*A project report on*

# TACKLING COMPLEX SCIENTIFIC QUESTIONS USING LARGE LANGUAGE MODEL

*Submitted in partial fulfillment for the award of the degree of*

## M Tech (Integrated) Computer Science and Engineering with Specialization in Business Analytics

*by*

**T S S ABINANDHAN KUMAR**

**19MIA1062**

**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

December, 2023

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)
CHENNAI

# DECLARATION

I here by declare that the thesis entitled "TACKLING COMPLEX SCIENTIFIC QUESTIONS USING LARGE LANGUAGE MODEL" submitted by me, for the award of the degree of M Tech (Integrated) Computer Science and Engineering with Specialization in Business Analytics, Vellore Institute of Technology, Chennai, is a record of bonafide work carried out by me under the supervision of "Dr. RAJESH R"

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

**Place: Chennai**

**Date: 18/12/2023**

**Signature of the Candidate**
T S S Abinandhan Kumar

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CHENNAI

**School of Computer Science and Engineering**

# CERTIFICATE

This is to certify that the report entitled **"TACKLING COMPLEX SCIENTIFIC QUESTIONS USING LARGE LANGUAGE MODEL"** is prepared and submitted by **T S S ABINANDHAN KUMAR** (**19MIA1062**) to Vellore Institute of Technology, Chennai, in partial fulfillment of the requirement for the award of the degree of **M Tech (Integrated) Computer Science and Engineering with Specialization in Business Analytics** programme is a bonafide record carried out under my guidance. The project fulfills the requirements as per the regulations of this University and in my opinion meets the necessary standards for submission. The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma and the same is certified.

Signature of the Guide:

Name: Dr./Prof. RAJESH R

Date: 18/12/2023

| | |
|---|---|
| Signature of the Examiner 1 | Signature of the Examiner 2 |
| Name: Dr. Ilavendan A | Name: Dr. Padmanaban R |
| Date: 18/12/2023 | Date: 18/12/2023 |

Approved by the Head of Department

# ABSTRACT

This initiative takes a multimodal approach to solving complex scientific questions. It includes twelve different kinds of analyses that address different aspects of feature engineering, model evaluation, and data exploration. Through the use of an intuitive interface, the system makes it possible for users to engage with the underlying models in a seamless manner, which promotes effective query processing and result retrieval. Important elements are the application of Large Language Models (LLMs) for deep contextual understanding and Long Short-Term Memory (LSTM) networks for complicated sequence comprehension, both of which are essential for producing accurate answers to hard scientific questions. This project establishes a complete framework that combines state-of-the-art analytical methods, user-centered design, and state-of-the-art language models to solve complex scientific problems.

# ACKNOWLEDGEMENT

It is my pleasure to express with deep sense of gratitude to **Dr RAJESH R**, **assistant professor**, SCOPE, Vellore Institute of Technology, Chennai, for his constant guidance, continual encouragement, understanding; more than all, he taught me patience in my endeavor. My association with him is not confined to academics only, but it is a great opportunity on my part of work with an intellectual and expert in the field of Business Analytics and Data science.

It is with gratitude that I would like to extend thanks to our honorable Chancellor, Dr. G. Viswanathan, Vice Presidents, Mr. Sankar Viswanathan, Dr. Sekar Viswanathan and Mr. G V Selvam, Assistant Vice-President, Ms. Kadhambari S. Viswanathan, Vice-Chancellor In-charge, Dr. V. S. Kanchana Bhaaskaran and Additional Registrar, Dr. P K Manoharan for providing an exceptional working environment and inspiring all of us during the tenure of the course.

Special mention to Dean, Dr. Ganesan R, Associate Dean Academics, Dr. Parvathi R and Associate Dean Research, Dr. Geetha S, SCOPE, Vellore Institute of Technology, Chennai, for spending their valuable time and efforts in sharing their knowledge and for helping us in every aspect.

In jubilant mood I express ingeniously my whole-hearted thanks to

**Dr Sivabalakrishnan M**, Head of the Department, Project Coordinator, Dr. **Yogesh C**, SCOPE, Vellore Institute of Technology, Chennai, for their valuable support and encouragement to take up and complete the thesis.

My sincere thanks to all the faculties and staff at Vellore Institute of Technology, Chennai, who helped me acquire the requisite knowledge. I would like to thank my parents for their support. It is indeed a pleasure to thank my friends who encouraged me to take up and complete this task.

**Place: Chennai**

**Date: 18/12/2023** **Name of the student**

T S S Abinandhan Kumar

# CONTENTS

**CHAPTER 6**

**CONCLUSION AND FUTURE ENHANCEMENTS**

## LIST OF ACRONYMS

1. **LLM – Large Language Model**
2. **LSTM – Long Short-Term Memory**
3. **GPT - Generative Pre-trained Transformer**
4. **NLP – Natural Language Processing**
5. **BERT - Bidirectional Encoder Representations from Transformers**
6. **RNN - Recurrent Neural Network**
7. **AI – Artificial Intelligence**
8. **XL-Net - Generalized Autoregressive Pretraining for Language Understanding**
9. **T5 - The Text-to-Text Transfer Transformer**
10. **ML – Machine Learning**
11. **CSS – Computational social systems**
12. **UI – User Interface**
13. **API – Application Interface**
14. **ToM – Theory of Mind**
15. **VAE – Variational Autoencoders**
16. **UX – User Experience**
17. **JS – Java Script**
18. **POST – Power On Self-Test**
19. **MAPE – Mean Absolute Percentage Error**
20. **CPU – Central Processing Unit**
21. **TPU – Tensor Processing Units**
22. **PC – Personal Computer**
23. **RMSE – Root Mean Squared Error**
24. **MAE – Mean Absolute Error**

# CHAPTER 1

# INTRODUCTION

## 1.1 INTRODUCTION TO LARGE LANGUAGE MODELS:

Large Language Models (LLMs) are sophisticated artificial intelligence models that can comprehend and produce language similar to that of humans by utilizing deep learning methods, especially neural networks. These models are able to learn complex patterns, correlations, and linguistic nuances since they have been trained on vast amounts of text data from many sources. By demonstrating exceptional ability in a range of language-related tasks, such as text generation, summarization, translation, sentiment analysis, question answering, and more, LLMs have transformed natural language processing tasks.

Transformer designs allow LLMs, such GPT (Generative Pre-trained Transformer) models, like GPT-3, to efficiently understand contextual information over long text sequences. They are incredibly skilled at comprehending and producing coherent and contextually relevant writing because they make use of self-attention processes, which enable them to concentrate on pertinent portions of the input text and grasp long-range dependencies.

These models are able to gain a thorough understanding of language patterns and nuances since they are usually pre-trained on large-scale corpora, which frequently involve internet-scale data. Then, optimizing for particular downstream activities improves their performance and flexibility for domain-specific applications. Due to their strong language comprehension skills, which allow for automation, information extraction, and the augmentation of human-generated material, language learning machines (LLMs) have had a substantial impact on a number of industries, including healthcare, finance, education, and customer service.

**features of Large Language Models (LLMs):**

- **Scalability**: LLMs with greater computational power, such as GPT models, are built to grow. They may perform better and understand language better if they are trained on bigger datasets and use more complex model architectures.

- **Transfer Learning:** A transfer learning paradigm is used in these models. They can be further refined on smaller, task-specific datasets for specific applications after being pre-trained on a variety of large, diverse datasets to learn general language patterns.

- **Versatility:** Because LLMs can understand and produce language that is similar to that of a human, they are versatile in the way that they can be applied to a wide range of natural language processing tasks, such as text generation, language translation, sentiment analysis, summarization, and more.

- **Ethical and Bias Considerations:** Because LLMs have been trained on a large amount of internet-scale data, it is possible that they will unintentionally pick up on and reinforce biases in the training data. In an effort to produce more moral and equitable language models, researchers and developers are actively addressing these biases.

- **Resource Intensity**: The deployment and maintenance of LLMs are resource-intensive due to the significant computational resources and energy required for training and fine-tuning them.

- **Continuous Advancements**: Research and development are being conducted to improve model architectures, enhance language understanding, reduce biases, and make these models more accessible and efficient. The field of LLMs is rapidly evolving.

- **Open Source and Community Contributions:** A lot of LLMs and related tools are available under an open-source license, which promotes community involvement and collaboration and helps to develop new ideas and advances the field.

- **Real-world Applications:** LLMs have the potential to revolutionize how people interact with technology and information by finding use in a variety of real-world scenarios, including content creation, chatbots, virtual assistants, content recommendation systems, and more.

## 1.2 RELATED WORK

- The article "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" by Devlin et al. discusses the shortcomings of the current natural language processing (NLP) models in terms of their ability to accurately capture bidirectional context. Because of their sequential structure, traditional NLP models—such as unidirectional and shallow bidirectional architectures—had trouble understanding the context of words within a sentence. The authors point out that earlier models, such as OpenAI GPT (Generative Pre-trained Transformer), used a left-to-right architecture that had limitations when it came to capturing bidirectional context. As a result, these models were unable to produce more accurate and contextually relevant word representations. Due to this unidirectional approach, models found it difficult to perform tasks like text classification, natural language inference, and question answering that required a deep understanding of language context.

  The paper's main focus is on the need for a more efficient method of pre-training natural language processing (NLP) models so that they can better understand and represent bidirectional context in text data. In order to address this problem, the concept of BERT (Bidirectional Encoder Representations from Transformers) was put forth. By utilizing bidirectional transformers, the model is able to obtain context from both left and right directions, which improves its comprehension of language context and semantics.

- The main issue is that current language models are not well-suited to novel tasks with sparse or nonexistent examples. Conventional language models needed a significant amount of task-specific data for optimal performance and had difficulty generalizing to different tasks without requiring significant fine-tuning or task-specific training.

  The authors point out that prior language models frequently had trouble picking up new tasks or learning from small amounts of data without requiring significant changes to their architecture or extra training steps. This presented a challenge in real-world situations where it's essential to quickly adapt or learn from small examples.

  The paper's main focus is on the need for language models with enhanced few-shot learning capabilities that can carry out a variety of tasks and learn from sparse examples or instructions. In response to this challenge, GPT-3 (Generative Pre-trained Transformer 3) was presented. It showed remarkable few-shot learning capabilities by understanding tasks with little context or examples, displaying greater adaptability and versatility across a variety of tasks without requiring a great deal of fine-tuning.

- The problem statement of Yang et al.'s paper "XLNet: Generalized Autoregressive Pretraining for Language Understanding" (2019) focuses on how current language models can't capture bidirectional context effectively while retaining the benefits of autoregressive models. Conventional autoregressive models have the drawback of having to pay attention to previous tokens during generation, which can cause context fragmentation and information loss. This is also the case for unidirectional Transformer-based models.

  The authors set out to create a new language model that would preserve the benefits of autoregressive modeling while also being able to capture bidirectional contexts more accurately. In an effort to overcome the drawbacks of the current models, they suggested XLNet, a generalized autoregressive pretraining technique. By taking into account every possible combination of the token sequence during pretraining, this approach aims to improve comprehension of text sequences and increases the model's ability to identify bidirectional dependencies.

The main goal of this paper's problem statement is to create a language model that can accurately represent bidirectional contexts without sacrificing autoregressive characteristics. With the use of a permutation-based training strategy, XLNet sought to overcome the drawbacks of conventional unidirectional or bidirectional models by capturing rich contextual information throughout sequences. The ultimate goal of this was to use a deeper comprehension of bidirectional dependencies within text sequences to enhance the model's performance in a variety of natural language understanding tasks.

- The problem statement of Raffel et al.'s paper "T5: Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer" (2020) centers on investigating the potential and boundaries of transfer learning through the use of a unified text-to-text Transformer architecture. The authors' goal was to create a single model that wouldn't need any special architectures or fine-tuning techniques in order to be used for a variety of text-related tasks.

  The main goal was to determine whether it would be possible and efficient to create a single text-to-text transformer model that could be used for a variety of natural language processing (NLP) tasks, including text classification, summarization, translation, and question answering. The Text-to-Text Transfer Transformer (T5) architecture was put forth by the authors. It functions in a text-to-text fashion, with each task being expressed as a text generation problem.

  This paper aims to evaluate the suitability of a unified transformer architecture for a variety of natural language processing tasks. T5 attempted to simplify the process of resolving various NLP tasks by framing them as text production tasks. By investigating this method, the authors hope to make it easier to apply transfer learning to different tasks and possibly achieve state-of-the-art results across a number of benchmarks using just one model architecture and training scheme.

- The problem statement of Khashabi et al. (2020)'s paper "UNIFIEDQA: Crossing Format Boundaries with a Single QA Model" addresses the constraints brought about by the various formats of question-answering (QA) datasets. The question formats of traditional QA datasets are frequently different, which makes it difficult to build a single model that can efficiently respond to questions in a variety of formats.

  The primary goal of the work is to develop a single question-answering model that can manage various question formats without the need for fine-tuning techniques or task-specific architectures. The authors set out to create a single quality assurance model that functions uniformly across various quality assurance datasets, irrespective of the formatting variations found in those datasets.

  Developing a model architecture and training methodology that can understand and react to questions in different formats, including multiple-choice, free-form, and cloze-style questions, is the problem this research aims to solve. The writers sought to promote the creation of flexible QA systems that exhibit strong performance over a range of QA datasets by standardizing the method for handling various QA formats.

## 1.3 LITERATURE REVIEW

- **"Language Models are Unsupervised Multitask Learners"**

Without explicit training on the WebText dataset's examples, language models trained on the dataset demonstrate unsupervised learning in tasks such as summarizing and answering questions, achieving 55 F1 on CoQA. Zero-shot task transfer is strongly influenced by the model's capacity; larger models, such as GPT-2, demonstrate state-of-the-art performance in language modeling on a variety of datasets. GPT-2 produces coherent text samples even when WebText is underfit, suggesting that natural language demonstrations alone may be used to develop task-performing language systems.

