A SMART Q&A SYSTEM FOR MODERN TOURISM USING GNNs AND LLMs

A PROJECT REPORT

Submitted by

Aditi Kannan	**(203002004)**
Dhasharadharami Reddy Bommana	**(203002021)**

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

ELECTRONICS AND COMMUNICATION ENGINEERING

Department of

Electronics and Communication Engineering

Sri Sivasubramaniya Nadar College of Engineering

(An Autonomous Institution, Affiliated to Anna University)

Rajiv Gandhi Salai (OMR), Kalavakkam – 603 110

MAY 2024

Sri Sivasubramaniya Nadar College of Engineering

(An Autonomous Institution, Affiliated to Anna University)

BONAFIDE CERTIFICATE

Certified that this project titled "**A SMART Q&A SYSTEM FOR MODERN TOURISM USING GNNs AND LLMs**" is the bonafide work of "**Aditi Kannan** (203002004) and **Dhasharadharami Reddy Bommana** (203002021)", who carried out the project work under my supervision.

Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

[illegible]

SIGNATURE
Dr. P. Vijayalakshmi
HEAD OF THE DEPARTMENT
Professor
Department of Electronics and Communication Engineering
SSN College of Engineering
Kalavakkam – 603110

[illegible] 26/4/24

SIGNATURE
Dr. M. P. Actlin Jeeva
SUPERVISOR
Assistant Professor
Department of Electronics and Communication Engineering
SSN College of Engineering
Kalavakkam – 603110

Submitted for project viva-voce examination held on....02:05:2024..........

EXTERNAL EXAMINER **INTERNAL EXAMINER**

ABSTRACT

This project introduces a novel Smart Question and Answer (Q&A) System for modern tourism in Chennai, India. Our system is designed to provide accurate and informative responses to user queries for 31 tourist attractions in Chennai and other relevant topics such as Chennai culture. Aligned with the growing demand for tailored travel experiences, the system contributes to India's tourism sector growth. A custom dataset comprising 320 questions was specifically curated for Chennai tourism. This dataset was divided into a training set of 270 questions and a test set of 50 questions. Leveraging the power of RoBERTa, a state-of-the-art large language model, we trained our system for 20 epochs on the custom dataset, achieving a final training loss of 0.0178. Subsequently, we evaluated the performance of our model on both the training and test datasets. Our system demonstrated robustness and accuracy, achieving an impressive accuracy rate of 88.125% on the test dataset. This highlights the effectiveness of our approach in providing reliable information to users seeking to explore Chennai.

ACKNOWLEDGEMENT

We would like to extend our sincere gratitude to everyone who gave us the chance to finish this project. We owe a great deal to Sri Sivasubramaniya Nadar College of Engineering for helping us with this project and for providing the resources.

We extend our sincere gratitude to **Dr. P Vijayalakshmi**, Head of the Department, for providing us with the initial project idea, valuable advice and guidance throughout the course of this project.

We would like to take this opportunity to acknowledge our sincere thanks to our guide and mentor **Dr. M. P. Actlin Jeeva**, for her constant support and encouragement towards the project.

We would also like to acknowledge the role of our panel supervisors, **Dr. R. Rajavel**, **Dr. Satyakam Baraha** and **Dr. N Prabagarane** who gave us valuable feedback and insights on ways to improve our project.

Finally, we would like to thank our parents, friends, and members of the technical and academic staff of the department for their ongoing interest in our project.

ADITI KANNAN

DHASHARADHARAMI REDDY BOMMANA

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATION	FULL FORM
GNNs	Graph Neural Networks
LLMs	Large Language Models
NLP	Natural Language Processing
EIC	Exploratory Inference Chain
TDIL	Text Datasets for Indian Languages
KG	Knowledge Graph
KBQA	Knowledge Base Question Answering
IQAGNN	Interpretable Question Answering method based on heterogeneous Graph Neural Networks
CQA	Community Question Answering
DPR	Dense Passage Retriever
OpenQA	Open-domain Question Answering
AMGN	Asynchronous Multi-grained Graph Network
GLoVe	Global Vectors
ASR	Automatic Speech Recognition
STT	Speech-to-Text
TTS	Text-to-Speech
BPE	Byte Pair Encoding
LMs	Language Models
HELM	Holistic Evaluation of Language Models

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

Modern tourism is evolving towards personalized and immersive journeys, demanding responsive and intelligent systems to cater to the diverse queries and needs of contemporary travelers. In this dynamic landscape, the demand for innovative technologies has surged. This project introduces a Smart Question and Answer (Q&A) System, leveraging advanced Large Language Models (LLMs) to meet this demand.

LLMs represent a groundbreaking advancement in natural language processing (NLP) technology, equipped with the ability to understand and generate human-like text with unprecedented accuracy and fluency. These models, such as BERT and GPT, have demonstrated remarkable proficiency across a wide range of language understanding tasks, including question answering, text generation, and sentiment analysis. By leveraging the vast amount of pre-existing textual data, LLMs have acquired a rich understanding of linguistic patterns, semantics, and contextual cues, enabling them to provide contextually relevant and accurate responses to user queries.

LLMs serve as linguistic powerhouses capable of understanding and generating human-like text, crucial in addressing the multifaceted nature of travel-related inquiries. The significance of this project lies in its potential to enhance existing technologies in the tourism sector, delivering personalized insights into tourist hotspots and cultural nuances.

With the power of LLMs, the smart Q&A system aims to provide precise responses tailored to individual traveler queries, contributing to the broader goals of the tourism sector, including increased revenue, employment, and GDP growth.

1.2 MOTIVATION

The necessity for implementing the smart question and answer System in modern tourism stems from the evolution of contemporary travel and its integral role in shaping India's economic landscape. With the tourism sector contributing a substantial ₹13.2 lakh crore (US$170 billion) in 2021, equivalent to 5.8% of the nation's GDP and fostering employment for 32.1 million individuals, the significance of this industry cannot be overstated. In response to the changing expectations of today's travelers, who increasingly prioritize personalized and immersive experiences, the project endeavors to address this paradigm shift. In response to the growing demand for personalized and immersive travel experiences, the project endeavors to leverage advanced LLMs to enhance the responsiveness and efficacy of tourism-related information systems.

1.3 OBJECTIVES

The project aims to:

- Develop a smart question and answer system for modern tourism, utilizing LLMs.
- Provide accurate responses to traveler queries, offering personalized information on tourist attractions, and other cultural insights.
- Enhance tourism experiences, and increase revenue and GDP through informed and customized travel interactions.

1.4 MAJOR CONTRIBUTIONS

This project introduces several key advancements in the area of question answering for Chennai tourism:

- **Domain-Specific Dataset Creation:** A novel dataset specifically tailored for question answering tasks related to Chennai tourism was created. This dataset likely includes questions and corresponding answers relevant to tourist interests in the city. The project also employs the powerful RoBERTa pre-trained language model for question answering. RoBERTa's ability to understand complex relationships between words makes it well-suited for this task. By utilizing this model, the QnA system achieves a high level of comprehension, leading to more accurate and informative answers.
- **Optimized Context Selection with Sentence Transformers:** The project implements Sentence Transformers to optimize the selection of relevant context for answering user questions. This likely involves leveraging sentence embeddings to identify the portion of the tourism dataset that best aligns with the user's query.
- **Integration of ASR system for better user experience:** To further enhance user experience and cater to a wider range of users, the project integrates an Automatic Speech Recognition (ASR) system. The ASR system transcribes spoken queries into text, which are then processed by the RoBERTa model to retrieve the most relevant answer from the Chennai tourism dataset. This voice-based interaction provides a more natural and user-friendly experience, particularly for users who may be unfamiliar with typing or who prefer hands-free interaction.

1.5 ROADMAP

The report highlights the following sections. In chapter 2, it provides a comprehensive literature review that serves as the foundation for informed decision-making and aids in identifying gaps, trends, and areas of innovation within the chosen field. The subsequent chapter, chapter 3, Exploration and Analysis of Data, provides a succinct overview on the type of data available that can potentially be used to train the system to be built. This discussion culminates in the reasons for choosing RoBERTa model and need for a curated dataset for the same.

Chapter 4 provides details on the meticulous curation of the dataset needed to train and test the system for the problem statement and the LLM chosen. Chapter 5 elaborates on the mechanics and performance of LLMs, focusing on their architecture, training methodologies, and capabilities in handling natural language understanding tasks. This discussion encompasses an exploration of state-of-the-art LLMs such as RoBERTa, highlighting its strengths, limitations, and potential applications in the context of tourism-related question answering systems. Additionally, this chapter also includes an evaluation of the performance of RoBERTa on the curated text corpus, providing insights into the effectiveness in generating accurate and contextually relevant responses to traveler queries. Chapter 6 focuses on building the ASR system to enhance user experience.

Finally, the project culminates in Chapter 7, which provides a summary of the results obtained through the application of LLMs in the development of the smart Q&A system. Additionally, insights obtained from these findings will be used to outline potential avenues for future research.

CHAPTER 2

LITERATURE REVIEW

2.1 INTRODUCTION

In the literature review section, we delve into the existing body of research work that forms the foundation of our project. We did analysis and examined relevant research to contextualize our study. By realizing key findings and identifying gaps in the current knowledge, we aim to establish a strong theoretical framework for our project and demonstrate its significance in advancing the field. This critical review can also serves as a bridge between existing research and the innovative contributions our project seeks to make.

2.2 LITERATURE REVIEW ON GNNS

L. Wu at al., (2021) provides comprehensive overview of the utilization of Graph Neural Networks (GNNs) in the domain of Natural Language Processing (NLP). It highlights the dominant role of deep learning in NLP tasks, emphasizing the shift from traditional token-based representations to graph structures. The survey introduces a new taxonomy for GNNs in NLP, categorizing research along axes of graph construction, graph representation learning, and graph-based encoder-decoder models. It discusses the surge of interest in developing GNNs for various NLP tasks, presenting applications such as sentence classification, semantic role labeling, relation extraction, machine translation, question generation, and summarization. The text also addresses challenges in utilizing GNNs for NLP, including automatic transformation of text sequences into graph-structured data and effective modeling of complex data.

In addition to the broader context of GNNs in NLP, the survey elaborates on their application in Knowledge Base Question Answering (KBQA). The paper specifically mentions the challenge of automatically transforming original text sequence data into highly graph-structured data and the importance of determining graph representation learning techniques. Additionally, it emphasizes the need for effective modeling of complex data, particularly in NLP tasks that involve mapping between graph-based inputs and highly structured output data such as sequences, trees, and graphs with multi-types in both nodes and edges. The survey contributes valuable insights into the intersection of GNNs and NLP, offering a detailed exploration of their applications in KBQA and shedding light on methodologies, challenges, and benchmarks in this domain.

Y. Wu et al., (2022) introduces an innovative Interpretable Question Answering method based on heterogeneous Graph Neural Networks (IQAGNN) for Community Question Answering (CQA). Addressing challenges in effective feature fusion and answer interpretability, IQAGNN constructs a tailored heterogeneous information network for CQA, facilitating the fusion of relationships among multi-type entities within communities. Leveraging a heterogeneous graph neural network, IQAGNN derives embeddings for these entities, enhancing its ability to capture nuanced relationships. The method predicts answers based on entity correlation and provides interpretation analyses using meta-paths, aiming to deliver both accurate and interpretable explanations.

Empirical evaluations on three real datasets demonstrate IQAGNN's superiority in interpretive question-answering research, emphasizing the significance of incorporating heterogeneous graph structures and neural networks in enhancing the interpretability of community-based question-answering systems. The paper suggests future directions in applying interpretability research to broader social mining tasks, extending IQAGNN's contributions to knowledge sharing within online communities.

The integration of pre-trained language models (LMs) and structured knowledge graphs (KGs) introduced by **Yasunaga et al., (2021)** has significantly shaped the landscape of question answering (QA). In the early stages, QA systems often relied on either LMs for contextual information or KGs for structured knowledge. Recent research has recognized the potential synergy of these two distinct sources, leading to comprehensive models that leverage both LMs and KGs for a more nuanced understanding of QA contexts. One of the key challenges addressed in the literature involves the fusion of information from LMs and KGs. QA-GNN introduces a novel approach through relevance scoring, aiming to identify and emphasize pertinent knowledge within KGs based on a given QA context. This strategy helps mitigate noise in KGs, enhancing the precision of QA systems by focusing on relevant information. Joint reasoning, facilitated by methods like GNNs, is identified as a pivotal strategy for integrating information from LMs and KGs. QA-GNN employs GNNs to enable simultaneous updates to representations, fostering a holistic approach to QA by jointly considering both textual and structured knowledge.

The contributions of QA-GNN are validated through a combination of quantitative and qualitative analyses. This aligns with a broader trend in QA research, where comprehensive evaluations provide a holistic perspective on a model's performance and capabilities. This dual approach to analysis ensures a thorough assessment of QA-GNN's effectiveness in handling diverse aspects of question answering tasks.

In the realm of Open-domain Question Answering (OpenQA), the synergy between robust passage retrieval and precise reading comprehension is crucial. **R. Li et al., (2022)** investigate the collaborative potential of Dense Passage Retriever (DPR) and Multi-hop Dense Retriever as efficient retrieval mechanisms alongside GNN-based readers for enhanced OpenQA. Introducing a novel training strategy, the framework leverages scores from dense retrievers and GNN-based readers as correction weights, mutually promoting their strengths. The Asynchronous Multi-grained Graph Network (AMGN) is extended to accommodate passage nodes and passage-level relationships, overcoming fixed-hop constraints through a Recurrent Neural Network-based question reformulation mechanism. Evaluation on established datasets such as Natural Questions, TriviaQA, and HotpotQA demonstrates competitive results compared to existing models. This collaborative framework showcases the potential for improved OpenQA by combining the strengths of dense retrievers and GNN-based readers, suggesting promising directions for future research in retrieval strategies, graph-based models, and applicability to diverse datasets and domains.

