Knowledge Extraction with Open-Source LLMs

A Challenge by EON

Data Analytics in Applications, Summer Semester 2024
Research Institute for Management, Logistics and Production
TUM School of Management

Fabian Arnold, Tuğçe Çağlayan, Şeyma Çakır, Özlem Şenel, and Özgür Kaan Varlık

***Abstract*— E.ON would like to consider open-source alternatives to commercial solutions for their virtual assistant E.ON GPT. This study evaluates BERT, RoBERTa, and ALBERT on question answering using the SQuAD 1.0 and SQuAD 2.0 datasets. We find that fine-tuning these open-source models demonstrates their capability to handle both answerable and unanswerable questions effectively. Specifically, the fine-tuned RoBERTa Large model provides highly accurate answers, showing potential as a viable alternative to commercial solutions. Question Answering systems can transform business operations by enabling natural language queries for data insights, improving customer support with fast and accurate responses, and enhancing internal knowledge management. The results highlight the advantages of open-source models, such as greater flexibility, cost-effectiveness, and data privacy across various business contexts.**

Index Terms*—*natural language processing, question answering, open-source, BERT, SQuAD

I. INTRODUCTION

With the introduction of ChatGPT in 2022, Generative Artificial Intelligence (GenAI) has seen massive growth with 92% of Fortune 500 companies using OpenAI's technology [1]. The GenAI market size is forecasted to continue to increase with a compounded annual growth rate of 46.47% from 2024-2030 resulting in a market volume of €329.36 billion by 2030 [2]. E.ON, a large operator of energy networks and energy infrastructure in Europe, is also developing Gen AI solutions for integrated knowledge and operational planning [3]. E.ON GPT is an internal generative artificial intelligence assistant based on OpenAI's GPT3.5 technology for E.ON employees which collects and provides data offering support with research activities and word processing [4]. The data challenge for E.ON consists of evaluating open-source models on question answering as alternatives to commercial solutions.

A. Natural Language Processing and Question Answering

Natural Language Processing (NLP) is the ability of computers to understand and communicate natural language. Question answering is an NLP task where given a question and context information in the form of a segment of text, an answer is extracted from the context. Question answering models are commonly used in conversational applications which can respond to questions in natural language using a knowledge base [5]. Providing question answering models with a domain-specific knowledge base and business documents, allows the model to quickly find information from large bodies of text facilitating efficient business intelligence.

B. Open-Source Models

E.ON describes multiple problems with the current commercial solution of OpenAI's GPT3.5 technology. The commercial solution is not always able to serve the requested bandwidth reducing deployable capacity. Furthermore, there is high cost scaling with user growth as well as limitations in fine-tuning models to adapt the model to E.ON's knowledge of energy networks and infrastructure. In contrast to commercial solutions, open-source models allow for greater flexibility and accessibility. Through the use of a knowledge base, open-source models can be tailored to and adapted to better serve the needs of users of E.ON GPT.

C. SQuAD Datasets

The SQuAD datasets were created at Stanford and have become a standard benchmark for evaluating question answering models. SQuAD 1.0 was created by crowdworkers from the United States and Canada on a set of Wikipedia articles. SQuAD 2.0 was later introduced because although models could often locate the correct answer they tended to make unreliable guesses on questions with no answer in the context. For example, a model that gets 86% F1 on SQuAD 1.0 achieves only 66% F1 on SQuAD 2.0 [6], [7].

D. CRISP-DM

This paper's structure is based on the CRISP-DM process model. CRISP-DM, which stands for Cross Industry Standard Process for Data Mining, is a popular framework for executing data mining projects. The model contains 6 project phases: business understanding, data understanding, data preparation, modeling, evaluation, and deployment. Business understanding aims to determine business objectives, assess the current situation, and produce a plan for the project. In the data understanding phase, the initial data is collected and then explored to verify the data quality. Data preparation is used to select, clean, and format the data. The modeling phase involves selecting models to test. The results are evaluated and the process is reviewed to determine the next steps, in the evaluation phase. Finally, deployment develops a deployment plan and produces a final report on the project [8]. As depicted in Fig. 1, the CRISP-DM model is an iterative model where the project is repeatedly refined through multiple iterations of the project phases.

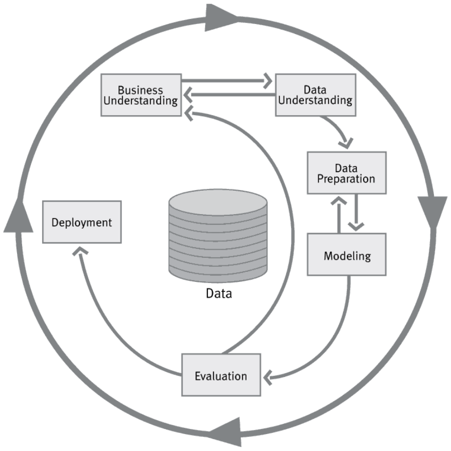

Fig. 1. CRISP-DM model

E. Overview

This paper is structured according to the CRISP-DM model: Section II covers a thorough overview of the business understanding regarding the data challenge. Next, section III describes and explores the datasets to provide data understanding. Section IV covers the data preparation and data cleaning used to obtain the training data for the models. Section V delves into model selection and justification, as well as hyperparameters and fine-tuning. Afterward, section VI presents the evaluation of the models, explaining the evaluation metrics and comparing the selected models. Section VII discusses the possible deployment options of the models. Section VIII covers the business implications of question answer models outlining potential use cases. Finally, section IX presents the conclusions drawn from the data challenge.

II. BUSINESS UNDERSTANDING

Starting with a thorough understanding of the business situation and objectives of E.ON and how the project helps E.ON is the foundation of a successful project. E.ON is one of Europe's largest operators of energy networks and energy infrastructure. In 2023, E.ON had an EBITDA of around €9.4 billion and served over 47 million retail customers. E.ON employs over 70,000 people all over Europe with the majority working in Germany, United Kingdom, Romania, and Hungary. In the Integrated Annual Report E.ON released in 2023 three strategic pillars were outlined: sustainability, digitalization, and growth [3]. As our project revolves around the viability of the use of open-source models for E.ON GPT, this project mainly addresses digitalization. However, since E.ON GPT facilitates business intelligence and analysis the project also contributes towards sustainability and growth.

A. Situation Assessment

E.ON GPT is currently using a commercial solution namely OpenAI's GPT3.5. Tabel I shows OpenAI's pricing of gpt-3.5-turbo tokens for input, output, and training.