- **"Language Models are Few-Shot Learners"**

In Brown et al.'s paper "Language Models are Few-Shot Learners" (2020), the potential of language models is examined, with a special emphasis on their capacity for few-shot learning—a machine learning paradigm in which models are trained on small amounts of data and subsequently perform tasks or make predictions with little further training. Overview of Language Models: The paper explores the state of language models today, highlighting important architectures such as Transformers as well as their evolution and advancements. It explains the evolution of these models from simple language comprehension tasks to more difficult ones, laying the foundation for the theory of few-shot learning.

**Gaps in Current Language Models:** It draws attention to the shortcomings of traditional language models in situations where they must be modified for novel contexts or domains with scant data. The authors point out the weaknesses in using currently available pre-trained models for tasks where it is difficult to fine-tune on small datasets.

**Examining the Potential of Language Models:** This paper examines how well language models, like GPT (Generative Pre-trained Transformer), can pick up new skills from a small number of examples in a variety of tasks, including text classification, question answering, and language translation. It describes tests and evaluations showing that the models can generalize from small amounts of data.

**Identification of Research Gaps:** The paper identifies gaps in our knowledge of how language models generalize, transfer knowledge, and adapt to new tasks with limited training data by examining few-shot learning abilities. It emphasizes the need for more thorough research into the processes that allow for few-shot learning and the ways in which models can be tailored for these kinds of situations.

- "**Evaluation of ChatGPT for NLP-based Mental Health Applications"**

The study "Evaluation of ChatGPT for NLP-based Mental Health Applications" investigates the possible applications of the language model ChatGPT for therapy and support related to mental health. This paper's literature review explores several important areas, including:

**Current Uses of Natural Language Processing (NLP) in Mental Health:** This survey examines the body of research on the use of NLP techniques in mental health support. It looks at the applications of NLP models in the field of mental health, such as sentiment analysis, emotion detection, therapy support, and chatbots that try to help people with mental health issues.

**Conversational agents and chatbots in mental health support and therapy** are the main topics of ChatGPT and Conversational Agents in Mental Health. In order to facilitate conversations about mental health, the survey addresses the possible advantages and disadvantages of using language models such as ChatGPT. These include the ability to comprehend user sentiments, respond with empathy, and provide relevant support.

**Research Gaps and Limitations:** The survey highlights research gaps and limitations related to NLP-based mental health applications in the current body of literature. This could involve questions about the moral application of AI in mental health, privacy and data security concerns, the accuracy of the technology in recognizing complex emotional states, and the demand for more individualized and sympathetic responses.

The **main theme** of the discussion is the assessment of ChatGPT, particularly as it relates to applications related to mental health. The survey describes the methods used to evaluate ChatGPT's comprehension of conversations pertaining to mental health, as well as its ability to deliver pertinent information, show empathy, and provide suitable support.

**Problems and Future Directions**: The survey identifies the difficulties in using ChatGPT for mental health applications, including the bias of the model, the absence of domain-specific expertise, and the requirement for ongoing learning and modification. Additionally, it suggests possible directions for further study and advancements in utilizing ChatGPT for better mental health assistance.

- "**Towards Automated Urban Planning: When Generative and ChatGPT-like AI Meets Urban Planning"**

**State of Urban Planning Technology at the Moment:** The study examines the methods and tools currently used in urban planning. It examines the software, data analytics, and traditional approaches now used in the industry, pointing out flaws and opportunities for development.

**Applications of Artificial Intelligence (AI) in Urban Planning**: This section reviews prior studies and AI applications in urban planning, covering a range of fields including simulations, data analysis, predictive modeling, and decision support systems. This covers research on the application of AI to land use prediction, infrastructure development, traffic management, and environmental impact assessments.

**Generative Models in Urban Planning:** The use of generative models, such as GANs (Generative Adversarial Networks), VAEs (Variational Autoencoders), and other comparable AI-based methods, in urban planning is explored in this survey. It goes over how these models create artificial intelligence, mimic urban environments, and support the processes of urban planning and development.

**Role of ChatGPT-like AI in Urban Planning:** ChatGPT-like AI's Place in Urban Planning The study looks into the possible applications of ChatGPT-like conversational AI models in urban planning. It looks at how these models can help with scenario talks, decision-making, public participation, and gathering input from the community for urban planners, architects, and policymakers.

**Vulnerabilities in Current Urban Planning Techniques**: The survey points out weaknesses and gaps in the technologies and methods used in urban planning today. This could involve shortcomings in data analysis, an inability to support decision-making in real time, difficulties engaging the community, and the requirement for more user-friendly and accessible tools for both planners and citizens.

**Theme - Integration of AI for Enhanced Urban Planning**: The integration of conversational AI akin to ChatGPT and generative AI models into urban planning practices is the main theme of "Integrated AI for Enhanced Urban Planning." It investigates how these technologies might work in concert to enhance decision-making, community engagement, and the general effectiveness of urban development procedures.

- **“Accelerating the integration of ChatGPT and other large- scale AI models into biomedical research and healthcare”**

The goal of the paper "Accelerating the Integration of ChatGPT and Other Large-Scale AI Models into Biomedical Research and Healthcare" is to give an overview of the current state of integration of cutting-edge AI models into the fields of biomedical research and healthcare. Specifically, the paper focuses on ChatGPT and related large-scale language models. This paper's literature survey is made up of various important parts:

**AI Integration in Biomedical Research and Healthcare:** The integration of artificial intelligence (AI) in biomedical research and healthcare settings is examined, with a focus on the function of large-scale language models such as ChatGPT. This covers patient-doctor interactions, clinical decision support systems, medical data analysis, natural language processing (NLP) apps, and other AI-driven healthcare solutions.

**Benefits and Advantages:** The poll outlines the possible gains and advantages of using ChatGPT and related AI models in healthcare and biomedical research. This could include more precise diagnosis, better patient care through tailored treatment regimens, effective information retrieval, and creative methods for examining medical literature.

**Limitations and Challenges**: It lists the restrictions and difficulties related to the application of AI models such as ChatGPT in the biomedical and healthcare fields. Data privacy, model interpretability, bias reduction, regulatory compliance (e.g., HIPAA), domain-specific fine-tuning requirements, and ethical issues with AI use in healthcare are a few examples of these concerns.

**Ethical and Regulatory Considerations:** The survey delves into the ethical ramifications, regulatory structures, and directives that oversee the implementation of artificial intelligence models in the healthcare sector. It covers concerns about patient privacy, permission, openness, and the moral application of AI-generated data to medical judgment.

**Gaps in the Biomedical Research's AI Integration:** It points out any holes or places where the current application of ChatGPT and related AI models in healthcare and biomedical research might be deficient. This includes restrictions on the range of medical specialties in which AI models can be applied, deficiencies in the quality or availability of data, and difficulties in integrating AI-based research into clinical practice.

**Theme:** Using AI to Integrate Healthcare Better The main focus is on using ChatGPT and other advanced AI models to improve healthcare and biomedical research. Through addressing obstacles, identifying opportunities, and emphasizing the possible influence on patient outcomes and healthcare delivery, it seeks to hasten the adoption of these models.

- “**Distinguishing Human Generated Text From ChatGPT Generated Text Using Machine Learning”**

Using machine learning techniques, the paper "Distinguishing Human Generated Text From ChatGPT Generated Text Using Machine Learning" addresses the crucial task of distinguishing text generated by AI models—specifically, ChatGPT—from text generated by humans. This paper's literature review most likely includes the following elements:

**Approaches and Methodologies Now in Use:** The survey starts off by looking at the approaches and methodologies currently in use for telling human-generated text from artificial intelligence-generated text. This covers an overview of conventional techniques, feature engineering, and more recent developments that use machine learning algorithms to identify AI-generated content and classify text.

**Text Generation Function of ChatGPT:** It explores the features and capabilities of ChatGPT and related language models, emphasizing their subtleties, patterns, and language generation capabilities. For the purpose of creating efficient techniques to distinguish AI-generated text from human-generated text, it is imperative to comprehend these characteristics.

**Machine Learning Techniques:** To address the issue of text classification and the distinction between content generated by humans and artificial intelligence (AI), this survey covers a range of machine learning algorithms, models, and techniques. This includes talking about natural language processing models, supervised learning, deep learning architectures, and ensemble techniques designed for this particular task.

**Datasets and Evaluation Metrics:** This section talks about the datasets that are frequently used to train and assess models that are intended to distinguish text generated by artificial intelligence (AI) from human text. It also looks at the evaluation metrics used to evaluate these models' performance, with a focus on metrics like accuracy, precision, recall, and F1-score.

**Gaps in Current Approaches**: The survey finds shortcomings or inadequacies in the approaches being used currently to differentiate text generated by AI from text created by humans. These gaps could include difficulties managing adversarial inputs, addressing biases in training datasets that impact model performance, or handling contextually similar content.

**Theme:** Identifying AI-Generated Text: The main focus is on creating efficient machine learning techniques that can reliably differentiate ChatGPT-generated content from human-written text. In order to address this pressing issue, the paper looks at possible directions for advancement in this field as well as offering insights into current methods.

- **"Awareness and acceptance of ChatGPT as a generative conversational AI for transforming education by Ghanaian academics: A two-phase study"**

**Educational Technology and AI Integration:** The current state of educational technology and the incorporation of artificial intelligence—particularly conversational AI such as ChatGPT—into educational environments are both examined in this survey. It explores the use of AI in education around the world, emphasizing its advantages, disadvantages, and possible applications.

**Knowledge and Attitude toward AI in Education:** It addresses prior research and scholarly works concerning teachers' knowledge, perspectives, and dispositions toward AI-driven teaching aids. Surveys and studies that look at teachers' openness, concerns, and readiness to incorporate AI technologies into their lessons may fall under this category.

**ChatGPT in Educational Settings:** The purpose of the survey is to find out more about how ChatGPT and other conversational AI models are used, recognized, and accepted in educational settings. It talks about earlier studies or projects that used AI to help students learn, teachers with their teaching, or learning activities.

**Pedagogical Implications and Gaps**: It examines the possible benefits, difficulties, and shortcomings of the existing implementations of ChatGPT in educational settings. This section could contain information about how well conversational agents powered by AI support learning objectives and tackle educational difficulties.

**Gaps in Adoption and Implementation:** The study pinpoints the shortcomings, restrictions, or impediments that prevent ChatGPT from being widely used in learning settings, especially in Ghanaian academic settings. It might cover topics like infrastructure constraints, cultural issues, ethical dilemmas, or ignorance that hinder the effective application of AI in education.

**Theme:** Using ChatGPT to Revolutionize Education Examining ChatGPT's awareness, acceptability, and potential transformative effects in Ghanaian academia is the main theme. The purpose of the paper is to close the gaps that have been found and provide insight into how prepared Ghanaian educators are to use and integrate AI tools like ChatGPT in the classroom.

- “**ChatGPT for Higher Education Professional Development: A Guide to Conversational AI”**

  **Professional Development in Higher Education**: This article examines the state of professional development initiatives currently offered to instructors in postsecondary educational establishments. The relevance, difficulties, and strategies used to improve teachers' abilities, expertise, and methods of instruction are covered in this section.

  **Conversational AI in Education:** With a particular focus on higher education, the survey examines prior research and literature on the application of conversational AI in educational settings. It includes studies on the ways in which conversational agents powered by AI can facilitate faculty support, professional development, and administrative tasks in higher education.

**ChatGPT and Educational Applications:** The paper probably explores the particular uses of ChatGPT or related conversational AI models in the context of faculty and staff professional development in higher education. It might go over case studies or examples of how educators are using ChatGPT to share knowledge, learn from one another, and facilitate training.

**Gaps in AI-Driven Conversational Agents for Professional Development**: This section highlights the potential role of AI-driven conversational agents in addressing the deficiencies or gaps in current professional development approaches in higher education. It might go over the restrictions or difficulties that traditional professional development programs present for educators and organizations.

**Theme:** Empowering Higher Education Professionals: Using ChatGPT or conversational AI, the main focus of this theme is on empowering professionals in higher education. The goal of the paper is to offer advice and insights on how to use AI-driven conversational agents to support faculty and staff in higher education settings, improve professional development programs, and improve teaching methodologies.

- “**ChatGPT for Computational Social Systems: From Conversational Applications to Human-Oriented Operating Systems”**

**Computational Social Systems (CSS):** In this section, the literature on CSS is reviewed, and the intersections between social systems and computational techniques such as artificial intelligence (AI), machine learning, and social computing are examined. It looks at studies on social interactions, modeling human behavior, and how technology shapes and comprehends societal dynamics.

**Conversational Applications in Computational Social Systems:** This survey explores earlier research and applications that make use of conversational AI in the context of CSS, such as ChatGPT. It might look at how conversational AI systems improve communication, help analyze social data, and help us understand social behavior.

**Gaps in CSS Applications:** This section highlights areas that need further investigation or work in the field of CSS in relation to the integration of chat applications such as ChatGPT. It may draw attention to shortcomings or difficulties in current studies or applications, especially in areas where conversational AI can make major contributions.

**Theme - Human-Oriented Operating Systems:** The integration of conversational AI, in particular ChatGPT, into computational social systems is the central theme, with the goal of creating more human-centered operating systems. The purpose of the paper is to demonstrate how utilizing AI in social computing can result in systems that are more cognizant of, flexible with, and able to communicate with people.

**Future Paths and Consequences**: It might suggest future paths and consequences for integrating ChatGPT or a related conversational AI with CSS. This section could go over possible uses, moral dilemmas, or difficulties in creating human-oriented operating systems and their wider social effects.

- "**Theory of Mind May Have Spontaneously Emerged in Large Language Models"**

**Theory of Mind (ToM) in AI and Psychology:** This section may summarize the literature in AI and Psychology on Theory of Mind, with an emphasis on how people interpret the mental states of others in order to interpret and forecast behavior. It might investigate how well AI can imitate or simulate this cognitive function.

**Large Language Models (LLMs) and Cognitive Capabilities**: The survey may explore the traits and development of LLMs, talking about both their cognitive and language-generation capacities. It could draw attention to research demonstrating characteristics of LLMs that point to a primitive method of comprehending or forecasting human behavior.

**AI and Emergence of Cognitive Abilities:** This section may discuss studies looking into the factors that lead to the emergence of unexpected cognitive behaviors or abilities in AI models, particularly LLMs. It might go over situations in which LLMs exhibit actions that suggest the beginnings of comprehending or forecasting mental states.

