Comparative Analysis of Transformer Architectures for Closed-Book Generative Question Answering Tasks

EESHAN PATEL

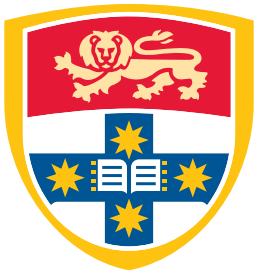
SID: 490517138

Supervisor: Dr. Mostafa Shahin

This thesis is submitted in partial fulfillment of
the requirements for the degree of
Bachelor of Advanced Computing

School of Computer Science
The University of Sydney
Australia

20 June 2024



THE UNIVERSITY OF SYDNEY

Student Plagiarism: Compliance Statement

I certify that:

I have read and understood the University of Sydney Student Plagiarism: Coursework Policy and Procedure;

I understand that failure to comply with the Student Plagiarism: Coursework Policy and Procedure can lead to the University commencing proceedings against me for potential student misconduct under Chapter 8 of the University of Sydney By-Law 1999 (as amended);

This Work is substantially my own, and to the extent that any part of this Work is not my own I have indicated that it is not my own by Acknowledging the Source of that part or those parts of the Work.

Name: Eeshan Patel

Signature: 

Date: June 20, 2024

Abstract

Question-answering (QA) tasks have been a significant area of research in artificial intelligence for the last two decades, yet closed-book question-answering tasks have received comparatively limited interest. This study explores various transformer architectures and assesses their efficacy for closed-book generative QA tasks, mainly focusing on their applications that could benefit corporations seeking tailored AI solutions. By fine-tuning two models, FLAN-T5-small and DistilGPT2, on QA pairs from NVIDIA documentation, the performance of these models was evaluated in a low-resource setting. The evaluation employed robust metrics, including automated tools and human assessments, to ensure a comprehensive model performance analysis.

Our study reveals six key findings. First, current research in QA primarily addresses extractive question answering, highlighting a gap in the study of generative approaches. Second, encoder-only models are found to be unsuitable for closed-book generative QA tasks, as they lack text generation capabilities. Third, encoder-decoder models perform better than decoder-only models in generating accurate and contextually relevant answers. Fourth, the enlargement of training datasets significantly enhances the efficacy of closed-book generative QA tasks. Fifth, the customization and refinement of training data using large language models prove beneficial for improving model training outcomes. Sixth, there is an urgent need for new evaluation metrics that accurately assess the correctness of answers, as current metrics fall short.

KEYWORDS: Question-answering (QA), closed-book QA, extractive QA, transformers, encoder-decoder, decoder-only

Acknowledgements

First and foremost, I would like to express my heartfelt gratitude to my supervisor, Dr. Mostafa Shahin. His constant support and guidance has been pivotal throughout my research journey. Dr. Shahin's wisdom and patience not only steered my academic path but also inspired me to challenge my own boundaries and think more creatively. His willingness to share his vast knowledge and thoughtful advice during critical moments has left a huge impact on my personal and professional development.

I am also incredibly thankful to Dr. Basem Suleiman for sparking my interest in the field of Artificial Intelligence and for his encouragement at the very start of this journey. His enthusiasm and support has been fundamental in steering me toward this exciting field.

I am immensely grateful to my family and friends for their unwavering love and encouragement. Their constant belief in my abilities and their emotional support have been my source of strength and resilience. The joyful breaks, caring messages, and thoughtful conversations have made this academic journey not only bearable but truly enjoyable and fulfilling.

Lastly, I owe a huge thank you to the University of Sydney for providing me with the opportunity to pursue my passions in such a vibrant and supportive academic environment. This experience has been transformative, broadening my horizons and allowing me to engage with a community of like-minded individuals and scholars. I am proud to be a part of this community, and I cherish the doors that have been opened for me here.

CONTENTS

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

In the early 1920s, Albert Einstein opened the door to the world of quantum physics so that an esoteric band of theoretical physicists could peer through and alter how humanity understood space and time (Bird, 2021). An American playwright, Eugene O'Neil, once famously said, "There is no present or future - only the past, happening over and over again - now." (Lish, 2023). Fast forward to the present day, a similar paradigm shift can be observed in the field of Question Answering (QA), an AI discipline revolving around information retrieval and natural language processing, which focuses on building systems that automatically answer questions posed by humans in a natural language (Satapathy, 2018). David Ferruci, a notable pioneer of modern QA systems, developed the IBM Watson System that not only defeated the human champions on a quiz show but also managed to unleash the potential of AI to the world in answering complex questions across diverse questions (Ferrucci et al., 2010). QA systems can potentially revolutionise several industries exponentially as they augment information accessibility, ameliorate efficiency and support decision-making processes. In today's digital age, QA systems are a scientific marvel that can transform customer service and support. These systems can automate responses to frequently asked questions, reducing response time and physical manpower drastically (Pimpalkar, 2023).

Closed-book question answering, a type of QA system, requires a model to directly answer a question without access to any external knowledge (Su et al., 2022). Able to function without external sources, closed-book question answering not only reduces the risk of data breaches drastically but also aids in providing instant responses to the customers, an essential feature of time-sensitive applications. Additionally, it can function in remote regions as it does not require internet connectivity. Transformer models, advanced architectures explicitly designed to handle such tasks, leverage deep learning techniques to understand and generate natural language based on their training data. Existing research delineates several critical challenges that transformer models confront in generating accurate answers in closed-book QA settings. A primary challenge is the limitation related to context: these models operate without the

aid of external documents, relying solely on internally embedded knowledge, which may lead to incomplete or incorrect answers, especially for specialised subjects (Brown et al., 2020). Another significant challenge is the knowledge cutoff; transformer models are trained on current datasets up to a particular date. Hence, they cannot provide information that has emerged since their last training session (Cheng et al., 2024).

Thus, this study compares the performance of different transformer models on question-answering tasks, categorised into Extractive QA, Open Generative QA, and Closed Generative QA (Wolf et al., 2019). While Extractive QA and Open Generative QA are beneficial, they depend heavily on large-scale, high-quality annotated datasets, which are often not readily available (Brown et al., 2020). Therefore, our research pivots to Closed Generative QA, where models generate answers without any external contexts, allowing us to assess the generative capabilities of the transformer models. Two different transformer architectures were fine-tuned on QA pairs using Supervised Learning (Acharya, 2023). This equips the models to generate precise and contextually relevant responses based on the training data provided without access to external databases. Specifically, encoder-decoder and decoder-only models are evaluated for their generative capabilities, with encoder-only models excluded due to their inherent constraints in generative tasks.

This study fine-tunes initial versions of FLAN-T5, an encoder-decoder model, and DistilGPT2, a decoder-only model, on NVIDIA Documentation Question and Answer pairs within a resource-constrained environment (Deepesh et al., 2023; Raffel et al., 2019; Wolf et al., 2019). This dataset is specifically selected for its domain-specific nature, allowing for rigorous testing of the model's capacity to acquire and apply new knowledge. Importantly, none of the models were pre-trained on this data, ensuring a fair comparison of their learning capabilities.

This research aims to identify the most effective transformer architecture for corporate use, enabling companies to choose the ideal model for crafting AI tailored to their company data. To ensure the integrity and reliability of our findings, robust evaluation metrics are employed, enhancing the precision of our comparative analysis and supporting the selection of an optimal model for the corporate world seeking to implement custom AI systems capable of answering company-specific queries (Chen et al., 2019).

1.1 Problem Definition

In recent times, AI has made rapid inroads in its sub-field of natural language processing. Despite that, traditional question-answering (QA) systems are severely restricted by their overreliance on highly annotated datasets and specific contextual information. This dependency often limits their functionality in environments where such resources are either sparse or unavailable, leading to a growing demand for research that focuses on enhancing transformer models to learn solely from question-answer pairs and generate accurate responses.

1.2 Aim & Objectives

1.2.1 Aim

The primary aim of this research is to evaluate the performance of different transformer model architectures on closed-book question-answering (QA) tasks where the models are trained solely on new knowledge obtained from question-answer pairs.

1.2.2 Objectives

- To compare the performance of different transformer model architectures and determine the most effective model for domain-specific closed-book QA tasks.
- To investigate the impact of size of fine-tuning data on the performance of transformer models for closed-book QA tasks.
- To propose and validate reliable evaluation metrics that integrate automatic and human-assessed approaches for a comprehensive assessment of the model's performance.

1.3 Research Questions

The following research questions served as a foundation for this research project:

- What are the primary challenges encountered by traditional question-answering (QA) (Extractive QA, Retrieve-and-Read) systems in generating accurate responses?
  The existing literature identified various critical challenges that traditional QA systems - especially systems employing retrieval and extraction methodologies - encounter. It was observed

that these systems predominantly depend on the existence of large-scale and high-quality annotated datasets along with the accessibility of relevant external documents. This dependency constrains their efficacy in scenarios where such resources are scarce or incomplete (Brown et al., 2020). Additionally, traditional QA systems often lack the capability to autonomously generate accurate answers due to their reliance on specific contexts (Dwivedi and Singh, 2013; Soares and Parreiras, 2020). They also face substantial difficulties when handling complex queries that necessitate the integration of information from multiple sources (Yang et al., 2018). Retrieval of irrelevant documents can significantly impede the system's ability to provide correct answers as accuracy of the document retrieval process plays a crucial role to generate accurate responses (Wang et al., 2024).

- What are the potential applications of closed-book QA models in corporate settings?
  Closed-book QA models hold various potential applications especially in corporate settings as it can not only enhance customer support frameworks but also bolster their information retrieval process. Corporations can implement QA systems that are trained exclusively on proprietary documentation to address inquiries concerning their products, services, and internal policies. This strategy not only augments the efficiency and consistency of customer interactions but also alleviates the burden on human support staff by providing prompt and precise information to the users (Olujimi and Ade-Ibijola, 2023).

- What are the strengths and limitations of various model architectures in managing domain-specific closed-book QA tasks?
  Different model architectures exhibit distinct strengths and limitations when deployed in domain-specific closed-book QA tasks. Encoder-decoder models, exemplified by T5 are proficient in understanding and generating contextually pertinent responses, thus demonstrating their versatility across diverse NLP tasks within a singular framework (Roberts et al., 2020). However, these models necessitate substantial computational resources and are prone to overfitting on smaller datasets which can compromise their generalisation capabilities (DatabaseCamp, 2023). Encoder-only models, like BERT, excel in text comprehension tasks such as information retrieval and sentence classification, attributed to their advanced bidirectional context encoding capabilities (Devlin et al., 2019a). Despite their effectiveness, these models are not tailored for text generation and their performance is heavily dependent upon the quality of their

pre-training data, potentially limiting their efficacy in specific domains (Raffel et al., 2019). On the other hand, decoder-only models such as GPT-3, are adept at producing fluent and coherent responses and exhibit robust few-shot learning abilities, enabling them to adapt swiftly to new tasks (Brown et al., 2020). Nonetheless, these models often struggle to maintain long-term context and are susceptible to generating erroneous information due to inherent biases and hallucinations (Marcus and Davis, 2020).

- How does the amount of data influence the performance of closed-book QA models?
  Existing research highlighted larger datasets typically lead to better model performance in neural networks and other machine learning algorithms due to improved generalisation and reduced overfitting (Brownlee, 2020). However, this field remains completely unexplored for closed-book QA tasks. For closed-book QA tasks, the models rely exclusively on the knowledge encoded within their parameters, and thus, the quantity and quality of training data becomes of paramount importance. In this uncharted territory, the effect of scaling data on these models needs further investigation. Increasing the dataset size can potentially provide a broader range of context and examples for the model to learn from, thereby enhancing its ability to generate accurate and contextually relevant responses.

- How can transformer models be effectively evaluated for their performance in closed-book question-answering tasks, and which evaluation metrics are most reliable?
  The evaluation of transformer models for closed-book QA tasks necessitates a combination of both automatic and human-assessed metrics. Common automatic metrics such as BLEU and ROUGE, which measure the n-gram overlap, are frequently used but have been criticised for their inability to fully grasp the semantic quality of responses, thereby reducing their reliability (Hsu et al., 2021; Lin, 2004; Papineni et al., 2002). Human evaluations, assessing criteria such as relevance, coherence, and informativeness, are considered the most reliable method for assessing model effectiveness. However, this approach is resource-intensive and may not be feasible for all studies (van der Lee et al., 2021). Thus, a balanced approach is required that integrates both automatic and human evaluations for a thorough assessment of transformer models in closed-book QA tasks.

1.4 Significance of the Research

Closed-book QA models can revolutionise customer support frameworks and bolster information retrieval frameworks in corporate environments. By conducting a systematic evaluation to identify the most efficient model architecture for domain-specific QA along with recommending robust evaluation metrics, this study provides significant insights that can transform the development of tailored QA systems for various corporate settings. Additionally, the findings of this study will help to guide future research in the field on model-selection and fine-tuning strategies, improving the performance and accuracy of QA systems trained solely on proprietary datasets.

1.5 Overview

Chapter 2 provides a detailed review of the literature on transformer models and QA systems, highlighting various techniques employed within these systems and the key metrics used for their evaluation. Chapter 3 outlines the methodology and implementation process used to fine-tune the models on domain-specific QA pairs to evaluate their performances. Chapter 4 presents the results of the experiments carried out, comparing the performance of different model architectures. Chapter 5 discusses the significance of the findings along with their larger implications. Additionally, it also highlights the contributions as well as the limitations of this project and proposes recommendations for future research in the field. Lastly, Chapter 6 concludes the study.

CHAPTER 2

RELATED WORK

This chapter reviews the current research in the question-answering (QA) domain, discussing various methodologies and technologies employed in this area. It comprises four primary areas: open-domain and extractive QA, closed-book and generative QA, transformer models for text generation, and evaluation metrics. Each section delves into aspects of question-answering, ranging from techniques employed for different QA tasks to advanced transformer models harnessing text generation capabilities. Furthermore, it reviews different metrics used to evaluate these systems, emphasising automated and human-centred methods. This comprehensive overview highlights the latest developments in question-answering research and identifies gaps and opportunities for future exploration in the QA field. The concept graph below provides an overview of this chapter, illustrating the key themes explored in the existing literature.

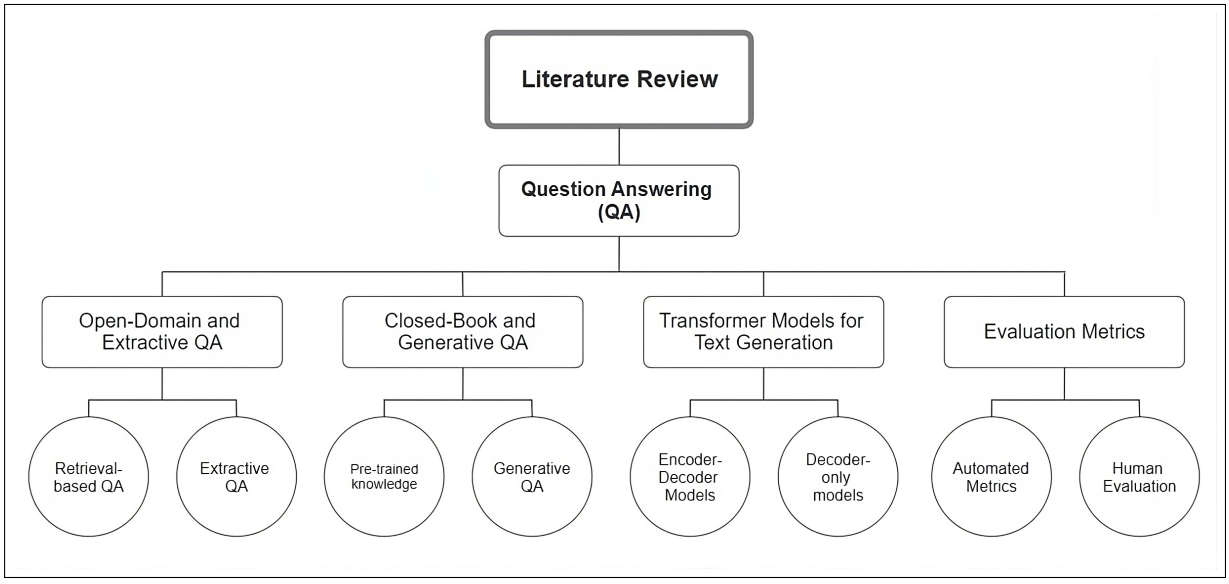


FIGURE 2.1: Concept Graph

2.1 Open-Domain and Extractive QA

Open-domain question-answering (QA) systems handle a wide range of questions without being restricted to a specific domain. These systems typically follow a retrieve-then-read paradigm, where a retriever identifies relevant passages from a large corpus, and a reader processes these passages to extract or generate the answer (Wang et al., 2024). Recent advancements in transformer models such as BERT, RoBERTa, and T5 have significantly enhanced the capabilities of both retrievers and readers (Devlin et al., 2019b; Liu et al., 2019; Raffel et al., 2019). The BERTserini system, developed by (Yang et al., 2019a), exemplifies this progression by integrating BERT with the Anserini information retrieval toolkit to identify answers from a vast corpus of Wikipedia articles efficiently. This approach proves remarkably effective when aggregating and retrieving necessary information scattered across various documents (Hsu et al., 2021).