In a thorough survey, **Wu et al. (2021)** explore the evolving role of Graph Neural Networks (GNNs) in Natural Language Processing (NLP), introducing a taxonomy for GNNs. They discuss the surge of interest in GNNs for various NLP tasks, highlighting challenges in working with graph-structured data. Wu et al. (2022) present IQAGNN, demonstrating its superiority in interpretive question-answering for CQA. Yasunaga et al. (2021) discuss the integration of pre-trained language models and knowledge graphs, introducing QA-GNN. Li et al. (2022) investigate the collaborative potential of DPR and GNN-based readers in OpenQA, showcasing competitive results. Haji et al. (2023) introduce the Exploratory Inference Chain framework, showing its superiority in multi-hop question answering. These diverse approaches reflect the multifaceted landscape of current research in enhancing question answering systems.

2.3 LITERATURE REVIEW ON LLMS

S. Haji et al., (2023) talk about The Exploratory Inference Chain (EIC) framework and how it enhances question answering by combining implicit language model processing with explicit inference chains. Experimental results validate EIC's superiority over existing approaches in multi-hop QA, achieving more accurate and knowledge-rich answers. The EIC framework generates information needed to answer a multi-hop question using keywords and then performs 1-hop inference for each keyword. If the inference is not sufficient, additional inferences are made. The final answer is an aggregation of all the references made and this can help produce a very relevant answer to the query given. This kind of ability will be helpful for applications like in tourism as there may be multiple sources to investigate before coming to a logical and definitive conclusion.

Vaswani. A at al. (2017), proposed the architecture of the fundamental unit that is used in every LLM today called the Transformer, which revolutionized machine translation and other sequence-to-sequence tasks. Traditionally, these tasks relied on recurrent neural networks (RNNs) that struggled with capturing long-range dependencies in sequences. The paper introduces two key attention mechanisms, Scaled Dot-Product Attention and Multi-Head Attention. While the paper delves into both concepts, it places significant emphasis on multi-head attention as it offers a more powerful and versatile approach compared to using scaled dot-product attention alone.

The Multi-Head Attention builds upon the idea of scaled dot-product attention but uses multiple "heads" in parallel. Each head learns different ways to attend to the input, allowing the model to capture diverse aspects of the relationship between elements in the sequence. This mechanism thus allows the model to dynamically focus on relevant parts of the input sequence when processing information and generating the output. By attending to words far apart in a sentence or prioritizing informative sections within a sequence, the Transformer achieves a deeper contextual understanding.

This approach offers several advantages like, the meaning can be derived from words that are far apart in a sentence and at higher efficiency. Attention allows the model to directly attend to relevant parts no matter how far away they are in the sentence. And not all parts of an input sequence are equally important. Attention allows the model to prioritize informative sections and allocate processing power efficiently. This is useful for long sequences like documents or code. The model can focus on crucial elements without getting distracted by less relevant details. The proposed transformer model differs from the attention model as the transformer also includes positional encoding and good positional encoding will save a lot of compute power.

In the domain of natural language processing (NLP), effectively representing text data is crucial for various tasks. One recent approach achieving impressive results is the Bidirectional Encoder Representations from Transformers (BERT) model, introduced by **Devlin et al. (2018)**. Unlike prior models that focus on unidirectional text processing, BERT leverages a unique pre-training approach based on transformers. This approach trains the model on unlabeled text by considering both the left and right context of each word simultaneously across all layers within the model's architecture. This bidirectional training allows BERT to capture richer and more nuanced relationships between words in a sentence, leading to a deeper understanding of the overall text.

This enhanced understanding offered by BERT translates to significant performance improvements on various NLP tasks. Devlin et al. (2018) demonstrate BERT's effectiveness by applying it to eleven different NLP benchmarks. These tasks encompass question answering, natural language inference, and sentiment analysis. Compared to previous state-of-the-art models, BERT achieves significant improvements on all evaluated tasks. Notably, it pushes the General Language Understanding Evaluation (GLUE) score to 80.5%, representing a 7.7% absolute improvement. Similarly, BERT achieves impressive gains on tasks like MultiNLI accuracy and question answering on the SQuAD datasets. These advancements highlight the power of BERT's bidirectional pre-training approach and its potential to revolutionize various NLP applications.

The introduction of BERT has sparked significant interest within the NLP community. Its ability to achieve state-of-the-art performance on a wide range of tasks with minimal architecture modifications makes it a versatile and powerful tool. As research in this area continues, BERT's pre-trained model and underlying concepts are likely to be further explored and adapted for even more advanced NLP applications.

Wei et al. (2022) address the challenge of enhancing zero-shot learning capabilities in large language models (LLMs). Their work introduces a novel approach called instruction tuning, which involves fine-tuning pre-trained LLMs on a collection of datasets described using natural language instructions. This method aims to improve the model's ability to perform unseen tasks without requiring specific training examples for those tasks.

The authors implemented instruction tuning on a 137B parameter pre-trained LLM, fine-tuning it on over 60 NLP datasets represented by natural language instruction templates. This fine-tuned model, named FLAN (Finetuned Language Models are Zero-Shot Learners), was then evaluated on unseen task types. The results demonstrate significant performance improvements compared to the unmodified pre-trained model. Notably, FLAN outperforms zero-shot GPT-3, a 175B parameter model, on 20 out of 25 evaluated datasets. Furthermore, FLAN surpasses even few-shot GPT-3 by a large margin on tasks like question answering and story completion.

To understand the key factors contributing to the success of instruction tuning, Wei et al. (2022) conducted ablation studies. These studies revealed that the number of datasets used for fine-tuning, the size of the pre-trained model, and the quality of the natural language instructions all play crucial roles in achieving optimal performance. This research suggests that instruction tuning offers a promising approach for enhancing the zero-shot learning abilities of LLMs, enabling them to tackle new tasks without extensive training data.

Nakano et al. (2021) present WebGPT, a novel approach for training large language models (LLMs) to answer long-form questions using a simulated web browsing environment. This method addresses the limitations of traditional LLM training, which often relies on static datasets and may struggle with factual accuracy or the ability to answer complex, open ended questions that require real-world information retrieval.

WebGPT achieves this by fine-tuning a pre-trained LLM, such as GPT-3, within a text-based web browsing environment. The model is trained using imitation learning, where it observes and learns from human demonstrations of how to search and navigate the web to find relevant information for answering questions. This allows WebGPT to develop the ability to access and process external information sources dynamically, enhancing its ability to answer complex questions and improve factual accuracy.

To facilitate human evaluation of factual accuracy, WebGPT is trained to collect references from the web pages it visits to support its answers. This transparency allows human evaluators to assess the validity of the information presented. The authors evaluate WebGPT on the ELI5 dataset, a collection of questions posed by Reddit users. Their best performing model, achieved through a combination of behavior cloning and reward-based optimization, surpasses human demonstrators in answer quality 56% of the time. Additionally, it outperforms the highest-voted answers from Reddit in 69% of cases. These results demonstrate the potential of WebGPT for training LLMs that can access and leverage real-world information to provide comprehensive and accurate answers to complex questions.

Chung et al. (2024) delve deeper into the potential of instruction fine-tuning, a technique that has shown promise in enhancing language model performance and generalizability. This approach involves fine-tuning pre-trained models on a collection of datasets presented as natural language instructions. Their research focuses on three key aspects of scaling instruction fine-tuning:

1. **Scaling the Number of Tasks:** The authors explore the impact of

expanding the number of tasks included in the fine-tuning process. Their findings suggest that a larger number of tasks leads to significant performance improvements.

2. **Scaling the Model Size:** The research investigates the influence of model size on instruction fine-tuning effectiveness. They demonstrate that larger models benefit even more from instruction fine-tuning, achieving even greater performance gains.
3. **Fine-tuning on Chain-of-Thought Data:** The study also examines the effectiveness of fine-tuning on "chain-of-thought" data, which provides insights into the reasoning steps the model takes to arrive at an answer. This approach further enhances performance across various scenarios.

The results presented by Chung et al. (2024) demonstrate that instruction fine-tuning, particularly when implemented with a large number of tasks, larger model sizes, and chain-of-thought data, leads to substantial performance improvements across various language model classes (including PaLM, T5, and U-PaLM), prompting setups (zero-shot, few-shot, and Chain-of-Thought), and evaluation benchmarks (MMLU, BBH, TyDiQA, MGSM, and open-ended generation). For instance, their Flan-PaLM 540B model, fine-tuned on 1.8K tasks, outperforms the baseline PaLM 540B by a significant margin (averaging a 9.4% improvement). Additionally, Flan-PaLM achieves state-of-the-art performance on several benchmarks. The study also contributes by releasing Flan-T5 checkpoints, demonstrating strong few-shot performance even when compared to much larger models. Overall, Chung et al. (2024) solidify the position of instruction fine-tuning as a powerful and versatile approach for enhancing the capabilities and usability of pre-trained language models.

Liang et al. (2022) recognize the growing importance of language models (LMs) across various language technologies. However, a comprehensive understanding of their capabilities, limitations, and potential risks remains elusive. To address this gap, they introduce the Holistic Evaluation of Language Models (HELM) framework, aiming to enhance the transparency and evaluation practices surrounding LLMs.

HELM adopts a multifaceted approach. First, it establishes a taxonomy encompassing a wide range of potential LM use cases (scenarios) and evaluation metrics (desiderata). This taxonomy acknowledges areas that require further exploration, such as question answering for under-represented dialects or trustworthiness metrics. Second, HELM utilizes a multi-metric strategy. It assesses seven key metrics – accuracy, calibration, robustness, fairness, bias, toxicity, and efficiency – across 16 core scenarios for various LM types whenever possible. This ensures a balanced evaluation that goes beyond just accuracy, revealing potential trade-offs between different metrics. Additionally, HELM incorporates seven targeted evaluations focused on specific aspects like reasoning and disinformation analysis, employing 26 targeted scenarios.

Finally, Liang et al. (2022) conduct a large-scale evaluation encompassing 30 prominent LMs, including open-access, limited-access, and closed models. This evaluation covers all 42 scenarios defined by HELM, with 21 of them being entirely new to mainstream LM evaluation. It's worth noting that prior to HELM, the average LM was evaluated on only 17.9% of the core HELM scenarios, with some prominent models lacking any shared evaluation scenarios. HELM significantly improves this landscape, achieving a 96% coverage rate. All 30 models are now benchmarked thoroughly on the same core scenarios and metrics under standardized conditions. This comprehensive evaluation yields 25 key findings, and for complete transparency, the authors release all raw model prompts and completions for public scrutiny, along with a versatile, modular toolkit. HELM is envisioned as a living benchmark, continuously evolving to incorporate new scenarios, metrics, and LM evaluations by the research community.

2.4 SUMMARY

Chapter 2 provides a comprehensive literature review, beginning with an introduction that emphasizes the importance of understanding existing research as the foundation for the project. This literature review meticulously dissects recent breakthroughs in Question Answering (QA) systems, particularly those that harness the power of Graph Neural Networks (GNNs) and Large Language Models (LLMs). These advancements offer valuable knowledge for the construction of a novel QNA system.

GNNs have emerged as a dominant force in the domain of Natural Language Processing (NLP) due to their remarkable ability to capture the intricate relationships between entities within a graph structure. The review meticulously examines how Wu et al. (2021) convincingly demonstrate the effectiveness of GNNs in tasks like question answering. Furthermore, Yasunaga et al. (2021) delve deeper into the domain of Knowledge Base Question Answering (KBQA), showcasing how QA-GNN integrates information gleaned from both LLMs and Knowledge Graphs (KGs) by leveraging GNNs.

This integration leads to superior performance in KBQA tasks. Li et al. (2022) train their sights on Open Domain Question Answering (OpenQA), exploring the collaborative potential of GNN-based readers with retrieval mechanisms. Their work highlights promising avenues for combining the strengths of different approaches, paving the way for the development of more robust QNA systems.

The review also sheds light on the continuous evolution of LLMs and their expanding capabilities in various NLP tasks. The introduction of the groundbreaking Transformer architecture by Vaswani et al. (2017) revolutionized the field of machine translation and other sequence-to-sequence tasks. Building upon this foundation, BERT (Devlin et al., 2018) further propelled LLM performance by leveraging a bidirectional pre-training approach on unlabeled text data. This approach empowers LLMs with a deeper understanding of language. Wei et al. (2022) introduce instruction tuning, a technique that enhances the zero-shot learning capabilities of LLMs by fine-tuning them on natural language instructions.

Additionally, Nakano et al. (2021) present WebGPT, a novel approach for training LLMs to tackle long-form questions by simulating a web browsing environment. Finally, Chung et al. (2024) explore methods for scaling instruction fine-tuning to achieve even greater performance gains.

By gleaning insights from these advancements in GNNs and LLMs, the development of a QNA system can be strategically guided. The system can leverage GNNs to model the relationships within a knowledge base, allowing for a more nuanced understanding of user queries. Furthermore, by incorporating fine-tuned LLMs, the system can harness the latest techniques for handling natural language and generating informative answers. This synergistic approach has the potential to yield a QNA system that is both accurate and versatile, capable of effectively handling complex questions and providing comprehensive responses.

CHAPTER 3

EXPLORATION AND ANALYSIS OF DATA

3.1 INTRODUCTION

In the domain of generative AI models and creating text corpus for the same, the effectiveness of any textual analysis depends upon the proper curation of a comprehensive and well-organized text corpus. This segment outlines the methodologies employed for data acquisition but also gives comments on the significance of rigorous cleaning strategies. By doing such corpus development, we can ensure that the textual data is not only voluminous but also refined to extract meaningful insights.

