	0. 521	0. 521	0.0000
Best TPE	0.521	0.521	0.0000

Table 8: BERT-base-cased with QLoRA on SQuAD 2.0

Optimization	Exact Match	F1 Score	BLEU Score
Raw	0.521	0. 521	0.0000
Best Random	0. 521	0. 521	0.0000
Best Grid	0. 521	0. 521	0.0000
Best TPE	0.521	0.521	0.0000

Table 9: Raw performance of German-finetuned models on GermanQuAD (no PEFT)

Model	Exact Match	F1 Score	BLEU Score
gelectra-base	0.527	0.707	0.3303
gBERT-base	0.529	0.714	0.3237
gBERT-large	0.624	0.821	0.3940