On the other hand, Extractive QA is focused on identifying the exact span of text from a given context paragraph to answer a specified question (Gao et al., 2023). This approach leverages models like BERT, which excel at understanding and pinpointing the specific segments of text that directly respond to the question (Yang et al., 2019b). Advancements in this field include a study by (Gholami and Noori, 2021), which explored Zero-Shot Open-Book Question Answering within a corpus of Amazon Web Services (AWS) technical documents. The study employed a practical two-step architecture, first using a retriever to identify the correct document, followed by an extractor to pinpoint the answers. This approach demonstrates the real-world application of Extractive QA and its potential for compelling question-answering.

However, a significant challenge in retrieval-based and extractive QA systems is their reliance on large-scale, high-quality annotated datasets (Le et al., 2023). The performance of these models heavily depends on the availability of substantial training data, which is necessary for the fine-tuning processes. Additionally, a notable drawback of these systems is that the answers they provide are often not user-friendly, as they are merely extracted from the contexts in which they are trained. For these QA systems, datasets like SQuAD (Rajpurkar et al., 2016), NewsQA (Trischler et al., 2016), and TriviaQA (Joshi et al., 2017) are commonly used to adapt pre-trained models to specific tasks. These datasets provide QA pairs along with the necessary context to answer the questions, helping models learn to identify relevant text spans and improve their accuracy. Nevertheless, collecting such extensive, high-quality data is time-consuming and resource-intensive, especially for new domains or languages.

Moreover, without ample data, these models cannot be fine-tuned effectively, thereby limiting their capacity to deliver accurate answers (Li et al., 2020). Lastly, these systems face a significant limitation in its inability to generate new information since it is confined to extracting answers from the provided context. If the necessary context is absent in the retrieved documents, the system will fail to produce a correct answer (Wang et al., 2024).

Therefore, while open-domain and extractive QA systems have demonstrated impressive capabilities, they are fundamentally limited by their reliance on the availability and quality of external documents. The need for large-scale, annotated datasets remains a significant barrier, and their inability to generate new information limits their effectiveness in scenarios where the context is incomplete or missing.

Consequently, there is a growing need to explore closed-book generative QA tasks. The models do not rely on extensive labelled data for these tasks and can leverage their generative capabilities to provide answers based on the knowledge encoded within their parameters. Additionally, they can be fine-tuned for new domains, enhancing their adaptability and effectiveness. This approach overcomes the drawbacks of extractive QA and provides a promising path for future research by generating answers even without explicit context.

2.2 Closed-Book and Generative QA

Closed-book question-answering (QA) refers to the task where the model generates answers without access to external documents or context at the time of inference (Ye et al., 2020). Instead, the model relies solely on the knowledge it has internalised during pre-training. Generative QA, particularly in closed-book settings, leverages the generative capabilities of language models to produce answers from their learned representations. Fine-tuning these models on QA pairs further enhances their ability to generate accurate and contextually relevant answers, even in new domains. This fine-tuning process involves training the model on domain-specific QA pairs, allowing it to adapt and internalise the nuances of the new domain, thereby improving its generative responses (Yagnik et al., 2024). However, limited research has been conducted in this area, highlighting a need for further exploration.

The study by (Roberts et al., 2020) explores how pre-trained models like T5 can internalise knowledge from vast text corpora. The findings suggest that these models can perform competitively in QA tasks by leveraging the knowledge encoded during pre-training. This study highlights the potential of

generative models to act as knowledge bases, providing answers based on pre-trained data alone. However, the lack of fine-tuning on QA pairs without additional contextual data limits their adaptability to new domains.

Another similar study evaluating medium-language models in zero-shot settings assessed models like GPT-3, where the models generate answers without additional training on specific QA pairs (Brown et al., 2020). The evaluation revealed that while these models could produce plausible answers, their performance varied significantly based on the complexity and specificity of the questions. Therefore, the study underscores the importance of fine-tuning to improve the reliability and accuracy of generative QA models, especially in closed-book scenarios where context is not available during inference.

One notable study by (Hsu et al., 2021) demonstrated the advantages of incorporating extractive capabilities into generative QA systems. This approach utilises extractive methods to pre-select relevant information from answer candidates, which a generative model then uses to synthesise responses. However, this method relies on the availability of extensive data, as it necessitates the generation of answer candidates to implement this hybrid strategy. This underscores the importance of exploring closed-book generative QA tasks, mainly when only QA pairs are available, to address the limitations associated with data-intensive requirements.

Another study by (Wang et al., 2021) evaluated the effectiveness of generative models fine-tuned on QA pairs. The findings indicated that while the encoder-decoder model BART could generate answers, its performance was suboptimal without contextual information, achieving only 1.5% accuracy in generating correct answers. Conversely, when fine-tuned with context, the models significantly improved their ability to produce accurate and relevant answers. This study suggests that while contextual fine-tuning enhances performance, the dependence on context is problematic due to the limited availability and difficulty in curating annotated data. Therefore, there is a pressing need to investigate further fine-tuning the models that do not rely on contextual data and to assess the generative capabilities of these models in such settings.

In the medical domain, a study explored the application of large language models for medical question-answering systems, comparing the performance of general-purpose models with those specialised in the medical domain (Yagnik et al., 2024). The study found that fine-tuning on domain-specific QA pairs improved accuracy. However, the results highlighted issues with reliability, underscoring the need for more robust training methodologies and evaluation metrics. This study emphasises the need

for effective fine-tuning strategies to fully leverage the generative model's capabilities, particularly in specialised domains.

Thus, a key challenge in closed-book QA is the reliance on pre-trained knowledge, which can be incomplete or outdated. Additionally, the generative nature of these models can lead to hallucinations, where the model generates plausible but incorrect answers. This highlights the need for more effective fine-tuning strategies and robust evaluation metrics that can sufficiently capture the accuracy and reliability of generated answers.

Consequently, further exploration of closed-book generative QA models is crucial. These models do not require extensive labelled data and can leverage their generative capabilities to provide answers based on the knowledge encoded within their parameters. Refining this approach makes it possible to overcome the limitations of extractive QA, enabling the generation of accurate answers even without explicit context. This presents a promising direction for future research.

2.3 Transformer models for Text generation

The generative capabilities of transformer models in QA systems are crucial to their functionality, especially when addressing closed-book generative tasks. Encoder-decoder and decoder-only models predominantly exhibit these capabilities. Encoder-decoder models, such as T5, utilise both components to handle complex tasks that require understanding the input context and generating appropriate output (Raffel et al., 2019). Decoder-only models like GPT-3 excel in text generation by predicting subsequent tokens, making them ideal for tasks requiring creative or extensive content generation (Brown et al., 2020).

While encoder-only models excel in extractive QA, they face significant limitations in generative tasks due to the absence of the decoder component (Raffel et al., 2019; Bandi et al., 2023; Caldarini, 2023). This architectural difference implies that while these models are highly effective at tasks requiring the precise extraction of information from a given context, they lack the ability to construct answers independently without contextual prompts. Researchers have demonstrated the effectiveness of these models in extractive QA through a multi-task learning framework, which enhances model generalisation by leveraging data from multiple domains. This approach has shown significant improvement over single-task baselines, reinforcing the suitability of encoder-only models for tasks that require detailed contextual understanding but not independent answer generation (Su et al., 2019).

Building on previous research, this work will focus on the application of encoder-decoder and decoder-only models for closed-book generative QA. These models offer the generative capabilities required for such tasks and are well-suited for generating coherent and contextually accurate answers without external inputs. Their inherent ability to synthesise information based on learned data makes them especially valuable for exploring new approaches in closed-book QA, where generating informative, accurate, and contextually relevant answers is paramount.

By harnessing the strengths of these advanced model architectures, this approach has the potential to revolutionise our understanding and enhance the efficacy of QA systems in scenarios where direct information is not readily available. This research paves the way for pushing the boundaries of what artificial intelligence can achieve in natural language understanding and response generation from question-answer pairs, instilling a sense of optimism for the future of AI in NLP.

2.4 Evaluation Metrics

The effectiveness of question-answering (QA) systems fundamentally relies on the evaluation metrics chosen to assess their performances. These metrics are essential for assessing how accurately and relevantly a model can answer the question. As highlighted by (Chen et al., 2019) in their comprehensive study, “Evaluating Question Answering Evaluation”, the selection of these metrics not only influences model development but also shapes our understanding of a model’s effectiveness and accuracy. Table 2.1 illustrates the various metrics utilised by different researchers, underscoring the absence of a standardised approach in the field.

Researcher	Count of Metrics Employed	Task	Evaluation Methods Utilised
(Hsu et al., 2021)	4	Hybrid QA (Extractive + Generative QA)	Accuracy, BLEU, Human Evaluation, ROUGE
(Le et al., 2023)	4	Retriever-Then-Read Paradigm QA	Exact Match, F1, Human Evaluation, Recall
(Li et al., 2020)	2	Extractive QA	Exact Match, F1
(Roberts et al., 2020)	2	Closed-book QA (No fine-tuning)	Exact Match, Human Evaluation
(Wang et al., 2021)	3	Closed-book Generative QA	Exact Match, F1, Human Evaluation
(Peinl and Wirth, 2023)	1	Closed-book Generative QA	Human Evaluation
(Su et al., 2019)	2	Extractive QA	Exact Match, F1
(Yagnik et al., 2024)	3	Closed-book Generative QA	BLEU, Human Evaluation, ROUGE

TABLE 2.1: Evaluation Metrics Employed by Various Researchers

This diversity in evaluation metrics, from automated metrics like ROUGE and F1 to more subjective methods such as Human Evaluation, illustrates the challenges in establishing a unified standard for evaluating QA systems. However, a significant issue remains: considerable metrics focus on lexical accuracy, often overlooking the nuanced meanings behind the answers. A study highlights that this approach can mistakenly mark semantically correct answers as wrong because they do not match word-for-word (Risch et al., 2021). This oversight can lead to underestimating a model's ability to process and respond to queries in a contextually relevant manner. The absence of evaluation standards that capture both the lexical precision and the semantic depth of answers complicates comparing the systems effectively. Therefore, advancing the evaluation metrics to include both dimensions would significantly enhance our understanding of each system's capabilities, ensuring a more comprehensive assessment across different scenarios.

2.5 Findings from Existing Literature

The existing literature on question-answering (QA) tasks highlights significant ongoing challenges. Open-domain and extractive QA systems heavily depend on high-quality annotated datasets, which are time-consuming and resource-intensive to compile, particularly for new domains or languages. This

reliance on extensive datasets restricts the effectiveness of these models when adequate data is unavailable. Additionally, their inability to generate new information confines them to merely extracting answers from provided contexts. In contrast, closed-book generative QA, which relies solely on knowledge encoded within model parameters, remains underexplored. By fine-tuning models to encode new knowledge directly from QA pairs, these models have the potential to surpass extractive QA systems in performance.

Additionally, the literature revealed that encoder-only transformer models lack text generative capabilities and the evaluation practices rely predominantly on lexical-based metrics, leading to potential biases and uncertain outcomes due to their failure to capture semantic depth. While human evaluations continue to provide valuable, in-depth assessments of model performance, offering a more comprehensive view of a system's capabilities, they are not always viable due to the extensive time and resources required.

CHAPTER 3

METHODOLOGY & IMPLEMENTATION

The primary objective of this study is to investigate how different transformer models perform on a domain-specific question-answering task, assessing their ability to generate precise and contextually relevant answers. Given the complex structure of this task, the methodology and implementation are integrated into a single cohesive chapter. This integration facilitates a thorough and seamless presentation of the various processes involved in executing the research. This chapter outlines the execution of the research, from the selection and preparation of the dataset to the training and evaluation of transformer models. Figure 3.1, presented below, offers a brief overview of this chapter. The code for this study is available on GitHub and can be accessed through this link[1].

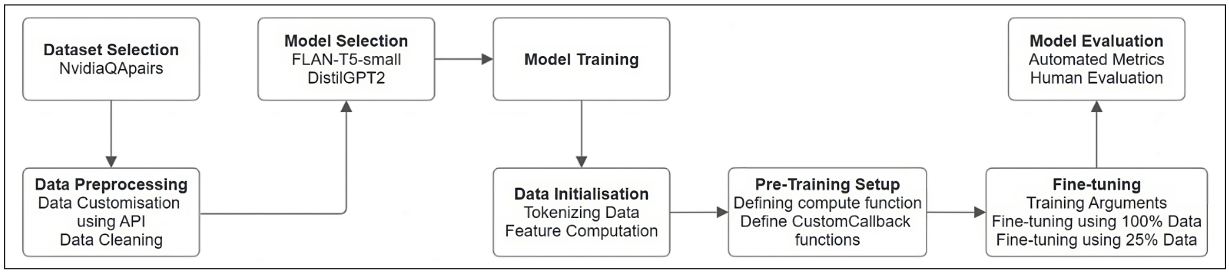


FIGURE 3.1: Process Flowchart Detailing the Methodology Employed in The Research

[1]See more details at GitHub.

3.1 Dataset

The dataset utilised in this study is a public collection of 7107 question-and-answer pairs sourced from Kaggle, which reflects domain-specific knowledge similar to corporate documentations (Deepesh et al., 2023). These pairs were originally generated from various NVIDIA documentation sources, such as development kits and guides. This dataset selection was intentional, aiming to closely mimic the real-world data environments of corporate entities that frequently rely on specialised, context-heavy informational resources. The dataset's domain-specific nature makes it an ideal candidate for training models intended for corporate deployment, facilitating an authentic assessment of model performance in business applications.

3.2 Data Pre-processing

3.2.1 Dataset Customisation

In preparation for practical model training, the dataset underwent a customisation phase where the original answers were condensed using the advanced capabilities of ChatGPT-4 (OpenAI, 2023). This step was essential for optimising the dataset to support the generation of concise answers, ensuring that only essential information was included in the training process. The dataset's public domain licence under Creative Commons Zero (CC0) ensures unrestricted use, facilitating the customisation process without legal constraints. This process was not only aimed at reducing time and computational resources but also at enhancing the focus and directness of the model outputs. Figure 3.2 illustrates the process of dataset customisation, detailing the use of the *get_short_answer* function. This function was applied to each row of the CSV data frame using a *lambda* function, where the prompt for the API requested the summarisation of answers into concise responses. The resulting short answers were stored in a new column, and the updated data frame was saved to a new CSV file. Manual validation of the generated answers was then conducted to ensure the quality and accuracy of the responses.