**Gaps in Our Understanding of ToM in LLMs**: It may be very important to pinpoint any existing knowledge or research methodology gaps regarding ToM in LLMs. This section may draw attention to gaps in our knowledge of how LLMs display behaviors resembling those of ToMs or to the difficulties in determining whether AI models truly exhibit cognitive abilities.

**Theme:** ToM's Emergence in LLMs The main focus is on the possible or conjectured appearance of Theory of Mind in large language models. It attempts to look into and talk about situations in which LLMs exhibit behaviors that can be seen as indicating a kind of mental state understanding or prediction.

**Future Research Directions and Implications:** Based on the possible emergence of Theory of Mind in LLMs, this section may suggest future research directions, methodologies, or implications. It might go over the technological, social, and ethical ramifications of AI models displaying Theory of Mind-like cognitive abilities.

- **"ChatGPT and a New Academic Reality: AI-Written Research Papers and the Ethics of the Large Language Models in Scholarly Publishing"**

**AI-Generated Content in Scholarly Publishing:** Examining the body of research on the incorporation of artificial intelligence (AI)-generated content—in particular, ChatGPT or large language models—into academic publications. Studies on the rise of AI-authored papers, their difficulties, and their possible effects on scholarly discourse and information sharing may fall under this category.

**Ethical Implications of AI-Authored Papers:** Examining the moral implications of research articles created by artificial intelligence. Debates about authorship, intellectual property, transparency, biases, and credibility in AI-generated content and its acceptance in academia may be covered in this section.

**High-quality, accurate, and dependable AI-generated research:** Comparing large-scale language model-generated research to human-authored papers for assessment of research quality and dependability. This might draw attention to knowledge gaps regarding the potential and constraints of AI-generated content in terms of fulfilling academic requirements.

**Perception and Acceptance in the Academic Community:** Investigating how academic publications composed by AI are viewed and accepted. This could entail talking about acceptance, skepticism, or the difficulties AI-authored research has in getting noticed.

**Theme**: AI's Ethical Aspects in Scholarly Publishing The main focus of the discussion is the moral conundrums and issues that the incorporation of AI-generated content, especially in academic publishing, raises. It explores potential solutions or guidelines for responsible AI use in scholarly research, as well as the moral and ethical implications.

**Gaps in Knowledge and Future Directions:** Determining the present state of knowledge, problems, or unresolved issues surrounding academic papers composed by artificial intelligence. This segment may suggest avenues for future investigation, models, or suggestions to tackle moral dilemmas and improve the incorporation of artificial intelligence in academic publications.

- "**Chatting about ChatGPT: How may AI and GPT impact academia and libraries?"**

**AI Impact on Academia:** Examining the body of research on the possible effects of artificial intelligence (AI), and more especially GPT-like models, on educational establishments. These might include papers that address how AI is incorporated into learning environments, how it affects teaching strategies, how students learn, and how academic research develops.

**AI Applications in Libraries:** Examining research on the use of AI in libraries, including ChatGPT, and other related applications. This could include research on AI-powered advances in content curation, chatbots for user support, automated cataloging, and information retrieval.

**Ethical Issues and Difficulties**: Examining the moral ramifications of using AI in libraries and educational environments. This could entail talking about intellectual property, data privacy, biases in AI-generated content, and the moral obligations of organizations using AI technology.

**User Experience and Engagement:** Investigating how AI, in particular conversational agents or chatbots driven by GPT models, affects how users interact with and use library services. This could include research on accessibility, user satisfaction, and how well AI-driven services fulfill user needs.

**Theme:** The Effects of AI on Libraries and Academics The main focus is on how AI technologies, particularly those that resemble GPT models, may affect education and libraries. It addresses how artificial intelligence (AI) is transforming education and information services, highlighting both the advantages and disadvantages.

**Gaps and Future Directions**: Determining the present state of knowledge or application of artificial intelligence in educational and library contexts. This section may include suggestions for responsible AI integration, directions for future research, or ways to overcome obstacles and maximize the advantages of AI in these fields.

- "**Study and Analysis of Chat GPT and its Impact on Different Fields of Study"**

**Multidisciplinary Impact Analysis:** Examining research on the use and effects of ChatGPT in a variety of domains (such as healthcare, education, finance, etc.). Research demonstrating the usefulness, difficulties, and possible drawbacks of ChatGPT in various fields may fall under this category.

**Use Cases and Applications:** Examining publications that go over particular use cases and real-world uses for ChatGPT across industries. This could entail studies showing how ChatGPT is used in various industries for decision-making, problem-solving, data analysis, customer service, etc.

**Limitations and Challenges:** Determining the gaps in the literature concerning the restrictions, difficulties, or disadvantages related to the application of ChatGPT in various fields. This could involve talking about biases, moral dilemmas, or circumstances in which ChatGPT might not function as well as it should.

**Theme:** The Significance of ChatGPT in Various Domains: Analyzing the complex effects of ChatGPT across numerous disciplines and industries is the main focus. The goal of the paper is to present a thorough analysis of ChatGPT's uses, advantages, drawbacks, and difficulties across several academic disciplines.

**Possible Future Directions:** Outlining prospective fields for investigation or research in the future where ChatGPT might be improved upon or used. This could entail suggesting methods to lessen restrictions, enhance precision, or make better use of ChatGPT in particular industries.

**Integration of ChatGPT in Various Fields:** This section focuses on the adoption and integration of ChatGPT in various fields and sectors, looks at how its use has changed over time, and discusses the implications for each particular area.

- "**Towards Human-Bot Collaborative Software Architecting with ChatGPT"**

**Human-Bot Collaborative Software Architecting:** Examining previous research and writings on collaborative settings between people and AI bots, particularly in software architecture, is the goal of this project on human-bot collaborative software architecture. This entails examining the ways in which ChatGPT or analogous AI models have been incorporated into the design, planning, and decision-making phases of the software development lifecycle.

**Applications in Software Architecture**: Examining studies and research papers that illustrate ChatGPT's uses and consequences in software architecture. Talks about how ChatGPT helps with design concepts, architectural patterns, code production, documentation, and advice and insights during the software development process could fall under this category.

**Finding Collaboration Gaps:** Evaluating the shortcomings and difficulties in the human-bot collaboration environment that exists today in software architecture. This entails examining areas in which ChatGPT's involvement might be improved, pointing out gaps in its present functionality, addressing moral issues, or speculating about situations in which the partnership might not succeed.

**Theme:** The main theme of the paper is "Human-AI Collaboration in Software Architecting," which focuses on the dynamics of human-bot cooperation specifically in relation to software architecture. This entails analyzing ChatGPT's potential advantages, difficulties, and contribution to the software development lifecycle.

**Improving Efficiency and Collaboration:** Talking about suggested approaches or tactics to make human architects and ChatGPT work together more effectively in order to increase software architecture productivity, accuracy, or creativity.

**Future Directions:** Determining possible lines of inquiry or advancement in the area of software architecture for human-bot cooperation in the future. This can entail suggesting brand-new methods, instruments, or frameworks to boost ChatGPT's efficiency when working on software architecture projects.

## 1.4 PROBLEM STATEMENT

The primary goal of this project is to address the ongoing difficulty of understanding and responding to complex scientific questions. The intricacy and breadth of research in many scientific fields frequently exceed the capacity of traditional systems. This project intends to close this gap by utilizing the enormous knowledge base, natural language understanding, and reasoning powers of Large Language Models (LLMs). The main issue is that the instruments currently in use are inadequate for efficiently processing and interpreting complex scientific questions that cut across disciplines. This project aims to revolutionize the way complex scientific problems are approached and solved by using LLMs to develop a novel solution that allows accurate comprehension, intelligent inference, and insightful responses to intricate scientific inquiries. The objective of this initiative is to provide researchers, academics, and professionals with an intelligent and flexible system that can accurately and thoroughly handle a variety of scientific challenges, thereby advancing the fields of science by improving knowledge acquisition and problem-solving.

## 1.5 PROJECT OVERVIEW

Using Large Language Models (LLMs) to tackle complex and multifaceted questions across various scientific domains is the goal of the project "Tackling Complex Scientific Questions using Large Language Models," which aims to transform scientific inquiry. This initiative's primary goal is to overcome the shortcomings of current methods for understanding and responding to complex scientific questions, which frequently call for contextual reasoning, domain-specific knowledge, and nuanced understanding. The goal of the research is to create an intelligent system that can process, analyze, and provide in-depth answers to complex scientific queries by utilizing the enormous language understanding, contextual learning, and knowledge base of LLMs.

The main goal of this project is to develop a flexible framework that will allow professionals, academics, and researchers to efficiently investigate and analyze complex scientific issues. Through the utilization of cutting-edge machine learning methodologies and cutting-edge language models, the system will enable precise understanding and thoughtful deduction, contributing to the advancement of knowledge and promoting innovations in scientific inquiry. The objective of this project is to democratize access to sophisticated language-based resources, enabling users to investigate intricate scientific questions and ultimately spurring innovation, discovery, and progress in a range of scientific fields. The project's goal is to push the limits of scientific research and problem-solving by providing a strong, clever, and adaptable platform to address the difficulties that come with answering complicated scientific queries.

## 1.6 CHALLENGES PRESENT IN THIS PROJECT:

**Complexity of Scientific Queries:** Handling complex scientific queries requires the system to comprehend and process various forms of scientific knowledge from various domains. This requires understanding multifaceted concepts.

**Knowledge Representation:** Ensuring that the system can access and use large amounts of diverse scientific knowledge is a major challenge, as is efficiently arranging and representing it within the Large Language Models (LLMs).

**Semantic Understanding:** The system must accurately identify complex scientific contexts in order to interpret the nuanced meanings, context, and subtle intricacies embedded within scientific questions. This requires a sophisticated semantic understanding.

**Scalability and Performance:** It is still difficult to manage and process massive amounts of scientific data effectively while preserving scalability and optimal performance, especially when it comes to the size and complexity of large-scale learning models (LLMs).

**Model Interpretability:** Because of the intrinsic complexity of the underlying models, it can be difficult to guarantee the transparency and interpretability of the decisions made by the LLM-based system, particularly in scientific contexts.

**Adaptability to Diverse Domains:** One of the challenges in creating a solution that is universally applicable is designing the system to accommodate a variety of scientific domains while preserving accuracy, relevance, and domain-specific understanding.

**LSTM vs LLM:** Model Complexity and Contextual Understanding in LSTM vs. LLM In comparison to Long Short-Term Memory Models (LSTMs), which concentrate on sequential data and have a particular architecture intended for memory retention, Large Language Models (LLMs) have a more comprehensive contextual understanding capacity. Because LLMs integrate diverse knowledge, complex language structures, and semantics, they can handle a wider range of complex scientific queries due to their larger scale operation. Nonetheless, addressing multi-modal scientific data, comprehending domain-specific jargon, and managing contextual subtleties in scientific texts continue to be significant obstacles for both LSTM and LLM architectures. Assessing their relative effectiveness in answering intricate scientific questions is essential to ascertaining their ideal application in various scientific fields.

**Metrics for Evaluation:** Because scientific problems are multifaceted and user requirements are diverse, it is difficult to design robust evaluation metrics that accurately measure the system's performance in answering complex scientific queries.

**Ethical Considerations:** Ensuring data privacy, bias mitigation, and ethical AI practices in scientific research contexts, as well as the responsible use and deployment of the system, are ongoing challenges.

## 1.7 OBJECTIVES

**Developing a User-Centric Interface:** Provide a user-friendly interface that enables users to enter challenging scientific queries and get precise answers.

**Integration of LSTM and LLM**: Examine and combine the powers of Large Language Models (LLMs) and Long Short-Term Memory (LSTM) networks to enhance the resolution of scientific queries.

**Optimization of LLM Architectures:** Test different pre-trained LLM architectures and optimize them with different tuning techniques to improve understanding of scientific questions.

**Increasing the Robustness of the Model:** To increase the robustness of the model, tackle issues like multi-modal data, diverse scientific contexts, and terminology specific to a particular domain.

**Assessment and Validation:** Perform comprehensive assessments, contrasting the effectiveness of hybrid, LLM, and LSTM models in answering challenging scientific questions.

**Scalability and Generalization:** Examine strategies for optimizing models' performance and guaranteeing their applicability to a wide range of scientific fields.

**Performance Metrics and Assessment**: Establish and put into practice assessment metrics to measure the effectiveness of the model and its relevance in answering scientific questions.

**User Interaction Analysis:** Using feedback and query patterns, analyze user interactions to improve the user experience and interface.

**Cross-Domain Applicability:** Evaluate models' applicability to various scientific fields, noting their advantages and disadvantages.

**Ethical Considerations and Bias Mitigation**: In order to achieve responsible deployment, examine ethical issues pertaining to data bias, fairness, and transparency in model predictions.

**Documentation and Knowledge Sharing:** To assist researchers and developers in using LLMs for scientific inquiry, produce thorough documentation and exchange best practices.

**Real-world Implementation:** Investigate ways that LLMs might support scientific research and decision-making in the real world.

**Cooperation and Community Engagement:** Promote cooperation with the scientific community to improve model applicability, gather a variety of viewpoints, and refine models.

**Model Explainability and Interpretability**: To help users comprehend the rationale behind model responses, strive for interpretability in model predictions.

**Continuous Model Improvement:** For long-term model enhancement, apply iterative updates, modifications, and adaptations based on continuing assessment and user input.

## 1.8 SCOPE OF THE PROJECT

The project "Tackling Complex Scientific Questions using Large Language Models" is a comprehensive investigation that uses state-of-the-art Natural Language Processing (NLP) methods. The ambitious goal of this project is to create an interface that is easy to use and intuitive so that complex scientific queries can be submitted. It is utilizing the capabilities of cutting-edge Large Language Models (LLMs) and combining them with Long Short-Term Memory (LSTM) networks to improve understanding and answering of difficult scientific questions. The scope of the project includes optimizing LLM architectures and addressing issues related to specialized terminology and various scientific contexts.