To do such operations we make use of LLMs and GNNs. Embedding generation using a Large Language Model (LLM) involves leveraging the model's pre-trained knowledge to transform input text into dense, continuous-valued vectors that encapsulate semantic information. In this process, the LLM utilizes its understanding of contextual relationships within language to create embeddings that encode nuanced meanings and relationships between words. By extracting features from the input text, the LLM generates embeddings that capture the underlying semantic structure, allowing for a meaningful representation of the input. These embeddings, often produced through techniques like tokenization and attention mechanisms within the LLM's architecture, can then be used for downstream natural language processing tasks such as sentiment analysis, document classification, or named entity recognition. The power of LLMs lies in their ability to produce contextually rich embeddings, enabling more nuanced and accurate representations of language semantics compared to traditional methods.

Graph Neural Network (GNN) involves capturing intricate relationships and dependencies within structured data, such as graphs. In the context of natural language processing, GNNs can be applied to model semantic connections between words or entities within a document. The GNN processes the graph representation of the textual data, where nodes correspond to words or entities, and edges represent the relationships or co-occurrences between them. Through iterative message-passing mechanisms, the GNN refines embeddings for each node based on the information from its neighbors, effectively encoding contextual relationships. This allows the model to generate embeddings that encapsulate the complex interplay of words or entities within the given text. The resulting embeddings can then be utilized for various tasks, including document classification, information retrieval, or even in the context of knowledge graphs to infer relationships and similarities between entities.

3.2 SAMANANTAR DATASET

Bridging the gap for under-resourced languages like those spoken in India is crucial for inclusive technological development. In this context, the Samanantar dataset by AI4Bharat emerges as a game-changer for researchers working on NMT and multilingual NLP tasks involving Indic languages.

Samanantar boasts the title of the largest publicly available collection of parallel sentences specifically designed for Indic languages. It encompasses a staggering eleven Indic languages, including Hindi, Bengali, Tamil, Telugu, and more. Each sentence in English within the dataset has a corresponding translation in one of these Indic languages, forming millions of sentence pairs. This structure provides the perfect training ground for NMT models, enabling them to learn the intricacies of translating text between English and various Indic languages. It's important to note that the dataset offers varying amounts of data for each language pair. While Hindi-English enjoys the most extensive collection, some language pairs might have slightly fewer sentences. However, the overall size and variety of the dataset cater to a broad range of research needs.

As of April 2024, the latest version offers a remarkable collection of around 49.7 million sentence pairs, constantly expanding the possibilities for research. Furthermore, the dataset is released under a permissive Creative Commons CC0 license, essentially placing it in the public domain.

Feature	**Description**
Dataset Name	Samanantar
Source	AI4Bharat
Focus	Parallel Corpora for Indic Languages
Number of Indic Languages	11
Indic Languages Covered	Assamese (as), Bengali (bn), Gujarati (gu), Hindi (hi), Kannada (kn), Malayalam (ml), Marathi (mr), Odia (or), Punjabi (pa), Tamil (ta) and Telugu (te).
Source Language	English
Data Format	Sentence Pairs
Licensing	Creative Commons CC0
Total Sentence Pairs	49.7 Million
Access Information	ai4bharat/samanantar · Datasets at Hugging Face

Table 1: Samanantar Dataset Description

3.3 EMBEDDING STRATEGIES USING LLM

Creating a sentence embedding by averaging the embeddings of each word using GloVe involves the following steps:

- **Load Pre-trained GloVe Model**: Start by loading a pre-trained GloVe model that has been trained on a large corpus, such as Wikipedia. Tokenize the Sentence: Break down the sentence into individual words or tokens.
- **Retrieve Word Embeddings**: For each token in the sentence, retrieve its corresponding GloVe word embedding from the pre-trained model.
- **Handle Out-of-Vocabulary Words**: If a word is not in the GloVe vocabulary, you can ignore it or initialize it with a zero or random vector.
- **Calculate the Average**: Sum up all the word embeddings and then divide by the number of words to get the average. This results in a single embedding vector that represents the entire sentence.
- **Normalize the Embedding**: Optionally, you can normalize the resulting sentence embedding to have a unit length, which can be useful for some applications.

This method is simple and computationally efficient but doesn’t produce good embeddings for sentences as it loses context in the process. Therefore, the embeddings were generated using SentenceTransformers. The steps to generate the word embeddings are as follows:

- **Load the Model**: Import the SentenceTransformer class and load the model using its name.
- **Prepare Sentences**: Create a list of sentences that you wish to convert into embeddings.
- **Generate Embeddings**: Call the encode method with your list of

sentences to obtain the embeddings. The resulting embeddings array contains the sentence embeddings, which can be used for various NLP tasks such as semantic similarity, clustering, or information retrieval. Both models work similarly in terms of the steps required to generate sentence embeddings.

The primary difference lies in the underlying Transformer model and the dimensionality of the embeddings produced. all-MiniLM-L6-v2 produces 384-dimensional embeddings, while all-mpnet-base-v2 produces 768-dimensional embeddings.

The embeddings produced by these models are useful for a wide range of applications that require understanding the semantic meaning of sentences such as semantic search, clustering, or semantic similarity measurements.

Given that the dataset lacked labeled domains for sentences, a methodical approach was taken. After pre-requisite text preprocessing steps, an unsupervised technique leveraging BERT Word Embedding (GLoVe) was employed.

Figure 1 shows a list of words whose embeddings are close to the notion of ‘TRAVEL’. Embeddings were created for each sentence by converting all the words in the sentence into an embedding and subsequently summarizing each sentence by computing the mean of the word embeddings.

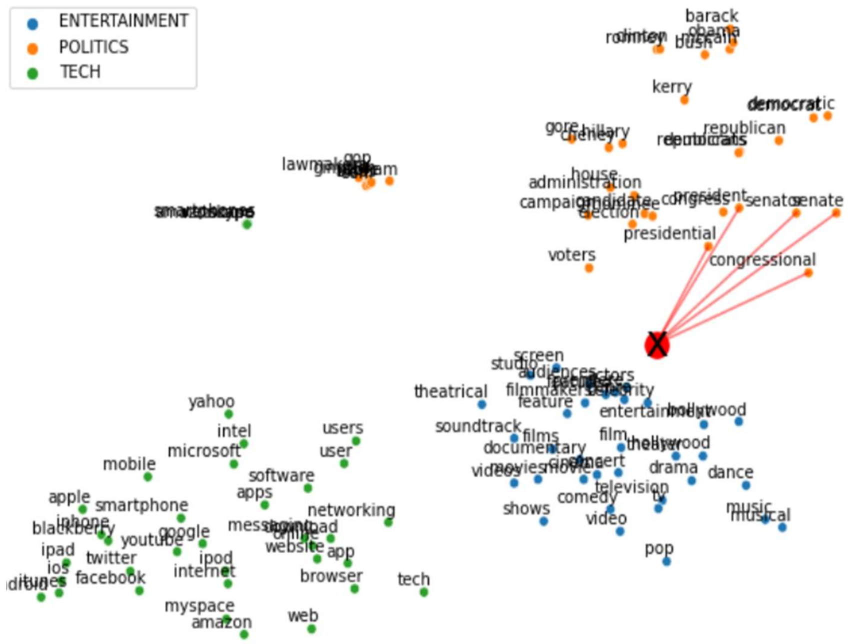
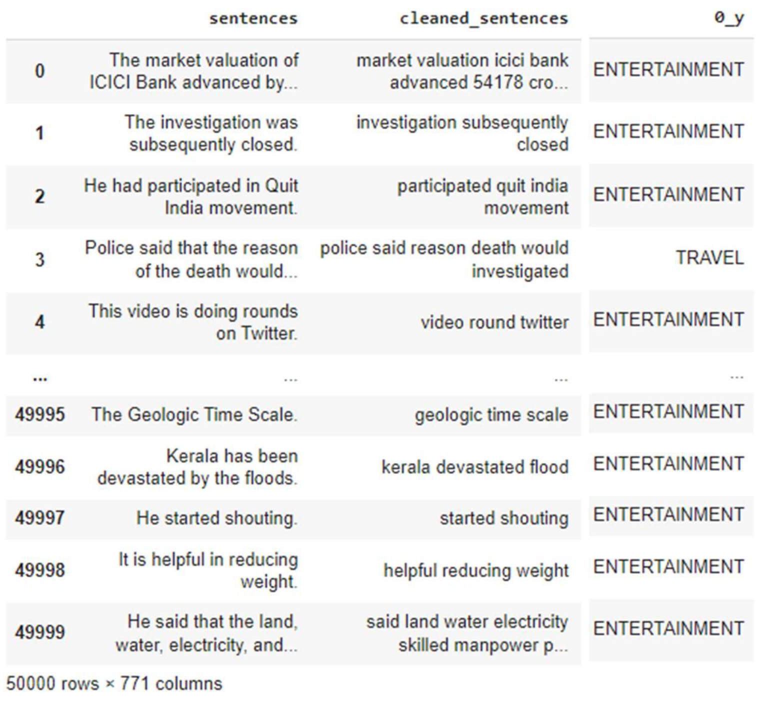


Figure 1: Graphical representation of clustering of few words.



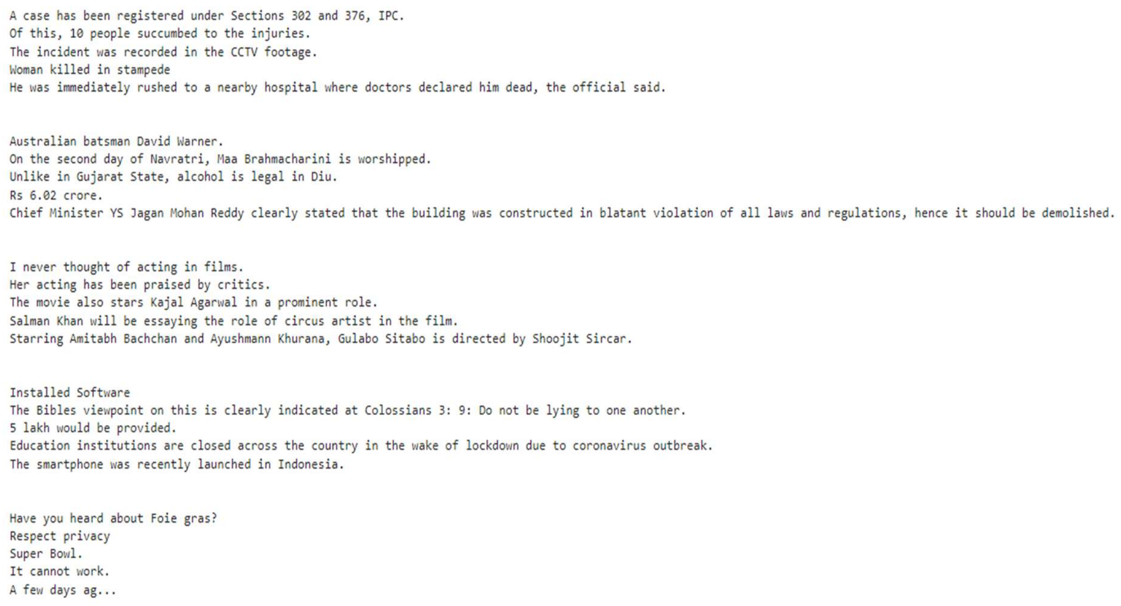
	sentences	cleaned_sentences	0_y
0	The market valuation of ICICI Bank advanced by...	market valuation icici bank advanced 54178 cro...	ENTERTAINMENT
1	The investigation was subsequently closed.	investigation subsequently closed	ENTERTAINMENT
2	He had participated in Quit India movement.	participated quit india movement	ENTERTAINMENT
3	Police said that the reason of the death would...	police said reason death would investigated	TRAVEL
4	This video is doing rounds on Twitter.	video round twitter	ENTERTAINMENT
...	...	...	...
49995	The Geologic Time Scale.	geologic time scale	ENTERTAINMENT
49996	Kerala has been devastated by the floods.	kerala devastated flood	ENTERTAINMENT
49997	He started shouting.	started shouting	ENTERTAINMENT
49998	It is helpful in reducing weight.	helpful reducing weight	ENTERTAINMENT
49999	He said that the land, water, electricity, and...	said land water electricity skilled manpower p...	ENTERTAINMENT

50000 rows × 771 columns

Figure 2: Sentence clustered together after finding the mean embedding of all constituent word embeddings.

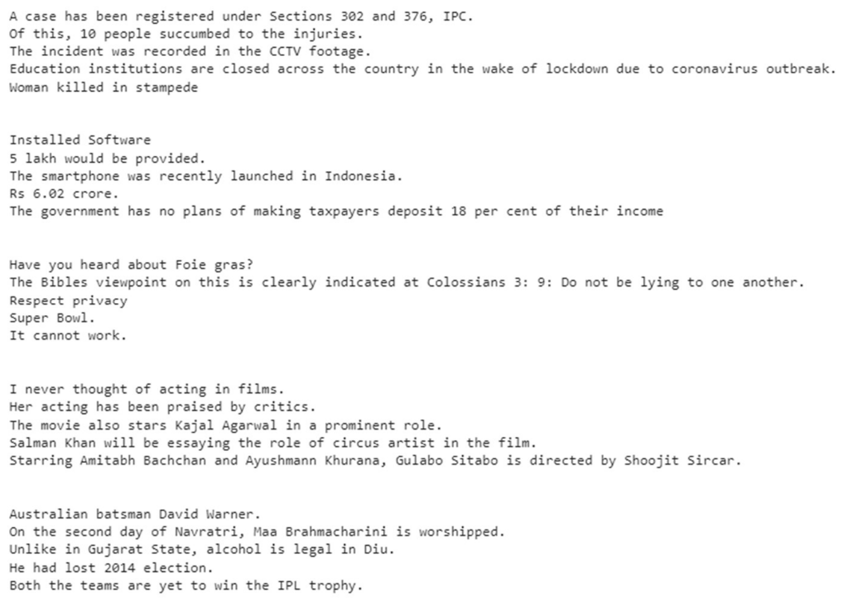
However, it was recognized in figure 2, this method resulted in a loss of sentence meaning during the embedding process. To address this limitation and preserve the semantic nuances of sentences, a specialized technique was adopted using Sentence Transformers.