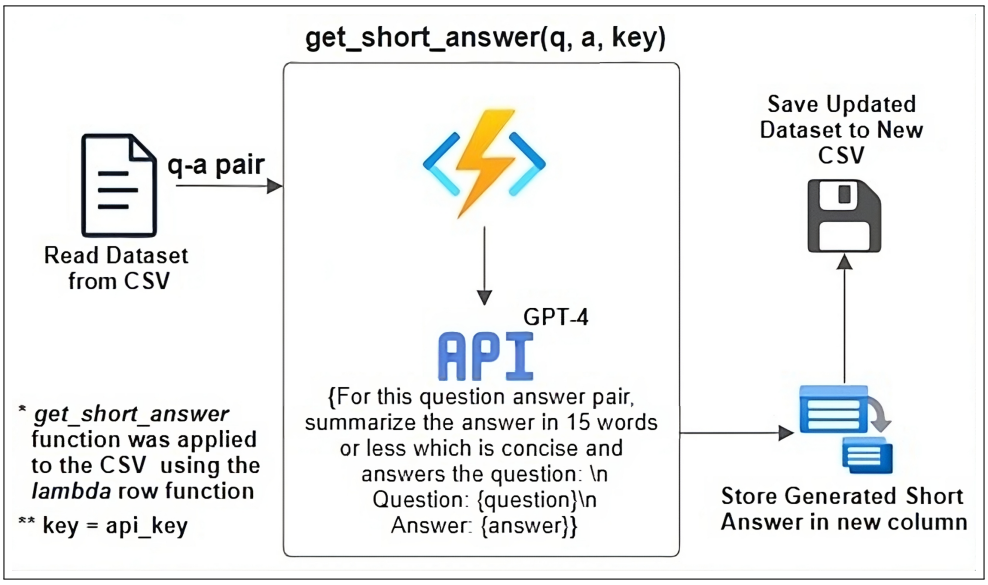


FIGURE 3.2: Workflow of the *get_short_answer* Function Utilizing the GPT-4 API for Dataset Customisation

Below are some exemplar pairs demonstrating the transition from detailed explanations to concise responses, facilitating more direct and efficient model training:

(1) **Question:** What is Hybridizer?
- **Original Answer:** Hybridizer is a compiler from Altimesh that enables programming GPUs and accelerators using C code or .NET Assembly.
- **Condensed Answer:** Hybridizer is a compiler for programming GPUs and accelerators using C or .NET Assembly.

(2) **Question:** What are the fundamental WMMA sizes in CUDA 9.0?
- **Original Answer:** The fundamental WMMA sizes in CUDA 9.0 are typically 16-by-16-by-16, corresponding to the size of the processing array in Tensor Cores.
- **Condensed Answer:** The fundamental WMMA sizes in CUDA 9.0 are typically 16-by-16-by-16.

(3) **Question:** How does the exceptionally high memory bandwidth of GPUs contribute to hash map acceleration?
- **Original Answer:** The exceptionally high memory bandwidth of GPUs enables the acceleration of data structures like hash maps by improving memory access efficiency.
- **Condensed Answer:** High GPU memory bandwidth accelerates hash maps by enhancing memory access efficiency.

(4) **Question:** What technology did the scientists use to develop EDDY?

- **Original Answer:** The scientists used NVIDIA Tesla K40 GPUs and CUDA technology to develop EDDY.
- **Condensed Answer:** EDDY was developed using NVIDIA Tesla K40 GPUs and CUDA technology.

3.2.2 Data Cleaning and Preparation

After curating the new dataset, the data preparation process began with a thorough cleaning procedure to ensure data uniformity and integrity, which is crucial for effective model training (Tae et al., 2019). The process involved identifying and removing duplicate entries to avoid potential bias caused by their imbalance. It continued with the normalisation of the text by converting all characters to lowercase to maintain consistency across the dataset. Additionally, the preparation meticulously eliminated excessive spaces, newline characters, and any leading or trailing white-space. This thorough preparation enhanced the dataset's quality and rendered it well-suited for model training and evaluation.

3.3 Model Selection

This section presents the transformer models selected for this research, focusing on their architectural differences and suitability for question-answering tasks. A thorough understanding of the various transformer architectures was crucial before selecting the appropriate models for the research objectives. Introduced in 2017 by Vaswani et al., transformers feature a sophisticated attention mechanism that processes input data through multiple layers, effectively transforming it into the desired output. This architecture incorporates an attention mask, crucial for its ability to manage long-range dependencies and contextual information. By focusing on relevant data points within the input, the attention mask ensures that the transformer does not allocate resources to unnecessary parts of the data, such as padded tokens. This capability is particularly important in complex tasks like question-answering, where accurately understanding and using the context and dependencies within the data are essential for generating correct responses. The architecture is visualized in Figure 3.3, which illustrates how different components of a transformer work together to process and generate information (Vaswani et al., 2017).

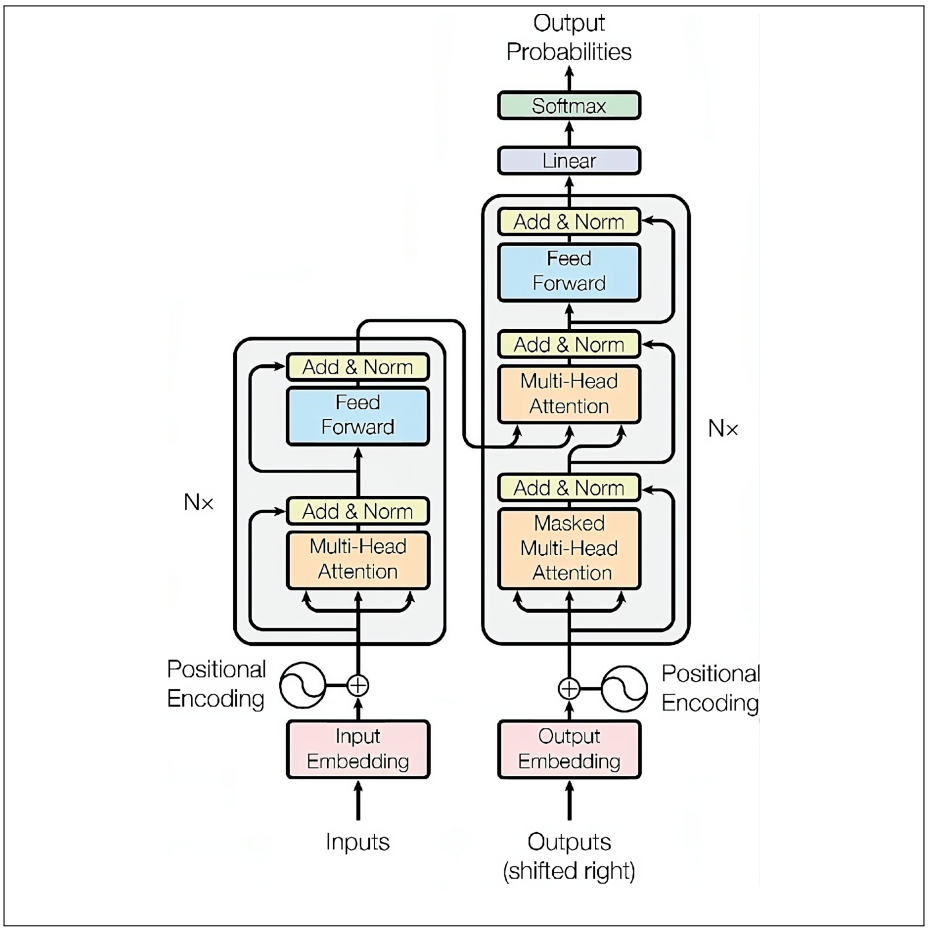


FIGURE 3.3: Transformer architecture (Vaswani et al., 2017)

These models are primarily divided into three types depending on their architecture:

- **Encoder-Decoder Models:** These models, a combination of both encoder and decoder components, leverage the full transformer structure depicted in Figure 3.3. This versatile configuration allows them to perform tasks that involve both understanding the input and generating new text based on that understanding. They are found to be effective for tasks like translation or summarisation, where the input needs to be accurately transformed into an output sequence.
- **Encoder-Only Models:** These models consist solely of the encoder blocks, as shown on the left side of the Figure 3.3. They process input text to create a contextual understanding, useful for tasks that require understanding but not generating text, such as text classification or feature extraction. The encoder layers use self-attention to analyse the input sequence in its entirety.
- **Decoder-Only Models:** These models consist solely of decoder blocks, as illustrated on the right side of Figure 3.3. They are specifically designed for text generation tasks. Decoder-only

models employ masked multi-head attention mechanisms, which restrict the model's attention to earlier positions in the output sequence. This design ensures that each token is generated sequentially, relying on the context accumulated from previously generated tokens.

3.3.1 Models Chosen for This Study

3.3.1.1 Encoder-Decoder Model

FLAN-T5-small, part of Google's expansive FLAN-T5 series, is designed to handle a variety of text-to-text generation tasks (Chung et al., 2022). With approximately 77 million parameters and 308 MB model size, FLAN-T5-small is engineered for efficient processing, making it particularly suitable for computational environments with limited resources (Wolf et al., 2019). This model benefits from a broad pre-training regime, encompassing various language understanding and generation tasks. Such comprehensive pre-training enhances its ability to be fine-tuned for specific domains, aligning perfectly with the needs of this study, which focuses on the answer generation for questions based on NVIDIA documentation. The encoder-decoder architecture of FLAN-T5-small enables it to effectively understand and transform input data into relevant and coherent outputs.

3.3.1.2 Encoder-Only Model

No encoder-only models were selected for this study. As the findings from Chapter 2 indicate, while encoder-only models, such as BERT, are highly effective for tasks requiring a deep understanding of input text (e.g., classification or feature extraction), they lack the inherent capability to generate text. This limitation is due to their architecture, which does not include a decoding component necessary for producing sequential outputs. For the objectives of this research, which involves generating text as a primary function, encoder-only models were confidently deemed unsuitable.

3.3.1.3 Decoder-Only Model

DistilGPT2, Developed by Hugging Face, is a distilled version of OpenAI's GPT2 model and contains approximately 88.2 million parameters (353 MB). Despite its reduced size, this model retains much of the functionality of the larger GPT2 model, optimised for reduced computational cost while maintaining its exciting rapid response generation capabilities (Wolf et al., 2019). The model is specifically tailored to prioritise rapid response generation, a crucial attribute for applications that demand

real-time processing, such as interactive chatbots for question-answering. The inclusion of DistilGPT2 in this study provides a valuable basis for comparison against the encoder-decoder model, highlighting the strengths and limitations of using a decoder-only model in tasks that require the generation of contextually accurate and relevant textual responses (Sanh et al., 2019).

To conclude, FLAN-T5-small and DistilGPT2 were selected for this study due to their substantial yet comparable parameter counts, with approximately 77 million and 88.2 million, respectively. This allowed for a balanced comparison of their text generation capabilities within similar computational constraints.

3.4 Model Training

3.4.1 Training Environment

The study was conducted on a machine equipped with an NVIDIA RTX 3080 GPU and 10 GB of GPU memory. Anaconda Navigator was used as the primary platform for managing necessary packages and environments (Anaconda Software Distribution, 2016). All significant computational tasks, including data processing and model training, were executed on Jupyter Notebooks. This interface was selected for its ease of use in documenting the code, detailing the outputs, and providing analytical comments in a cohesive notebook format, facilitating a clear and transparent depiction of the research process.

As the research progressed, the 10 GB of GPU memory was insufficient for the scale of training and evaluation required. However, the challenge was addressed by procuring additional resources. Specifically, an NVIDIA RTX 6000 Ada GPU with 48 GB of memory was rented from Vast.ai, an online platform providing scalable cloud computing resources (Vast.ai, n.d.). This upgrade significantly enhanced computational capacity, allowing for more extensive training sessions and complex model evaluations, thereby supporting the thorough exploration and validation of the research objectives.

3.4.2 Data Initialisation

The dataset, cleaned and prepared as outlined in the data cleaning section, resulted in 6,956 entries. It was then divided into three sets: training, validation, and testing, constituting 70%, 20%, and 10% of the data, respectively. This division resulted in 4,869 entries for training, 1,391 for validation,

and 696 for testing. A fixed seed was used during the splitting process to ensure reproducibility and consistency in the results. Subsequently, these subsets were organised into a DatasetDict, a structured format that facilitates efficient data handling and batch processing during model training and evaluation. Essential libraries such as 'pandas' for data manipulation were imported to support these operations (pandas development team, 2020). Furthermore, the data was prepared for fine-tuning using two custom preprocessing functions for both the model architectures: encoder-decoder and decoder-only models.

3.4.2.1 Pre-processing function for FLAN-T5-small

A custom pre-processing function, *tokenizer_for_t5*, was employed to tokenize question and answer pairs for the sequence-to-sequence training of the FLAN-T5-small model. This function, utilising the T5Tokenizer from the transformers library, tokenized questions with a maximum length of 90 and answers with 64, with padding and truncation to manage sequence length effectively (Wolf et al., 2019). These specific tokenization parameters were crucial for managing sequence length effectively and ensuring that the model's input was precisely tailored for its learning. The padded labels were replaced with -100 before feature computation to exclude them from loss calculations. The comprehensive steps and configurations are detailed in the pre-processing workflow, as depicted in Figure 3.4.

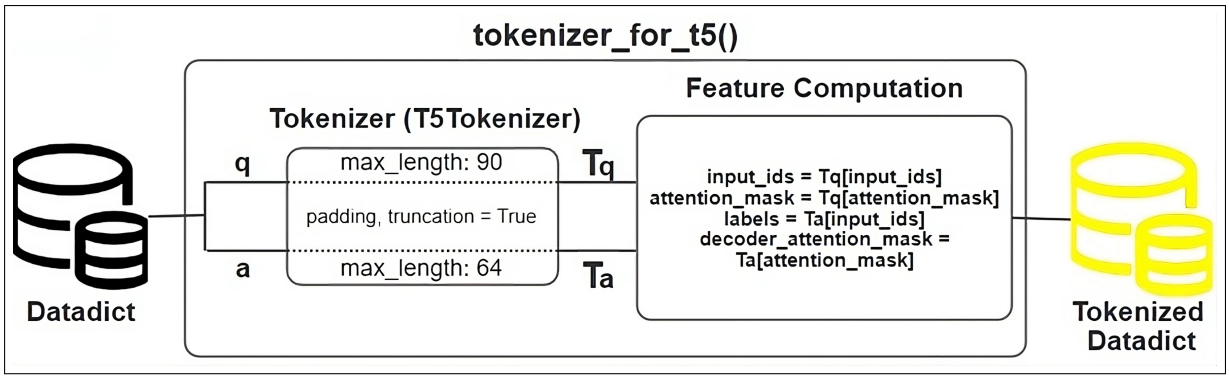


FIGURE 3.4: Data Pre-processing Pipeline for FLAN-T5-small

The critical components generated by *tokenizer_for_t5* included:

- **Input IDs:** These served as the numerical representation of the questions, essential for the model to process and understand the textual inputs.
- **Attention Mask:** This mask was generated for the questions. The primary function of the attention mask was to identify which tokens should be considered by the model's attention mechanism during processing. It differentiates between actual data tokens and padding tokens.

- **Labels:** Generated from the tokenized answers, labels were used as targets in the model's training process, guiding the model in learning the correct outputs associated with the given inputs.
- **Decoder Attention Mask:** This mask was generated for the answers. Its primary role was to direct the mode's decoder to focus on relevant sections of the answers, ensuring that attention was only given to actual data tokens rather than padding (Raffel et al., 2019).

This preprocessing function was mapped to transform the raw text data from the DatasetDict into a structured, tokenized DatasetDict, which was prepared for model training. This systematic conversion ensured that each element was methodically organised and readily accessible for the model, optimising the efficiency of the training process.