Additionally, it concentrates on assessing and verifying the effectiveness and precision of LLMs and LSTM-LLM hybrids in answering scientific questions in a variety of fields. The project also intends to explore the ethical issues related to the application of these models, as well as their scalability and generalization. Part of the project's broad scope is a collaborative engagement with the scientific community and a commitment to ongoing improvement and documentation. Its goal is to guarantee interpretability and explainability in model predictions while opening the door for practical applications of LLMs in a variety of scientific research and decision-making domains.

# CHAPTER 2

# USER INTERFACE DEVELOPMENT

## 2.1 DESIGNING USER CENTRIC INTERFACE

server.js × | .env.example | .env

server.js > ...

```javascript
import express from "express";
import * as dotenv from "dotenv";
import cors from "cors";
import { Configuration, OpenAIApi } from "openai";

dotenv.config();

const port = 5000;

const configuration = new Configuration({
    apiKey: process.env.OPENAI_API_KEY,
});

const openai = new OpenAIApi(configuration);

const app = express();
app.use(cors());
app.use(express.json());

app.get('/', async (req, res) => {
    res.status(200).send({
        message: 'THIS INTERFACE IS BUILT BY ABINANDHAN KUMAR AND ITS OPEN TO ALL FOR',
    })
});

app.post('/', async (req, res) => {
    try {
        const prompt = req.body.prompt;

        const response = await openai.createCompletion({
            model: "text-davinci-003",
            prompt: `${prompt}`,
            temperature: 0,
            max_tokens: 3000,
            top_p: 1,
            frequency_penalty: 0.5,
            presence_penalty: 0,
        });

        res.status(200).send({
            bot: response.data.choices[0].text
        });
    } catch (error) {
        console.log(error);
        res.status(500).send(error);
    }
```

JS server.js × $ .env.example ● .env

JS server.js > ...

```javascript
app.get('/', async (req, res) => {
    res.status(200).send({
        message: 'THIS INTERFACE IS BUILT BY ABINANDHAN KUMAR AND ITS OPEN TO ALL FOR',
    })
});

app.post('/', async (req, res) => {
    try {
        const prompt = req.body.prompt;

        const response = await openai.createCompletion({
            model: "text-davinci-003",
            prompt: `${prompt}`,
            temperature: 0,
            max_tokens: 3000,
            top_p: 1,
            frequency_penalty: 0.5,
            presence_penalty: 0,
        });

        res.status(200).send({
            bot: response.data.choices[0].text
        });
    } catch (error) {
        console.log(error);
        res.status(500).send(error);
    }
})

app.listen(port,
    () => console.log(`Server is running on http://localhost:${port}`)
);
```

JS script.js ×

JS script.js > ...

```
import bot from "./assets/bot.svg";
import user from "./assets/user.svg";

const form = document.querySelector('form');
const chatContainer = document.querySelector('#chat_container');
const serverApi = "http://localhost:5000/";

let loadInterval;

/**
 * This will show ... as a loading animation when processing
 * @param {object} element
 */
function loader(element) {
  element.textContent = '';

  //every 300ms  it will add '.'
  loadInterval = setInterval(() => {
    element.textContent += '.';

    //resetting textContent
    if (element.textContent === '....') {
      element.textContent = '';
    }
  }, 300);
}

/**
 * When Ai has an answer answer will write letter by letter
 * @param {object} elemet
 * @param {string} text
 */
function typeText(elemet, text) {
  let index = 0;

  let interval = setInterval(() => {
    if (index < text.length) {
      elemet.innerHTML += text.charAt(index);
      index++;
    } else {
      clearInterval(interval);
    }
  }, 20);
}

/**
```

```
/**
 * Will generate unique ID for question
 * @returns unique id string
 */
function generateUniqueId() {
  const timeStamp = Date.now();
  const randomNumber = Math.random();
  const hexadecimalString = randomNumber.toString(16);

  return `id-${timeStamp}-${hexadecimalString}`
}

/**
 * Generate chat line among bot and user
 * @param {boolean} isAi
 * @param {string} value
 * @param {string} uniqueId
 * @returns template string of code
 */
function chatStripe(isAi, value, uniqueId) {
  return (
    `
      <div class="wrapper ${isAi && 'ai'}">
        <div class="chat">
          <div class="profile">
            <img src="${isAi ? bot : user}"
              alt="${isAi ? 'bot' : 'user'}" />
          </div>
          <div class="message" id=${uniqueId}>${value}</div>
        </div>
      </div>
    `
  )
}

/**
 * When submit button
 * @param {event} e
 */
const handleSubmit = async (e) => {
  e.preventDefault();

  const data = new FormData(form);

```

JS script.js ×

JS script.js > ...

```javascript
    //User's chat stripe
    chatContainer.innerHTML += chatStripe(false, data.get('prompt'));
    form.reset();

    //BotChat stripe
    const uniqueId = generateUniqueId();
    chatContainer.innerHTML += chatStripe(true, " ", uniqueId);
    chatContainer.scrollTop = chatContainer.scrollHeight;

    const messageDiv = document.getElementById(uniqueId);

    loader(messageDiv);

    //Fetch data from server
    const response = await fetch(serverApi, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json'
      },
      body: JSON.stringify({
        prompt: data.get('prompt')
      }),
    })

    //Clear interval and add empty string to message div
    clearInterval(loadInterval);
    messageDiv.innerHTML = '';

    if (response.ok) {
      const data = await response.json();
      const parsedData = data.bot.trim();

      typeText(messageDiv, parsedData);
    } else {
      const err = response.text();
      messageDiv.innerHTML = "Something went wrong!";
      console.log(err);
    }
  }

  /**
   * add event listners and callback functions in enterkey pressed
   */
  form.addEventListener('submit', handleSubmit);
  form.addEventListener('keyup', (e) => {
    if (e.keyCode === 13 && !e.shiftKey) {
      handleSubmit(e);
    }
  })
```

The project's user interface development is not complete without the included code snippets. Through a user-centric interface, they enable complex scientific queries to be submitted and processed, thereby facilitating interaction between users and the backend system.

**User Interface Interaction Code**

The functions that control user input, message display, and communication with the AI-driven backend are defined in the JavaScript code snippet. An overview of its features is provided below:

Event Handling: User interactions are effectively managed by event listeners for form submission and important events (such as hitting Enter).

Chat Interface: The chat-Stripe function creates chat message containers and uses layout structures and unique identifiers to differentiate between responses generated by AI and user input.

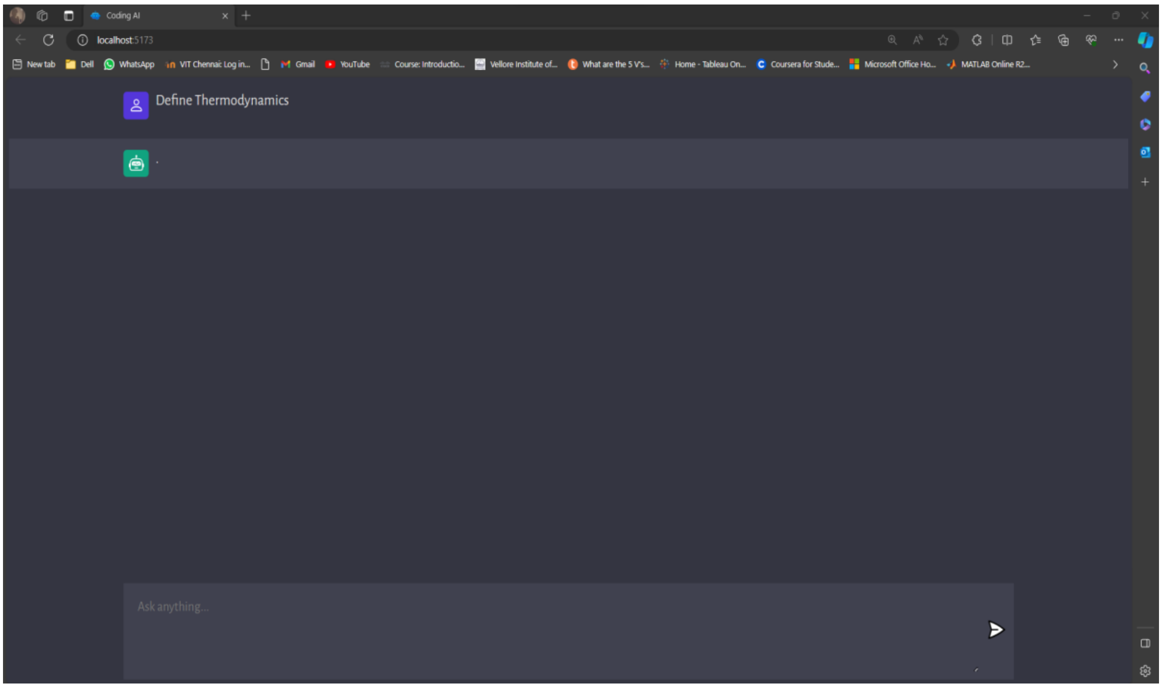
Message Loading Animation: The loader function informs users during wait times by displaying a loading animation as the system responds to user queries.

Server Communication: To handle the user's query, the code uses the defined server API (http://localhost:5000/) to send asynchronous requests to the server-side code through the Fetch API.

Code on the server side (Express / Node.js)
In order to handle POST requests sent from the client-side interface, a server is established using the provided Node.js/Express code. This is an explanation:

**API Handling**: The server handles responses for different interactions by having endpoints for GET and POST requests. POST Request Processing: To generate completions for user prompts, the POST endpoint uses the OpenAI API to process incoming requests. Based on user input, it generates AI responses using OpenAI's language model.
Response Handling: The AI generates a response and sends it back to the client-side interface to be displayed in the chat interface after receiving data from the OpenAI model.

**INTERFACE:**




**Creating Reactions:** Contextual Understanding: By utilizing its extensive training on a plethora of text data, ChatGPT analyzes and comprehends the context when presented with a prompt or message.

Based on probability Prediction the model predicts the most likely word or series of words to come next based on the context that has been supplied.

**Dynamic Generation:** It creates text in a dynamic way by taking the context into account and coming up with responses that make sense and are pertinent to the context.

**Natural Text Generation (Natural Language Generation):** In order to enable ChatGPT to generate responses that are both coherent and appropriate for the context, the system focuses on producing text that closely resembles human-written language.

**Restrictions and Ethical Issues:** Contextual Limitations: Although ChatGPT is good at producing text, there are situations when the responses it generates are not contextual or do not convey the meaning that was intended.

**Ethical Usage:** Because it can produce text that resembles that of a human, there are ethical questions surrounding its use. As a result, its deployment must be done responsibly to stop abuse and the spread of false information.

## 2.2 USER EXPERIENCE TESTING AND IMPROVEMENT STRATEGIES

In order to make sure that the developed system is both effective and user-friendly, user experience (UX) testing and enhancement strategies are essential. The following techniques and tactics are some that can be used:

- **Usability Testing**: Low-fidelity prototypes or wireframes should be used for early testing in order to obtain input on the interface's design and core features. Interactive Testing: Run thorough tests on the fully operational system, watching how users interact with it to spot any confusing or inefficient areas.

- **Gathering User Input** via Surveys and Questionnaires Through structured surveys or questionnaires, learn about the preferences and opinions of users with an emphasis on their levels of satisfaction and suggestions for improvement.

- **Interviews and Group Discussions:** To learn more about user experiences, preferences, and pain points, conduct group discussions or one-on-one interviews.

- **Analytics and Data Analysis**: User Behavior Analysis: Make use of user interaction tracking tools (such as heatmaps and click-through rates) to find frequently used features, high-engagement areas, and typical user paths.

- **A/B testing:** Examine several interface iterations to ascertain which functionality or design option outperforms the others in terms of predetermined metrics.

- **Iterative Design and Refinement, Iterative Development:** Apply modifications in response to gathered input and insights, progressively enhancing the interface.

- **Usability Heuristics:** To find and fix usability problems, compare the interface to accepted usability heuristics (such as Nielsen's heuristics).

- **Accessibility Testing:** By following accessibility guidelines (such as WCAG) and using assistive technology during testing, you can make sure the interface is usable by people with disabilities.

- **Testing for inclusivity**: Ensure that the user interface takes into account the various cultural, linguistic, and educational backgrounds of its users.

- **Performance Enhancement:** Response Time and Loading Time: Test and fine-tune the interface's functionality to guarantee fast load times and responsiveness on a range of devices and network configurations.

- **Continuous Improvement**, **Feedback Loop:** To promote continuous improvements, set up a feedback loop that is constantly gathering, analyzing, and putting user input into practice.

- **Principles of User-Centric Design**, **Persona creation and Empathy Mapping**: To match the needs and preferences of users with the interface design, use empathy maps and user personas.

In order to understand context and produce text responses that resemble those of a human, ChatGPT is a sophisticated language generation model that uses a great deal of pre-training and fine-tuning. It also continuously learns from interactions to enhance its performance.

# CHAPTER 3
# DATA EXPLORATION AND FEATURE ENGINEERING

## 3.1 PREPROCESSING SCIENTIFIC DATASETS

Scientific datasets are carefully chosen collections of structured data with a primary focus on scientific fields. One example of such a dataset is the Open-Book-QA dataset. Gaining a thorough understanding of the domain-specific content, structure, and purpose of scientific datasets is essential. This is a synopsis:

**Content and Extent, Scientific Knowledge**: A variety of information unique to scientific fields, including physics, biology, chemistry, and so forth, is included in scientific datasets.
Factual data, established concepts in the field, theories, principles, and experimental findings are all included in these datasets.

**Structure and Form, Structured Data**: To organize information in a methodical way, datasets usually make use of formats like tables, graphs, or text.
Metadata describing the variables, data types, sources, and collection methods may be present in scientific datasets.

**Goal and Application, Research and Analysis**: These datasets are used by researchers for a variety of tasks, including modeling, analysis, testing hypotheses, and verifying scientific theories.
**Question-Answering and Understanding:** Certain datasets, such as OpenBookQA, are made for tasks involving answering questions in order to evaluate a user's capacity for understanding and reasoning about scientific concepts.

**Difficulties and Features:**
**Complexity:** Scientific datasets frequently contain intricate information that calls for specialized knowledge in the field to properly interpret and analyze.

**Multifaceted Understanding**: Comprehending such datasets requires an understanding of the domain's connections, relationships, and context in addition to the basic data.