Two distinct models were employed for this purpose: all-MiniLM-L6-v2, with a model size of 80MB and dimensions of (384,), and all-mpnet-base-v2, boasting a larger size of 420MB and dimensions of (768,). Notably, the research demonstrated that the larger model consistently outperformed its smaller counterpart, providing more detailed and refined embeddings for each sentence.



```
A case has been registered under Sections 302 and 376, IPC.
Of this, 10 people succumbed to the injuries.
The incident was recorded in the CCTV footage.
Woman killed in stampede
He was immediately rushed to a nearby hospital where doctors declared him dead, the official said.


Australian batsman David Warner.
On the second day of Navratri, Maa Brahmacharini is worshipped.
Unlike in Gujarat State, alcohol is legal in Diu.
Rs 6.02 crore.
Chief Minister YS Jagan Mohan Reddy clearly stated that the building was constructed in blatant violation of all laws and regulations, hence it should be demolished.


I never thought of acting in films.
Her acting has been praised by critics.
The movie also stars Kajal Agarwal in a prominent role.
Salman Khan will be essaying the role of circus artist in the film.
Starring Amitabh Bachchan and Ayushmann Khurana, Gulabo Sitabo is directed by Shoojit Sircar.


Installed Software
The Bibles viewpoint on this is clearly indicated at Colossians 3: 9: Do not be lying to one another.
5 lakh would be provided.
Education institutions are closed across the country in the wake of lockdown due to coronavirus outbreak.
The smartphone was recently launched in Indonesia.


Have you heard about Foie gras?
Respect privacy
Super Bowl.
It cannot work.
A few days ag...
```

Figure 3: all-MiniLM-L6-v2 embeddings clusters



```
A case has been registered under Sections 302 and 376, IPC.
Of this, 10 people succumbed to the injuries.
The incident was recorded in the CCTV footage.
Education institutions are closed across the country in the wake of lockdown due to coronavirus outbreak.
Woman killed in stampede


Installed Software
5 lakh would be provided.
The smartphone was recently launched in Indonesia.
Rs 6.02 crore.
The government has no plans of making taxpayers deposit 18 per cent of their income


Have you heard about Foie gras?
The Bibles viewpoint on this is clearly indicated at Colossians 3: 9: Do not be lying to one another.
Respect privacy
Super Bowl.
It cannot work.


I never thought of acting in films.
Her acting has been praised by critics.
The movie also stars Kajal Agarwal in a prominent role.
Salman Khan will be essaying the role of circus artist in the film.
Starring Amitabh Bachchan and Ayushmann Khurana, Gulabo Sitabo is directed by Shoojit Sircar.


Australian batsman David Warner.
On the second day of Navratri, Maa Brahmacharini is worshipped.
Unlike in Gujarat State, alcohol is legal in Diu.
He had lost 2014 election.
Both the teams are yet to win the IPL trophy.
```

Figure 4: all-mpnet-base-v2 embeddings clusters

The results in figures 3 and 4 appear to be better than the earlier approach but still are not up to the mark.

3.4 GRAPHICAL REPRESENTATION OF TEXT

In the exploration of GNNs and their application to NLP tasks, a foundational understanding of graph representation and graph representation learning was established. A graph, defined as a mathematical representation of interconnected objects, consists of nodes representing entities and edges denoting relationships between these entities. Graph representation learning emerged as a crucial process aimed at acquiring low-dimensional vector representations (embeddings) for nodes or edges in a graph. This learning process seeks to capture both structural and semantic information within the graph, laying the groundwork for diverse downstream tasks such as node classification, link prediction, and graph clustering.

In the exploration of GNNs and their application to NLP tasks, a foundational understanding of graph representation and graph representation learning was established. A graph, defined as a mathematical representation of interconnected objects, consists of nodes representing entities and edges denoting relationships between these entities. Graph representation learning emerged as a crucial process aimed at acquiring low-dimensional vector representations (embeddings) for nodes or edges in a graph. This learning process seeks to capture both structural and semantic information within the graph, laying the groundwork for diverse downstream tasks such as node classification, link prediction, and graph clustering.

However, the application of GNNs to the Text Datasets for Indian Languages (TDIL) dataset and the Samanantar dataset encountered challenges. The discovery was made that GNNs inherently rely on supervised learning, and the available datasets contained predominantly raw data with limited information suitable for GNNs. To address this limitation, a bespoke dataset was curated, leveraging information from the Wikipedia page for Chennai tourism. Knowledge graphs were constructed from this data, intending to provide meaningful information for GNN learning. Unfortunately, the generated knowledge graphs fell short of offering substantial insights conducive to effective GNN training. This realization emphasized the intricacies and challenges involved in aligning GNNs with specific datasets and underscored the importance of data quality and relevance in facilitating meaningful learning outcomes for these sophisticated models.

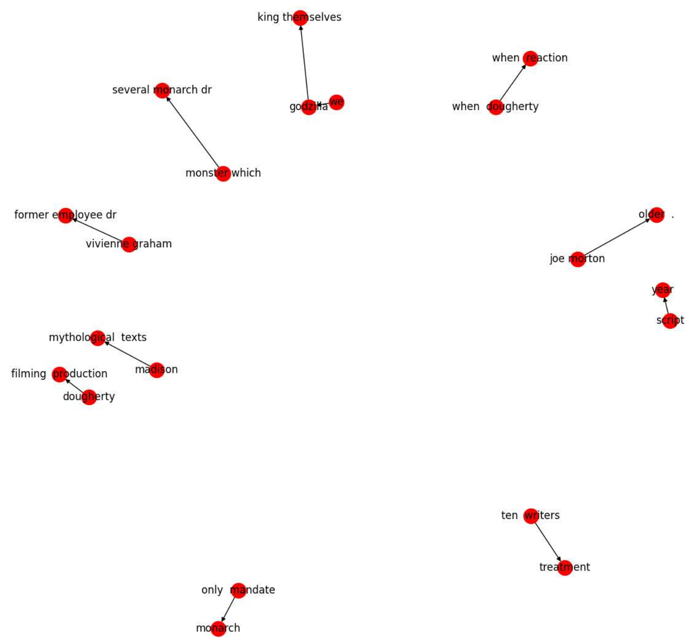


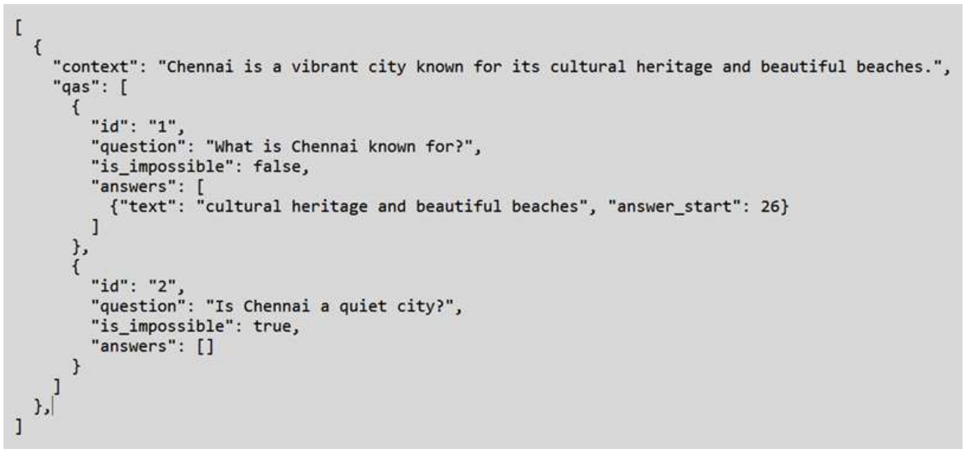
Figure 5: Example of generated knowledge graph

Thus, the unsupervised approach is underperforming and going for a custom database that includes questions, answers and context will be a better approach. The database will have to be in the format that is accepted by the LLM chosen, which will be done as a part of future work.

3.5 USING ALBERT ON SAMANANTAR DATASET

ALBERT's input data structure is a list of dictionaries or a JSON file, with each dictionary representing a unique context and its associated questions. Key attributes include "context" for the text providing context and "qas" for a list of dictionaries containing question details. Within each "qas" dictionary, crucial attributes like "id," "question," "is_impossible," and "answers" define the question's unique identifier, text, answerability, and correct answers.

This organized format enables ALBERT's efficient processing of diverse contexts and questions, supporting its natural language understanding capabilities. The separation of context and questions allows ALBERT to focus on learning relationships between information and queries, while attributes like "is_impossible" enhance its handling of questions without feasible answers. The use of dictionaries ensures a clear and standardized representation, facilitating seamless integration with ALBERT's pre-training and fine-tuning processes, ultimately improving its performance in comprehending and responding to questions within a given context.



```
[
  {
    "context": "Chennai is a vibrant city known for its cultural heritage and beautiful beaches.",
    "qas": [
      {
        "id": "1",
        "question": "What is Chennai known for?",
        "is_impossible": false,
        "answers": [
          {"text": "cultural heritage and beautiful beaches", "answer_start": 26}
        ]
      },
      {
        "id": "2",
        "question": "Is Chennai a quiet city?",
        "is_impossible": true,
        "answers": []
      }
    ]
  },
]
```

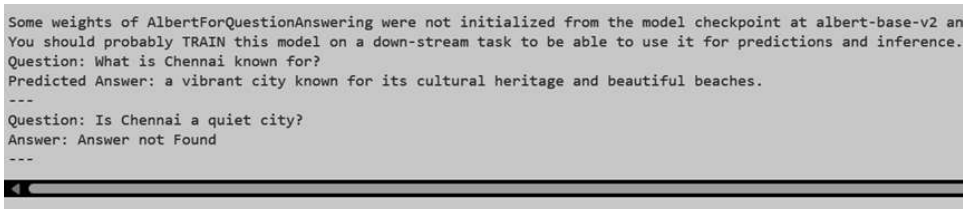
Figure 6: Example of Dataset structure for ALBERT

3.6 DATA PREPROCESSING FOR ALBERT

Data preprocessing for ALBERT involves several key steps to prepare the input text for effective utilization by the model. Initially, tokenization is employed to break down the input text into subword tokens, a method ALBERT utilizes efficiently to handle a large vocabulary. The process employs the AlbertTokenizer from the transformers library, specifically designed for ALBERT's subword tokenization. Following tokenization, the numerical conversion of tokenized text is crucial, accomplished by mapping each token to its corresponding index in the model's vocabulary using the convert_tokens_to_ids method of the tokenizer.

To manage variable-length sequences, padding tokens are added to ensure uniform sequence lengths, and sequences exceeding the model's maximum input length are truncated. These steps are essential for maintaining consistency in input format.

The AlbertTokenizer's padding and truncation options are leveraged for this purpose. Additionally, special tokens like [CLS] (start of sequence) and [SEP] (end of sequence) are incorporated into the input. These tokens play a crucial role in conveying the structural information of the input to the model. The add_special_tokens option in the tokenizer facilitates the seamless addition of these special tokens, enhancing ALBERT's understanding of the input's contextual and structural aspects.



```
Some weights of AlbertForQuestionAnswering were not initialized from the model checkpoint at albert-base-v2 an
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
Question: What is Chennai known for?
Predicted Answer: a vibrant city known for its cultural heritage and beautiful beaches.
---
Question: Is Chennai a quiet city?
Answer: Answer not Found
---
```

Figure 7: Sample output of albert-base-v2 model

3.7 INSIGHTS OBTAINED

Upon thorough exploration and experimentation with various methodologies and models, several key insights have emerged, shaping the direction and scope of our project. These insights are pivotal in guiding our decision-making process and optimizing the efficiency and effectiveness of our Smart Question and Answer (Q&A) System.

Selection of RoBERTa Model: After experimenting with multiple language models, including ALBERT, we have concluded that RoBERTa stands out as the most suitable choice for our task. RoBERTa, based on BERT architecture, has demonstrated superior performance in natural language understanding tasks due to its robust pre-training objectives and larger training corpora. Its ability to capture intricate linguistic patterns and contextual nuances aligns perfectly with the complexity of tourism-related inquiries. Moreover, RoBERTa's versatility and adaptability make it well-suited for handling the diverse range of queries encountered in the tourism domain.

Importance of Curated Dataset: Our exploration highlighted the critical role of dataset curation in the development of a high-performing Q&A system. While existing datasets such as Samanantar provide valuable resources, they often lack specificity and relevance to our target domain. Therefore, creating a curated dataset tailored specifically to tourism in Chennai becomes imperative. This curated dataset ensures that the model is trained on relevant and representative examples, thereby enhancing its accuracy and relevance to real-world queries.

Consideration of Model Complexity: While Graph Neural Networks (GNNs) offer intriguing possibilities for textual representation and understanding, we have determined that their integration may introduce unnecessary complexity and hinder the model's performance for our specific task. The additional computational overhead and time complexity required for training and inference with GNNs may outweigh the potential benefits, particularly considering the relatively straightforward nature of our Q&A system. Thus, we have opted to streamline our approach and focus on maximizing the efficiency and effectiveness of the RoBERTa-based model.

3.8 SUMMARY

Chapter 3 explores available datasets and methodologies for preparing textual data for the Smart Question and Answer (Q&A) System for Chennai tourism. It begins with the significance of corpus development and discusses data acquisition methodologies, emphasizing the utilization of Large Language Models (LLMs) and Graph Neural Networks (GNNs). The chapter examines the Samanantar dataset and its limitations in embedding strategies using LLMs. However, due to the lack of labeled domains for sentences, unsupervised techniques using BERT Word Embedding (GloVe) were employed for sentence embedding creation. This approach resulted in some loss of sentence meaning.