3.4.2.2 Pre-processing function for DistilGPT2

For the training of the DistilGPT2 model, a distinct pre-processing function, *tokenizer_for_gpt* was implemented to accommodate its architectural differences compared to FLAN-T5-small. This function joined the question and answer pair into a single string format - "Question: question Answer: answer", which is essential for the GPT model's learning mechanism that focuses on predicting the next token in the sequence (Radford et al., 2019). The concatenated strings were tokenized to a maximum length of 155 with GPT2Tokenizer, with padding and truncation set to true to manage sequence length effectively (Wolf et al., 2019). The comprehensive steps and configurations are detailed in the pre-processing workflow, as depicted in Figure 3.5.

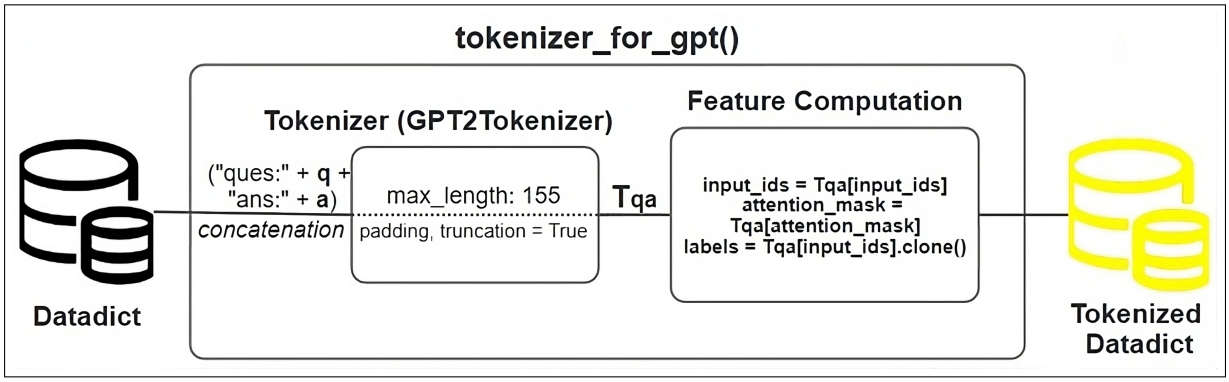


FIGURE 3.5: Data Pre-processing Pipeline for DistilGPT2

The critical components generated by this *tokenizer_for_gpt* included:

- **Input IDs:** These served as the numerical representation of the concatenated question and answer pairs, essential for the model to process and understand the textual inputs.
- **Attention Mask:** This mask was generated during the tokenization of the concatenated pairs. Its primary role was to identify which tokens should be considered by the model's attention mechanism during processing, differentiating between actual data and padding tokens. This feature optimises computational efficiency by focusing the model's processing power on relevant content.
- **Labels:** These labels were used as targets during the training process. The model automatically shifts them by one position during fine-tuning, which is essential since the GPT architecture is designed to predict the next token in a sequence.

This preprocessing function transformed the raw text data from the DatasetDict into a structured, tokenized DatasetDict prepared for DistilGPT2 model training.

3.4.3 Pre-Training Setup

3.4.3.1 Compute Metrics Function

This function was designed to evaluate various performance metrics for the predictions generated by the models. This function uses *eval_pred* as input, which comprises of the model's predictions and the corresponding labels - actual answers (Wolf et al., 2019). The metrics integrated in this function were categorised into three categories: lexical-based metrics, similarity metrics and operational efficiency metrics, all of which are discussed in detail in the model evaluation section. Additionally, several advanced methods were defined alongside this function to support the complex calculations required for some metrics.

Including a wide range of metrics was crucial for identifying the most effective metrics for evaluating closed generative question-answering tasks. Given the absence of standardised evaluation metrics in the field, this approach enabled a thorough assessment of metrics tailored to measure model performance accurately in specific domains. Lastly, this function was important to implement before the training process to monitor the performance over the epochs and avoid overfitting - a common challenge where a model learns the training data overly well but performs poorly on new data (Ying, 2019).

3.4.3.2 Custom Callback Functions

Three custom callback functions were implemented to support the training process and to ensure robust monitoring of the model's learning progression.

(1) **LossTrackingCallback:** This function was implemented to track the training and validation losses over the epochs. The recorded data assisted in effectively diagnosing the model's training dynamics, enabling the identification of any tendencies towards overfitting. As a result, appropriate modifications were made throughout the training process based on the visualised graphs generated from this tracked data.

(2) **MemoryCleanupCallback:** This function was implemented to manage the GPU resources effectively, which is crucial when handling extensive datasets or conducting prolonged training sessions. It ensured that GPU memory was cleared after each epoch, preventing memory overflow and maintaining the stability of the training process.

(3) **GenerationControlCallback:** This callback managed adjustments to the model's text generation settings during the evaluation phase. It modified parameters such as generation length, beam search, and temperature in the output to optimise the quality of text generation during evaluation (Wolf et al., 2019).

These functions were crucial in establishing a solid foundation for the training phase. They ensured that the models were not only optimised for good performance but also met the stringent analytical demands of the study, thereby enabling a comprehensive and effective evaluation of their capabilities.

3.4.4 Fine-tuning Approach

This study used a supervised learning approach to fully fine-tune two language models, FLAN-T5-small and DistilGPT2. Supervised learning involves training pre-trained models on a labeled dataset, enabling the models to learn to predict the correct labels for each input (Acharya, 2023). This method was particularly suited for adapting these models which pre-trained on extensive, diverse corpora to the specific task of handling question-answers. This section outlines the configuration and fine-tuning procedures for both models.

3.4.4.1 Training Arguments

A consistent set of training arguments was defined to ensure a fair comparison between both models. This approach was crucial to ascertaining that any differences in performance would be attributed directly to the intrinsic capabilities and architecture of the models rather than variations in training configurations.

The *num_train_epochs* was set to 5, a balance chosen to allow for comprehensive learning while also preventing the models from over-fitting to the training examples. This helped maintain the model's ability to generalise well to new, unseen data.

To optimise the use of computational resources, the *per_device_train_batch_size* was configured at 8 and the *per_device_eval_batch_size* at 4. These settings ensured efficient learning and GPU utilisation without overwhelming the system's memory, which is crucial for maintaining high computation speeds and system stability.

The *warmup_steps* parameter was configured to 500. This strategy helps mitigate the primacy effect, where the model might disproportionately learn from the initial training examples. By starting with a lower learning rate and gradually increasing it, the model adjusts more effectively, learning from the entirety of the data. The learning rate was not fixed to a single value to avoid the risk of overshooting optimal parameter updates with a high learning rate or stagnating progress with a low learning rate. The warmup phase allowed for a more flexible and adaptive learning process, enhancing the model's ability to converge effectively.

A *weight_decay* of 0.01 was implemented as a regularisation technique to prevent the model weights from growing too large, which could lead to over-fitting. By penalising large weights, the model maintained uniform weight distributions, enhancing its ability to perform well across the entire dataset.

Both models employed an *epoch-based* evaluation and save strategy to monitor and preserve their progress throughout training systematically. This setup allowed the evaluation of the model's performance at the end of each training epoch, providing regular feedback on its performance and facilitating periodic checkpoints for model recovery and further analysis.

Due to potentially higher memory demands, *gradient_accumulation_steps* and *eval_accumulation_steps* were set to 2 for the DistilGPT2 model. This configuration allows the model to process larger batches of data in smaller, more manageable segments, effectively reducing the instantaneous memory load.

These training parameters were meticulously chosen to support the models learning efficiencies and robustness, ensuring that each model could be evaluated fairly and effectively under controlled environments.

3.4.4.2 FLAN-T5 Fine-tuning

The fine-tuning process for the FLAN-T5-small model, a crucial step in effectively leveraging the model's capabilities, began with retrieving the pre-trained FLAN-T5-small model from the Hugging Face model hub, along with its corresponding tokenizer (Wolf et al., 2019). The DataCollator for *Seq2Seq* tasks was then initialised, playing a key role in batch formation for training and dynamic padding of batches during both the training and evaluation phases.

Custom callback functions were initialised to augment the training regimen, as discussed in the previous subsection. These included callbacks for monitoring the loss metrics, efficiently managing GPU memory, and adjusting generation settings during evaluations. Each played a significant role in enhancing the model's training process and its adaptability during the fine-tuning stage.

Further, a *Trainer* instance was initialised, serving as the core management system for the FLAN-T5-small model's training. This instance was configured with the model, training arguments, the prepared training and validation datasets, the tokenizer, and the data collator. Additionally, the *compute_metrics* function was passed into the *Trainer* to continuously monitor the model's performance, allowing for real-time assessment of various performance metrics throughout the training process.

The fine-tuning was carried out by applying the *train* method on the *Trainer* instance, which guided the extensive training of the model. Upon completion, the model was saved for inference and evaluation. Detailed results of this training process are presented in Chapter 4.

3.4.4.3 DistilGPT2 Fine-tuning

Building on the procedures established for the FLAN-T5 model, the fine-tuning of the DistilGPT2 model was executed similarly. The pre-trained DistilGPT2 model and its corresponding tokenizer were loaded. Additionally, necessary components such as the data collator and custom callback functions were set up to enhance training efficiency. Concurrently, the *Trainer* instance was configured with the same training arguments to ensure consistency across experiments. A critical modification for the DistilGPT2 model involved implementing gradient accumulation and evaluation steps necessitated by

the model's increased memory demands. This strategy enabled processing larger batch sizes in smaller segments, thus effectively managing GPU memory and preventing computational overloads (Rotenberg, 2020). Following these preparations, the model underwent training and the fine-tuned model was saved.

Throughout the training process, the continuous feedback from the *compute_metrics* function demonstrated consistent improvement across various performance metrics, confirming the model's effective adaptation and enhanced learning in response to the task. Chapter 4 presents detailed results of this training process.

3.4.4.4 Training on Reduced Dataset Size

As part of the experimental design, training was also conducted using only 25% of the dataset, with the same training configurations as those used for the complete dataset experiments. This approach was intended to ensure consistency in the experimental setup and facilitate a comparison of the effects of reduced data size on model training. The subset selected for this scaled training was randomly chosen to preserve the diversity of question complexities and domain varieties in the entire dataset. The outcomes of the performance of this scaled training will be presented in the next chapter.

3.5 Model Evaluation

This section outlines the evaluation metrics and methodologies applied to measure the models' performance, as depicted in Figure 3.6. A diverse array of metrics drawn from various studies and findings highlighted in the literature review were selected for the analysis. These include lexical-based assessments, similarity measures, operational evaluations, and human evaluation, each offering insights into different aspects of the model's performance. This extensive evaluation criteria were designed to capture the nuanced capabilities of encoder-decoder and decoder-only models for generative question-answering tasks.

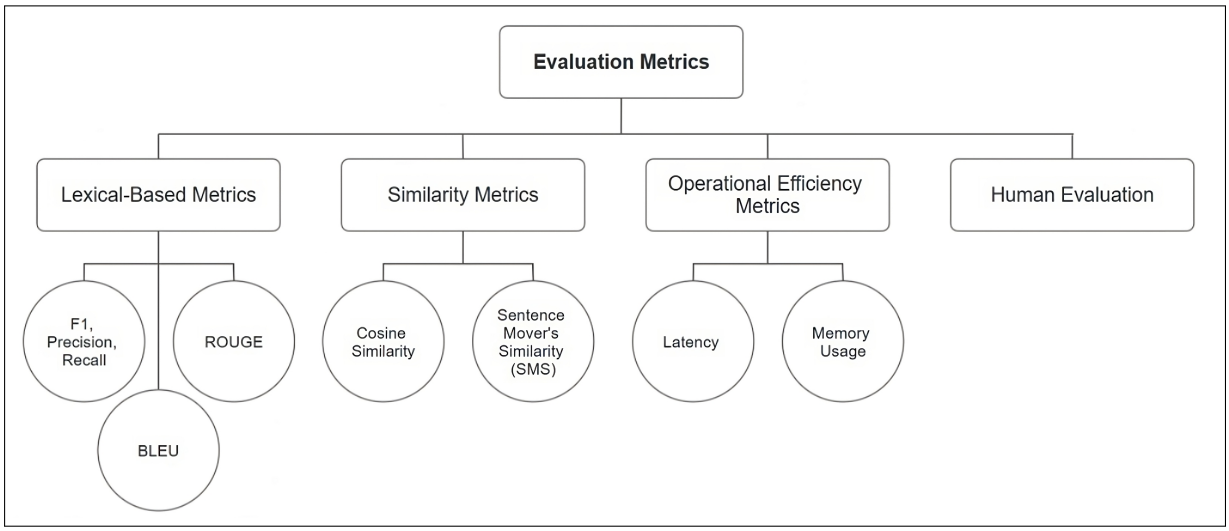


FIGURE 3.6: Overview of Model Evaluation Metrics

3.5.1 Lexical-Based Metrics

Lexical-based metrics have proven extremely valuable for evaluating natural language processing models. These metrics assess the correspondence of lexical items - words or phrases in the generated text - against those in a reference text. They are particularly effective for their speed in providing quantifiable feedback and are widely utilised in various research to assess translation and text generation tasks (Lee et al., 2023).

In this study, lexical-based metrics such as F1, Precision, and Recall were used, which are renowned for their effectiveness in numerous studies. Automated metrics like ROUGE and BLEU were also employed for their reliability in quickly assessing model performance. Each of these metrics is discussed in detail below.

3.5.1.1 F1, Precision, and Recall

Researchers extensively utilise F1, Precision, and Recall metrics to evaluate models within the domain of question-answering. In this study, these metrics were employed at the word level to assess the extent of overlap between words in the generated answers and the actual answers.

To implement these metrics within the research, a custom function named *precision_recall_fscore_support* was defined. This function was integrated into the *compute_metrics* function, allowing for effective assessment of these values during the validation and evaluation phases, ensuring that each model's performance underwent rigorous testing. The average of F1, Precision, and Recall scores was calculated to ensure comprehensive evaluation, reflecting the model's overall accuracy in reproducing exact words from the reference answers.

3.5.1.2 ROUGE

ROUGE, an acronym for Recall-Oriented Understudy for Gisting Evaluation, assesses the quality of generated summaries by comparing them to reference summaries. It quantifies the overlap of various units, such as n-grams, word sequences, and word pairs, between the generated text and reference texts ((Lin, 2004).

For this research, the focus was specifically on ROUGE-1 and ROUGE-L. ROUGE-1 measures the overlap of unigrams between the generated and reference text, providing a direct count of common words, thus gauging the extent of the exact match. ROUGE-L, based on the longest common subsequence (LCS), offers a more flexible evaluation metric since it does not require predefined n-gram sizes. The choice to focus on ROUGE-L alongside ROUGE-1 is supported by their relevance in assessing the structural integrity of text as outlined in a study by (Chen et al., 2019). According to the study, ROUGE-L is advantageous for its ability to measure longer sequences of coherent text, making it particularly useful for evaluating answers generated by question-answering systems.

This work calculated the ROUGE scores using the *Rouge* class in the *compute_metrics* function to integrate ROUGE evaluations within the framework. This allowed for an automated assessment of how close the model's generated answers were to the actual answers. After each evaluation phase, the average ROUGE scores were computed, providing a balanced view of how well the generated text matched the reference answers in granularity and fluency.

3.5.1.3 BLEU

BLEU, short for Bilingual Evaluation Understudy, was originally developed to assess machine translation accuracy by comparing a candidate translation to one or more reference translations (Papineni et al., 2002). Despite its roots in translation, BLEU has been broadly adopted for a variety of NLP tasks,

including text summarisation and question-answering. The strength of BLEU lies in its ability to perform quick and inexpensive evaluations. By counting matching n-grams in the candidate and reference texts, BLEU provides a quantitative measure of text alignment and accuracy.