**Uses and Progress:**

**Increasing Knowledge**: Accurate comprehension of scientific datasets can result in new developments in science, technology, and real-world applications in a variety of fields.

**Enhancing Algorithms and Models**: Open-Book-QA datasets are useful for benchmarking and creating sophisticated AI models that are able to understand and reason about scientific content.

**Combining and Examining:**

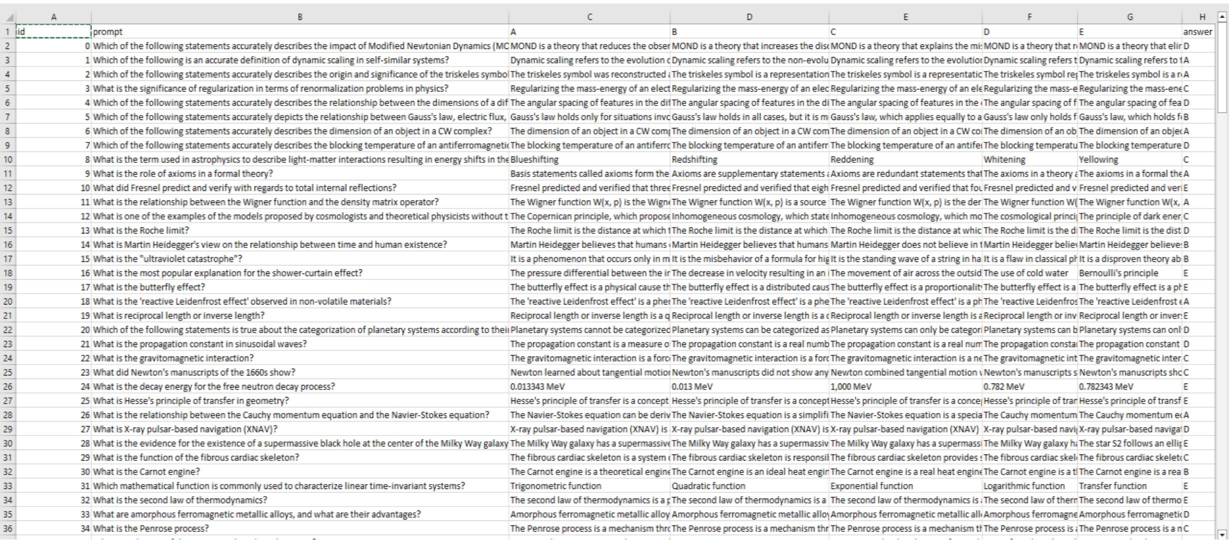
**Interdisciplinary Connections:** Datasets frequently facilitate integration between various scientific fields, opening up new avenues for investigation and analysis.
Statistical

**Computational Analysis:** To extract knowledge, trends, and conclusions from these datasets, researchers run statistical and computational analyses.

Comprehending scientific datasets necessitates a combination of domain knowledge, proficiency in data analysis, and the capacity to evaluate and utilize scientific data for investigation, creativity, and issue resolution in a range of scientific domains.

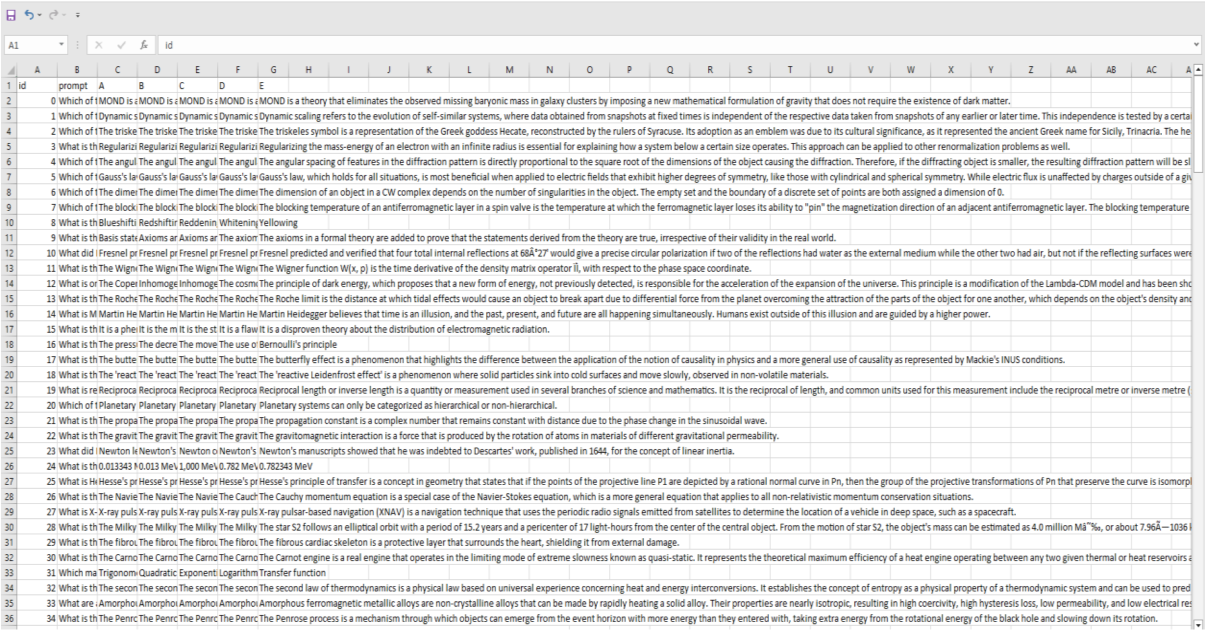
## 3.2 DATASET DESCRIPTION

## TRAIN DATASET:

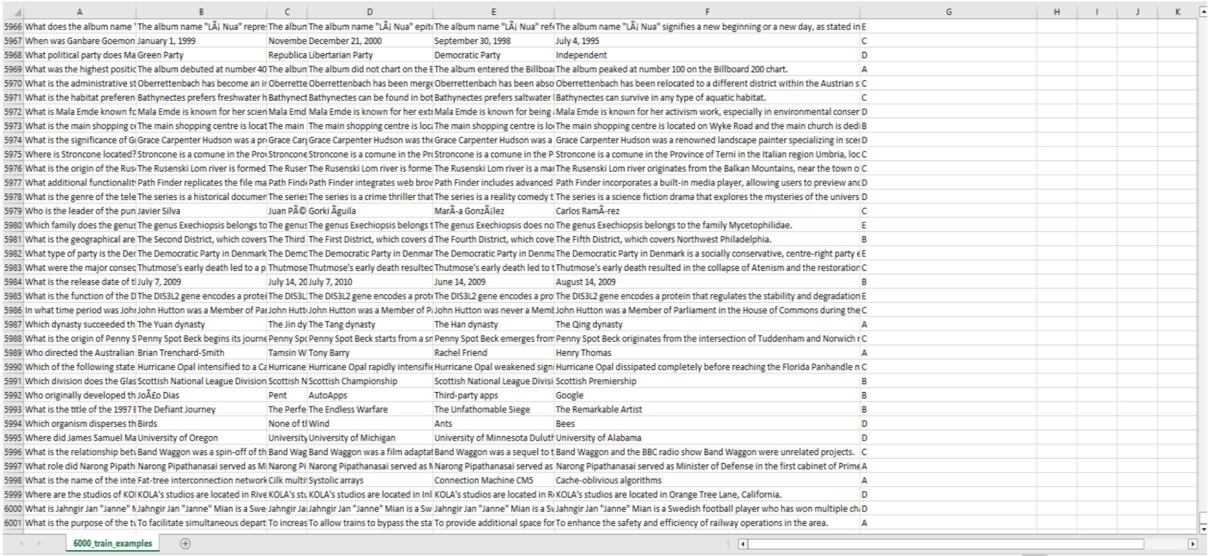


| id | prompt | A | B | C | D | E | answer |
|---|---|---|---|---|---|---|---|
| 0 | Which of the following statements accurately describes the impact of Modified Newtonian Dynamics (MC | MOND is a theory that reduces the obser | MOND is a theory that increases the disc | MOND is a theory that explains the mi | MOND is a theory that r | MOND is a theory that elir | D |
| 1 | Which of the following is an accurate definition of dynamic scaling in self-similar systems? | Dynamic scaling refers to the evolution c | Dynamic scaling refers to the non-evolu | Dynamic scaling refers to the evolutior | Dynamic scaling refers t | Dynamic scaling refers to t | A |
| 2 | Which of the following statements accurately describes the origin and significance of the triskeles symbol | The triskeles symbol was reconstructed | The triskeles symbol is a representation | The triskeles symbol is a representatic | The triskeles symbol re | The triskeles symbol is a r | A |
| 3 | What is the significance of regularization in terms of renormalization problems in physics? | Regularizing the mass-energy of an elect | Regularizing the mass-energy of an elec | Regularizing the mass-energy of an ele | Regularizing the mass-e | Regularizing the mass-ene | C |
| 4 | Which of the following statements accurately describes the relationship between the dimensions of a dif | The angular spacing of features in the dif | The angular spacing of features in the di | The angular spacing of features in the | The angular spacing of f | The angular spacing of fea | D |
| 5 | Which of the following statements accurately depicts the relationship between Gauss's law, electric flux, | Gauss's law holds only for situations invc | Gauss's law holds in all cases, but it is m | Gauss's law, which applies equally to a | Gauss's law only holds f | Gauss's law, which holds f | B |
| 6 | Which of the following statements accurately describes the dimension of an object in a CW complex? | The dimension of an object in a CW com | The dimension of an object in a CW com | The dimension of an object in a CW co | The dimension of an ob | The dimension of an objec | A |
| 7 | Which of the following statements accurately describes the blocking temperature of an antiferromagnetic | The blocking temperature of an antiferrc | The blocking temperature of an antiferr | The blocking temperature of an antifei | The blocking temperatu | The blocking temperature | D |
| 8 | What is the term used in astrophysics to describe light-matter interactions resulting in energy shifts in the | Blueshifting | Redshifting | Reddening | Whitening | Yellowing | C |
| 9 | What is the role of axioms in a formal theory? | Basis statements called axioms form the | Axioms are supplementary statements a | Axioms are redundant statements that | The axioms in a theory a | The axioms in a formal the | A |
| 10 | What did Fresnel predict and verify with regards to total internal reflections? | Fresnel predicted and verified that three | Fresnel predicted and verified that eigh | Fresnel predicted and verified that fou | Fresnel predicted and v | Fresnel predicted and veri | E |
| 11 | What is the relationship between the Wigner function and the density matrix operator? | The Wigner function W(x, p) is the Wign | The Wigner function W(x, p) is a source | The Wigner function W(x, p) is the der | The Wigner function W( | The Wigner function W(x, | A |
| 12 | What is one of the examples of the models proposed by cosmologists and theoretical physicists without t | The Copernican principle, which propose | Inhomogeneous cosmology, which state | Inhomogeneous cosmology, which mo | The cosmological princi | The principle of dark ener | C |
| 13 | What is the Roche limit? | The Roche limit is the distance at which t | The Roche limit is the distance at which | The Roche limit is the distance at whic | The Roche limit is the d | The Roche limit is the dist | D |
| 14 | What is Martin Heidegger's view on the relationship between time and human existence? | Martin Heidegger believes that humans | Martin Heidegger believes that humans | Martin Heidegger does not believe in t | Martin Heidegger belie | Martin Heidegger believe | B |
| 15 | What is the "ultraviolet catastrophe"? | It is a phenomenon that occurs only in m | It is the misbehavior of a formula for hig | It is the standing wave of a string in ha | It is a flaw in classical ph | It is a disproven theory ab | B |
| 16 | What is the most popular explanation for the shower-curtain effect? | The pressure differential between the ir | The decrease in velocity resulting in an | The movement of air across the outsid | The use of cold water | Bernoulli's principle | E |
| 17 | What is the butterfly effect? | The butterfly effect is a physical cause th | The butterfly effect is a distributed caus | The butterfly effect is a proportionalit | The butterfly effect is a | The butterfly effect is a ph | E |
| 18 | What is the 'reactive Leidenfrost effect' observed in non-volatile materials? | The 'reactive Leidenfrost effect' is a phe | The 'reactive Leidenfrost effect' is a phe | The 'reactive Leidenfrost effect' is a ph | The 'reactive Leidenfros | The 'reactive Leidenfrost e | A |
| 19 | What is reciprocal length or inverse length? | Reciprocal length or inverse length is a q | Reciprocal length or inverse length is a c | Reciprocal length or inverse length is a | Reciprocal length or inv | Reciprocal length or inver | E |
| 20 | Which of the following statements is true about the categorization of planetary systems according to thei | Planetary systems cannot be categorized | Planetary systems can be categorized as | Planetary systems can only be categori | Planetary systems can b | Planetary systems can onl | D |
| 21 | What is the propagation constant in sinusoidal waves? | The propagation constant is a measure o | The propagation constant is a real numb | The propagation constant is a real num | The propagation consta | The propagation constant | D |
| 22 | What is the gravitomagnetic interaction? | The gravitomagnetic interaction is a forc | The gravitomagnetic interaction is a forc | The gravitomagnetic interaction is a ne | The gravitomagnetic int | The gravitomagnetic inter | C |
| 23 | What did Newton's manuscripts of the 1660s show? | Newton learned about tangential motio | Newton's manuscripts did not show any | Newton combined tangential motion | Newton's manuscripts s | Newton's manuscripts shc | C |
| 24 | What is the decay energy for the free neutron decay process? | 0.013343 MeV | 0.013 MeV | 1,000 MeV | 0.782 MeV | 0.782343 MeV | E |
| 25 | What is Hesse's principle of transfer in geometry? | Hesse's principle of transfer is a concept | Hesse's principle of transfer is a concept | Hesse's principle of transfer is a conce | Hesse's principle of tran | Hesse's principle of transf | E |
| 26 | What is the relationship between the Cauchy momentum equation and the Navier-Stokes equation? | The Navier-Stokes equation can be deriv | The Navier-Stokes equation is a simplifi | The Navier-Stokes equation is a specia | The Cauchy momentum | The Cauchy momentum e | A |
| 27 | What is X-ray pulsar-based navigation (XNAV)? | X-ray pulsar-based navigation (XNAV) is | X-ray pulsar-based navigation (XNAV) is | X-ray pulsar-based navigation (XNAV) | X-ray pulsar-based navi | X-ray pulsar-based naviga | D |
| 28 | What is the evidence for the existence of a supermassive black hole at the center of the Milky Way galaxy | The Milky Way galaxy has a supermassive | The Milky Way galaxy has a supermassiv | The Milky Way galaxy has a supermassi | The Milky Way galaxy ha | The star S2 follows an ellip | E |
| 29 | What is the function of the fibrous cardiac skeleton? | The fibrous cardiac skeleton is a system | The fibrous cardiac skeleton is responsil | The fibrous cardiac skeleton provides | The fibrous cardiac skel | The fibrous cardiac skelet | C |
| 30 | What is the Carnot engine? | The Carnot engine is a theoretical engine | The Carnot engine is an ideal heat engir | The Carnot engine is a real heat engine | The Carnot engine is a t | The Carnot engine is a rea | B |
| 31 | Which mathematical function is commonly used to characterize linear time-invariant systems? | Trigonometric function | Quadratic function | Exponential function | Logarithmic function | Transfer function | E |
| 32 | What is the second law of thermodynamics? | The second law of thermodynamics is a p | The second law of thermodynamics is a | The second law of thermodynamics is | The second law of therm | The second law of thermo | E |
| 33 | What are amorphous ferromagnetic metallic alloys, and what are their advantages? | Amorphous ferromagnetic metallic alloy | Amorphous ferromagnetic metallic alloy | Amorphous ferromagnetic metallic all | Amorphous ferromagne | Amorphous ferromagnetic | D |
| 34 | What is the Penrose process? | The Penrose process is a mechanism thro | The Penrose process is a mechanism thr | The Penrose process is a mechanism th | The Penrose process is | The Penrose process is a n | C |