TABLE I
PRICING TABLE FOR OPENAI'S GPT-3.5-TURBO MODEL [9]

Model	Input Tokens	Output Tokens	Training Tokens
gpt-3.5-turbo	$3.00 / 1M	$6.00 / 1M	$8.00 / 1M

In comparison, although exact costs are hard to estimate, using open-source models will require a large upfront cost but will have a lower cost per token after the model is set up. Therefore, in regards to cost, the decision between using an open-source model or OpenAI's API depends largely on the expected utilization of the model. Another important factor other than cost to consider when deciding between open-source and commercial solutions is data privacy as well as vendor lock-in. For commercial solutions when fine-tuning on a knowledge base, sensitive or valuable business information may have to be excluded due to the risk of exposing data which would decrease the value that the resulting model can provide. Furthermore, any fine-tuning on commercial solutions would most likely lead to vendor lock-in due to the inability to transfer the model to another commercial provider. In comparison, a fine-tuned open-source model can easily change hosting providers or even be self-hosted.

Based on the business objectives and the situation assessment above, the project aims to compare the performance of several open-source models on the SQuAD datasets to present an alternative to the current commercial solution.

III. DATA UNDERSTANDING

For our project, we utilized the Stanford Question Answering Dataset (SQuAD), versions 1.0 and 2.0. These datasets are widely used benchmarks for evaluating the performance of machine learning models in question-answering tasks.

A. Data Characteristics

SQuAD1.0, introduced by Rajpurkar et al. in 2016 [6], is a large-scale reading comprehension dataset consisting of questions posed by crowdworkers on a set of Wikipedia articles. The key characteristics of SQuAD1.0 are as follows:

- **Size:** The dataset contains over 100,000 question-answer pairs.
- **Structure:** Each entry in the dataset includes a paragraph from a Wikipedia article, a question about the paragraph, and a span of text from the paragraph that contains the answer.
- **Answer Type:** All questions have an answer that is a continuous span of text within the corresponding paragraph.

The SQuAD2.0 was introduced in 2018 by Rajpurkar et al [7]. In addition to adding more than 50,000 unanswerable questions, it includes all the questions from SQuAD1.0. The purpose of these additional questions is to evaluate the model's ability to distinguish between answerable and unanswerable questions, hence elevating the dataset's complexity and robustness. The key characteristics of SQuAD2.0 are:

- **Size:** The dataset includes over 150,000 questions, original SQuAD1.0 questions with the new unanswerable ones.
- **Structure:** Each entry contains a paragraph, a question, and the answer span if it exists similar to SQuAD1.0. For unanswerable questions, the answer span is left empty.
- **Answer Type:** The dataset includes both answerable questions (with text spans as answers) and unanswerable questions (where no span exists in the paragraph).
- **Data Source:** Both datasets take their paragraphs from a diverse range of Wikipedia articles to provide a broad range of topics and contexts. This diversity helps in the training of robust, generalized models across a variety of domains.
- **Annotation Process:** Crowdworkers generated the questions and responses in both datasets. For SQuAD1.0, workers were instructed to create questions using the given paragraphs as a guide. For SQuAD2.0, more workers were tasked with generating questions that weren't covered by the paragraphs.

B. Data Structures

Each entry in the SQuAD1.0 dataset consists of a series of articles, with each article containing several paragraphs. Each paragraph includes:

- **title:** The title of the article.
- **paragraphs:** An array of paragraphs within the article.
- **context:** The context passage from which questions are derived.
- **qas:** An array of question-answer sets for the paragraph.
- **id:** A unique identifier for each question.
- **question:** The question posed about the context.
- **answers:** An array of answers for the question.
- **text:** The text of the answer.
- **answer start:** The starting character position of the answer in the context.

Each entry in the SQuAD 2.0 dataset follows a similar structure. However, is_impossible could take true value and answers, an array of answers for the question, is empty if is_impossible is true in SQuAD 2.0.

IV. DATA PREPARATION

The data preparation process involves several key steps, including data cleaning, tokenization, and preprocessing. The objective is to convert raw SQuAD data into a structured format suitable for model training and evaluation.

A. Data Cleaning

Initially, we load the SQuAD 2.0 dataset in JSON format and extract the essential fields: titles, contexts, questions, and answers. To ensure data quality:

1) **Filtering Out No-Answer Examples:** For SQuAD 1.0, question-answer pairs with missing answers are removed. In contrast, for SQuAD 2.0, questions without answers are crucial to train the model in recognizing unanswerable questions.
2) **Creating a Dictionary for Tokenization:** The cleaned data is transformed into a structured dictionary including:
   - **id:** Unique identifier for each question-answer pair.
   - **title:** The title of the article.
   - **context:** The passage from which the answer is derived.
   - **question:** The question posed about the context.
   - **answers:** Contains text (the answer itself) and answer_start (the position of the answer within the context).

   This structured dictionary is then used to create Dataset objects for training and validation.

B. Tokenization

Tokenization is a critical step in the data preparation process. It involves transforming unprocessed text into a list of tokens so that machine learning models can process it. Tokenization standardizes the input text, ensuring consistency across many data samples, which helps in the model's ability to detect and successfully generalize patterns, making this step crucial. Furthermore, natural language text can contain a wide range of punctuation, special characters, and variants of the same word. Tokenization helps manage this variability by breaking down text into consistent units. Furthermore, different models require input in specific tokenized formats. Proper tokenization ensures compatibility with the model architecture, enabling effective training and inference.

1) **Model Choice and Tokenizers:**
   - **Albert:** For Albert models, we used the 'albert-base-v2' tokenizer.
   - **Roberta:** For Roberta models, we used the 'facebook/roberta-base' tokenizer.
   - **Bert:** For Bert models, we used the 'bert-base-uncased' tokenizer.
2) **Tokenization Process:** Tokenization transforms unprocessed text into token sequences that the models can understand. This involves handling long contexts by using a sliding window method. By producing several training instances from a single long context, the sliding window method helps cover different context segments and improves the model's capacity to learn from a variety of data.

C. Preprocessing

1) **Training Data:** The preprocess_training_examples function aligns token positions with answer spans in the context. Key tasks include:
   - **Token Alignment:** Mapping the character offsets of answers to token positions within the context.

- **Start and End Position Calculation:** Determining the start and end positions of the answer within the tokenized sequence, handling cases where answers span multiple tokens.

2) **Validation Data:** The preprocess_validation_examples function prepares the validation data for evaluation by:
   - **Tokenization:** Converting questions and contexts into token sequences.
   - **Handling Long Contexts:** Applying the sliding window technique to ensure comprehensive coverage of the context.
   - Mapping: Adjusting offset mappings to reflect token positions accurately, facilitating precise evaluation.

Finally, the processed datasets are organized into DatasetDict objects for training and validation, ready for model training and performance evaluation. By using the appropriate tokenizers for each model and carefully preparing the data, we ensure that the models are trained and validated on clean, consistent, and well-structured information, ultimately enhancing their capability to address a wide range of question-answering tasks.