To address this, Sentence Transformers were utilized, with the larger all-mpnet-base-v2 model producing more detailed embeddingsDespite efforts with Sentence Transformers and GNNs, challenges in aligning GNNs with datasets are noted, highlighting the importance of data quality. ALBERT model utilization is explored, but concerns related to model performance lead to the selection of RoBERTa for superior performance. The chapter underscores the importance of dataset curation and streamlining model complexity.

In summary, Chapter 3 informs the project's direction, emphasizing the selection of RoBERTa, dataset curation's significance, and the need to streamline model complexity.

CHAPTER 4

DATASET CREATION AND DATA PREPROCESSING FOR ROBERTA

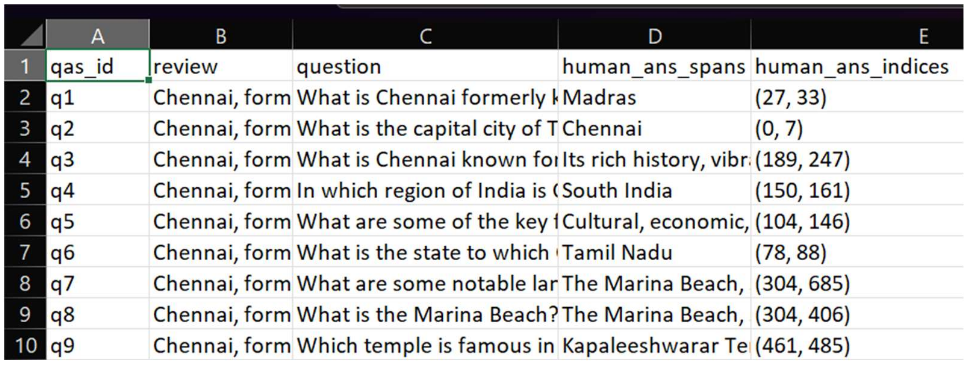
4.1 INTRODUCTION

Creating a robust and tailored dataset is paramount for optimizing the performance of RoBERTa. RoBERTa, derived from BERT architecture, has emerged as a leading contender in natural language understanding tasks due to its advanced pre-training objectives and extensive training corpora. Unlike its predecessor, RoBERTa is trained with larger batches, longer sequences, and diverse data sources, allowing it to capture intricate linguistic patterns and contextual nuances more effectively. With its transformer-based architecture and bidirectional self-attention mechanism, RoBERTa excels in understanding and generating human-like text across a wide range of tasks, including question answering, text generation, and sentiment analysis. Leveraging RoBERTa's capabilities requires a curated dataset specifically tailored to the nuances of tourism-related inquiries in Chennai, ensuring that the model is equipped with the requisite knowledge and context to deliver accurate and informative responses.

4.2 INPUT DATA STRUCTURE FOR ROBERTA

Generally, datasets for fine-tuning LLMs for the task of question answering is in the form of question answer pairs, and supervised learning methods are employed for LLM to comprehend the context and answer questions appropriately. The data is usually in JSON or CSV format.

Input data for this system is in CSV format where each row is a question answer pair with the relevant context. It has attributes like "review" (context) and "qas_id" (unique identifier) for the question, the "question", "human_ans_spans" which is correct answer to the question and “human_ans_indices” which includes the indices of where the answer starts and ends in the context paragraph.



	A	B	C	D	E
1	qas_id	review	question	human_ans_spans	human_ans_indices
2	q1	Chennai, form	What is Chennai formerly k	Madras	(27, 33)
3	q2	Chennai, form	What is the capital city of T	Chennai	(0, 7)
4	q3	Chennai, form	What is Chennai known for	Its rich history, vibr	(189, 247)
5	q4	Chennai, form	In which region of India is (	South India	(150, 161)
6	q5	Chennai, form	What are some of the key t	Cultural, economic,	(104, 146)
7	q6	Chennai, form	What is the state to which	Tamil Nadu	(78, 88)
8	q7	Chennai, form	What are some notable lar	The Marina Beach,	(304, 685)
9	q8	Chennai, form	What is the Marina Beach?	The Marina Beach,	(304, 406)
10	q9	Chennai, form	Which temple is famous in	Kapaleeshwarar Te	(461, 485)

Figure 8: Example image of curated dataset

4.3 DATASET DESCRIPTION

The dataset contains 320 rows of question answer pairs, from which 270 pairs were used as the training data and 50 pairs picked uniformly throughout the dataset were used as testing data. The dataset contains questions and answers from 31 tourist attractions in and around Chennai. The tourist attractions from which questions were formed is listed in Table 1 given below. The text corpus was obtained from various sources across Google, including Wikipedia, famous travel websites like MakeMyTrip, Goibibo, and official Tamil Nadu Government website for tourism. The dataset introduced in this report is manually curated for the specific use-case of Tourism in Chennai.

S.No	Topic	No. of questions from each topic
1	Chennai Culture	14
2	Guindy National Park	12
3	Vandalur Zoo	11
4	Madras Crocodile Bank	9
5	Vedanthangal Bird Sanctuary	13
6	Adyar Eco Park	11
7	Marina Beach	14
8	Kovalam Beach	7
9	Elliot's Beach	6
10	Blue Flag Beach Muttukadu	8
11	Muttukadu Boathouse	7
12	Fort St. George	13
13	Government Museum Chennai	10
14	DakshinaChitra Heritage Museum	12
15	Chennai Rail Museum	10
16	Birla Planetarium	8
17	Chennai Lighthouse	7
18	Valluvar Kottam	13
19	Chepauk Stadium	9
20	Napier Bridge	8
21	MGR Film City	12
22	Shopping Malls	14
23	T Nagar	9
24	Parry's Corner	5
25	Moore Market	7
26	Santhome Church	11
27	Annai Vailankanni Shrine	8
28	Thousand Lights Mosque	6
29	Kapaleeshwarar temple	12
30	Parthasarathy Temple	14
31	Mahabalipuram	13
32	Shore Temple	7

Table 2: List of topics based on which Dataset was curated

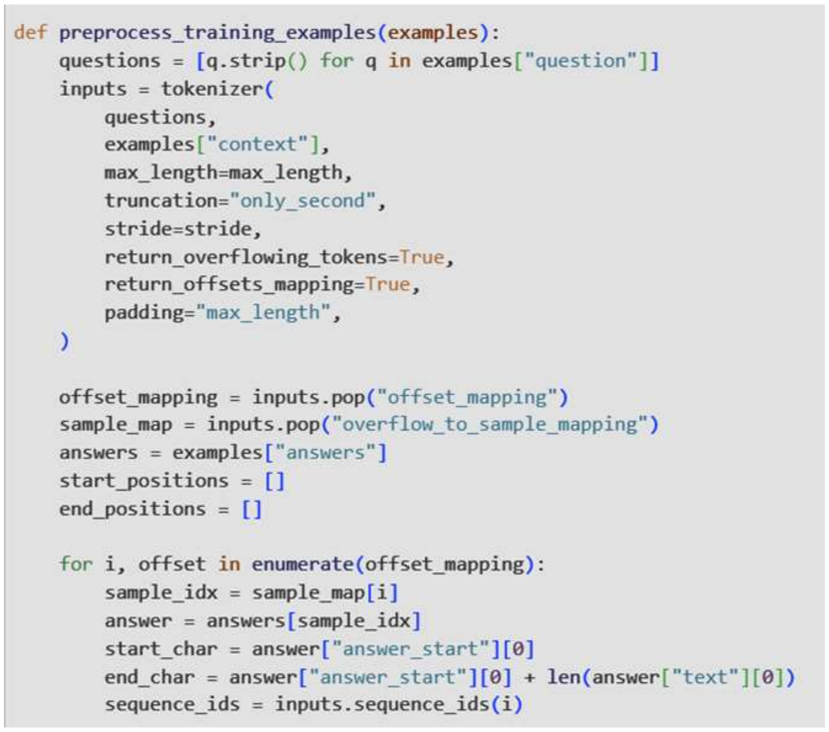
4.4 DATA PREPROCESSING FOR ROBERTA

Tokenization: This is the process of breaking down the input text into individual tokens (words, subwords, or characters). RoBERTa's tokenizer uses a Byte-Pair Encoding (BPE) algorithm. BPE is a subword tokenization technique. Unlike splitting text at word boundaries, it analyzes the training data and identifies frequently occurring sequences of characters (pairs of characters for starters). These pairs are then treated as single units (subwords) and added to the tokenizer's vocabulary.

Padding: Padding is added to ensure all sequences have the same length. The padding="max_length" argument ensures that all tokenized sequences are padded to the maximum length specified by max_length.

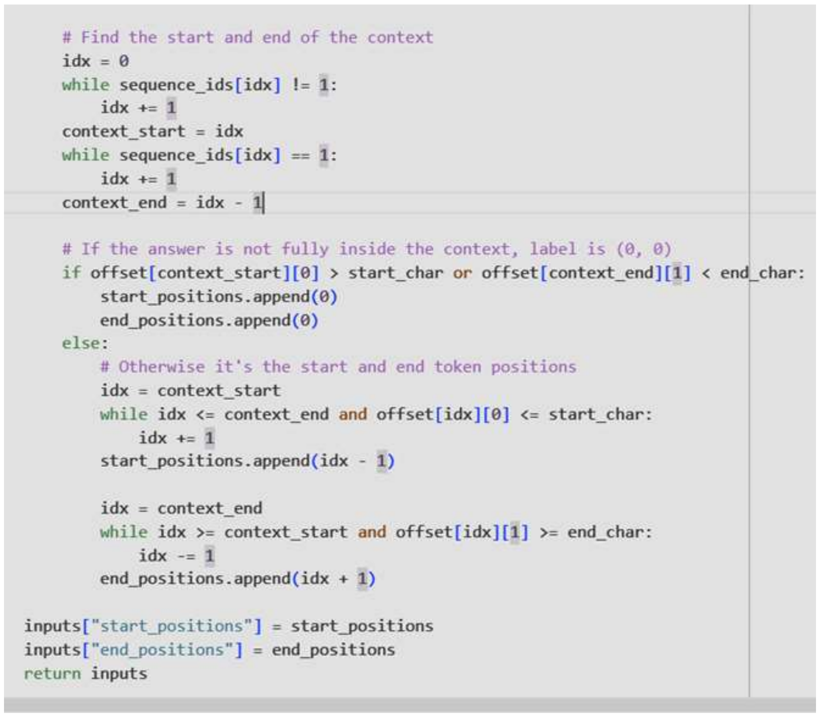
Handling Long Sequences: The return_overflowing_tokens=True ensures that overflow tokens are returned. This is useful for handling long sequences by dividing them into multiple segments with a certain overlap defined by stride.

Extracting Answer Spans and Finding Answer Positions: The return_offsets_mapping=True argument returns the mapping between tokenized tokens and their corresponding character positions in the original text. This mapping is essential for identifying the start and end positions of the answer spans in the tokenized sequences. The code iterates over the tokenized sequences and calculates the start and end positions of the answer spans within the context. It uses the character offsets provided by offset_mapping to map the answer span from the original text to the tokenized sequences. If the answer span is not fully contained within the context, it's labeled as (0, 0).



```python
def preprocess_training_examples(examples):
    questions = [q.strip() for q in examples["question"]]
    inputs = tokenizer(
        questions,
        examples["context"],
        max_length=max_length,
        truncation="only_second",
        stride=stride,
        return_overflowing_tokens=True,
        return_offsets_mapping=True,
        padding="max_length",
    )

    offset_mapping = inputs.pop("offset_mapping")
    sample_map = inputs.pop("overflow_to_sample_mapping")
    answers = examples["answers"]
    start_positions = []
    end_positions = []

    for i, offset in enumerate(offset_mapping):
        sample_idx = sample_map[i]
        answer = answers[sample_idx]
        start_char = answer["answer_start"][0]
        end_char = answer["answer_start"][0] + len(answer["text"][0])
        sequence_ids = inputs.sequence_ids(i)
```

Figure 9: Code for pre-processing the curated dataset



```python
        # Find the start and end of the context
        idx = 0
        while sequence_ids[idx] != 1:
            idx += 1
        context_start = idx
        while sequence_ids[idx] == 1:
            idx += 1
        context_end = idx - 1

        # If the answer is not fully inside the context, label is (0, 0)
        if offset[context_start][0] > start_char or offset[context_end][1] < end_char:
            start_positions.append(0)
            end_positions.append(0)
        else:
            # Otherwise it's the start and end token positions
            idx = context_start
            while idx <= context_end and offset[idx][0] <= start_char:
                idx += 1
            start_positions.append(idx - 1)

            idx = context_end
            while idx >= context_start and offset[idx][1] >= end_char:
                idx -= 1
            end_positions.append(idx + 1)

    inputs["start_positions"] = start_positions
    inputs["end_positions"] = end_positions
    return inputs
```

Figure 10: Code for pre-processing the curated dataset (continued)

4.5 SUMMARY

To enhance the question-answering system for Chennai tourism, a meticulous dataset creation and pre-processing process was undertaken specifically for the Roberta language model. Roberta's strength lies in its ability to grasp intricate language patterns, making it a perfect choice for this task.

The dataset is formatted as a CSV file containing 320 question-answer pairs, along with relevant context for each question. This data is meticulously divided into two sets: a training set of 270 examples and a testing set of 50 examples, allowing the model to learn from a comprehensive set of questions while reserving a separate set for unbiased evaluation. The questions themselves delve into 31 tourist attractions in and around Chennai, ensuring broad coverage of the city's tourism landscape. To gather the text corpus, the authors consulted various reputable online sources, including Wikipedia, popular travel websites like MakeMyTrip and Goibibo, and the official Tamil Nadu government tourism website.

Before feeding the data into Roberta, several pre-processing steps, such as Tokenization, Padding, Handling Long Sequences and Extracting Answer positions were implemented.