This work calculated the BLEU scores using the *sentence_bleu* function from the NLTK library (Loper and Bird, 2002). The scores were computed in the *compute_metrics* function by comparing the common n-grams in the predicted answers to those in the reference answers. Following the assessments, an average BLEU score was determined, summarising its ability to align n-grams with those in the reference answers and thereby gauging its overall linguistic accuracy.

3.5.2 Similarity Metrics

Similarity metrics are essential in assessing the semantic alignment between the generated text and reference text. These metrics extend beyond mere lexical overlaps to measure the meaningfulness of the generated answers. This study focused on two key similarity metrics as detailed below.

3.5.2.1 Cosine Similarity

Cosine similarity is a mathematical metric used to measure the similarity between two vectors by measuring the cosine of the angle between them. These vectors represent the semantic embeddings of the entire sentence. It helps determine how closely the output produced by the model aligns with the expected semantic orientation of the original answer. Essentially, it evaluates the overall orientation of vectors, indicating the degree of similarity in terms of content, but does not account for the structural variations within the sentences (Miesle, 2023). A cosine similarity score of 1 indicates perfect alignment, 0 signifies no similarity, and -1 represents complete dissimilarity.

For calculating cosine similarity, embedding vectors were generated using the sentence transformer model *'all-MiniLM-L6-v2'* (Aarsen et al., 2021b). This model was used to encode the predicted answers and actual answers into high-dimensional vectors. Cosine similarity between these vectors was then calculated. This process resulted in a series of cosine similarity scores, which were averaged to provide an overall measure of semantic alignment between the predicted and actual answers.

3.5.2.2 Sentence Mover's Similarity (SMS)

Sentence Mover's Similarity measures the minimal semantic distance that words in one sentence must 'move' in semantic space to align with the words in another sentence. This is computed using the Earth Mover's Distance (EMD), which assesses the 'cost' required to align the semantic spaces of two sets of word embeddings (Kusner et al., 2015). A higher SMS score indicates that less adjustment is needed, signifying a closer semantic relationship between the compared texts. The score is derived by subtracting the normalised EMD from 1, where a score near one indicates high similarity. SMS proves invaluable in fields such as question-answering and text summarisation, where understanding nuanced meanings and contextual dependencies is crucial.

For the Sentence Mover's Similarity (SMS) calculation, two functions were defined prior to the *compute_metric*. The first function, *get_word_embeddings*, retrieves embeddings for each word in a sentence using the pre-trained sentence transformer model *'all-mpnet-base-v2'* (Aarsen et al., 2021a). This function was defined to convert textual data into a format suitable for semantic comparison. Subsequently, the *sentence_movers_similarity* function calculates SMS by utilising the embeddings from the first function. It constructs a cost matrix representing the semantic distances between each word's embedding in one sentence and those in another. The Earth Mover's Distance (EMD) is then computed to determine the minimum cost required to transform one sentence's semantic structure into another. This transformation cost is normalised and converted into a similarity score. The final SMS scores were computed across multiple generated and actual answer pairs and averaged to provide a robust measure of the model's ability to generate semantically relevant responses.

3.5.3 Operational Efficiency Metrics

Operational efficiency metrics, specifically Latency and Memory Usage, are crucial for evaluating the practical deployment capabilities of machine learning models. These metrics help assess the model's deployability in real-world scenarios.

3.5.3.1 Latency

Latency measures the time required for a model to complete a task. It reflects the model's speed and operational efficiency. During this study, latency was monitored throughout the training and evaluation phases for both the models to analyse their performance speed.

For latency measurement, the start time was captured at the initiation of the *compute_metrics* function. At the conclusion of this function, the end time was recorded, and latency was calculated as the elapsed time between these two points. This provided a direct measure of the model's operational time efficiency.

3.5.3.2 Memory Usage

Memory Usage is a critical metric that measures the amount of RAM utilised during its training and evaluation stages. This measure is essential for assessing the model's resource efficiency and evaluating its potential for deployment in constrained memory resources. Memory usage was tracked through the *compute_metrics* function, which measured it at the end of metric computations. This process involved retrieving the Resident Set Size (RSS) from the system to get the total RAM used by the model. The RSS was converted to megabytes to measure memory demands during the training and evaluation phases.

3.5.4 Human Evaluation

Human evaluation stood out as the gold standard metric for evaluating the outputs of generative models in the context of question-answering tasks, from the findings of literature review (Peinl and Wirth, 2023; Wang et al., 2021; Yagnik et al., 2024). It is widely regarded as the most reliable method for evaluation since automated metrics can not account for the correctness of the answers.

In this study, a structured approach for human evaluation was adopted by categorising the predicted answers based on their correctness into three distinct levels: Correct, Partially Correct, and Incorrect. Each category is defined as follows:

- **Correct:** The predicted answer fully addresses the question, providing all necessary details found in the actual answer, demonstrating complete understanding and relevance.
- **Partially Correct:** The predicted answer addresses the question and includes some key details from the actual answer. It captures the essence of the question but may omit one or more significant aspects, showing a partial understanding.
  This category is particularly significant for generative question-answering tasks as it highlights the model's ability to address key concepts required to answer the question, even if it does not completely address every aspect of the expected answer.

- **Incorrect:** The predicted answer fails to address the question adequately or misses most of the essential details. This includes vague or generic responses that do not convey specific information relevant to the question.

To quantitatively assess the outcomes of this qualitative evaluation, each response was assigned a score based on its category: 1 for Correct, 0.5 for Partially Correct, and 0 for Incorrect. The average of these scores was then calculated to provide a robust measure of the model's accuracy from a human perspective. This approach allowed us to gauge the effectiveness of the models in producing contextually appropriate and accurate responses.

CHAPTER 4

RESULTS

This chapter presents the fine-tuning results for the FLAN-T5-small (encoder-decoder model) and DistilGPT2 (decoder-only model). The chapter highlights training and validation loss curves for models trained on the entire dataset, providing insights into their learning effectiveness. The validation section presents graphs highlighting the performance of evaluation metrics over 5 epochs. Lastly, the evaluation section outlines the results of fine-tuned models for both the models, using both 25% and 100% of the training data, providing a comprehensive analysis of their performance for closed-book generative question-answering.

4.1 Training and Validation Loss Curves

4.1.1 FLAN-T5-small

The training results of the FLAN-T5-small model were captured in Figure 4.1, which depicted the trends in training and validation losses over the observed epochs. The graph indicated a significant reduction in training loss, signalling efficient learning, while the validation loss decreased gradually and stabilised, reflecting strong adaptability to new data. As the training progressed, the training and validation losses began to converge, suggesting an optimal balance between learning and generalisation, with minimal risk of over-fitting. This graphical representation was crucial for assessing the model's performance dynamics and the effectiveness of the training approaches. Throughout the training process, the *compute_metrics* function provided ongoing assessments of various performance metrics, which showed improvement in initial epochs and stabilisation towards the end. Details of these metrics would be presented in the subsequent section.

FIGURE 4.1: FLAN-T5-small: Training and Validation Loss Curve

4.1.2 DistilGPT2

The training outcomes for the DistilGPT2 model were captured in Figure 4.2, which depicted the patterns of training and validation losses throughout the training epochs. The graph illustrated a significant drop in training loss, indicating robust learning during the initial training phase. Concurrently, the validation loss decreases gradually before stabilising, indicating a steady improvement in the model's ability to generalise effectively. Moreover, the influence of gradient accumulation was observable in the graph, where some data points were absent, resulting from adjustments that led to fewer logged updates per epoch under the configured training parameters. This was necessitated by the DistilGPT2 model due to its high memory demands. These dynamics were crucial for evaluating the overall efficacy of the training regimen and the model's ability to handle unseen data efficiently.

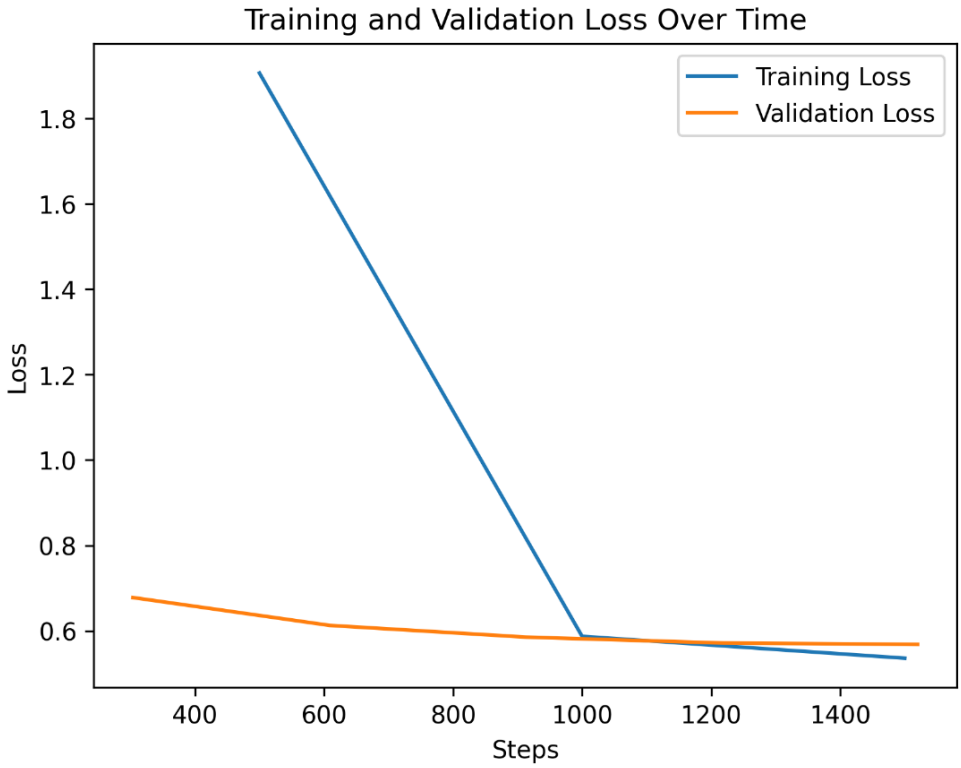


FIGURE 4.2: DistilGPT2: Training and Validation Loss Curve

4.2 Validation Graphs

This section illustrates the evolution of evaluation metrics over five epochs during the training phase of the models when trained on the entire dataset. To ensure a clear and consistent comparison throughout the section, results for the FLAN-T5-small model are depicted in red, while those for the DistilGPT2 model are depicted in blue.

4.2.1 F1, Precision, and Recall

The evolution of the F1-score, precision, and recall metrics across five training epochs for both the FLAN-T5-small and DistilGPT2 models was captured in Figure 4.3. This visualisation highlighted their gradual improvement and subsequent stabilisation over time, reflecting effective learning and generalisation. The use of different line styles - dotted for precision, dashed for recall, and solid for the F1-scores aided in distinguishing the individual progress of each metric clearly. Notably, the DistilGPT2 model consistently exhibited superior performance across all metrics, demonstrating its greater efficacy in aligning its generated answers with the actual answers.

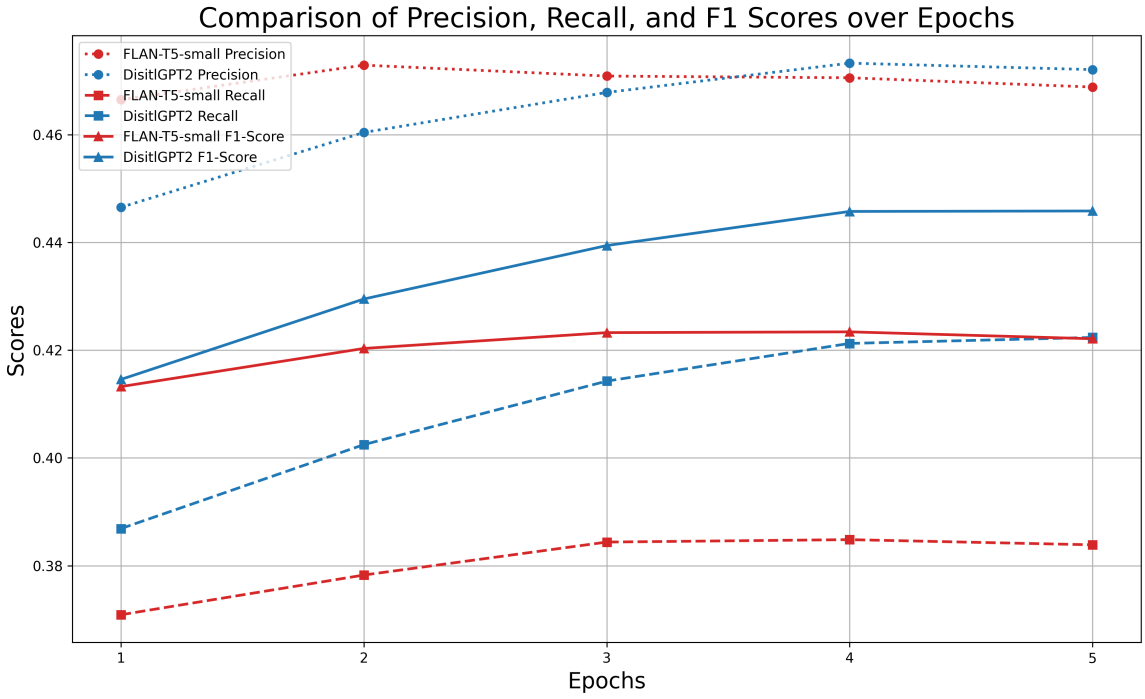


FIGURE 4.3: Comparison of Precision, Recall, and F1 Scores for FLAN-T5-small and DistilGPT2

4.2.2 ROUGE

The evolution of ROUGE scores during training were captured in Figure 4.4, which demonstrated the progression of ROUGE-1 and ROUGE-L metrics for both the models. The visualisation underscores the progressive improvement and stabilisation of these scores, with a noticeable convergence observed by the final epoch. The use of different line styles facilitated a clear distinction in these metrics trends: dashed lines for ROUGE-1 and solid lines for ROUGE-L. This distinction aided in observing the particular strengths of each model in capturing the exact unigram matches and the longest common subsequences (LCS). The analysis provided direct comparative insights into their performance during the validation phase. Notably, towards the last few epochs, the DistilGPT2 model demonstrated superior performance in achieving higher word match rates, reflecting its greater precision in aligning with the reference texts.