V. MODELING

In this section, we discuss the fine-tuning process on the SQuAD dataset using several transformers based models. For pre-trained models, we are going to use BERT, RoBERTa, and ALBERT, which are well-known and were trained on various NLP tasks. First, we begin with an explanation of the model choices and their configurations, followed by a rundown of the preprocessing steps that are relevant for the training (in particular the changes made moving from SQuAD 1.0 to 2.0). We also cover the loss function and optimizer setup used. Finally, we give an overview of the network architectures of the selected models, focusing on their relevance to our task at hand.

A. Model Selection and Justification

Transformer-based models have shown state of the art performance for sequence transduction tasks [10], and to investigate these architectures further in the context of our question-answering task, we will examine three more modern transformer-based model implementations: BERT (Bidirectional Encoder Representations from Transformers), RoBERTa (Robustly Optimized BERT Pretraining Approach), ALBERT (A Lite BERT). The reason why we decided to go with these models is because of their efficiencies in many NLP tasks, and their proven performance on the Stanford Question Answering Dataset benchmark [11].

- BERT is a foundational model which introduced the concept of bidirectional training of transformer models. This means that with BERT's architecture, it is possible to consider the context from both directions. The base model, bert-base-uncased, is a 12-layer transformer with 110 million parameters, making it a decent performer yet still computationally inexpensive for fine-tuning on SQuAD [10].
- RoBERTa is another model that improves upon BERT by optimizing the pretraining phase. It uses more data and requires a longer training schedule, removes the next-sentence prediction task from pretraining, and changes the masking pattern applied to the training samples dynamically. These modifications lead to better performance, as evidenced by RoBERTa's superior results in many NLP benchmarks (and also in our applications!). The model we selected, which is roberta-base, provides a good balance between performance and computational efficiency [12].
- ALBERT aims to reduce the occupied memory and increase the training speed of BERT. By using parameter-sharing techniques across layers (i.e. cross-layer parameter sharing) and factorized embedding parameterization, it reduces the number of parameters while maintaining a decent performance. The albert-base-v2 model, with fewer parameters than BERT, is particularly attractive for environments with limited resources as the number of parameters is significantly lower for this model [13].

B. Preprocessing

Through preprocessing, we need to make sure that the data is in the right format that conforms with the model structure. To this end, several steps were implemented:

1) **Tokenization:** Because models like BERT, RoBERTa, and ALBERT require inputs in the form of tokenized IDs rather than raw text, this step is a must.
2) **Sequence Length:** The maximum sequence length is set to 384, as we want to make sure that inputs are well within the model capacities and they can be used with the SQuAD dataset [10].
3) **Document Stride:** A document stride of 128 is used. This helps the sliding window approach to take overlapping parts of the context, as it ensures that answers near the boundaries of the windows aren't missed by the models.
4) **Handling Null Answers:** The move to SQuAD 2.0 adds questions that don't have answers to the dataset. To handle this, a special token position is used to represent the no answer. Of course, this step makes the training set more complex, forcing the model to learn the differences between answerable and unanswerable questions as well.

C. Loss Function and Optimizer

The loss function and optimizer are crucial for the training process. They influence how well the model learns from the data and eventually generalizes to unseen data. An overview of the loss functions and optimizer we used are given as follows:

Loss Function: For the question-answering task at hand, we opted for the Cross-Entropy Loss, specifically designed to operate on the predictions of the start and end position of the answers. It measures the performance of a classification model whose output is a probability value between 0 and

1 and it is calculated separately for the start and end positions and then averaged. This step ensures that the model learns to correctly predict both the beginning and the end of the answer span within the context. For SQuAD 2.0, an additional binary cross-entropy loss is used to handle the classification task of determining whether a question is answerable.

Optimizer: We used the AdamW (Adam with Weight Decay) optimizer for the training. Adam is a well-known optimizer that incorporates the best of the two models: Stochastic Gradient Descent with Momentum and RMSProp. It introduces a moment term to the update step for faster convergence and divides the learning rate by the square root of sum of squares of the gradients to keep the training stable. AdamW is an improved version of the Adam optimizer, incorporating weight decay directly into the optimization process, which helps to prevent overfitting.

According to Loshchilov and Hutter [14], L_2 regularization in Adam is typically implemented with the following modification:

$$g_t = \nabla f(\theta_t) + w_t \theta_t \tag{1}$$

where w_t is the rate of the weight decay at time t. AdamW, on the other hand, adjusts the weight decay term to appear in the gradient update:

$$\theta_{t+1,i} = \theta_{t,i} - \eta \left(\frac{1}{\sqrt{\hat{v}_t} + \epsilon} \cdot \hat{m}_t + w_t \theta_{t,i} \right), \forall t \tag{2}$$

where:

- θ_t represents the parameters at step t,
- η is the learning rate,
- $\hat{m}_t$ is the bias-corrected first moment estimate (mean of gradients),
- $\hat{v}_t$ is the bias-corrected second moment estimate (uncentered variance of gradients),
- ϵ is a small constant to avoid division by zero

The learning rate is adapted similarly to RMSProp, where it is divided by the square root of the second moment estimate, providing adaptive learning rates that stabilize training.

The combination of Cross-Entropy Loss and the AdamW optimizer is used frequently with fine tuning transformer based models, as studies have shown its effectiveness in achieving quick convergence and showing strong generalization performance [14].

D. Network Architectures

The choice of network architecture is also another important factor when it comes to the performance of a model on NLP tasks. Transformer-based models have revolutionized the field with their ability to handle long dependencies in text and capture contextual information. Unlike traditional RNNs and LSTMs, transformers make use of self-attention mechanisms that allow them to weigh the importance of different words in a sentence regardless of their positions, leading to more accurate and relevant predictions. This type of innovation enables transformers to shine in various sequential tasks [18]. Below, we present the common settings shared among all selected architectures, followed by a table summarizing the total parameters for each model:

TABLE II
COMMON VALUES FOR NETWORK ARCHITECTURES

Parameter	Value
Layers	12
Hidden Size	768
Attention Heads	12

TABLE III
NUMBER OF PARAMETERS FOR EACH MODEL

Model	Total Parameters
BERT	110 million
ALBERT	12 million
RoBERTa-base	125 million
RoBERTa-large	355 million

BERT uses a stack of 12 transformer layers with multi-head self-attention mechanisms and feed-forward neural networks. Each layer processes the input sequence, allowing bidirectional context understanding.

Fig. 2. Architecture description of BERT

ALBERT reduces the number of parameters significantly through techniques like cross-layer parameter sharing and factorized embedding parameterization, while keeping the performance relatively similar, by sharing weights across layers and factorizing embedding parameters.

Fig. 3. Architecture description of ALBERT

E. Hyperparameters and Configuration

The choice of hyperparameters is another valuable topic we need to explore to further optimize the performance of our models. Here, we discuss the hyperparameters used and the intuiton behind our selection.