Through this meticulous dataset creation and pre-processing, a robust and informative resource was crafted specifically for the Chennai tourism Q&A system. This empowers the fine-tuned Roberta model to deliver accurate and insightful responses to user queries, ultimately enhancing the overall user experience.

CHAPTER 5

FINE-TUNING AND EVALUATION

5.1 INTRODUCTION

Fine-tuning is a technique in machine learning, particularly deep learning, used to adapt a pre-trained model for a specific task. It's essentially a way to leverage the knowledge gained by a model on a vast amount of general data and specialize it for your particular application.

A model trained on a massive dataset of text or images, allowing it to learn general features and patterns. This could be a large language model (LLM) like RoBERTa trained on a huge corpus of text and code, or an image recognition model trained on millions of images. The fine-tuning process involves taking that pre-trained model and adjusting its internal parameters to focus on a specific task. The core idea is to reuse the valuable knowledge the model already has as a foundation and build upon it for your specific use case.

There are two main ways to approach fine-tuning, freezing layers of the pre-trained model, which capture general features, are frozen. This means their weights are not updated during fine-tuning. These layers act as a strong foundation of knowledge. And training upper layers responsible for more task-specific learning, are fine-tuned with your specific dataset. This allows the model to adapt to the new domain and problem you're interested in.

Benefits of fine-tuning are that it is efficient compared to training a model from scratch, fine-tuning is significantly faster and requires less computational power, and improved performance towards specialization in your task, often leading to better performance than using a generic model. The pre-trained knowledge helps it understand the nuances of your specific data. Fine-tuning leverages the power of transfer learning. Even with limited data for your specific task, the model can benefit from the general knowledge it learned during pre-training.

5.2 FINE-TUNING ROBERTA

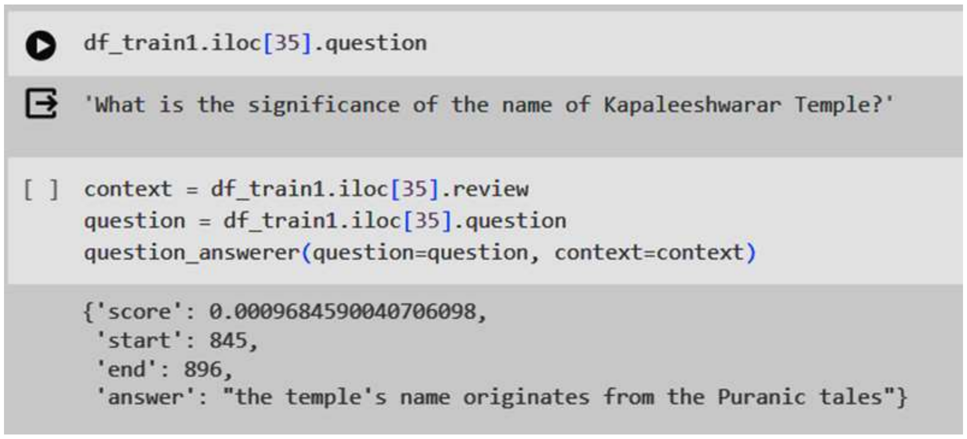
To enhance our question-answering system with domain-specific knowledge of Chennai tourism, we adopted a fine-tuning approach using the pre-trained Roberta large language model. Microsoft Azure's cloud infrastructure provided the computational resources necessary for this process. Our curated Chennai tourism dataset contained 320 question-answer (QnA) pairs, meticulously divided into a training set of 270 examples and a testing set of 50 examples as mentioned in previous chapter. This strategic split ensured the model's ability to learn from a comprehensive set of training examples while reserving a separate set for unbiased evaluation. The fine-tuning process entailed iteratively exposing the pre-trained Roberta model to our tourism-focused QnA pairs for 20 epochs. With each epoch, the model refined its internal parameters to adeptly handle the task of retrieving relevant answers to tourism-related inquiries. This tailored training culminated in a loss of 0.0178 at the end of 20 epochs, indicative of the model's significant progress in minimizing errors. A comprehensive table detailing the training loss across all epochs is incorporated within the report to visualize the model's learning curve and track its improvement over time.

Epoch	Training Loss
1	1.2009
2	0.6775
3	0.4106
4	0.275
5	0.2142
6	0.1426
7	0.0973
8	0.0575
9	0.0611
10	0.0405
11	0.0434
12	0.0289
13	0.0312
14	0.0279
15	0.02
16	0.0264
17	0.02
18	0.0152
19	0.0183
20	0.0178

Table 3: Outline of Training Loss for 20 epochs

5.3 EVALUATION OF ROBERTA

To assess the effectiveness of the fine-tuned Roberta model in addressing user queries pertaining to Chennai tourism, a comprehensive evaluation was conducted. The model's performance was measured on both the training set (270 question-answer pairs) and the testing set (50 question-answer pairs). The evaluation metrics employed in this project will be elaborated upon in the following section.



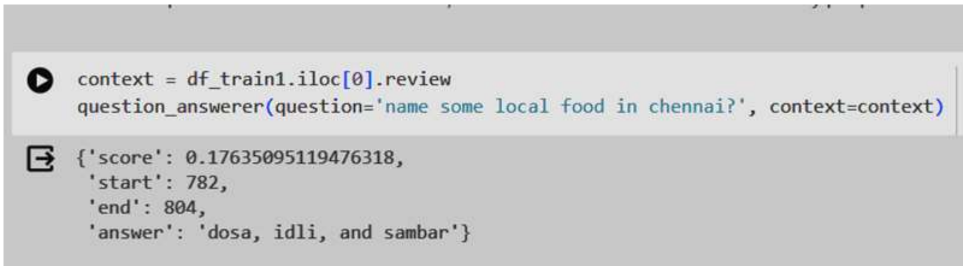
```
df_train1.iloc[35].question
```

```
'What is the significance of the name of Kapaleeshwarar Temple?'
```

```
[ ] context = df_train1.iloc[35].review
    question = df_train1.iloc[35].question
    question_answerer(question=question, context=context)
```

```
{'score': 0.0009684590040706098,
 'start': 845,
 'end': 896,
 'answer': "the temple's name originates from the Puranic tales"}
```

Figure 11: Question Answering for test dataset.



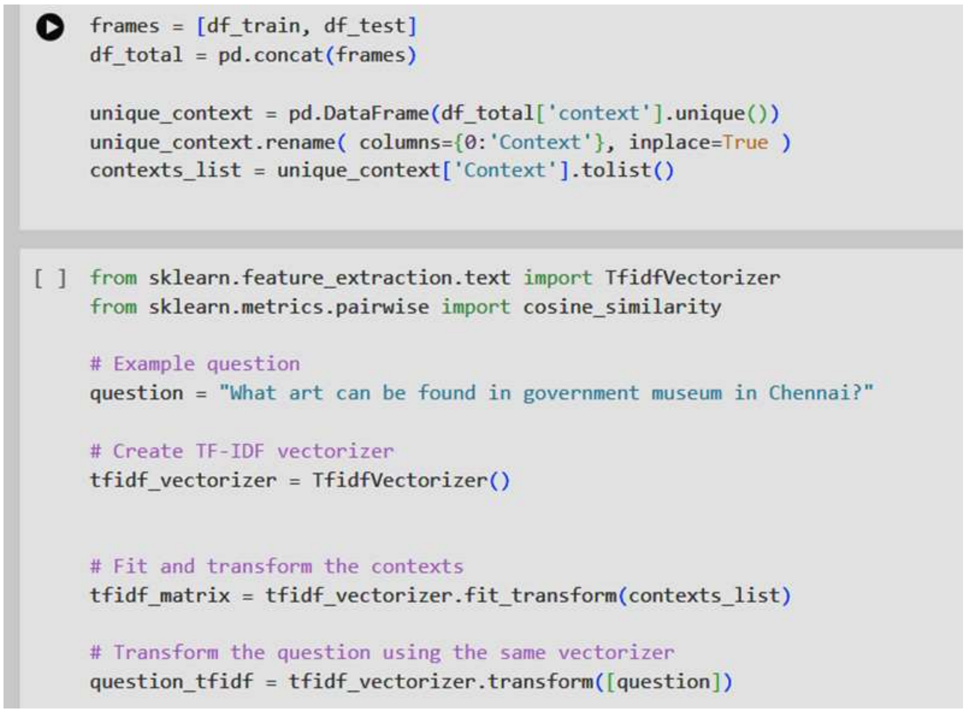
```
context = df_train1.iloc[0].review
question_answerer(question='name some local food in chennai?', context=context)
```

```
{'score': 0.17635095119476318,
 'start': 782,
 'end': 804,
 'answer': 'dosa, idli, and sambar'}
```

Figure 12: Answering new question not present in train or test datasets.

5.4 CHOOSING MOST RELEVANT CONTEXT USING TF-IDF SCORE

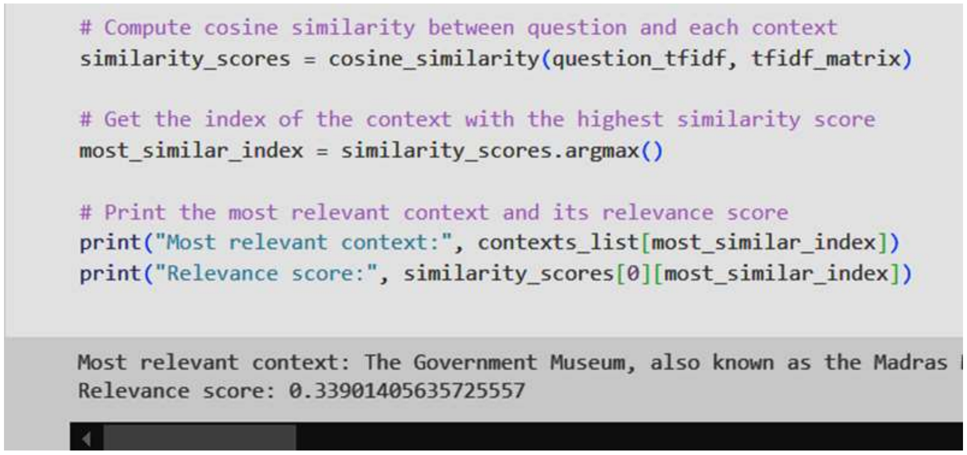
The TF-IDF (Term Frequency-Inverse Document Frequency) score is a metric used to evaluate a word's importance within a document relative to a collection of documents. It considers two factors: how frequently a term appears within a specific document (Term Frequency) and how uncommon that term is across the entire document collection (Inverse Document Frequency). Words that occur often in a single document but rarely appear in others receive high TF-IDF scores, indicating their significance for that specific document's content. This helps prioritize relevant keywords and filter out generic terms when searching for information or analyzing text data.



```
frames = [df_train, df_test]
df_total = pd.concat(frames)

unique_context = pd.DataFrame(df_total['context'].unique())
unique_context.rename( columns={0:'Context'}, inplace=True )
contexts_list = unique_context['Context'].tolist()
```

```
[ ] from sklearn.feature_extraction.text import TfidfVectorizer
    from sklearn.metrics.pairwise import cosine_similarity

    # Example question
    question = "What art can be found in government museum in Chennai?"

    # Create TF-IDF vectorizer
    tfidf_vectorizer = TfidfVectorizer()


    # Fit and transform the contexts
    tfidf_matrix = tfidf_vectorizer.fit_transform(contexts_list)

    # Transform the question using the same vectorizer
    question_tfidf = tfidf_vectorizer.transform([question])
```

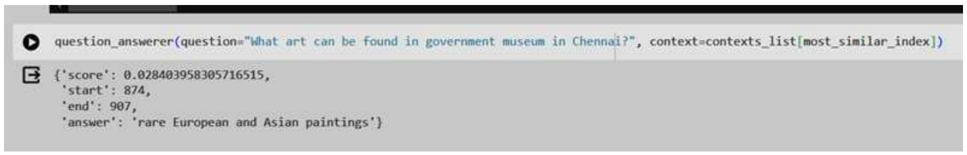
Figure 13: Code for Obtaining most relevant context from the list of contexts to answer question appropriately using TF-IDF Score.



```
# Compute cosine similarity between question and each context
similarity_scores = cosine_similarity(question_tfidf, tfidf_matrix)

# Get the index of the context with the highest similarity score
most_similar_index = similarity_scores.argmax()

# Print the most relevant context and its relevance score
print("Most relevant context:", contexts_list[most_similar_index])
print("Relevance score:", similarity_scores[0][most_similar_index])
```

```
Most relevant context: The Government Museum, also known as the Madras
Relevance score: 0.33901405635725557
```

Figure 14: Code for Obtaining most relevant context from the list of contexts to answer question appropriately using TF-IDF Score (continued)



```
question_answerer(question="What art can be found in government museum in Chennai?", context=contexts_list[most_similar_index])
```

```
{'score': 0.028403958305716515,
 'start': 874,
 'end': 907,
 'answer': 'rare European and Asian paintings'}
```

Figure 15: Answering a new question after choosing relevant context

However, this approach presented challenges due to character limitations imposed by Hugging Face Hub (maximum 512 characters per context). In the case of tourist attractions with extensive descriptions, this necessitated splitting the information into multiple context paragraphs. TF-IDF, by design, focuses on individual words and their frequency within a document. This limitation became apparent when dealing with these segmented contexts, as TF-IDF struggled to differentiate between context paragraphs describing the same attraction. Consequently, selecting the most relevant context paragraph to answer a specific question proved difficult.

To address this challenge, the researchers opted for sentence transformers. Unlike TF-IDF's emphasis on individual words, sentence transformers analyze the meaning of entire sentences. This semantic understanding empowered the model to effectively compare the user's question with each context paragraph. By assessing the overall meaning and semantic similarity between the question and each context, the sentence transformers ensured that the most pertinent passage was selected for answer retrieval. This approach significantly improved the accuracy of context selection and ultimately led to better overall performance in the question answering task.