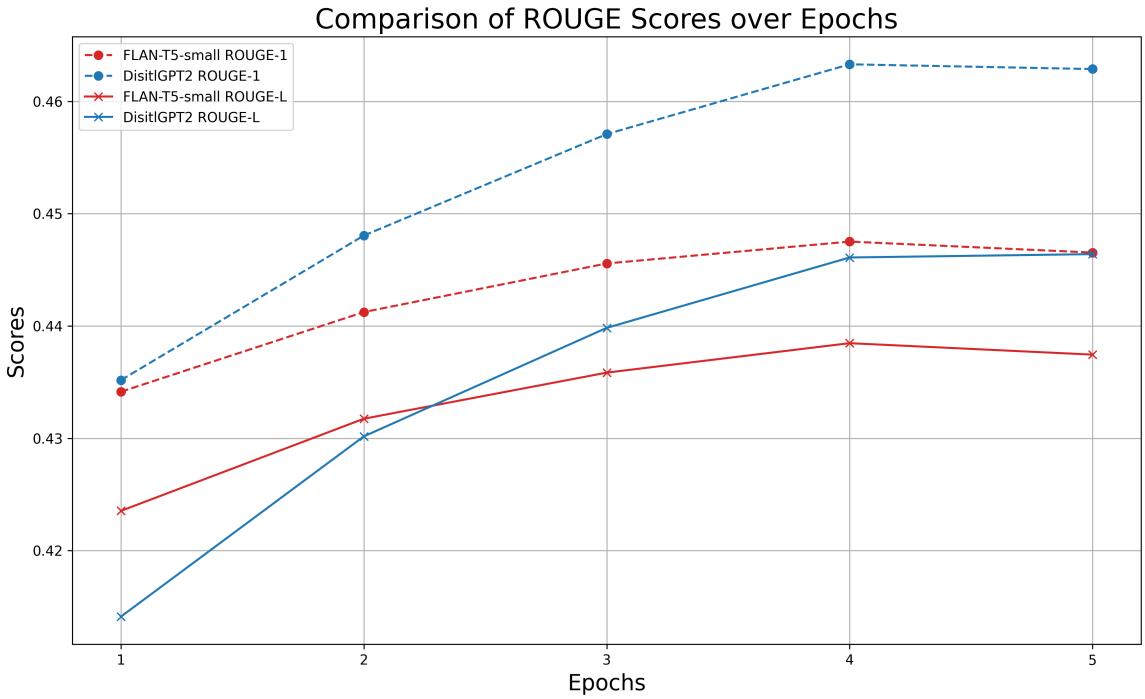


FIGURE 4.4: Comparison of ROUGE-1 and ROUGE-L Scores for FLAN-T5-small and DistilGPT2

4.2.3 BLEU

The evolution of BLEU scores during training was captured in Figure 4.5, which highlighted significant differences in performance between the two models. The FLAN-T5-small model's BLEU scores remained relatively constant and low throughout the epochs, indicating minimal improvement in its ability to generate answers which precisely match the word sequences found in the reference answers. In contrast, the DistilGPT2 model demonstrated substantial and consistent improvement, with scores progressively increasing from the first to the fifth epoch. This upward trend underscored DistilGPT2's capabilities to produce responses closely aligned with the exact wording of the reference answers. However, it is important to note that even the highest scores achieved by DistilGPT2 were modest, with the peak score around 0.15, reflecting the challenging nature of achieving good word-for-word alignment, as the theoretical maximum score is 1.0.

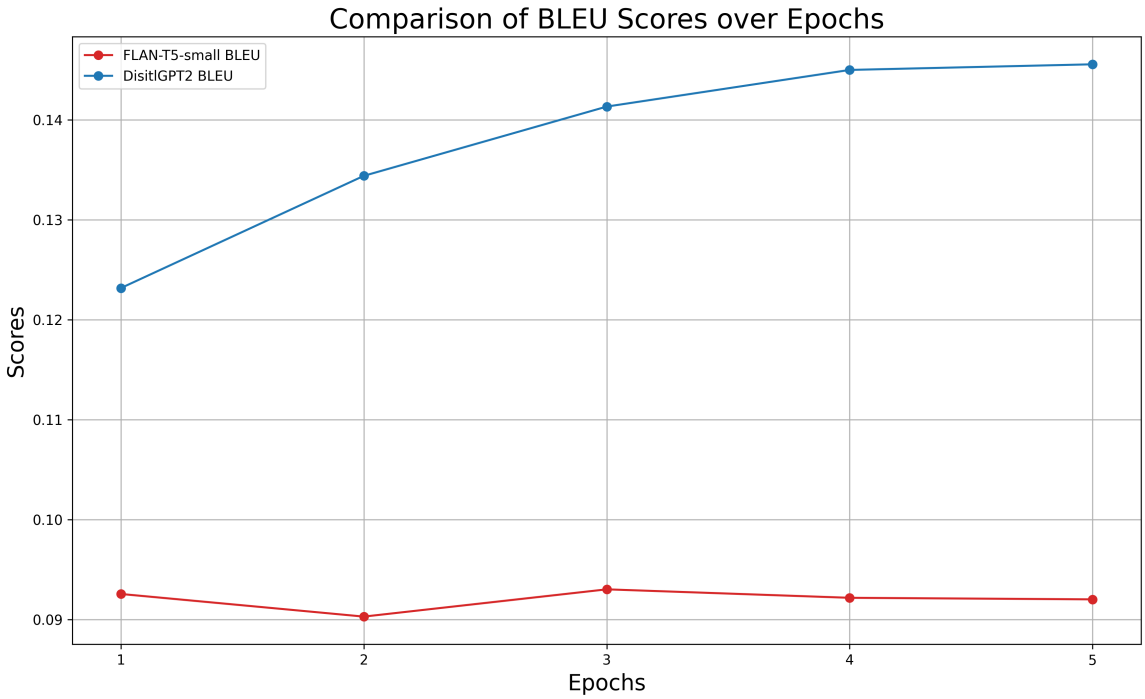


FIGURE 4.5: Comparison of BLEU Score for FLAN-T5-small and DistilGPT2

4.2.4 Cosine Similarity

Cosine similarity scores, which assess the semantic closeness between generated and actual answers, were tracked across five training epochs for the FLAN-T5-small and DistilGPT2 models, as shown in Figure 4.6. The DistilGPT2 model exhibited a steady increase in these scores through the first four epochs, stabilising in the fifth, which highlighted its consistent improvement in semantic accuracy. Conversely, the FLAN-T5-small model showed only slight improvements early on and plateaued at a lower level compared to DistilGPT2, indicating its limited capacity for semantic alignment with the reference texts.

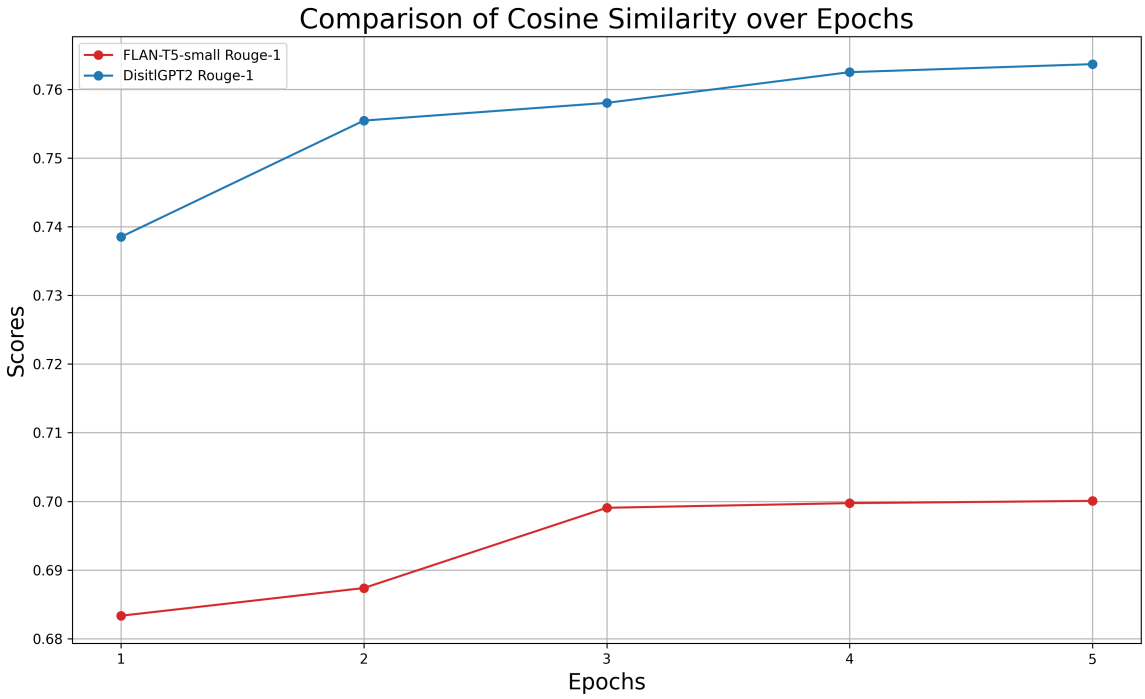


FIGURE 4.6: Comparison of Cosine Similarity for FLAN-T5-small and DistilGPT2

4.2.5 Sentence Mover's Similarity (SMS)

The graph, captured in Figure 4.7, depicted the differences in Sentence Mover's Similarity (SMS) scores between the FLAN-T5-small and DistilGPT2 models. Throughout the training epochs, the FLAN-T5-small model consistently maintained high SMS scores, indicating its robust ability to preserve the semantic integrity of the responses. In contrast, while the DistilGPT2 model exhibited some improvement, it remained significantly lower compared to that of FLAN-T5-small. The variations in SMS scores underscored the model's differing capabilities to accurately reflect the contextual nuances of the generated answers in comparison to the actual answers, a critical element in closed book generative question answering tasks.

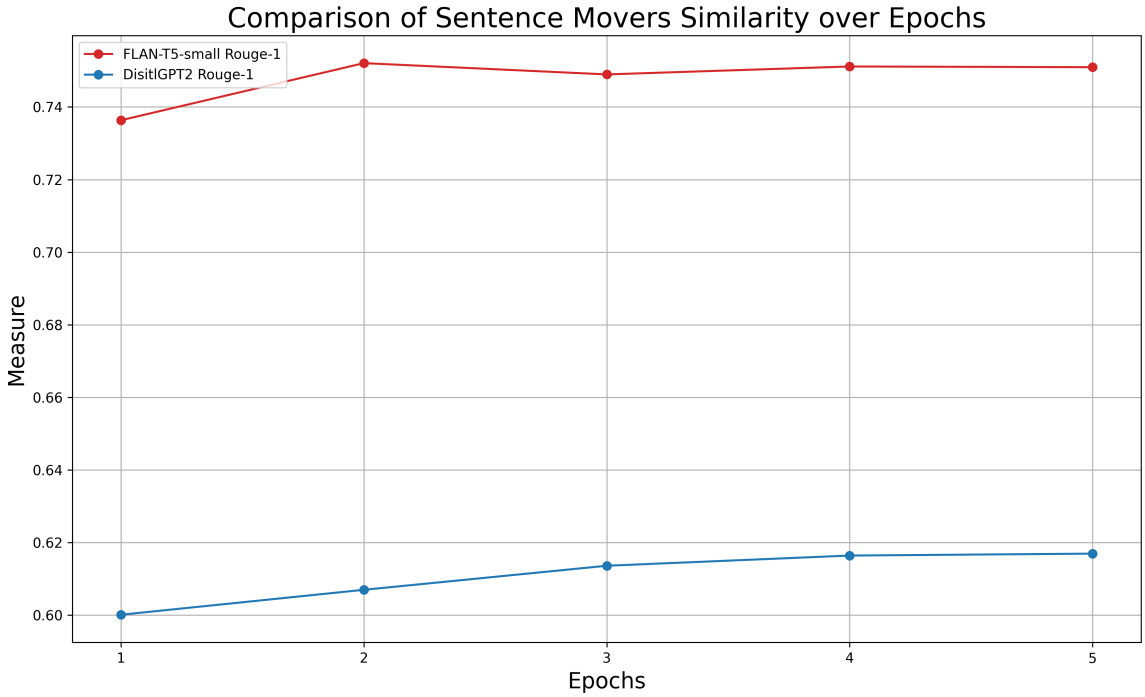


FIGURE 4.7: Comparison of Sentence Mover's Similarity (SMS) for FLAN-T5-small and DistilGPT2

4.2.6 Latency and Memory Usage

The graphs below, captured in Figure 4.8, presented the average latency and memory usage per epoch for both the FLAN-T5-small and DistilGPT2 models during the validation process. These metrics were crucial for assessing the operational efficiency of the models.

Latency measures the time required for the models to perform validation in each epoch. The FLAN-T5-small model exhibited significantly higher latency, with an average of 892.48 seconds, compared to 288.35 seconds for the DistilGPT2 model. This indicated that DistilGPT2 was approximately three times faster, demonstrating superior time efficiency during validation. In contrast, memory usage reflected the amount of RAM utilised by the models. DistilGPT2 had a substantially higher memory requirement, consuming an average of 43530.38 MB, which was roughly three times the memory usage of FLAN-T5-small, which averaged at 13815.72 MB. This indicated that while DistilGPT2 was comparatively faster, it demanded significantly more computational resources in terms of memory. These findings highlighted the trade-offs between processing speed and memory consumption during the validation process.

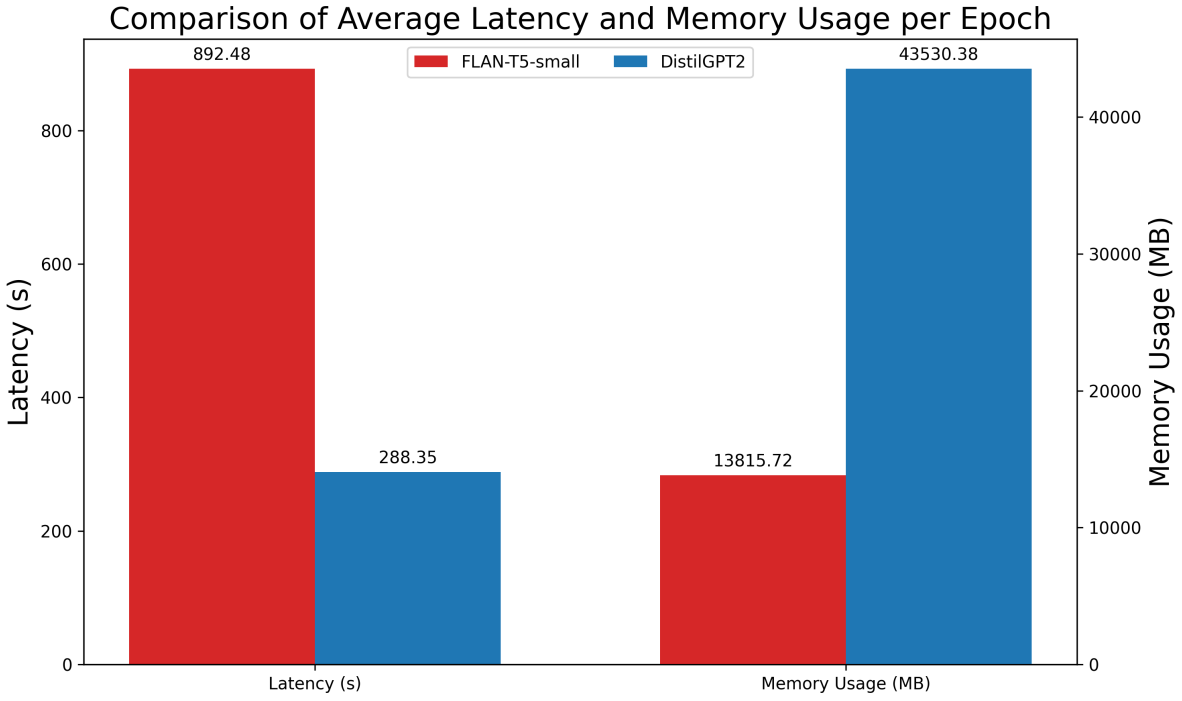


FIGURE 4.8: Comparison of Average Latency and Memory Usage for FLAN-T5-small and DistilGPT2

4.3 Evaluation of Fine-tuned Models

This section presents the evaluation results of the fine-tuned models of both architectures, FLAN-T5-small and DistilGPT2. Each model was trained on both 25% and 100% of the dataset and evaluated on a test set comprising 10% of the entire data, which was not used during training. The test set included a total of 696 question-answer pairs. The evaluation metrics covered various aspects, including lexical-based analysis, similarity measures, operational efficiency, and human evaluation, providing a holistic view of the performance of the fine-tuned models.

4.3.1 Lexical-Based Metrics

4.3.1.1 Precision, and Recall, F1

The table 4.1 presents a detailed comparison of precision, recall, and F1 scores for both the FLAN-T5-small and DistilGPT2 models when fine-tuned on 25% and 100% of training data. These metrics evaluate the overlap of words between the model's generated responses and the actual answers.