1) **Learning Rate:** We set the learning rate to 2e-5. This value is commonly used for fine-tuning transformer models and for our model, it strikes a balance between speed and stability.
2) **Batch Size:** We use a per-device batch size of 16. This size is within the generally used range and we were still able to fir the model into our GPU memory.
3) **Number of Epochs:** The models are trained for 3 epochs. Fine-tuning for a small number of epochs is generally sufficient for fine tuning, however we observed that training loss went down significantly in-between epochs and validation loss was mostly stable.

RoBERTa's architecture is similar to BERT's but optimized for training efficiency. It uses dynamic masking, larger batch sizes, and more data.

Fig. 4. Architecture description of RoBERTa

4) **Evaluation Strategy:** The evaluation is set to happen at the end of each epoch. This strategy helps to monitor the loss on the validation set regularly and enables us stop early if the model starts to overfit.
5) **Weight Decay:** We apply a weight decay of 0.01 as a regularization in order to prevent overfitting. Using weight decay adds a penalty to the loss function based on weight magnitudes, encouraging the model to go for smaller weights which can lead to better generalization.
6) **Logging:** Logging is done every 500 steps so that the training progress can be watched and issues can be diagnosed early on.
7) **Best Model Metric:** The metric preferred for selecting the best model is the evaluation loss. This ensures that the model setup with the lowest validation loss is chosen.
8) **FP16 Precision:** Training is conducted with mixed precision (FP16) to increase the computational speed and reduce memory usage without affecting model performance.
9) **Warmup Steps:** A warmup ratio of 0.1 is chosen, which means that 10% of total training steps are used for warmup. This improves the level of stabilization in the training by gradually increasing the learning rate at the beginning.

F. Fine-Tuning Process

The idea behind the fine tuning pipeline we built is basically to modify pretrained models like BERT, RoBERTa, or ALBERT to our specific question answering task using the SQuAD datasets. We start by tokenizing the data, ensuring the contexts and questions are formatted correctly for the model to handle. During training, the model uses the hyperparameters we defined to learn the mappings from tokenized inputs to answer spans within the context, more intuitively, where in the context the answer to the question falls into. Gradient descent optimization iteratively adjusts the model's weights to minimize the loss function, which measures the difference between the predicted and actual answers. Postprocessing transforms the model outputs into readable text by extracting answer spans from tokenized contexts. As the final step, we evaluate the performance using the official evaluation script calculating metrics such as exact match and F1 score. Consequently, by comparing the performance of BERT, RoBERTa, and ALBERT on the SQuAD datasets, we can determine the most suitable model for deployment in real-world applications.

G. Understanding Fine-Tuning

Fine tuning involves taking a pre-trained model, which has learned general language features, and adapting it to a specific task. Pretrained models capture patterns from large datasets, understanding both low level features like words and syntax and high level concepts like sentence structure and meaning.

In fine tuning, we leverage this existing knowledge through the concept of transfer learning. Instead of training from scratch, we further train the pretrained model on our specific dataset, making slight adjustments to its parameters. This process tailors the model's general understanding to our specific task, enhancing performance efficiently and effectively.

VI. EVALUATION

This section discusses the model evaluation metrics and the process of choosing the right model for our task. First, we show how to calculate F1, exact match metrics, and additional metrics used for a better comparison of models. Then, we compare the models by using these metrics. Finally, we indicate the selected model based on our comparison.

A. F1 Score

The F1 score provides a balance between precision and recall. For question-answering models such as our task, precision is determined as the fraction of the words in the prediction that emerges in the ground truth while recall is determined as the fraction of words in the ground truth that emerges in the prediction [15].

$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3}$$

To obtain a deeper insight into our models, the official evaluation script for this task is executed and different versions of F1 scores are calculated as HasAns F1 and NoAns F1. HasAns F1 measures the score for answerable questions and NoAns F1 score measures the score for answerable questions. The exact formulas for these metrics can be seen below:

$$\text{HasAns F1} = \frac{\sum \text{F1 scores for answerable questions}}{\text{Total number of answerable questions}} \tag{4}$$

$$\text{NoAns F1} = \frac{\sum \text{F1 scores for unanswerable questions}}{\text{Total number of unanswerable questions}} \tag{5}$$

B. Exact Match Score

Exact Match (EM) score measures the percentage of predictions that exactly match the ground truth [15]. This metric helps us to understand how precise our model's answers are without any deviation.

$$\text{EM} = \frac{\text{Number of exact matches}}{\text{Total number of questions}} \tag{6}$$

HasAns and NoAns versions of this metric have also been calculated. While HasAns Exact score measures for answerable questions, NoAns Exact score measures for unanswerable questions.

$$\text{HasAns Exact} = \frac{\sum \text{EM scores for answerable questions}}{\text{Total number of answerable questions}} \tag{7}$$

$$\text{NoAns Exact} = \frac{\sum \text{EM scores for unanswerable questions}}{\text{Total number of unanswerable questions}} \tag{8}$$

C. BLEU Score

BLEU (Bilingual Evaluation Understudy) is the measurement of the correlation between n-grams of the prediction and the ground truth. It is initially planned for machine learning tasks, however it can also be operated for question-answering tasks. To counter favoring extremely short outputs, brevity penalty is calculated. Here is the formula for the Brevity Penalty (BP) and BLEU score [16]:

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ \exp(1 - \frac{r}{c}) & \text{if } c \leq r \end{cases} \tag{9}$$

Where:

c is the length of prediction.
r is the length of ground truth.

$$\text{BLEU} = \text{BP} \times \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{10}$$

Where:

BP is the brevity penalty
p_n is the precision of n-grams
w_n is the weight assigned to n-gram precision

D. ROUGE Score

ROUGE (Recall Oriented Understudy for Gisting Evaluation) compares the predictions of our model with the ground truth by calculating the overlap between them. The formula for ROUGE-N score can be seen below [17]:

$$\text{ROUGE-N} = \frac{\sum_{\text{gram}_n \in \text{Reference Summaries}} \text{Count}_{\text{match}}(\text{gram}_n)}{\sum_{\text{gram}_n \in \text{Reference Summaries}} \text{Count}(\text{gram}_n)} \tag{11}$$

Where:

n$_n$ represents the length of the n-gram.
Count is the maximum number of n-grams overlapping.

For this task, we focused on three metrics from this set of scores. ROUGE - 1 score analyzes the overlap of unigrams which means single words, ROUGE - 2 score focuses on the overlap of bigrams which means two consecutive words and lastly ROUGE - L score calculates longest common subsequence (LCS) which overlaps between the prediction and the ground truth [17].