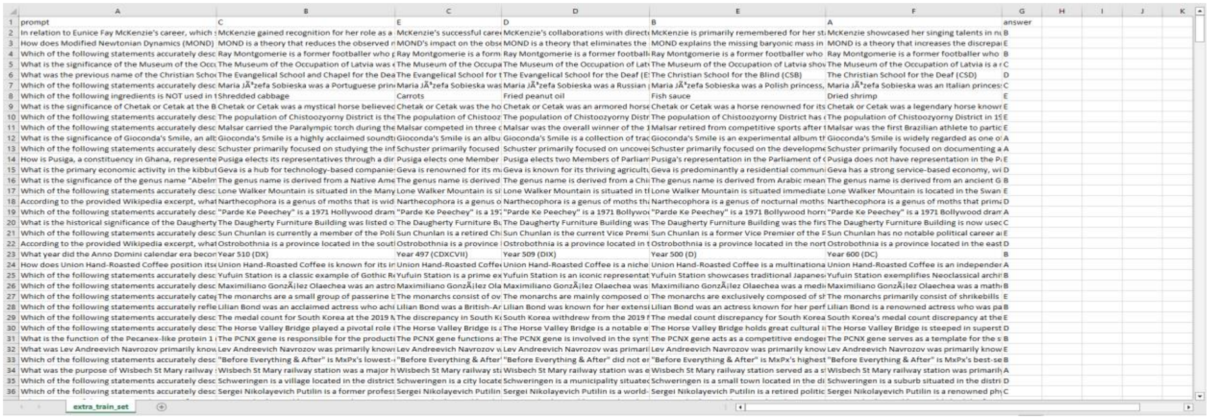
## TEST DATASET:



## 6000 TRAIN EXAMPLES DATASET:



## EXTRA TRAIN DATASET:



Below are descriptions of the columns in the dataset that are provided:

For every question-answer pair, there is an ID.
Questions from the dataset are included in the prompt.
Multiple-choice options A through E are offered as possible responses to the related question.
Provides the right response (A, B, C, D, or E) to the given question.
For every row in the collection:

ID: A special number given to every question-answer combination.
Prompt: The scientific questions posed are in this column.
A, B, C, D, and E options: The five options (A, B, C, D, and E) for each question indicate potential responses to the given question.
Indicates which option (A, B, C, D, or E) most closely matches the accurate response to the question.
The dataset comprises five possible answers (A, B, C, D, and E) for multiple-choice questions on scientific subjects.

# CHAPTER 4
# LONG SHORT-TERM MEMORY (LSTM) ANALYSIS

## 4.1 UNDERSTANDING LSTM NETWORKS

Recurrent neural network (RNN) architectures with Long Short-Term Memory (LSTM) networks are intended to capture long-term dependencies and address the vanishing gradient issue that conventional RNNs encounter. Because LSTMs are particularly good at processing and predicting sequences, they can be used for a variety of sequential data-related tasks, such as time series analysis, speech recognition, and natural language processing.

**Key Components of LSTM Networks:**

**Memory Units:** Long sequences of information are maintained by memory cells found in LSTMs. These cells enable the network to store and discard data according to how pertinent it is to the current task.

**Gates:** The Forget Gate selects which data from the cell state should be deleted.
**Input Gate:** Adds new data to change the state of the cell.
**Output Gate**: Determines what data should be output in accordance with the altered cell state.

**State of Cell:** symbolizes the network's "memory". It traverses time and is capable of carrying data while the sequence is being processed.

**How LSTMs Works:**

**Forget Gate Operation:** The forget gate generates numbers between 0 and 1 by passing the current input and the prior hidden state through a sigmoid function. The amount of the prior cell state that should be forgotten is determined by these values.

**Function of an Input Gate:** What new data is stored in the cell state is determined by the input gate. A sigmoid layer and a tanh layer are involved. The tanh layer generates a vector of potential new values, and the sigmoid layer selects which values (between 0 and 1) will be updated.

**Refreshing the Cell State**: The input gate selects new information, which is added after irrelevant information has been erased to update the current cell state.

**Operation of Output Gate**: Based on the updated cell state, the output gate determines what the next hidden state should be. The prediction and the hidden state for the subsequent time step both use the hidden state, which is a filtered version of the cell state.

**Benefits of LSTMs:**

- **Long-Term Dependency Handling:** Long-Term Dependency Trees (LSTMs) are well-suited for tasks where context over longer sequences is important because they can capture and remember long-term dependencies in sequences.

- **Decreased Vanishing Gradient Issue:** By reducing the vanishing gradient issue that conventional RNNs encounter, the architecture's gating mechanism enhances training on longer sequences.

  In order to use neural networks effectively for tasks requiring memory and context across sequences, it is essential to have a basic understanding of LSTM networks and how they handle sequential data.

## 4.2 IMPLEMENTATION AND TRAINING:

+ Code + Text

Connect


PROJECT TITLE

TACKLING COMPLEX SCIENTIFIC QUESTIONS USING LARGE LANGUAGE MODEL

NAME : T S S Abinandhan Kumar

REGISTER NUMBER : 19MIA1062

```
[ ] import numpy as np
    import pandas as pd
```

IMPORTING LIBRARIES

```
[ ] import keras
    import tensorflow as tf
    import tensorflow_hub as hub
    import tensorflow_text as text
    from tensorflow.keras.utils import to_categorical
    from sklearn.metrics import classification_report
    import random
    from sklearn.model_selection import KFold
    from tensorflow.keras.layers import concatenate
```

```
[ ] LABEL_DICT = {"A":0,'B':1,'C':2,'D':3,'E':4}
```

```
def read_data(path,check_test= False):
    df = pd.read_csv(path)
    X_Prompt = tf.convert_to_tensor(df.prompt.to_list())
    X_A = tf.convert_to_tensor(df.A.to_list())
    X_B = tf.convert_to_tensor(df.B.to_list())
    X_C = tf.convert_to_tensor(df.C.to_list())
    X_D = tf.convert_to_tensor(df.D.to_list())
    X_E = tf.convert_to_tensor(df.E.to_list())

    Selects = {'A':X_A,"B":X_B,"C":X_C,'D':X_D,'E':X_E}

    Labels = labels_one_hot = None
```

```
[ ]
    if not check_test:
        labels = df.answer.to_list()
        labels_one_hot = [LABEL_DICT.get(i) for i in labels]

        label_A = []
        label_B = []
        label_C = []
        label_D = []
        label_E = []

        for i in labels:
            if i =='A':
                label_A.append(0.8)
                label_B.append(0.05)
                label_C.append(0.05)
                label_D.append(0.05)
                label_E.append(0.05)
            if i =='B':
                label_B.append(0.8)
                label_A.append(0.05)
                label_C.append(0.05)
                label_D.append(0.05)
                label_E.append(0.05)

            if i =='C':
                label_C.append(0.8)
                label_A.append(0.05)
                label_B.append(0.05)
                label_D.append(0.05)
                label_E.append(0.05)
            if i =='D':
                label_D.append(0.8)
                label_A.append(0.05)
                label_B.append(0.05)
                label_C.append(0.05)
                label_E.append(0.05)
            if i =='E':
                label_E.append(0.8)
                label_A.append(0.05)
                label_B.append(0.05)
                label_C.append(0.05)
                label_D.append(0.05)
        label_A = tf.convert_to_tensor(label_A)
        label_B = tf.convert_to_tensor(label_B)
        label_C = tf.convert_to_tensor(label_C)
        label_D = tf.convert_to_tensor(label_D)
        label_E = tf.convert_to_tensor(label_E)


        Labels  = {'A': label_A,'B': label_B,'C': label_C,'D': label_D,'E': label_E}

    return {'Prompt':X_Prompt,'Selects':Selects,'labels':Labels,'label_onehot':labels_one_hot}
```

Libraries To handle and manipulate data, import pandas and num-py.
Deep learning frameworks for neural network construction include Keras and TensorFlow.
TensorFlow Hub is a repository for machine learning modules that can be reused.

**TensorFlow Text:** An extension for TensorFlow designed to manipulate textual data.

**To categorical**: Keras utility function for labels encoded one-hot.

**Classification report:** Scikit-learns output for a report on classification.

**random**: For data shuffles or random number generation.

**K-Fold:** A scikit-learn tool for data splitting for cross-validation.

**concatenate:** Layer concatenation using the Keras function.

**Defined Functions**
**LABEL_DICT:** A dictionary that associates numerical values (0, 1, 2, 3, 4) with labels (A, B, C, D, and E).

**Read data (path, check test=False):** This function uses the path parameter to specify a CSV file from which to read data. It loads several columns with text data related to a question and its options ('prompt', 'A', 'B', 'C', 'D', and 'E').

**Contributions:**
**path:** The CSV file's path.
**Check test:** a boolean flag designating whether test data is being checked.
Take Actions:

loads the information into a Pandas DataFrame (df) from the CSV file.
Creates TensorFlow tensors (X_Prompt, X_A, X_B, X_C, X_D, X_E) from text data columns ('prompt', 'A', 'B', 'C', 'D', 'E').
makes the tensors for options A, B, C, D, and E available in a dictionary (Selects).

**Managing Labels:**

If the training or validation data's check_test returns False:

extracts labels from the DataFrame's "answer" column.

creates one-hot encoded tensors (label_A, label_B, label_C, label_D, label_E) from the categorical labels.

Gives each label a weight based on which particular option is the right response (0.05 for incorrect answers and 0.8 for the right answer).

assembles these weighted labels into a dictionary (Labels).

**Results:**

yields a dictionary that includes:

Tensor for the prompt text is called "Prompt."

"Selects": Tensor dictionary for options A, B, C, D, and E.

'labels': If check_test returns False, then weighted label tensors for each choice.

'label_onehot': Labels encoded one-hot (provided check_test returns False).

```
def read_data_binary(path,test = False):
    df = pd.read_csv(path)
    X_Prompt = df.prompt.to_list()
    X_A = df.A.to_list()
    X_B = df.B.to_list()
    X_C = df.C.to_list()
    X_D = df.D.to_list()
    X_E = df.E.to_list()

    labels_num = None
    if not test:
        labels_old = df.answer.to_list()
        labels_num = [LABEL_DICT.get(i) for i in labels_old]

    questions= []
    answers = []
    labels = []
    label_num_new = []
    for i in range(len(X_Prompt)):
        for j in range(5):
            questions.append(X_Prompt[i])
        answers.append(X_A[i])
        answers.append(X_B[i])
        answers.append(X_C[i])
        answers.append(X_D[i])
        answers.append(X_E[i])
        if not test:
            one_hot_label = np.full((5,),0.35)
            one_hot_label[labels_num[i]] =1.0
            for j in one_hot_label:
                labels.append(j)
            for i in range(5):
                label_num_new.append(labels_num[i])

    questions = tf.convert_to_tensor(questions)
    answers = tf.convert_to_tensor(answers)
    labels = tf.convert_to_tensor(labels)

    return {'Prompt':questions,'Answers':answers,'labels':labels,'label_nums':labels_num}
```

**Summary of Function:**

**Reading and Preparing Data:**

opens the CSV file that the path argument specifies.

loads each choice (A, B, C, D, and E) and the prompt's data columns into distinct lists.

In the absence of test mode:

utilizes the LABEL_DICT mapping to extract and translate categorical labels into numerical representations.

**Restructuring Data:**

Creates distinct lists (questions and answers) by copying the prompt text five times and adding the options (A, B, C, D, and E) for each question in the correct order.

In the absence of test mode:

creates labels (one-hot encoded labels) for every possible response.

label_num_new, a list of label numbers, is created.