FLAN-T5-small

- When fine-tuned on 25% of data, the model achieved a precision of 0.47, a recall of 0.38, and F1 score of 0.42.
- When fine-tuned on the entire dataset, there was a slight improvement, with precision increasing to 0.48, recall to 0.40, and F1 score to 0.44.

DistilGPT2

- When fine-tuned on 25% of data, the model achieved a precision of 0.44, a recall of 0.39, and an F1 score of 0.41.
- When fine-tuned on the entire dataset, the metrics improved notably with the precision increasing to 0.48, recall to 0.44, and F1 score to 0.46.

TABLE 4.1: Precision, and Recall, F1 scores for FLAN-T5-small and DistilGPT2 Models Fine-Tuned on 25% and 100% of Dataset

	Fine Tuned Model on 25% Data			**Fine Tuned Model on 100% Data**		
Model/Metrics	**Precision**	**Recall**	**F1 Score**	**Precision**	**Recall**	**F1 Score**
Flan-T5-small	0.47	0.38	0.42	0.48	0.40	0.44
DistilGPT2	0.44	0.39	0.41	0.48	0.44	0.46

4.3.1.2 ROUGE and BLEU Scores

The table 4.2 provides a comparative analysis of ROUGE-1, ROUGE-L, and BLEU scores for both the models. These metrics assess how closely the model's generated answers match with the reference answers in terms of wording and sequence.

FLAN-T5-small

- **ROUGE-1 and ROUGE-L:** When fine-tuned on 25% of the data, the model scored 0.44 on ROUGE-1 and 0.43 on ROUGE-L. Upon training on the full dataset, scores slightly improved to 0.46 for ROUGE-1 and 0.45 for ROUGE-L, indicating a slight increase in the overlap of content words.
- **BLEU:** The BLEU scores showed minimal progress, moving from 0.09 when trained on 25% of the data to 0.10 on the full dataset, indicating a slight improvement in the exact matching of word sequences.

DistilGPT2

- **ROUGE-1 and ROUGE-L:** DistilGPT2 exhibited a similar pattern with a ROUGE-1 score of 0.43 and ROUGE-L score of 0.41 when fine-tuned on 25% data, and improving to 0.48 for ROUGE-1 and 0.46 for ROUGE-L with full dataset training, reflecting a better replication of content words.
- **BLEU:** The improvement in BLEU score was more notable, starting at 0.12 on 25% training data and increasing to 0.15 on full training data, indicating better accuracy in duplicating the exact word sequences.

These results underline the difficulty of achieving high BLEU scores with the best results still below 0.2, indicating the challenge of exact word-to-word matching sequences.

TABLE 4.2: ROUGE-1, ROUGE-L and BLEU scores for FLAN-T5-small and Distil-GPT2 Models Fine-Tuned on 25% and 100% of Dataset

	Fine Tuned Model on 25% Data			**Fine Tuned Model on 100% Data**		
Model/Metrics	**ROUGE-1**	**ROUGE-L**	**BLEU**	**ROUGE-1**	**ROUGE-L**	**BLEU**
Flan-T5-small	0.44	0.43	0.09	0.46	0.45	0.1
DistilGPT2	0.43	0.41	0.12	0.48	0.46	0.15

4.3.2 Similarity Metrics

4.3.2.1 Cosine Similarity and Sentence Mover's Similarity (SMS)

The table 4.3 outlines the results for Cosine Similarity and Sentence Mover's Similarity (SMS). These metrics assess how closely the responses generated by the models align semantically with the actual answers, reflecting each model's ability to grasp and produce the intended meanings.

FLAN-T5-small

- **Cosine Similarity**: Achieved scores of 0.68 and 0.71, showing moderate enhancement as the training data increased.
- **Sentence Mover's Similarity (SMS)**: Improved from 0.73 to 0.75, indicating better comprehension and alignment.

DistilGPT2

- **Cosine Similarity**: Scored 0.74 and 0.77, demonstrating notable improvement in semantic correlations.
- **Sentence Mover's Similarity (SMS)**: Resulted in an increase from 0.60 to 0.63, reflecting weaker performance in managing sentence-level semantic structures compared to FLAN-T5-small.

TABLE 4.3: Cosine Similarity and Sentence Mover's Similarity (SMS) for FLAN-T5-small and DistilGPT2 Models Fine-Tuned on 25% and 100% of Dataset

Model/Metrics	Fine Tuned Model on 25% Data		Fine Tuned Model on 100% Data	
	Cosine Similarity	SMS	Cosine Similarity	SMS
Flan-T5-small	0.68	0.73	0.71	0.75
DistilGPT2	0.74	0.60	0.77	0.63

4.3.3 Operational Efficiency Metrics

4.3.3.1 Latency and Memory Usage

The table 4.4 details latency and memory usage for the FLAN-T5-small and DistilGPT2 models during their evaluation phase on test data. This comparison highlights the performance of the fine-tuned models on different dataset sizes in terms of processing speed and resource consumption, offering insights into their operational efficiency.

FLAN-T5-small

- **Latency**: Remained fairly consistent, with times of 422 seconds for the fine-tuned model on 25% data and a slight increase to 442 seconds for full data.
- **Memory Usage**: Showed minimal increase from 8533 MB when tuned on partial data to 8584 MB when trained on the entire dataset.

DistilGPT2

- **Latency**: Marked a significant improvement in processing speed, reducing from 400 seconds on partial data to 148 seconds when trained on the full dataset.
- **Memory Usage**: Despite a slight reduction in memory consumption from 23378 MB to 23042 MB, it suggested substantial computational resources were still required for the decoder-only model.

4.3.3.2 Inference Time

The table 4.5 details the average inference time per single question for the FLAN-T5-small and DistilGPT2 models when generating 5 random answers from the test set. This comparison highlights

TABLE 4.4: Latency and Memory Usage for FLAN-T5-small and DistilGPT2 Models Fine-Tuned on 25% and 100% of Dataset

Model/Metrics	Fine Tuned Model on 25% Data		Fine Tuned Model on 100% Data	
	Latency (s)	Memory Usage (mb)	Latency (s)	Memory Usage (mb)
Flan-T5-small	422	8533	442	8584
DistilGPT2	400	23378	148	23042

the performance of the fine-tuned models on different dataset sizes in terms of inference speed, providing insights into their operational efficiency during real-time predictions.

FLAN-T5-small

- **Inference Time**: The average inference time per question showed a slight decrease from 0.51 seconds for the model fine-tuned on 25% of the data to 0.42 seconds for the model fine-tuned on 100% of the data, indicating a marginal improvement in response time with the full dataset.

DistilGPT2

- **Inference Time**: The average inference time per question remained relatively stable, with 0.21 seconds for the model fine-tuned on 25% of the data and 0.22 seconds for the model fine-tuned on 100% of the data, demonstrating consistent performance regardless of the dataset size.

TABLE 4.5: Inference Time for FLAN-T5-small and DistilGPT2 Models Fine-Tuned on 25% and 100% of Dataset

Model/Inference	Inference Time (s)	
	25% Data	100% Data
Flan-T5-small	0.51	0.42
DistilGPT2	0.21	0.22

4.3.4 Human Evaluation

Results from human evaluation were most significant compared to the other evaluation metrics for our task. They provided a direct measure of each model's performance in producing contextually accurate answers - a critical aspect of our task that other automated metrics could not fully address. The results from both the models are presented in the following subsections.

4.3.4.1 Comparison of Answer Correctness

The table 4.6 shows the human evaluation results for FLAN-T5-small and DistilGPT2 models fine-tuned on 25% and 100% data sets. It details the number of incorrect, partially correct, and correct responses for each model and training dataset size, providing insights into their performance in generating accurate answers.

FLAN-T5-small:

- **Fine-tuned on 25% of data:** The model recorded 10 fully correct responses, 90 partially correct, and 596 incorrect.
- **Fine-tuned on the full dataset:** The performance improved, with 19 fully correct responses, 151 partially correct, and a decrease in incorrect responses to 526.

DistilGPT2:

- **Fine-tuned on 25% of data:** The model started with 8 fully correct responses and 61 partially correct.
- **Fine-tuned on the full dataset:** The numbers modestly increased to 14 fully correct and 80 partially correct, with incorrect answers decreasing to 602.

TABLE 4.6: Human Evaluation Results for FLAN-T5-small and DistilGPT2 Models Fine-Tuned on 25% and 100% of Dataset

	Fine Tuned Model on 25% Data			**Fine Tuned Model on 100% Data**		
Model/Metrics	**Incorrect**	**Partially Correct**	**Correct**	**Incorrect**	**Partially Correct**	**Correct**
Flan-T5-small	596	90	10	526	151	19
DistilGPT2	627	61	8	602	80	14

4.3.4.2 Accuracy

Accuracy was calculated based on the answer correctness figures, as defined in Chapter 3. This metric provides a quantitative assessment of each model's performance in generating correct answers. The table 4.7 illustrates the findings in detail.

FLAN-T5-small:

- Fine-tuned on 25% of data: Achieved an accuracy of 7.9%.
- Fine-tuned on the full dataset: Showed a notable improvement with accuracy rising to 13.57%.

DistilGPT2:

- Fine-tuned on 25% of data: Achieved an accuracy of 5.53%.
- Fine-tuned on the full dataset: Displayed a moderate increase in accuracy to 7.75%.

TABLE 4.7: Accuracy for FLAN-T5-small and DistilGPT2 Models Fine-Tuned on 25% and 100% of Dataset

Model/Data	**Dataset (25%)**	**Dataset (100%)**
Flan-T5-small	7.9	13.57
DistilGPT2	5.53	7.75

The Figure 4.9 below visually compares the performance of both models across different training dataset sizes. This highlights that increasing the amount of training data leads to improved accuracy for both models. Notably, the FLAN-T5-small model consistently outperforms DistilGPT2, demonstrating a greater capability in generating correct answers.

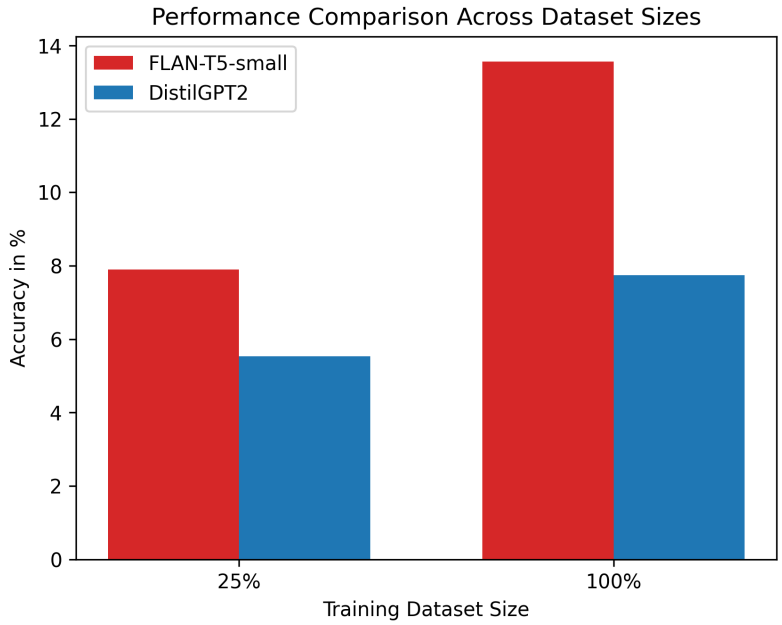


FIGURE 4.9: Visual Representation of Accuracy for FLAN-T5-small and DistilGPT2 Models Fine-Tuned on 25% and 100% of Dataset

CHAPTER 5

DISCUSSION

This chapter provides a summary of the project's key findings and offers an in-depth interpretation of the results obtained from comparing two different transformer architectures. Additionally, it also mentions the contributions and limitations of the project and proposes future research directions to explore this field further.

5.1 Summary of Key Findings

The analysis conducted in this study aimed to compare different architectures of transformer models and their effectiveness for the task in hand, closed-book generative question-answering. The encoder-decoder model (FLAN-T5-small) and decoder-only model (DistilGPT2) were fine-tuned on question-answer pairs from a domain specific dataset of NVIDIA documentation and were compared using a comprehensive set of evaluation metrics to assess their performance. However, it was noted that all the metrics employed were not equally effective for the comparison of these models. Only two evaluation metrics, human evaluation and SMS (Sentence Mover's Similarity), provided significant insights into the practical effectiveness of each model in generating relevant and accurate responses.

The analysis identified that the model with an encoder-decoder architecture could learn more effectively from sole pairs of question answer pairs compared to the model with decoder-only. The metrics suggested that the answers generated by the encoder-decoder model were more accurate and closer to the actual answers. Specifically, FLAN-T5-small exhibited a more nuanced understanding of complex queries, a critical advantage for practical applications in corporate settings where precise information retrieval is essential.

5.2 Implications of the Findings

After fine-tuning the selected models, both models demonstrated robust learning and generalisation capabilities which can be observed in the Figures 4.1 and 4.2. A notable difference for training loss between the two models was that while the FLAN-T5-small showed a steady decrease, DistilGPT2 showed a drastic drop. This drastic drop could be attributed to several factors, primarily centering on differences in model architectures and additional training arguments. As a distilled version of the larger GPT-2 model, DistilGPT2 may have inherited pre-optimised traits that facilitate faster convergence, especially in the initial stages of training. Additionally, the lower values recorded in the metrics for DistilGPT2 were due to the use of specific training arguments such as *gradient_accumulation_steps* and *eval_accumulation_steps*, which were essential for managing its high memory demands. However, despite these differences, the overall trend for both models was observed to be somewhat similar, with each converging towards the end epochs. Subsequently, the validation loss for both models was found to be reducing uniformly. This suggested that, regardless of the initial variations in loss reduction rates, both models eventually stabilised and demonstrated comparable generalisation capabilities as training progressed. These graphs played a pivotal role in mitigating the risk of over-fitting or under-fitting the models and also offered a valuable framework for interpreting the training and validation losses. This allowed for a thorough comparison of the model's performances, providing insights into their learning dynamics and efficiency throughout the training process.

The validation graphs, which analysed the evaluation metrics over five epochs on the validation dataset, showed that most metrics stabilised by the final epoch for both the models (Section 4.2). This stabilisation ensured that the learning process was effective and robust, confirming that both models were trained adequately. Notably, it was observed that for DistilGPT2, the epochs average execution time was approximately 3 times faster compared to FLAN-T5-small model but it also required 3 times more memory for training. This highlighted the trade-offs between computational speed and resource demands for both the models. However, later when the average inference time for these models was calculated, it was observed that the difference was not significant in real time. This suggests that despite the disparities during training, both models are capable of delivering prompt responses in real-world applications, maintaining efficiency across different operational contexts.

The metrics used to evaluate the models were categorised into four distinct groups to assess how different architectural designs performed on the test set. This approach was adopted due to the absence of standardised evaluation metrics in existing literature (Table 2.1). By establishing a clear set of metrics,

this project aims to provide a benchmark that future studies can use to consistently measure and compare the performance of various model architectures.

For lexical-based metrics, the Precision, Recall, F1-Score, ROUGE-1, ROUGE-L and BLEU scores were calculated. These metrics were included for their widespread use in the field for evaluating the performances of these models (Table 2.1) (Hsu et al., 2021; Le et al., 2023; Li et al., 2020; Su et al., 2019; Yagnik et al., 2024; Wang et al., 2021). Exact Match, a common metric for evaluation question-answering tasks, that compares the predicted answer to the reference answers and returns a boolean output either 1 and 0, was excluded from this study The reason for this was because the task at hand is generative, and a model in such a scenario is not expected to produce the exact same outputs but to generate responses based on what it has learned. If a model were to generate identical data to what it was trained on, it could indicate a problem, such as over-fitting, where the model is too closely fitted to the training data and may not perform well on new, unseen data. The values shown in table 4.1 and 4.2 indicate that DistilGPT2 generally outperformed FLAN-T5-small for lexical-based metrics. This implied that DistilGPT2 was more effective at generating answers that closely match the expected phrases and key terminology, showing a higher degree of word overlap with the actual reference answers compared to FLAN-T5-small. Furthermore, an improvement was observed in scores from 25% to 100% of the data, highlighting the importance of size of the dataset in training. It showed that with more data, the models can learn more comprehensive representations and thus, perform better in terms of linguistic alignment with reference texts. However, these metrics did not take correctness or meaning of the answers into consideration, outlining a major loophole in the evaluation. Thus, semantic similarity metrics were established that not only rely on the words but also the semantic meanings behind the generated answers (Risch et al., 2021).