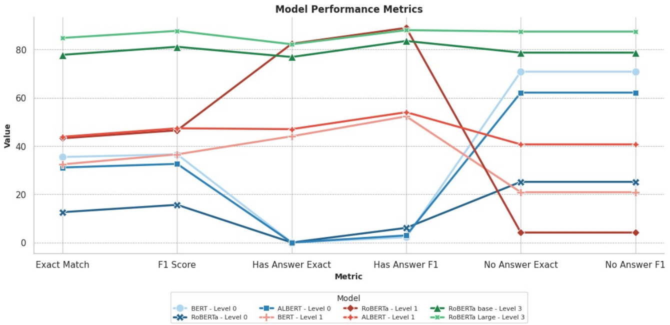

Fig. 5. Overview of Metrics for all Models

E. Comparison of Models

First, here is an overview for all the metrics calculated for all the models that are focused on for this task. As it can be seen, the metrics for the same levels emerge relatively close values compared to the metrics for the same type of models.

For closer examination, all the metrics will be compared for comparison and selection of the best one.

Level 0: First, let's look at the metrics for Level 0:

TABLE IV
F1 AND EM SCORES FOR LEVEL 0

Models	BERT	RoBERTa	ALBERT
F1	36.55	15.63	32.59
EM	35.45	12.60	31.10
HasAns F1	2.20	6.09	2.97
HasAns EM	0.00	0.01	0.00
NoAns F1	70.80	25.15	62.12
NoAns EM	70.80	25.15	62.12

It can be said that all of the models are performing better for NoAns metrics. However, the cause of this may be that there are many blank values in the models' predictions. The HasAns metrics are very low that approves our inference. Overall, the models' performances definitely aren't enough for deployment and there is so many room for improvement.

TABLE V
BLUE AND ROUGE SCORES FOR LEVEL 0

Models	BERT	RoBERTa	ALBERT
BLUE	0.03	0.04	0.03
ROUGE - 1	0.06	0.07	0.07
ROUGE - 2	0.03	0.04	0.03
ROUGE - L	0.06	0.06	0.07

When looking at the additional metrics, the values are really similar and low which is another indicator that we should improve the models.

Level 1: Second, the metrics for level 1 will be compared:

TABLE VI

F1 and EM Scores for Level 1

Models	BERT	RoBERTa	ALBERT
F1	36.54	46.48	47.33
EM	32.42	43.20	43.82
HasAns F1	52.29	88.94	54.00
HasAns EM	44.03	82.35	46.98
NoAns F1	20.84	4.15	40.67
NoAns EM	20.84	4.15	40.67

When comparing the values for level 1, it can be seen that RoBERTa significantly performs better for HasAns values. Since the metrics that we give weight to our HasAns metrics for level 1, proceeding with RoBERTa is the most logical option.

TABLE VII

BLUE and ROUGE Scores for Level 1

Models	BERT	RoBERTa	ALBERT
BLUE	0.16	0.74	0.23
ROUGE - 1	0.57	0.82	0.70
ROUGE - 2	0.33	0.51	0.43
ROUGE - L	0.57	0.82	0.70

After looking at additional metrics, it is more clear that RoBERTa outperforms other models in Level 1 and should be chosen as the best model for the development of the task.

Level 3: Last but not least, let's compare the metric values for level 3:

TABLE VIII

F1 and EM Scores for Level 3

Models	RoBERTa	RoBERTa Large
F1	81.09	87.70
EM	77.78	84.78
HasAns F1	83.50	88.00
HasAns EM	76.87	82.15
NoAns F1	78.69	87.40
NoAns EM	78.69	87.40

As can be seen from the values above, RoBERTa Large outperforms RoBERTa for every metric that is calculated. Also, it can be said that even though the RoBERTa Large model can be improved further, these metrics are promising for the purpose of the task.

TABLE IX

BLUE and ROUGE Scores for Level 3

Models	RoBERTa	RoBERTa Large
BLUE	0.64	0.68
ROUGE - 1	0.83	0.85
ROUGE - 2	0.52	0.54
ROUGE - L	0.83	0.85

After analyzing additional metrics, it is clear that RoBERTa Large's performance is higher when compared to RoBERTa.

F. Model Selection

After calculating four metrics and comparing our models at every level in a very comprehensive dimension, it can be stated that RoBERTa Large model's performance are better than other models and should be chosen as the final model. After the selection of the model, we can proceed with deploying our model.

VII. Deployment

The deployment is the final step in the CRISP-DM process. Even though the objective of this project is to optimize a question-answering model for knowledge extraction, in this section, the possible deployment options of the model will be briefly explained. Before a deep dive of the options, it is important to recall the current challenges E.ON has faced. Currently, E.ON partners with Microsoft Azure, incorporating Azure OpenAI models into its platform (E.ON GPT) and a family of custom solutions. The current situation presents several challenges: the capacity of available commercial offerings is limited; obtaining the desired bandwidth (tokens per minute) for an acceptable user experience is not always feasible; the evaluated solutions are costly, and the expenses escalate with the number of users; adapting models is challenging and fine-tuning becomes restricted or even impossible; and there is no option to host a specialized model based on a foreign base model.

During data processing and modeling phases, Google Colab is preferred as a platform because it provides free access to powerful GPUs and TPUs, which significantly speeds up the training process for large models like LLMs used in level 1 and level 3. So, users can easily train models without the need for expensive hardware investments. Moreover, Google Colab notebooks can easily be shared, which makes it useful for collaborative projects [19]. However, one of the most important and useful features of Google Colab is its easy integration with Hugging Face. It allows models to be loaded directly into Colab notebooks and make it easier to work on custom datasets and fine-tuning models. When it comes to model deployment, users can effortlessly push their fine-tuned models to the Hugging Face Model Repository, enabling them to host their models for deployment and additional fine-tuning [20]. After the evaluation phase is completed, the Roberta Large Fine-Tuned Model is selected and uploaded to the Hugging Face Model Repository for deployment.

It is important to understand the capabilities of the Hugging Face Platform in terms of pre-trained model deployment. Hugging Face provides the Transformers library, which comes with highly informative documentation. It is mostly useful for NLP projects since it provides functions for pre- and post-processing of text data. It is compatible with different frameworks, such as PyTorch, TensorFlow, and JAX. Therefore, users can choose the framework that best fits their needs or even switch between them if required [21]. Additionally, pipelines in this library make it easier to load models and predict outcomes using just a few lines of code. The pipeline requires a model, tokenizer, and task definition from users as input [22]. Hugging Face Inference Endpoints make it easy to load the model from pre-trained models in the repository without needing to manage the underlying infrastructure [23]. As a proof of concept, the Roberta Large Fine-tuned model is loaded from Hugging Face to Google Colab for predictions. The user can interact with the model in a basic way that it is possible to ask custom questions with context. Additionally, Hugging Face Inference Endpoints integrate with external services to facilitate model deployment. Users can use a custom container image hosted on an external service, such as Docker Hub, AWS ECR, Azure ACR, or Google GCR. It ensures consistent performance and reliability by automatically adjusting resources based on traffic which known as autoscalling. [23] Autoscaling enables the flexible adjustment of the number of endpoint replicas running models, depending on the volume of traffic and accelerator use. By implementing autoscaling, it is possible to reduce costs and effectively handle fluctuating workloads without compromising on high availability. Furthermore, it is feasible to tailor the scaling thresholds, behavior, and metrics according to your precise specifications. Users also need to consider model initialization time to make sure its effectiveness [25].