INPUT TRAIN

```
path_train = '/content/drive/MyDrive/kaggle-llm-science-exam/extra_train_set.csv'
result = read_data(path_train,False)
X_Prompt = result.get('Prompt')
X_A = result.get('Selects').get('A')
X_B = result.get('Selects').get('B')
X_C = result.get('Selects').get('C')
X_D = result.get('Selects').get('D')
X_E = result.get('Selects').get('E')

Selects = [result.get('Selects')[i] for i in result.get('Selects') ]
Selects.insert(0,X_Prompt)
labels = [result.get('labels')[i] for i in result.get('labels') ]
labels_one_hot = result.get('label_onehot')
labels_one_hot_train = tf.one_hot(labels_one_hot, depth=5)
```

```
[ ]
path_val = '/content/drive/MyDrive/kaggle-llm-science-exam/train.csv'
result_val = read_data(path_val ,False)
X_Prompt_val  = result_val .get('Prompt')
X_A_val  = result_val .get('Selects').get('A')
X_B_val  = result_val .get('Selects').get('B')
X_C_val  = result_val .get('Selects').get('C')
X_D_val  = result_val .get('Selects').get('D')
X_E_val  = result_val .get('Selects').get('E')

Selects_val  = [result_val .get('Selects')[i] for i in result_val .get('Selects') ]
Selects_val .insert(0,X_Prompt_val )
labels_val  = [result_val .get('labels')[i] for i in result_val .get('labels') ]
labels_one = result_val .get('label_onehot')
labels_one_hot_val = tf.one_hot(labels_one, depth=5)
```

## LOAD MODEL BERT

```
bert_preprocess = hub.KerasLayer("https://tfhub.dev/tensorflow/bert_en_uncased_preprocess/3")
bert_encoder = hub.KerasLayer("https://tfhub.dev/tensorflow/bert_en_uncased_L-12_H-768_A-12/4")
```

## BUILD MODEL

```
text_Prompt = tf.keras.layers.Input(shape=(), dtype=tf.string)
text_A = tf.keras.layers.Input(shape=(), dtype=tf.string)
text_B = tf.keras.layers.Input(shape=(), dtype=tf.string)
text_C = tf.keras.layers.Input(shape=(), dtype=tf.string)
text_D = tf.keras.layers.Input(shape=(), dtype=tf.string)
text_E = tf.keras.layers.Input(shape=(), dtype=tf.string)

encoder_prompt_input = bert_preprocess(text_Prompt)
encoder_A_input = bert_preprocess(text_A)
encoder_B_input = bert_preprocess(text_B)
encoder_C_input = bert_preprocess(text_C)
encoder_D_input = bert_preprocess(text_D)
encoder_E_input = bert_preprocess(text_E)

encoder_prompt = concatenate(
        tuple([bert_encoder(encoder_prompt_input)['encoder_outputs'][i] for i in range(-4, 0)]),
        name = 'hidden_states_prompt_1',
        axis = -1
    )[:, 0, :]
encoder_A = concatenate(
        tuple([bert_encoder(encoder_A_input)['encoder_outputs'][i] for i in range(-4, 0)]),
        name = 'hidden_states_A',
        axis = -1
    )[:, 0, :]
encoder_B = concatenate(
        tuple([bert_encoder(encoder_B_input)['encoder_outputs'][i] for i in range(-4, 0)]),
        name = 'hidden_states_B',
        axis = -1
    )[:, 0, :]
encoder_C = concatenate(
        tuple([bert_encoder(encoder_C_input)['encoder_outputs'][i] for i in range(-4, 0)]),
        name = 'hidden_states_C',
        axis = -1
    )[:, 0, :]
encoder_D = concatenate(
        tuple([bert_encoder(encoder_D_input)['encoder_outputs'][i] for i in range(-4, 0)]),
        name = 'hidden_states_D',
        axis = -1
    )[:, 0, :]
encoder_E = concatenate(
        tuple([bert_encoder(encoder_E_input)['encoder_outputs'][i] for i in range(-4, 0)]),
        name = 'hidden_states_E',
        axis = -1
    )[:, 0, :]
print(encoder_E.shape)
```

\+ Code + Text

```
print(encoder_E.shape)

reshape_prompt = tf.reshape(encoder_prompt, (-1, 4, 768))
reshape_A = tf.reshape(encoder_A, (-1, 4, 768))
reshape_B = tf.reshape(encoder_B, (-1, 4, 768))
reshape_C = tf.reshape(encoder_C, (-1, 4, 768))
reshape_D = tf.reshape(encoder_D, (-1, 4, 768))
reshape_E = tf.reshape(encoder_E, (-1, 4, 768))
print('reshape: ', reshape_A.shape)
# layer A
concat_encoder_A = concatenate([reshape_prompt,reshape_A,reshape_B,reshape_C,reshape_D,reshape_E],axis = 1)
LSTM = tf.keras.layers.LSTM(units=32,dropout=0.2, recurrent_dropout=0.2,return_sequences=True)(concat_encoder_A)
Dense_prompt_A = tf.keras.layers.Dense(32,activation = 'relu')(LSTM)
Dense_prompt_A =  tf.keras.layers.Dropout(0.3)(Dense_prompt_A)
Dense_prompt_A = tf.keras.layers.LayerNormalization()(Dense_prompt_A)
Linear_A =  tf.keras.layers.Dropout(0.3)(Dense_prompt_A)
classifer = keras.layers.Flatten()(Linear_A)
classifer = tf.keras.layers.Dense(5,activation = 'linear')(classifer)
classifer =  tf.keras.layers.Dense(5,activation = 'softmax')(classifer)

model = keras.Model(inputs=[text_Prompt, text_A,text_B,text_C,text_D,text_E], outputs=classifer)
model.compile(optimizer = 'adam',loss = 'categorical_crossentropy',metrics = ['accuracy'])
model.summary()
```

```
(None, 3072)
reshape:  (None, 4, 768)
Model: "model_8"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_49 (InputLayer)          [(None,)]            0           []

 input_50 (InputLayer)          [(None,)]            0           []

 input_51 (InputLayer)          [(None,)]            0           []

 input_52 (InputLayer)          [(None,)]            0           []

 input_53 (InputLayer)          [(None,)]            0           []

 input_54 (InputLayer)          [(None,)]            0           []

 keras_layer_2 (KerasLayer)     {'input_word_ids':   0           ['input_49[0][0]',
                                (None, 128),                      'input_50[0][0]',
                                 'input_mask': (Non               'input_51[0][0]',
                                e, 128),                          'input_52[0][0]',
                                 'input_type_ids':                'input_53[0][0]',
                                (None, 128)}                      'input_54[0][0]']

 keras_layer_3 (KerasLayer)     {'default': (None,   109482241   ['keras_layer_2[0][0]',
                                768),                             'keras_layer_2[0][1]',
                                 'sequence_output':               'keras_layer_2[0][2]',
                                 (None, 128, 768),                'keras_layer_2[0][0]',
                                 'pooled_output': (               'keras_layer_2[0][1]',
                                None, 768),                       'keras_layer_2[0][2]',
                                 'encoder_outputs':               'keras_layer_2[0][0]',
                                 [(None, 128, 768),               'keras_layer_2[0][1]',
```

**Synopsis of the Model:**

**Input Layers**

The summary does not specify the shapes of the six input layers (input_49, input_50, input_51, input_52, input_53, and input_54) that are used.

**Transformer Layers with prior training:**

In order to handle text input (input_word_ids, input_mask, input_type_ids), two KerasLayer instances (keras_layer_2, keras_layer_3) are used, which most likely represent a pre-trained Transformer-based model (like BERT or something similar).

Sequence and pooled representation outputs are indicated by the output shapes from these layers.

a large number of connections showing the data flow from these layers in various configurations.

**Layers of Concatenation:**

For every choice (prompt, A, B, C, D, and E), the outputs from multiple Transformer layers are combined using multiple Concatenate layers.
It appears that information from various Transformer outputs is combined by these concatenated layers (hidden_states_prompt_1, hidden_states_A, hidden_states_B, hidden_states_C, hidden_states_D, and hidden_states_E).

**Cutting Procedures:**

Although they are carried out, slicing operations (tf.__operators__.getitem_X) are not specifically mentioned in the summary that is supplied.

**Rearranging Layers:**

The data is reshaped into the desired dimensions by multiple tf.reshape_X layers ((None, 4, 768)).
These reshaping processes most likely prepare the data for the network's later layers.

**Layer of Recursion (LSTM):**

To extract temporal patterns from the concatenated outputs, an LSTM layer is used.
The sequence length of 24 and the hidden state size of 32 are indicated by the 32 units (None, 24, 32) of the LSTM layer.

**Dropout and Dense Layers:**

The LSTM layer is followed by dropout layers (dropout_16, dropout_17) and dense layers (dense_19, dense_20).
The Dropout layers help with regularization to avoid overfitting, while the Dense layers are probably used to learn non-linear mappings.

**Layer Normalization:**

To normalize the activations, apply LayerNormalization (layer_normalization_8).
Layer Flattening:
The data from the LSTM layer is reshaped into a single vector using a flatten layer (flatten_6).

**Layer of Output:**

For classification tasks, the output layer is represented by the final Dense layer (dense_20) with 5 units, which is probably used to predict among the 5 options (A, B, C, D, and E).

**Summary Details:**

**Total Parameters**

shows the total number of parameters (trainable and non-trainable) in the model.

**Qualifiable and Qualifiable Parameters:**

displays the model's trainable and non-trainable parameter counts, showing which parameters are fixed or derived from pre-trained layers and which are being learned during training (Trainable params).

```
y_predict = model.predict(Selects)
top_three_indices = (-y_predict).argsort(axis = 1)[:, :3].tolist()
top_max = np.argmax(y_predict,axis = -1)
report = classification_report(labels_one_hot, top_max)
print(report)
```

```
count = 0
for i in range(len(labels_one_hot)):
    if labels_one_hot[i] in top_three_indices[i]:
        count +=1
print(count)
```

```
y_predict = model.predict(Selects_val)
top_three_indices = (-y_predict).argsort(axis = 1)[:, :3].tolist()
top_max = np.argmax(y_predict,axis = -1)
report = classification_report(labels_one, top_max)
print(report)
```

```
count = 0
for i in range(len(labels_one)):
    if labels_one[i] in top_three_indices[i]:
        count +=1
print(count)
```

**DATA TEST**

```
path_test = '/kaggle/input/kaggle-llm-science-exam/test.csv'
result_bianry = read_data_binary(path_test,True)
Questions = result_bianry.get('Prompt')
Answers = result_bianry.get('Answers')
Labels = result_bianry.get('labels')
Label_nums = result_bianry.get('label_nums')
Selects = []
Selects.append(Questions)
Selects.append(Answers)
```

```
path_test = '/content/drive/MyDrive/kaggle-llm-science-exam/test.csv'
result = read_data(path_test,True)
X_Prompt = result.get('Prompt')
X_A = result.get('Selects').get('A')
X_B = result.get('Selects').get('B')
X_C = result.get('Selects').get('C')
X_D = result.get('Selects').get('D')
X_E = result.get('Selects').get('E')

Selects = [result.get('Selects')[i] for i in result.get('Selects') ]
Selects.insert(0,X_Prompt)
```

```
[ ]
    df_resutl =  pd.read_csv("/content/drive/MyDrive/kaggle-llm-science-exam/sample_submission.csv")
    x = ['A','B','C','D',"E"]
    Label_test = [' '.join([x[int(j)] for j in i]) for i in top_three_indices]

    for i in range(len(Label_test)):
        df_resutl.at[i, 'prediction'] = str(Label_test[i])
    df_resutl.to_csv('LSTM_Output.csv', index=False)
```

Double-click (or enter) to edit

OUTPUT:

| | A | B |
|---|---|---|
| 1 | id | prediction |
| 2 | 0 | B C E |
| 3 | 1 | B E C |
| 4 | 2 | C B E |
| 5 | 3 | B D E |
| 6 | 4 | B E D |
| 7 | 5 | B E D |
| 8 | 6 | B E D |
| 9 | 7 | B D C |
| 10 | 8 | D B C |
| 11 | 9 | B D E |
| 12 | 10 | B E C |
| 13 | 11 | B D C |
| 14 | 12 | B E D |
| 15 | 13 | B C D |
| 16 | 14 | B D E |
| 17 | 15 | B E D |
| 18 | 16 | B D C |
| 19 | 17 | B C D |
| 20 | 18 | B C D |
| 21 | 19 | B E C |
| 22 | 20 | B D E |
| 23 | 21 | B D E |
| 24 | 22 | B C E |
| 25 | 23 | B E C |
| 26 | 24 | B D C |
| 27 | 25 | B C D |
| 28 | 26 | B D E |
| 29 | 27 | B C D |
| 30 | 28 | B D E |
| 31 | 29 | B D C |
| 32 | 30 | B E C |
| 33 | 31 | B D E |
| 34 | 32 | B D E |
| 35 | 33 | B D C |
| 36 | 34 | B E C |

LSTM_Output

# CHAPTER 5
# LLM ANALYSIS FOR COMPLEX SCIENTIFIC QUESTIONS

## 5.1 LEVERAGING LARGE LANGUAGE MODELS

**Transformative Impact**: Accurate and contextually aware text understanding has been made possible by large language models such as GPT (Generative Pre-trained Transformer), which have completely changed NLP tasks.

**Record-Breaking Scale:** These models are trained on enormous datasets that cover a wide range of linguistic patterns, domains, and contexts.

**Wide Range of Applications**: Making use of these models goes beyond standard language tasks and includes sentiment analysis, translation, summarization, and more.

**Fine-tuning Capability:** Models can be improved to perform better on domain-specific problems by fine-tuning them on particular datasets or tasks.

**Semantic Understanding**: They enable more thorough text comprehension by capturing complex syntactic structures, semantic relationships, and contextual details.

**Decreased Annotation Dependency:** Because large language models are naturally able to learn from large corpora and become versatile for a variety of tasks, they minimize the need for extensive annotated data.

**Generation and Creativity:** These models can help with dialogue systems, story generation, and content creation. They are capable of producing coherent text as well as creative content.

**Ethical Challenges:** Careful curation and monitoring are required because ethical issues can arise from potential biases or the creation of misleading information.

**Resource Intensiveness**: These models are difficult for smaller teams or organizations to use or train because they demand a lot of computer power.

**Benefits of Transfer Learning:** Pre-trained models provide strong foundations for transfer learning, enabling quicker convergence and enhanced performance on tasks that come after.

**Constant Improvements:** Research is still being done to increase interpretability, decrease biases, increase efficiency, and scale up models.

**Real-world Applications:** These models are used in summarization tools, content recommendation engines, chatbots for customer service, and more.

**Multilingual Capabilities:** Certain models are particularly good at handling several languages, which promotes international communication and makes cross-language tasks easier.

**Reducing Language Barriers:** Broad language models help reduce language barriers by promoting information access and communication amongst linguistically diverse populations.