In this study, similarity metrics such as cosine similarity and sentence mover's similarity were employed because they offer a more nuanced evaluation than simple word-to-word comparisons (Chen et al., 2019; Risch et al., 2021). These metrics extend beyond basic lexical comparisons by generating embeddings and undertaking complex calculations, such as determining angles between embedding vectors and computing cost matrices. However, despite their ability to provide a deeper understanding of semantic relationships between generated and reference texts, these metrics are not widely used in the field for evaluating question-answering tasks. The primary reason for their limited adoption could be because of their high computational demands since they require generation of vector embeddings and sophisticated calculations, which can be a significant barrier, especially in larger datasets or real-time

applications. The cosine similarity results indicated that DistilGPT2 achieved higher scores, suggesting that the vectors representing its generated answers were more closely aligned with those of the target answers. This alignment demonstrates a closer semantic relationship, indicating that DistilGPT2's responses are more in line with the expected semantic orientation of the original answers. However, FLAN-T5-small scored higher in SMS compared to DistilGPT2 (Table 4.3) indicating that it retained the overall sentence structure and meaning more effectively than DistilGPT2. This highlighted that FLAN-T5-small is more suitable for question answering tasks requiring a deep understanding of context, such as complex query answering where the relation between different parts of a sentence plays a crucial role. It was observed that on scaling the training data, the metrics improved, further outlining the importance of training data size.

Operational metrics - not widely used in the field to compare models - were assessed and it was observed that latency for both models was identical during the evaluation phase when trained on 25% of the data. Although DistilGPT2's latency improved dramatically when it was trained on 100%, it required almost 3 times the memory compared to that of FLAN-T5-small. This emphasises the trade-off between time and resources. The latency for inference time for both models was recorded to be less than half a second, implying their relatively fast essence for real word application.

Human evaluation, which is often regarded as the most reliable and important metric for question-answering tasks, was employed in this study due to the task's complexity and the frequent use of human judgement in scenarios where automated metrics cannot accurately determine answer correctness. Human evaluation provided a direct measure of the model's performance from a user's perspective. From the results of human evaluation, we observed that FLAN-T5-small delivered more fully correct answers and fewer incorrect answers when fine-tuned on the full dataset compared to DistilGPT2 (Table 4.6). This suggested that while DistilGPT2 may be faster and more lexically accurate, FLAN-T5-small provided with responses that are contextually more reliable, which makes it preferable for tasks where answer correctness is important. This highlighted that the encoder-decoder model was better suited for this task with its accuracy being almost twice as that of DistilGPT2 (Table 4.7 and Figure 4.9) .

The accuracy, even for our best performing model, was only 13.57%. This low performance is primarily due to the small size of models employed in the study, a result of the limited computational resources available for this research. To ensure that this did not affect our comparison, both the selected models had roughly the same amount of model parameters (77 million for FLAN-T5-small and 88

million for DistilGPT2). This issue is further discussed in the limitations section of the project along with future research directions to be explored.

Thus, a comprehensive analysis across various evaluation metrics and conditions showed that while DistilGPT2 excelled in processing speed and lexical accuracy, FLAN-T5-small offered superior performance in terms of delivering contextually appropriate responses and maintaining semantic integrity. Based on the results, FLAN-T5-small, an encoder-decoder model, was deemed more effective overall for closed-book generative question-answering task.

5.3 Contributions

The major contributions of this thesis are:

- This thesis provides a comprehensive comparative analysis of two distinct transformer architectures - encoder-decoder models and decoder-only models - and their ability to handle closed-book QA tasks. This analysis helps to understand performance of the models despite their architectural differences.
- This thesis suggests a customisation phase for the dataset where the original answers are condensed using the advanced capabilities of large language models like ChatGPT-4 to support the generation of concise answers and ensuring that only essential information is included in the training process.
- This thesis demonstrates the impact of training data volume on model performance, offering insights into how quantity of training data can affect the outcomes in closed-book QA tasks.
- This thesis also examines the model performance using various evaluation metrics, ranging from lexical, similarity, and operational efficiency metrics. It further highlights the drawbacks of existing evaluation metrics such as F1, ROUGE, etc., and suggests better and more efficient metrics to evaluate their performance.

5.4 Limitations

A major limitation of the study is the use of small and distilled versions of the models due to limited resources. As the study focuses and compares only these two models, it does not entirely capture the

capabilities or limitations inherent in these transformer architectures. Additionally, the study was conducted on only one dataset, which imposes dataset specificity constraints and limits the generalisability of the findings. Furthermore, results in terms of operational efficiency - latency and memory usage - are heavily dependent on the hardware used for such experiments. Thus, different results might be observed with better hardware configurations, having an impact on the scalability of this research project.

5.5 Future Research Directions

Future research should focus on incorporating larger versions of these models to ensure better accuracy and further enhance existing customer support frameworks. Testing the models on datasets from diverse domains will help to understand the adaptability and robustness of these models across different domains and whether they can handle different types of data efficiently. Another area for future research will be to apply different techniques such as instruction tuning that aligns the model's responses more closely with the desired task-specific instructions and thus, examine the effect it has on performance for the task performed above. Additionally, future research should focus on developing new evaluation metrics that not only compare generated answers based on word overlap but also assess their correctness. These innovative metrics could represent a breakthrough for the field, providing a more comprehensive and accurate measure of model performance. Lastly, research should be carried out to gauge the potential of these models to engage in state-of-the-art learning systems such as interactive learning and incremental learning where the models can adapt based on feedback or new data and in turn increase the practical utility of such dynamic environments.

CHAPTER 6

CONCLUSION

This study conducted an in-depth investigation into closed-book generative question-answering systems, aiming to determine the most effective transformer architecture for this task. The study rigorously compared the performance of two models, FLAN-T5-small and DistilGPT2, trained on the NvidiaDocsQAPairs dataset. The findings indicate that the FLAN-T5-small, an encoder-decoder model, surpasses the decoder-only DistilGPT2 model in terms of generating contextually relevant responses and maintaining semantic integrity, albeit with marginally slower response times. These capabilities can be leveraged in corporate environments to develop tailored, AI-driven solutions that enhance user interactions.

However, the research also revealed limitations due to the use of distilled and smaller model versions, which may not fully capture the capabilities of more comprehensive models. Furthermore, the study's reliance on a single dataset could limit the generalisability of the results.

Looking forward, the study suggests several directions for future research. Testing larger model versions and employing multiple datasets could provide deeper insights into the robustness and adaptability of these architectures. Establishing new evaluation metrics that go beyond lexical similarity could enhance the assessment of responses for correctness and relevance, pushing the field forward. Moreover, incorporating adaptive learning techniques could allow these models to refine their answers based on feedback, enhancing their practical utility in dynamic real-world applications.

This thesis makes several significant contributions, including a detailed comparative analysis of two transformer architectures and their efficacy in a closed-book QA setting. It underscores the importance of dataset customisation for training efficiency and highlights the critical role of training data volume in influencing model performance. By pushing the boundaries on evaluation metrics and suggesting future research pathways, this work serves as a foundational reference for enhancing closed-book question-answering systems.

Bibliography

T. Aarsen, N. Reimers, and O. Espejel. 2021a. Sentence-transformers: all-mpnet-base-v2. Hugging Face.

T. Aarsen, J. Wigton, N. Reimers, G. Becquin, O. Espejel, and J. Gante. 2021b. Sentence-transformers: all-minilm-l6-v2. Hugging Face.

Rishiraj Acharya. 2023. Fine-tuning llms: Supervised fine-tuning and reward modelling. Community Article.

Anaconda Software Distribution. 2016. Conda. [Computer software].

Ajay Bandi, Pydi Venkata Satya Ramesh Adapa, and Yudu Eswar Vinay Pratap Kumar Kuchi. 2023. The power of generative ai: A review of requirements, models, input-output formats, evaluation metrics, and challenges. *Future Internet*, 15:260.

Kai Bird. 2021. *American Prometheus, Chapter-1*. Atlantic Books.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *ArXiv*, abs/2005.14165.

Jason Brownlee. 2020. Impact of dataset size on deep learning model skill and performance estimates. *Machine Learning Mastery*.

Guendalina Caldarini. 2023. Bert vs. gpt-3: Comparing two powerhouse language models. *Towards NLP*.

Anthony Chen, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Evaluating question answering evaluation. In *Conference on Empirical Methods in Natural Language Processing*.

Jeffrey Cheng, Marc Marone, Orion Weller, Dawn Lawrie, Daniel Khashabi, and Benjamin Van Durme. 2024. Dated data: Tracing knowledge cutoffs in large language models. *ArXiv*, abs/2403.12958.

Hyung Won Chung, Le Hou, S. Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Wei Yu, Vincent Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed Huai hsin Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models. *ArXiv*, abs/2210.11416.

DatabaseCamp. 2023. T5 model. DatabaseCamp ML Blog.

G. Deepesh, S. Bavana, and S. Kalakonda. 2023. Nvidia documentation question and answer pairs.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019a. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019b. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*.

Sanjay K. Dwivedi and Vaishali Singh. 2013. Research and reviews in question answering system. *Procedia Technology*, 10:417–424. First International Conference on Computational Intelligence: Modeling Techniques and Applications (CIMTA) 2013.

David A. Ferrucci, Eric W. Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John M. Prager, Nico Schlaefer, and Chris Welty. 2010. Building watson: An overview of the deepqa project. *AI Mag.*, 31:59–79.

Ge Gao, Hung-Ting Chen, Yoav Artzi, and Eunsol Choi. 2023. Continually improving extractive qa via human feedback. *ArXiv*, abs/2305.12473.

Sia Gholami and Mehdi Noori. 2021. Zero-shot open-book question answering. *ArXiv*, abs/2111.11520.

Chao-Chun Hsu, Eric Lind, Luca Soldaini, and Alessandro Moschitti. 2021. Answer generation for retrieval-based question answering systems. In *Findings*.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *ArXiv*, abs/1705.03551.

Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. 2015. From word embeddings to document distances. In *International Conference on Machine Learning*.

Hung Le, Le Minh Nguyen, Jiaying Ni, and Shogo Okada. 2023. Constructing a closed-domain question answering system with generative language models. *2023 15th International Conference on Knowledge and Systems Engineering (KSE)*, pages 1–6.

Seungjun Lee, Jungseob Lee, Hyeonseok Moon, Chanjun Park, Jaehyung Seo, Sugyeong Eo, Seonmin Koo, and Heu-Jeoung Lim. 2023. A survey on evaluation metrics for machine translation. *Mathematics*.

Zhongli Li, Wenhui Wang, Li Dong, Furu Wei, and Ke Xu. 2020. Harvesting and refining question-answer pairs for unsupervised qa. In *Annual Meeting of the Association for Computational Linguistics*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Annual Meeting of the Association for Computational Linguistics*.

Titian Lish. 2023. The first man. *The Eugene O'Neill Review*, 44(2):237–241. Review of the production directed by Eric Fraisher Hayes, Museum of San Ramon Valley, Danville, CA, January 13-15, 2023.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. *ArXiv*, cs.CL/0205028.

Gary Marcus and Ernest Davis. 2020. Gpt-3, bloviator: Openai's language generator has no idea what it's talking about. *MIT Technology Review*.

Phil Miesle. 2023. What is cosine similarity: A comprehensive guide.

Peter Adebowale Olujimi and Abejide Ade-Ibijola. 2023. Nlp techniques for automating responses to customer queries: a systematic review. *Discover Artificial Intelligence*, 3.

OpenAI. 2023. Chatgpt (march 14 version). Large language model.

The pandas development team. 2020. pandas-dev/pandas: Pandas.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Annual Meeting of the Association for Computational Linguistics*.

Ren'e Peinl and Johannes Wirth. 2023. Evaluation of medium-large language models at zero-shot closed book generative question answering. *ArXiv*, abs/2305.11991.

Saurabh Pimpalkar. 2023. The rise of generative ai: Transforming question answering systems.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Conference on Empirical Methods in Natural Language Processing*.

Julian Risch, Timo Moller, Julian Gutsch, and Malte Pietsch. 2021. Semantic answer similarity for evaluating question answering models. *ArXiv*, abs/2108.06130.

Adam Roberts, Colin Raffel, and Noam M. Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Conference on Empirical Methods in Natural Language Processing*.

Raz Rotenberg. 2020. What is gradient accumulation in deep learning? backpropagation process of neural networks explained.

Ranjan Satapathy. 2018. Question answering in natural language processing [part-i].

Marco Antônio Calijorne Soares and Fernando Silva Parreiras. 2020. A literature review on question answering techniques, paradigms and systems. *J. King Saud Univ. Comput. Inf. Sci.*, 32:635–646.

Dan Su, Mostofa Patwary, Shrimai Prabhumoye, Peng Xu, Ryan J. Prenger, Mohammad Shoeybi, Pascale Fung, Anima Anandkumar, and Bryan Catanzaro. 2022. Context generation improves open domain question answering. In *Findings*.

Dan Su, Yan Xu, Genta Indra Winata, Peng Xu, Hyeon-Jin Kim, Zihan Liu, and Pascale Fung. 2019. Generalizing question answering system with pre-trained language model fine-tuning. In *Conference on Empirical Methods in Natural Language Processing*.

Ki Hyun Tae, Yuji Roh, Young H. Oh, Hyunsub Kim, and Steven Euijong Whang. 2019. Data cleaning for accurate, fair, and robust models: A big data - ai integration approach. *Proceedings of the 3rd*

International Workshop on Data Management for End-to-End Machine Learning.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2016. Newsqa: A machine comprehension dataset. In *Rep4NLP@ACL*.

Chris van der Lee, Albert Gatt, Emiel van Miltenburg, and Emiel J. Krahmer. 2021. Human evaluation of automatically generated text: Current trends and best practice guidelines. *Comput. Speech Lang.*, 67:101151.

Vast.ai. n.d. Vast.ai. Accessed: 2024-04-20.

Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Neural Information Processing Systems*.

Cunxiang Wang, Pai Liu, and Yue Zhang. 2021. Can generative pre-trained language models serve as knowledge bases for closed-book qa? *ArXiv*, abs/2106.01561.

Dingmin Wang, Qiuyuan Huang, Matthew Jackson, and Jianfeng Gao. 2024. Retrieve what you need: A mutual learning framework for open-domain question answering. *Transactions of the Association for Computational Linguistics*, 12:247–263.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Niraj Yagnik, Jay Jhaveri, Vivek Sharma, Gabriel Pila, Asma Ben, and Jingbo Shang. 2024. Medlm: Exploring language models for medical question answering systems. *ArXiv*, abs/2401.11389.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy J. Lin. 2019a. End-to-end open-domain question answering with bertserini. In *North American Chapter of the Association for Computational Linguistics*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019b. Xlnet: Generalized autoregressive pretraining for language understanding. In *Neural Information Processing Systems*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing*.

Qinyuan Ye, Belinda Z. Li, Sinong Wang, Benjamin Bolte, Hao Ma, Xiang Ren, Wen tau Yih, and Madian Khabsa. 2020. Studying strategically: Learning to mask for closed-book qa. *ArXiv*, abs/2012.15856.

Xue Ying. 2019. An overview of overfitting and its solutions. *Journal of Physics: Conference Series*, 1168.