The deployment's simple framework is illustrated in Fig. 6:

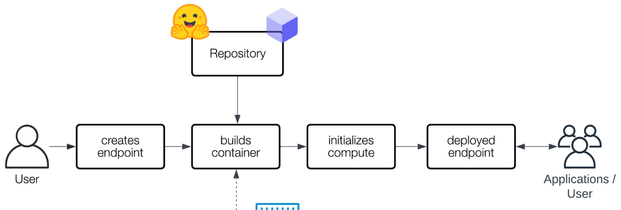

Fig. 6. HF Inference Endpoints Creation Flow, adapted from [23]

Evaluating specific requirements, existing infrastructure, and cost management strategies will help determine the best platform for deploying fine-tuned models for customized tasks. Also, Hugging Face provides detailed documentation of integration, configuration, and price for each platform (see Appendix I for detailed pricing). Last but not least, Hugging Face Endpoints provides different security levels, including public, protected, and private endpoints (see Appendix II for detailed structure).

Before deploying the model, users should carefully determine the purpose for which they will use the model. Since the intended use-case is the main factor in determining the requirements of the deployment architecture. For example, the time range for response time may be longer for some use-cases, while for others it may be quite significant and affect customer satisfaction. Therefore, it is very important to define Key Performance Indicators (KPIs) in advance. Finally, having a robust system that allows continuously monitor the system and make additional adjustments to the model as needed should be the main consideration.

VIII. Business Implications and Techniques

Question-Answering (QA) models can improve a wide range of applications in organizations. So far, QA has been explained as the process of providing context and questions to LLMs as input and getting answers. It is known as open QA. However, considering the user's experience, it would be more convenient if the user directly asks questions to the QA model without providing context, with the system retrieving the context in the background. It is known as closed QA [24]. For closed-QA systems, Retrieval-Augmented Generation (RAG) is a common technique used in the industry [26].

A. RAG

RAG basically consists of two steps: receiver and generator. The receiver searches documents to find relevant context based on the user's query. It uses advanced retrieval techniques to ensure that the most relevant information is selected from the vast amount of data. It can be performed in different ways like sparse vector search, dense embedding, or full-text search. The generator is basicly the combination of text-to-text LLM and prompt engineering. It takes the input and generates contextually relevant answer [27].

The basic framework of RAG is shown in Fig. 7.

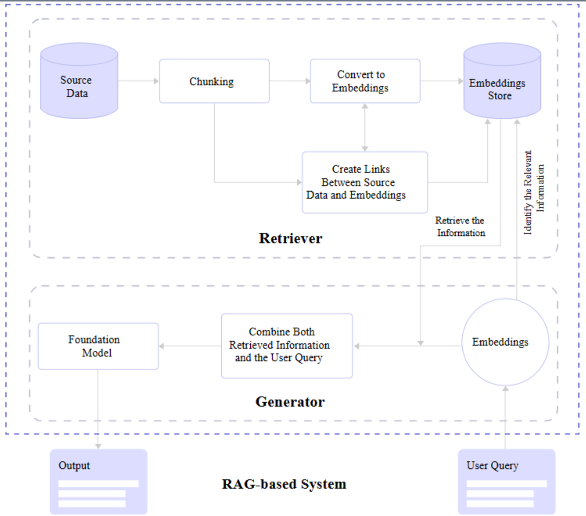

Fig. 7. Rag Architecture. Adapted from [28]

QA systems can differ in the way answers are generated. With extractive QA, the model extracts the answer from a context and provides it directly to the user. In the model training phase, the models are optimized for extractive QA by training on question-answer pairs. On the other hand, generative QA is based on generating free text directly from context. In RAG, generator models are an example of generative QA [24]. Fine-tuning and Retrieval-Augmented Generation (RAG) are different but complementary techniques that improve the user experience and operational efficiency of large language models (LLMs) used in QA. Each has different goals, mechanisms, advantages, and disadvantages, but they can work together to provide significant benefits. While training RAG models is beyond the scope of this project, as a proof of concept, the sample code demonstrates how a fine-tuned extractive QA model can be integrated with RAG.

B. Use Cases

QA systems can change the nature of business intelligence by allowing users to ask questions in natural language and get immediate and relevant answers. This reduces the need for technical knowledge. Users can get information from databases without the technical knowledge of complex queries like SQL. For example, an executive might ask, "Why did sales decline in Q3 2020?" and get insights that combine data trends and contextual analysis, such as "Sales declined due to lower marketing spend and increased competition" [29].

QA models can improve customer support by providing fast and accurate answers to issues raised by customers [24]. These models can be integrated into chatbots or virtual assistants to answer questions. So, it reduces manpower requirements. For example, a customer support chatbot can answer questions like, "How do I reset my password?" or "What is the status of my order?". Model can search documentation from the company's knowledge base and return relevant information.

Within organizations, QA systems can manage and retrieve information from large internal databases [24]. For example, they can help employees find answers to questions about company policies, procedures, or technical documentation. This practice increases efficiency by reducing the time spent searching for information. An example might be an employee asking, "What is the process for requesting leave?" and receiving a step-by-step guide from the HR database.

QA models offer significant benefits by enabling natural language querying and improving access to information. Their integration with business intelligence, customer support, and internal knowledge management systems demonstrates their potential to drive efficiency and innovation in many areas.

IX. CONCLUSIONS

In this project, we believe that we managed to successfully meet the main objectives of evaluating open source language models as viable alternatives to commercial solutions for E.ON GPT. We demonstrated that BERT, RoBERTa, and ALBERT models can achieve high performance in question-answering tasks by fine-tuning SQuAD datasets. Especially our final model, Roberta Large Fine-Tuned, provides accurate and efficient answers to both has-answer and no-answer questions. This shows that open source models can replace commercial solutions, offering greater flexibility, cost-effectiveness, and data privacy.