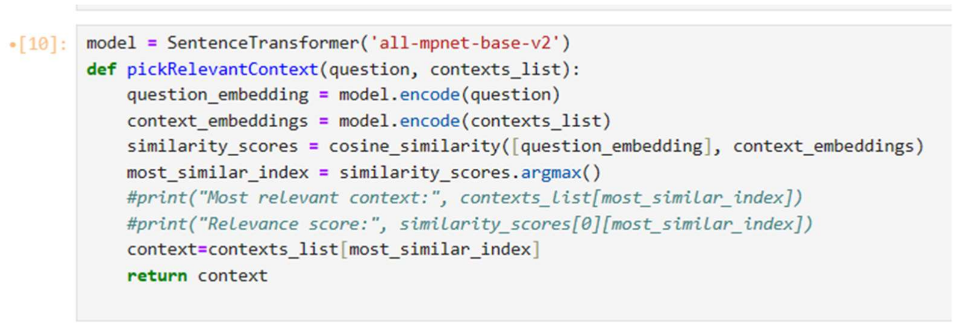
5.5 CHOOSING RELEVANT CONTEXT USING SENTENCE TRANSFORMERS

The specific sentence transformer chosen for this project was the **“all-mpnet-base-v2”** model. The "all-mpnet-base-v2" variant specifically utilizes the pretrained Microsoft MPNet model, further enhancing its semantic understanding capabilities.

Sentence transformers, particularly all-mpnet-base-v2, addressed the challenges faced with TF-IDF as listed below:

Semantic Similarity Assessment: Sentence transformers don't simply compare word frequencies; they calculate the semantic similarity between the user's question and each context paragraph. This similarity score reflects how closely the meaning of the question aligns with the meaning of the context, ensuring selection of the most thematically relevant passage.

Context Disambiguation: When dealing with segmented descriptions of a single tourist attraction, sentence transformers can effectively differentiate between contexts based on their overall meaning. They are not misled by the presence of similar keywords across different paragraphs, leading to a more accurate context selection process.



```
•[10]: model = SentenceTransformer('all-mpnet-base-v2')
       def pickRelevantContext(question, contexts_list):
           question_embedding = model.encode(question)
           context_embeddings = model.encode(contexts_list)
           similarity_scores = cosine_similarity([question_embedding], context_embeddings)
           most_similar_index = similarity_scores.argmax()
           #print("Most relevant context:", contexts_list[most_similar_index])
           #print("Relevance score:", similarity_scores[0][most_similar_index])
           context=contexts_list[most_similar_index]
           return context
```

Figure 16: Code for choosing most relevant context using Sentence Transformers

5.6 SIMILARITY SCORES FOR MODEL EVALUATION

To assess the effectiveness of Roberta's answer retrieval, **Jaccard similarity** and **Levenshtein distance** were employed to compare the model's generated answers with the expected answers (ground truth) within the 320 question-answer pairs of the training and testing sets.

The Jaccard similarity score evaluates the overlap between the sets of words in the expected and retrieved answers, providing a measure of how many words they share proportionally.

Jaccard Similarity formula:

$$\boxed{J(A, B) = |A \cap B| / |A \cup B|}$$

where:

- J(A, B) represents the Jaccard similarity score between sets A and B.
- $|A \cap B|$ denotes the cardinality (number of elements) of the intersection between sets A and B (elements common to both sets).
- $|A \cup B|$ denotes the cardinality of the union between sets A and B (all elements in either set or both sets).

Conversely, Levenshtein distance calculates the minimum number of edits (insertions, deletions, or substitutions) needed to transform the retrieved answer into the expected answer. This metric reflects the edit operations required to achieve an exact match.

Evaluating performance using both Jaccard similarity and Levenshtein distance offers a comprehensive understanding of answer accuracy. High Jaccard scores indicate a strong overlap in content between the expected and retrieved answers, while low Levenshtein distances suggest minimal edits are necessary for an exact match.

Question	Expected Answer	RoBERTa Answer	Jaccard_similarity	Levenshtein_distance
What is Chennai formerly known as?	Madras	Madras	1	0
What is the capital city of Tamil Nadu?	Chennai	Chennai	1	0
What is Chennai known for?	Its rich history, vibrant culture, and architectural landmarks	rich history, vibrant culture, and architectural landmarks	0.875	4
In which region of India is Chennai located?	South India	South India	1	0
What are some of the key features of Chennai?	Cultural, economic, and educational center	cultural, economic, and educational center	0.666666667	1

Table 4: Example of how the system was tested using similarity scores

Out of 320 questions, the model was able to answer 282 questions accurately. This implies that the model shows 88.125% accuracy while testing.

5.7 SUMMARY

This chapter details the process of fine-tuning a Roberta language model for a question-answering system focused on Chennai tourism. Fine-tuning leverages a pre-trained model's knowledge (like Roberta's understanding of general language) and adapts it to a specific task (answering tourism questions). This is achieved by Freezing the pre-trained model's core layers (containing general knowledge) and by Fine-tuning the upper layers to handle the new task (tourism Q&A).

The pre-trained Roberta model was fine-tuned using Microsoft Azure's cloud resources. The curated dataset of 320 question-answer pairs (270 training, 50 testing) focused on Chennai tourism was used. The fine-tuning process involved iteratively exposing Roberta to the Q&A pairs for 20 epochs, resulting in a significant loss reduction (0.0178) by the end, indicating the model's ability to learn the task.

Initially, TF-IDF (Term Frequency-Inverse Document Frequency) was used to identify the most relevant context passage for answering a question. TF-IDF considers word frequency within a document.However, due to character limitations (max 512 per context) and splitting descriptions into multiple paragraphs, TF-IDF struggled to differentiate between contexts for the same attraction. This led to challenges in selecting the best context. To overcome TF-IDF's limitations, sentence transformers were employed. These models analyze the meaning of entire sentences, not just individual words.

Jaccard similarity and Levenshtein distance were used to compare the model's generated answers with the expected answers (ground truth) in the 320 Q&A pairs. The model had 88.125% accuracy during evaluation using the similarity scores mentioned above.

CHAPTER 6

ASR AND TTS SYSTEM

6.1 INTRODUCTION

ASR stands for Automatic Speech Recognition. It's a technology that allows computers to convert spoken language into text. ASR has become a crucial part of many applications, making our interactions with technology more natural and hands-free.

ASR systems typically involve two main stages, Acoustic Modeling, which breaks down speech into its basic building blocks (phonemes) and analyzes them statistically to identify the most likely sounds. And Language Modeling, which considers grammar and word probabilities to refine the recognized text based on context, making the final output more natural.

6.2 STT AND TTS SYSTEMS EMPLOYED

There are several methods available for both Speech-to-Text (STT) and Text-to-Speech (TTS), each with its own advantages and limitations. And the following lists out all the possible methods and our rationale behind our choice of approach is given at the end.

Speech-to-Text (STT):

Acoustic Modeling breaks down speech into its fundamental components like phonemes (basic units of sound). Uses statistical models (e.g., Hidden Markov Models - HMMs) to analyze these components and predict the most likely word sequence. Commonly used in speaker-independent systems (work for various speakers).

Language Modeling takes into account grammar rules and word probabilities to refine the recognized text based on context. Helps eliminate unlikely word sequences and improve accuracy.

Deep Learning Approaches utilize powerful neural networks like recurrent neural networks (RNNs) or convolutional neural networks (CNNs) to learn complex patterns from large speech datasets. Can achieve high accuracy, especially with sufficient training data.

Hybrid Approaches combine traditional acoustic modeling with deep learning techniques for better performance. Leverage the strengths of both methods to achieve optimal results.

Text-to-Speech (TTS):

Rule-Based Synthesis uses pre-recorded speech units (phonemes or diphones) and rules to concatenate them into words and sentences. Offers good control over pronunciation but can sound robotic and unnatural.

Statistical Parametric Synthesis analyzes large speech databases to learn statistical models of speech parameters like pitch, amplitude, and spectral features. Generates speech that sounds more natural than rule-based approaches.

Deep Learning Synthesis employs deep neural networks trained on large amounts of speech data to directly generate speech waveforms. Can produce very natural-sounding speech, often indistinguishable from human speech.

Additional Considerations:

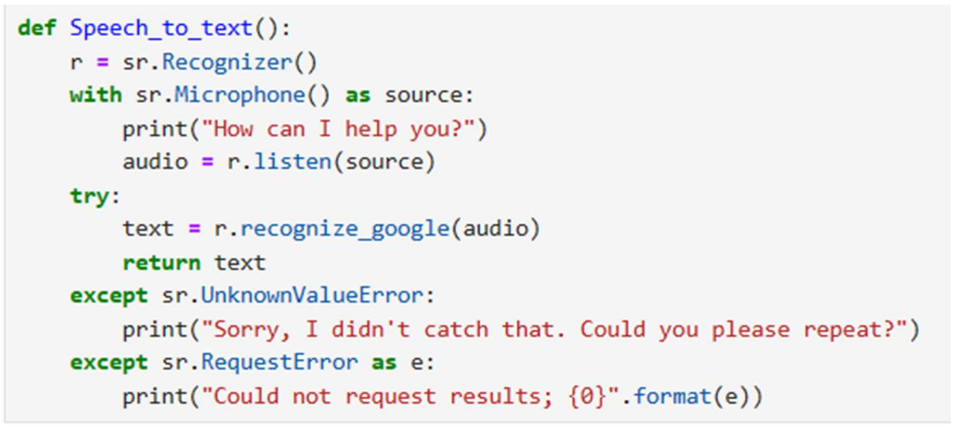
Speaker Dependence: Some STT systems are speaker-dependent, requiring training on a specific speaker's voice for better accuracy.

Language Support: The availability of STT and TTS for different languages varies. More commonly spoken languages typically have better support.

Accuracy: The accuracy of both STT and TTS depends on the chosen method, quality of training data, and background noise levels.

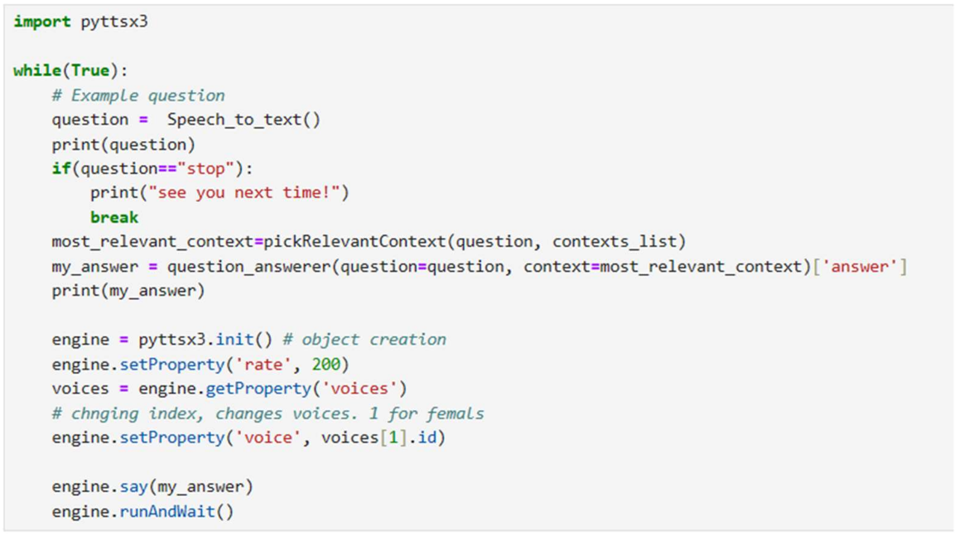
Computational Resources: Deep learning approaches often require significant computational resources for training and running.

Hence, for STT and TTS we chose Google Cloud's Speech-to-Text and Text-to-Speech to avoid straining our computational resources and avoid training or fine-tuning the offline deep-learning based approaches which again often is speaker dependent. And since network connectivity has improved and become cheaper over the years, using cloud services is the way to go and it also provides very good accuracy and language support for different languages. Python modules such as **pyttsx3**, **pyaudio** were used for speech output whereas **speech_recognition** module was used for speech input.