On the other hand, the limitations of the project need to be considered in order to provide a sustainable solution. Fine-tuning large models requires significant computational power, which not all organizations have access to. Based on E.ON's current use of Open AI solutions, we can assume that it has access to infrastructure with these resources. However, even after fine-tuning the models on SQuAD datasets, E.ON may require additional field-specific data to further optimize performance for its internal processes. This could potentially expose E.ON's data to security vulnerabilities. Although open source models provide better control over the data, pre-processing data and fine-tuning models raise privacy risks. Furthermore, handling real-time queries and scaling the model to a large number of users can be challenging and may require more optimization than available solutions. In addition, E.ON, a German-based company, likely uses German texts in its data, whereas we trained our model on an English data set.

There are some possible next steps and improvements that E.ON can apply to address the limitations mentioned above. To implement our solution and any other projects E.ON may have, investing in more powerful GPUs or TPUs can be a beneficial option because it provides faster and more efficient computational power. Additionally, collecting and integrating more domain-specific datasets will improve the model's accuracy and relevance in specific business contexts, also German data can be considered. However, E.ON should carefully consider data confidentiality and develop security measures to protect sensitive information during data pre-processing and training. Finally, E.ON can establish a continuous monitoring system to regularly update and monitor the performance of deployed models, ensuring their accuracy and relevance. Focus should be on optimizing models for real-time query processing and scalability.

Our proposed approach to testing and validating the business objectives starts with defining KPIs that align with business goals. We would use these KPIs to compare the performance of open-source models with the existing commercial solution in terms of accuracy, response time, user satisfaction, and cost savings. Before deploying the model or switching between alternatives, a detailed cost analysis including initial setup costs, ongoing maintenance, and scalability expenses is required. On the other hand, user satisfaction will be placed after deployment. We can perform A/B testing to compare open source models with commercial solutions in real-world scenarios. We can conduct user testing sessions with E.ON employees to gather feedback on the performance and usability of new models. Surveys and interviews can help measure user satisfaction. We can also quantify metrics such as F1 score, precision, recall, and average response time.Through continuous monitoring and performance evaluation, E.ON can ensure the accuracy and relevance of these models, ultimately can have reliable knowledge extraction and management from analogue and digital documents for natural language queries for data insights, improving customer support with fast and accurate responses, and enhancing internal knowledge management, or any possible applications.

X. ACKNOWLEDGMENT

We would like to thank E.ON for the opportunity to collaborate on this project. This research provided us with valuable knowledge in both the theoretical foundations and practical applications of Natural Language Processing (NLP) and question answering. Their collaboration is essential for the practical implementation of this project. Also, we would also like to express our sincere gratitude to the Data Analytics in Applications course and its organizers, Benedikt Blümelhuber, Sebastian Junker, and Niklas Stepanek from the Research Institute for Management, Logistics, and Production at the TUM School of Management.

APPENDIX

APPENDIX I
PRICING

When creating an Endpoint, the type of instance to deploy can be selected and the model can be scaled based on an hourly rate. At the end of the subscription period, the user or organization account will be charged for the compute resources used to launch and run successfully deployed Endpoints [30].

The hourly pricing for all available instances for Inference Endpoints and examples of how costs are calculated are listed below.

TABLE X

CPU INSTANCES [30]

Provider	Instance Type	Instance Size	Hourly Rate	vCPUs	Memory	Architecture
aws	intel-icl	x1	$0.032	1	2 GB	Intel Ice Lake
aws	intel-icl	x2	$0.064	2	4 GB	Intel Ice Lake
aws	intel-icl	x4	$0.128	4	8 GB	Intel Ice Lake
aws	intel-icl	x8	$0.256	8	16 GB	Intel Ice Lake
aws	intel-spr	x1	$0.033	1	2 GB	Intel Sapphire Rapids
aws	intel-spr	x2	$0.067	2	4 GB	Intel Sapphire Rapids
aws	intel-spr	x4	$0.134	4	8 GB	Intel Sapphire Rapids
aws	intel-spr	x8	$0.268	8	16 GB	Intel Sapphire Rapids
azure	intel-xeon	x1	$0.060	1	2 GB	Intel Xeon
azure	intel-xeon	x2	$0.120	2	4 GB	Intel Xeon
azure	intel-xeon	x4	$0.240	4	8 GB	Intel Xeon
azure	intel-xeon	x8	$0.480	8	16 GB	Intel Xeon
gcp	intel-spr	x1	$0.070	1	2 GB	Intel Sapphire Rapids
gcp	intel-spr	x2	$0.140	2	4 GB	Intel Sapphire Rapids
gcp	intel-spr	x4	$0.280	4	8 GB	Intel Sapphire Rapids
gcp	intel-spr	x8	$0.560	8	16 GB	Intel Sapphire Rapids

TABLE XI

GPU INSTANCES [30]

Provider	Instance Type	Instance Size	Hourly Rate	GPUs	Memory	Architecture
aws	nvidia-a10g	x1	$1	1	24 GB	NVIDIA A10G
aws	nvidia-t4	x1	$0.5	1	14 GB	NVIDIA T4
aws	nvidia-t4	x4	$3	4	56 GB	NVIDIA T4
aws	nvidia-l4	x1	$0.8	1	24 GB	NVIDIA L4
aws	nvidia-l4	x4	$3.8	4	96 GB	NVIDIA L4
aws	nvidia-a100	x1	$4	1	80 GB	NVIDIA A100
aws	nvidia-a10g	x4	$5	4	96 GB	NVIDIA A10G
aws	nvidia-a100	x2	$8	2	160 GB	NVIDIA A100
aws	nvidia-a100	x4	$16	4	320 GB	NVIDIA A100
aws	nvidia-a100	x8	$32	8	640 GB	NVIDIA A100
gcp	nvidia-t4	x1	$0.5	1	16 GB	NVIDIA T4
gcp	nvidia-l4	x1	$1	1	24 GB	NVIDIA L4
gcp	nvidia-l4	x4	$5	4	96 GB	NVIDIA L4
gcp	nvidia-a100	x1	$6	1	80 GB	NVIDIA A100
gcp	nvidia-a100	x2	$12	2	160 GB	NVIDIA A100
gcp	nvidia-a100	x4	$24	4	320 GB	NVIDIA A100
gcp	nvidia-a100	x8	$48	8	640 GB	NVIDIA A100
gcp	nvidia-h100	x1	$12.5	1	80 GB	NVIDIA H100
gcp	nvidia-h100	x2	$25	2	160 GB	NVIDIA H100
gcp	nvidia-h100	x4	$50	4	320 GB	NVIDIA H100
gcp	nvidia-h100	x8	$100	8	640 GB	NVIDIA H100

TABLE XII

ACCELERATOR INSTANCES [30]

Provider	Instance Type	Instance Size	Hourly Rate	Accelerators	Accelerator Memory	RAM	Architecture
aws	inf2	x1	$0.75	1	32 GB	14.5 GB	AWS Inferentia2
aws	inf2	x12	$12	12	384 GB	760 GB	AWS Inferentia2
gcp	tpu	1x1	$1.38	1	16 GB	44 GB	Google TPU v5e
gcp	tpu	2x2	$5.5	4	64 GB	186 GB	Google TPU v5e
gcp	tpu	2x4	$11	8	128 GB	380 GB	Google TPU v5e

APPENDIX II
SECURITY AND PRIVACY

For further information see:

- Security and Compliance [31]
- Hugging Face Privacy Policy [32]

The basic architecture of privacy levels shown in Fig. 8

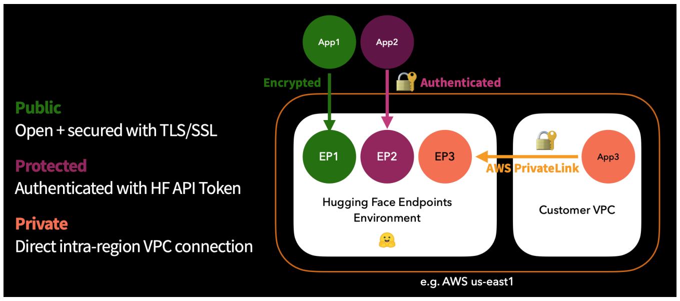
App1
App2
Encrypted
Authenticated
Public
Open + secured with TLS/SSL
Protected
Authenticated with HF API Token
Private
Direct intra-region VPC connection
EP1
EP2
EP3
AWS PrivateLink
App3
Hugging Face Endpoints Environment
Customer VPC
e.g. AWS us-east1

Fig. 8. Architecture of Privacy Levels. Adapted from [31]

REFERENCES

[1] Murgia, M., & Hammond, G. "OpenAI on track to hit $2bn revenue milestone as growth skyrockets", 2023. Available: https://www.ft.com/content/81ac0e78-5b9b-43c2-b135-d11c47480119 (accessed Jul. 31, 24).

[2] Statista. "Outlook for Generative AI Worldwide". Available: https://www.statista.com/outlook/tmo/artificial-intelligence/generative-ai/worldwide (accessed Jul. 31, 24).

[3] E.ON. "Annual Report 2023". Available: https://annualreport.eon.com/en.html

[4] E.ON. "E.ON's Generative Artificial Intelligence Goes Live", 2023. Available: https://www.eon.com/en/about-us/media/press-release/2023/eons-generative-artificial-intelligence-goes-live.html (accessed Jul. 31, 24).

[5] Hugging Face. "Question Answering". https://huggingface.co/tasks/question-answering (accessed Jul. 31, 24).

[6] Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P. "SQuAD: 100,000+ Questions for Machine Comprehension of Text", 2016. Available: https://arxiv.org/abs/1606.05250.

[7] Rajpurkar, P., Jia, R., Liang, P. *Know What You Don't Know: Unanswerable Questions for SQuAD*, 2018. Available: https://arxiv.org/abs/1806.03822.

[8] Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., Wirth, R. (1999). "CRISP-DM 1.0: Step-by-step data mining guide". Available: https://www.the-modeling-agency.com/crisp-dm.pdf (accessed Jul. 31, 24).

[9] OpenAI. "OpenAI API Pricing". Available: https://openai.com/api/pricing/ (accessed Jul. 31, 24).

[10] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," arXiv preprint arXiv:1810.04805, 2018.

[11] S. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuAD Explorer," *Stanford Question Answering Dataset (SQuAD) Explorer*, 2016. [Online]. Available: https://rajpurkar.github.io/SQuAD-explorer/

[12] Y. Liu, 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 preprint arXiv:1907.11692, 2019.

[13] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, “ALBERT: A lite BERT for self-supervised learning of language representations,” arXiv preprint arXiv:1909.11942, 2020.

[14] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” arXiv preprint arXiv:1711.05101, 2019.

[15] Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P. (2016). SQuAD: 100,000+ Questions for Machine Comprehension of Text. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (pp. 2383-2392).

[16] Papineni, K., Roukos, S., Ward, T., 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 (ACL), 311-318.

[17] Lin, C.-Y. (2004). ROUGE: A Package for Automatic Evaluation of Summaries. Text Summarization Branches Out: Proceedings of the ACL-04 Workshop, 74-81.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in Advances in Neural Information Processing Systems, 2017, pp. 5998–6008.

[19] Google Research, "Welcome to Colab," Google Colaboratory. https://colab.research.google.com/ (accessed Jul. 30, 24)

[20] I. S., "Integrating a Hugging Face Model with Google Colab," LinkedIn, May 23, 2024 Available: https://www.linkedin.com/pulse/integrating-hugging-face-model-google-colab-indrajit-swain–btzrc/ (accessed Jul. 30, 2024).

[21] "Transformers," Hugging Face. https://huggingface.co/docs/transformers/en/index (accessed Jul. 30, 2024).

[22] "Pipelines for inference," Hugging Face. https://huggingface.co/docs/transformers/v4.18.0/en/pipeline-tutorial (accessed Jul. 30, 2024).

[23] "Inference Endpoints," Hugging Face. https://huggingface.co/docs/inference-endpoints/index (accessed Jul. 30, 2024).

[24] F. Chiusano, "Two minutes NLP — Quick intro to Question Answering," Medium, Mar. 05, 2022. [Online]. Available: https://medium.com/nlplanet/two-minutes-nlp-quick-intro-to-question-answering-124a0930577c.

[25] “Autoscaling,” Hugging Face. https://huggingface.co/docs/inference-endpoints/autoscaling (accessed Jul. 31, 2024).

[26] K. Martineau, “What is retrieval-augmented generation?,” IBM Research, May 01, 2024. https://research.ibm.com/blog/retrieval-augmented-generation-RAG.

[27] “What is RAG? - Retrieval-Augmented Generation AI Explained - AWS,” Amazon Web Services, Inc. https://aws.amazon.com/what-is/retrieval-augmented-generation/.

[28] “A Simple Guide To Retrieval Augmented Generation Language Models — Smashing Magazine,” Smashing Magazine, Jan. 26, 2024. Avilable: https://www.smashingmagazine.com/2024/01/guide-retrieval-augmented-generation-language-models/.

[29] R. Djiroun, M. A. Guessoum, K. Boukhalfa, and E. H. Benkhelifa, “Natural language why-question answering system in business intelligence context,” Cluster Computing, May 2024, doi: 10.1007/s10586-024-04327-4.

[30] “Pricing,” Hugging Face. https://huggingface.co/docs/inference-endpoints/pricing (accessed Jul. 31, 2024).

[31] “Security & Compliance,” Hugging Face. https://huggingface.co/docs/inference-endpoints/security (accessed Jul. 31, 2024).

[32] “Privacy Policy – Hugging Face,” Hugging Face. https://huggingface.co/privacy (accessed Jul. 31, 2024).