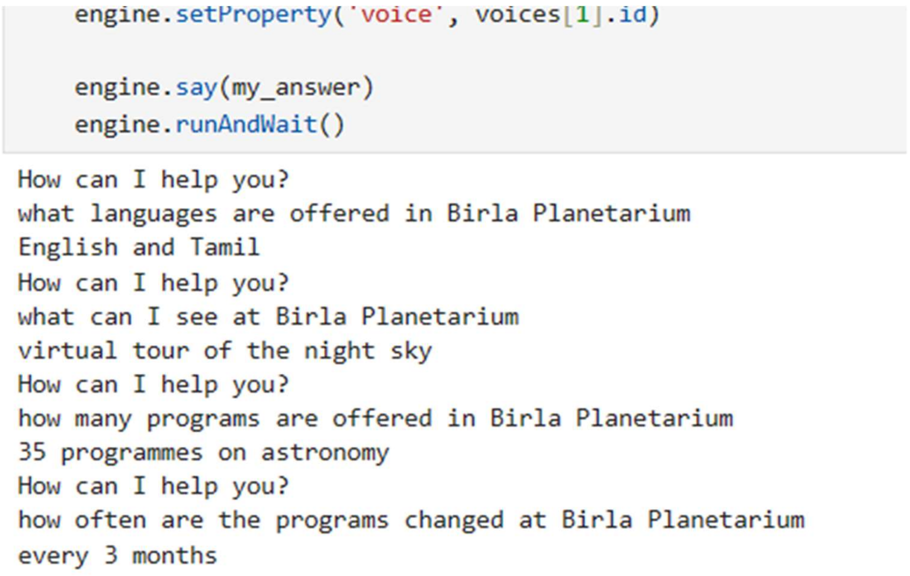
```python
def Speech_to_text():
    r = sr.Recognizer()
    with sr.Microphone() as source:
        print("How can I help you?")
        audio = r.listen(source)
    try:
        text = r.recognize_google(audio)
        return text
    except sr.UnknownValueError:
        print("Sorry, I didn't catch that. Could you please repeat?")
    except sr.RequestError as e:
        print("Could not request results; {0}".format(e))
```

Figure 17: Python code for converting speech to text



```
import pyttsx3

while(True):
    # Example question
    question =  Speech_to_text()
    print(question)
    if(question=="stop"):
        print("see you next time!")
        break
    most_relevant_context=pickRelevantContext(question, contexts_list)
    my_answer = question_answerer(question=question, context=most_relevant_context)['answer']
    print(my_answer)

    engine = pyttsx3.init() # object creation
    engine.setProperty('rate', 200)
    voices = engine.getProperty('voices')
    # chnging index, changes voices. 1 for femals
    engine.setProperty('voice', voices[1].id)

    engine.say(my_answer)
    engine.runAndWait()
```

Figure 18: Question Answering pipeline using speech input and speech output

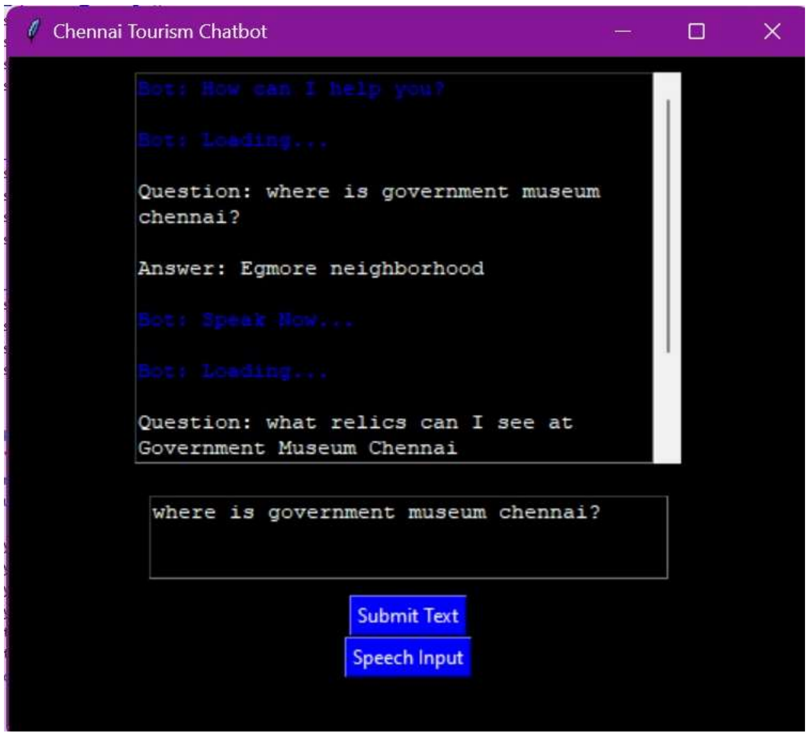


```
    engine.setProperty('voice', voices[1].id)

    engine.say(my_answer)
    engine.runAndWait()
```

```
How can I help you?
what languages are offered in Birla Planetarium
English and Tamil
How can I help you?
what can I see at Birla Planetarium
virtual tour of the night sky
How can I help you?
how many programs are offered in Birla Planetarium
35 programmes on astronomy
How can I help you?
how often are the programs changed at Birla Planetarium
every 3 months
```

Figure 19: Output of Question Answering pipeline

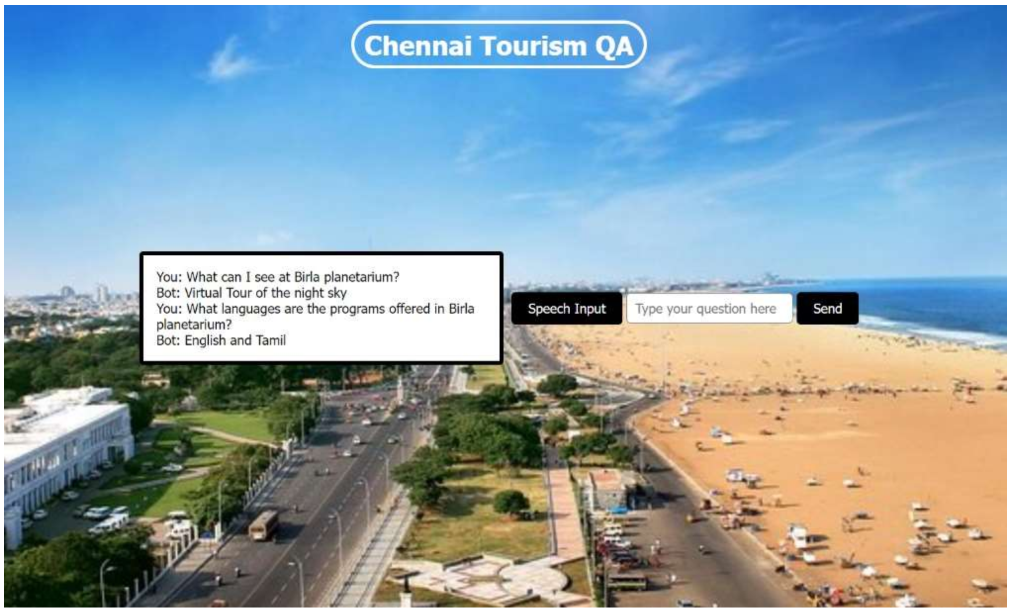
It was also observed that the STT module had some difficulties recognizing Indian names such as “Kapaleeshwarar Temple” and “Parthasarathy Temple”, although the model worked pretty efficiently for Indian accent.



```
Bot: How can I help you?

Bot: Loading...

Question: where is government museum
chennai?

Answer: Egmore neighborhood

Bot: Speak Now...

Bot: Loading...

Question: what relics can I see at
Government Museum Chennai
```

```
where is government museum chennai?
```

Figure 20: User Interface for using the chatbot on edge



```
You: What can I see at Birla planetarium?
Bot: Virtual Tour of the night sky
You: What languages are the programs offered in Birla
planetarium?
Bot: English and Tamil
```

Figure 21: Webpage built for the chatbot

6.3 SUMMARY

This chapter explores integrating Automatic Speech Recognition (ASR) and Text-to-Speech (TTS) functionalities into the Chennai tourism question-answering system. ASR allows computers to understand spoken language and convert it to text. ASR systems typically involve two stages, Acoustic Modelling and Language Modelling.

Acoustic Modeling breaks down speech into fundamental units (phonemes) and analyzes them statistically to identify the most likely sounds. Language Modeling considers grammar and word probabilities to refine the recognized text based on context, improving naturalness.

Considering factors like computational resources, training requirements, and language support, Google Cloud's Speech-to-Text and Text-to-Speech services were chosen for this project. This avoids the need for training or fine-tuning deep learning models (often speaker-dependent) and straining local computational resources.

Python modules like pyttsx3, pyaudio, and speech_recognition were used for speech input, output, and text conversion, respectively.The chapter showcases the code for speech-to-text conversion and the overall question-answering pipeline with speech input and output functionalities.

The STT module faced some difficulties recognizing specific Indian names like "Kapaleeshwarar Temple" and "Parthasarathy Temple," despite working well with an Indian accent overall. This highlights potential areas for future improvement in speech recognition accuracy for regional names and accents.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

7.1 CONCLUSIONS

This study successfully developed a question-answering (Q&A) system specifically designed to address tourism inquiries related to Chennai, India. The system leverages the power of fine-tuning a large language model (LLM), Roberta, and integrates Automatic Speech Recognition (ASR) and Text-to-Speech (TTS) functionalities. This approach provides a user-friendly and informative experience for tourists, enabling them to interact with the system naturally through spoken language.

To effectively answer tourism-related questions, we fine-tuned Roberta on a curated dataset specifically focused on Chennai. This dataset comprised 320 question-answer (Q&A) pairs meticulously divided into a training set of 270 examples and a testing set of 50 examples. The fine-tuning process involved iteratively exposing Roberta to these Q&A pairs for 20 epochs. This targeted training resulted in a significant loss reduction to 0.0178 at the end of 20 epochs. Loss is a metric used to measure the difference between the model's predicted output and the actual answer (ground truth). A lower loss indicates the model is learning effectively and minimizing errors.

To assess the effectiveness of the fine-tuned Roberta model, a comprehensive evaluation was conducted. The model's performance was measured on both the training set (270 Q&A pairs) and the testing set (50 Q&A pairs) not seen during training. This ensures an unbiased evaluation of the model's ability to generalize to unseen data.

We employed two key metrics to evaluate answer accuracy: Jaccard similarity and Levenshtein distance. Jaccard similarity measures the overlap between the sets of words in the expected answer (ground truth) and the model's generated answer. It provides a value between 0 and 1, with 1 indicating complete overlap (identical answers) and 0 signifying no overlap.

Levenshtein distance, on the other hand, calculates the minimum number of edits (insertions, deletions, or substitutions) required to transform the model's answer into the expected answer. A lower Levenshtein distance suggests minimal editing is needed for an exact match.

The model achieved an impressive accuracy of 88.125% on the combined testing and training sets based on the Jaccard similarity and Levenshtein distance metrics. This demonstrates the model's ability to retrieve relevant and accurate answers to user queries concerning Chennai tourism. Therefore, this paves the way for a user-centric and informative Q&A system that empowers tourists to explore Chennai with ease.

7.2 FUTURE WORK

While the ASR module performed well with an Indian accent, there's room for improvement in recognizing specific regional names. Further training on datasets rich in such names could enhance its accuracy. Moreover, the training dataset can be expanded to include more places for tourism other than Chennai, expanding our region of focus.

Multilingual Support can also be a future area of research. Expanding the system's language capabilities to cater to tourists from diverse backgrounds would significantly broaden its reach and usefulness.

Considering India has 22 official languages, multilingual support for these languages can also be provided. Moreover, by regularly incorporating new tourism information and user feedback into the fine-tuning process can ensure the system remains up-to-date and delivers the most relevant information to users.

By addressing these areas for future work, this question-answering system has the potential to become an invaluable resource for tourists exploring Chennai, offering a seamless and informative experience throughout their journey.

REFERENCES

1 Haji, S., Suekane, K., Sano, H. and Takagi, T., 2023, February. Exploratory Inference Chain: Exploratorily Chaining Multi-hop Inferences with Large Language Models for Question-Answering. In 2023 IEEE 17th International Conference on Semantic Computing (ICSC) (pp. 175-182). IEEE.

2 Kakwani, D., Kunchukuttan, A., Golla, S., Gokul, N.C., Bhattacharyya, A., Khapra, M.M. and Kumar, P., 2020, November. IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages. In Findings of the Association for Computational Linguistics: EMNLP 2020 (pp. 4948-4961).

3 Wu, L., Chen, Y., Shen, K., Guo, X., Gao, H., Li, S., Pei, J. and Long, B., 2023. Graph neural networks for natural language processing: A survey. Foundations and Trends® in Machine Learning, 16(2), pp.119-328.

4 Li, R., Wang, L., Jiang, Z., Hu, Z., Zhao, M. and Lu, X., 2022. Mutually improved dense retriever and GNN-based reader for arbitrary-hop open-domain question answering. Neural Computing and Applications, 34(14), pp.11831-11851.

5 Khattab, O., Potts, C. and Zaharia, M., 2021. Relevance-guided supervision for openqa with colbert. Transactions of the association for computational linguistics, 9, pp.929-944.

6 Yasunaga, M., Ren, H., Bosselut, A., Liang, P. and Leskovec, J., 2021. QA-GNN: Reasoning with language models and knowledge graphs for question answering. arXiv preprint arXiv:2104.06378.

7 Wu, Y., Zhou, Q., Yin, H. and Liu, D., 2022, July. An Interpretable Question Answering Method Based on Heterogeneous Graph Neural Networks. In 2022 International Conference on Computer Engineering and Artificial Intelligence (ICCEAI) (pp. 576-580). IEEE.

8 Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A. and Agarwal, S., 2020. Language models are few-shot learners. Advances in neural information processing systems, 33, pp.1877-1901.

9 Liang, Z., Khot, T., Bethard, S., Surdeanu, M. and Sabharwal, A., 2022. Better retrieval may not lead to better question answering. *arXiv preprint arXiv:2205.03685*.

10 Banerjee, P., Pal, K.K., Mitra, A. and Baral, C., 2019. Careful selection of knowledge to solve open book question answering. *arXiv preprint arXiv:1907.10738*.

11 Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need. Advances in neural information processing systems. *arXiv preprint arXiv:1706.03762*

12 Devlin, J., Chang, M.W., Lee, K. and Toutanova, K., 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805.*

13 Wei, J., Bosma, M., Zhao, V., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., & Le, Q. V. (2022). Finetuned Language Models are Zero-Shot Learners. In International Conference on Learning Representations (pp. 1-9). OpenReview.

14 Nakano, R., Hilton, J., Balaji, S., Wu, J., Long, O., Kim, C., Hesse, C., Jain, S., Kosaraju, V., Saunders, W., Jiang, X., Cobbe, K., Eloundou, T., Krueger, G., Button, K., Knight, M., Chess, B., & Schulman, J. (2021). WebGPT: Browser-assisted question-answering with human feedback. ArXiv, abs/2112.09332. CorpusID:245329531

15 Chung, H.W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y., Wang, X., Dehghani, M., Brahma, S. and Webson, A., 2024. Scaling instruction-finetuned language models. Journal of Machine Learning Research, 25(70), pp.1-53.

16 Liang, P., Bommasani, R., Lee, T., Tsipras, D., Soylu, D., Yasunaga, M., Zhang, Y., Narayanan, D., Wu, Y., Kumar, A. and Newman, B., 2022. Holistic evaluation of language models. arXiv preprint arXiv:2211.09110.