Regulators may be closely monitoring the use and implementation of these models in relation to issues of fairness, privacy, and disinformation.

**Impact on interdisciplinary fields**: These models extend beyond natural language processing and have applications in fields such as clinical text analysis in healthcare, risk analysis in finance, and contract analysis in law.

**Constant Learning:** Over time, models can remain relevant by being gradually updated to reflect changing linguistic trends.

**Collaborative Development**: Community-driven enhancements are encouraged by open-source projects and partnerships, which encourage creativity and accessibility.

**Semantic Understanding:** By allowing systems to comprehend context, sarcasm, intent, and sentiment, large language models facilitate interactions that are more akin to those of humans.

**Augmented Content Creation**: By using these models to brainstorm, draft, and improve their content, writers, marketers, and content producers can streamline their workflows.

**Handling Data Scarcity**: Pre-trained models provide a foundation for developing efficient models with smaller datasets in situations where there is a shortage of labeled data.

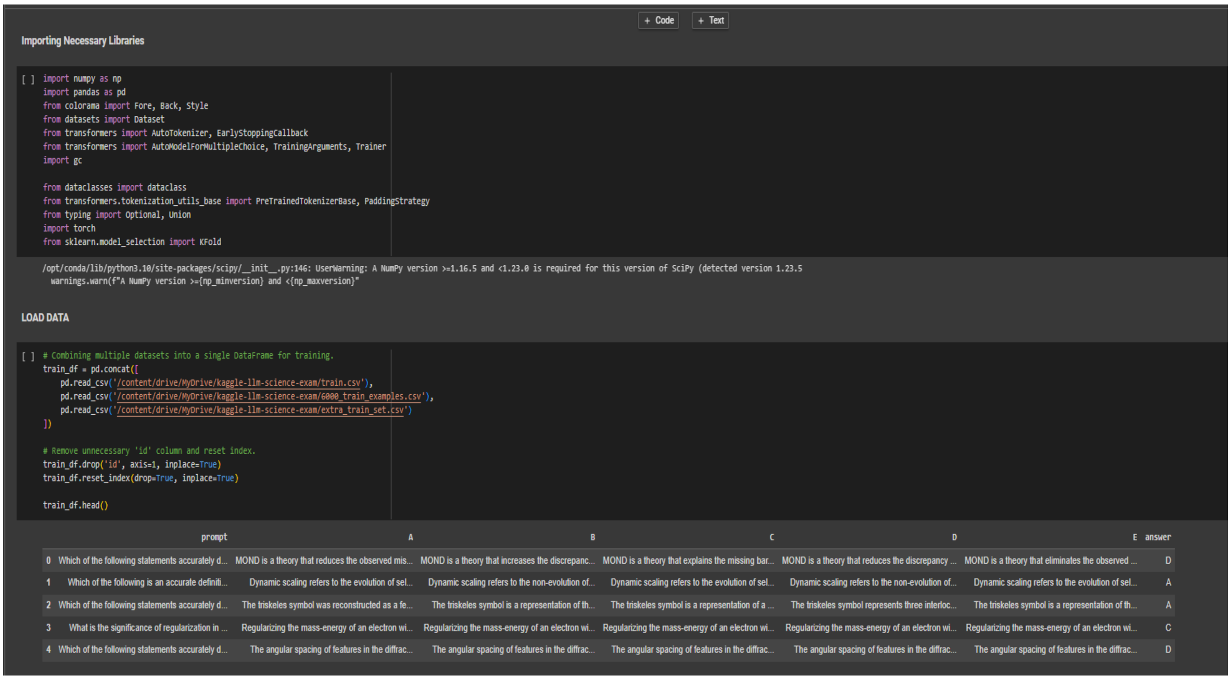
Bias Mitigation: In order to promote justice and inclusivity in their applications, efforts are made to mitigate biases in these models.

**Education and Research:** They enable scholars to investigate language generation and comprehension by acting as instructional tools and supports.

**Market Competition:** As more complex models are developed, rivalry between established tech companies and emerging players in the AI space is heightened.

**Future Developments:** As hardware, algorithms, and data become more accessible, it is expected that language models will become even more potent and contextually aware, creating new avenues for AI-driven applications.

## 5.2 IMPLEMENTATION AND RESULTS



+ Code + Text

Importing Necessary Libraries

```
[ ] import numpy as np
    import pandas as pd
    from colorama import Fore, Back, Style
    from datasets import Dataset
    from transformers import AutoTokenizer, EarlyStoppingCallback
    from transformers import AutoModelForMultipleChoice, TrainingArguments, Trainer
    import gc

    from dataclasses import dataclass
    from transformers.tokenization_utils_base import PreTrainedTokenizerBase, PaddingStrategy
    from typing import Optional, Union
    import torch
    from sklearn.model_selection import KFold
```

```
/opt/conda/lib/python3.10/site-packages/scipy/__init__.py:146: UserWarning: A NumPy version >=1.16.5 and <1.23.0 is required for this version of SciPy (detected version 1.23.5
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

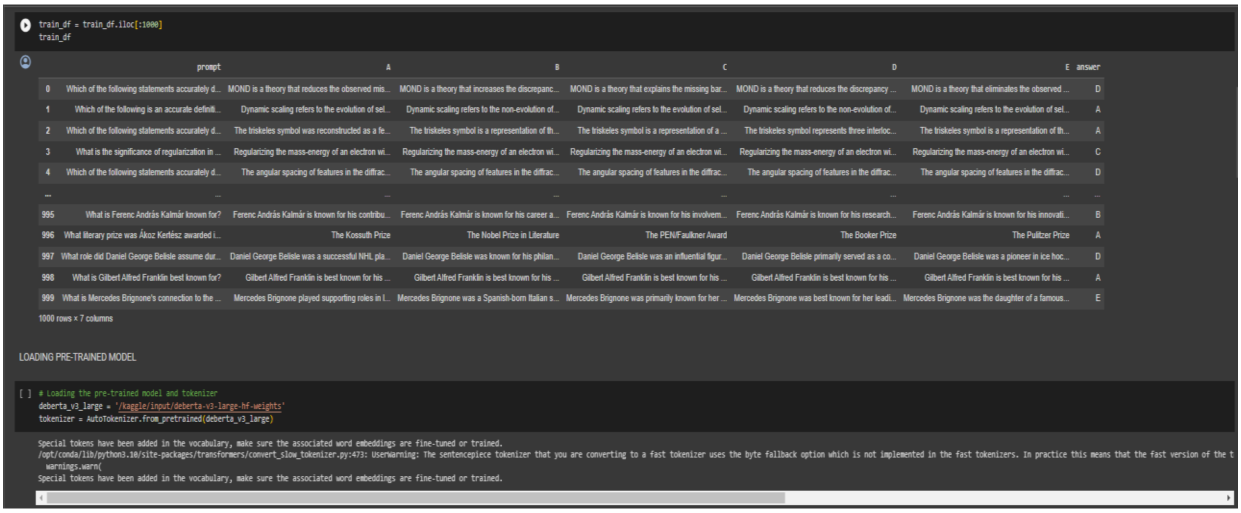
LOAD DATA

```
[ ] # Combining multiple datasets into a single DataFrame for training.
    train_df = pd.concat([
        pd.read_csv('/content/drive/MyDrive/kaggle-llm-science-exam/train.csv'),
        pd.read_csv('/content/drive/MyDrive/kaggle-llm-science-exam/6000_train_examples.csv'),
        pd.read_csv('/content/drive/MyDrive/kaggle-llm-science-exam/extra_train_set.csv')
    ])

    # Remove unnecessary 'id' column and reset index.
    train_df.drop('id', axis=1, inplace=True)
    train_df.reset_index(drop=True, inplace=True)

    train_df.head()
```

| | prompt | A | B | C | D | E | answer |
|---|---|---|---|---|---|---|---|
| 0 | Which of the following statements accurately d... | MOND is a theory that reduces the observed mis... | MOND is a theory that increases the discrepanc... | MOND is a theory that explains the missing bar... | MOND is a theory that reduces the discrepancy ... | MOND is a theory that eliminates the observed ... | D |
| 1 | Which of the following is an accurate definiti... | Dynamic scaling refers to the evolution of sel... | Dynamic scaling refers to the non-evolution of... | Dynamic scaling refers to the evolution of sel... | Dynamic scaling refers to the non-evolution of... | Dynamic scaling refers to the evolution of sel... | A |
| 2 | Which of the following statements accurately d... | The triskeles symbol was reconstructed as a fe... | The triskeles symbol is a representation of th... | The triskeles symbol is a representation of a ... | The triskeles symbol represents three interloc... | The triskeles symbol is a representation of th... | A |
| 3 | What is the significance of regularization in ... | Regularizing the mass-energy of an electron wi... | Regularizing the mass-energy of an electron wi... | Regularizing the mass-energy of an electron wi... | Regularizing the mass-energy of an electron wi... | Regularizing the mass-energy of an electron wi... | C |
| 4 | Which of the following statements accurately d... | The angular spacing of features in the diffrac... | The angular spacing of features in the diffrac... | The angular spacing of features in the diffrac... | The angular spacing of features in the diffrac... | The angular spacing of features in the diffrac... | D |



```
train_df = train_df.iloc[:1000]
train_df
```

| | prompt | A | B | C | D | E | answer |
|---|---|---|---|---|---|---|---|
| 0 | Which of the following statements accurately d... | MOND is a theory that reduces the observed mis... | MOND is a theory that increases the discrepanc... | MOND is a theory that explains the missing bar... | MOND is a theory that reduces the discrepancy ... | MOND is a theory that eliminates the observed ... | D |
| 1 | Which of the following is an accurate definiti... | Dynamic scaling refers to the evolution of sel... | Dynamic scaling refers to the non-evolution of... | Dynamic scaling refers to the evolution of sel... | Dynamic scaling refers to the non-evolution of... | Dynamic scaling refers to the evolution of sel... | A |
| 2 | Which of the following statements accurately d... | The triskeles symbol was reconstructed as a fe... | The triskeles symbol is a representation of th... | The triskeles symbol is a representation of a ... | The triskeles symbol represents three interloc... | The triskeles symbol is a representation of th... | A |
| 3 | What is the significance of regularization in ... | Regularizing the mass-energy of an electron wi... | Regularizing the mass-energy of an electron wi... | Regularizing the mass-energy of an electron wi... | Regularizing the mass-energy of an electron wi... | Regularizing the mass-energy of an electron wi... | C |
| 4 | Which of the following statements accurately d... | The angular spacing of features in the diffrac... | The angular spacing of features in the diffrac... | The angular spacing of features in the diffrac... | The angular spacing of features in the diffrac... | The angular spacing of features in the diffrac... | D |
| ... | ... | ... | ... | ... | ... | ... | ... |
| 995 | What is Ferenc András Kalmár known for? | Ferenc András Kalmár is known for his contribu... | Ferenc András Kalmár is known for his career a... | Ferenc András Kalmár is known for his involvem... | Ferenc András Kalmár is known for his research... | Ferenc András Kalmár is known for his innovati... | B |
| 996 | What literary prize was Ákoz Kertész awarded i... | The Kossuth Prize | The Nobel Prize in Literature | The PEN/Faulkner Award | The Booker Prize | The Pulitzer Prize | A |
| 997 | What role did Daniel George Belisle assume dur... | Daniel George Belisle was a successful NHL pla... | Daniel George Belisle was known for his philan... | Daniel George Belisle was an influential figur... | Daniel George Belisle primarily served as a co... | Daniel George Belisle was a pioneer in ice hoc... | D |
| 998 | What is Gilbert Alfred Franklin best known for? | Gilbert Alfred Franklin is best known for his ... | Gilbert Alfred Franklin is best known for his ... | Gilbert Alfred Franklin is best known for his ... | Gilbert Alfred Franklin is best known for his ... | Gilbert Alfred Franklin is best known for his ... | A |
| 999 | What is Mercedes Brignone's connection to the ... | Mercedes Brignone played supporting roles in I... | Mercedes Brignone was a Spanish-born Italian s... | Mercedes Brignone was primarily known for her ... | Mercedes Brignone was best known for her leadi... | Mercedes Brignone was the daughter of a famous... | E |

1000 rows × 7 columns

LOADING PRE-TRAINED MODEL

```
[ ] # Loading the pre-trained model and tokenizer
    deberta_v3_large = '/kaggle/input/deberta-v3-large-hf-weights'
    tokenizer = AutoTokenizer.from_pretrained(deberta_v3_large)
```

```
Special tokens have been added in the vocabulary, make sure the associated word embeddings are fine-tuned or trained.
/opt/conda/lib/python3.10/site-packages/transformers/convert_slow_tokenizer.py:473: UserWarning: The sentencepiece tokenizer that you are converting to a fast tokenizer uses the byte fallback option which is not implemented in the fast tokenizers. In practice this means that the fast version of the t
  warnings.warn(
Special tokens have been added in the vocabulary, make sure the associated word embeddings are fine-tuned or trained.
```

This little piece of code loads a pre-trained DeBERTa v3 large model and the tokenizer that goes with it using the Hugging Face transformers library, which is widely used for tasks related to natural language processing.

**Model Filling:**

The path or identifier for the DeBERTa v3 large model is probably stored in the variable deberta_v3_large. This path may be a model identifier that the Hugging Face library recognizes or it may be a directory path.

**Initialization of Tokenizer:**

- Tokenizer for the DeBERTa v3 large model is loaded using the AutoTokenizer class from the transformers library.
- The tokenizer is initialized using the pre-trained DeBERTa v3 large model, which is supplied by the deberta_v3_large variable, using the AutoTokenizer.from_pretrained method.
- This tokenizer is in charge of tokenizing the input text data, turning words or sentences into tokens that the DeBERTa model can comprehend.

**Library of Hugging Face Transformers:**

- The Hugging Face transformers library, which provides an easy-to-use interface for working with a variety of pre-trained language models, such as DeBERTa, BERT, GPT, and others, is utilized by the code.
- Tokenizers, pre-trained models, and tools for adjusting, inferring, and deploying these models are all conveniently accessible through the library.

**Useful Points to Remember:**

- In order to use the DeBERTa v3 large model for downstream NLP tasks like text classification, sentiment analysis, question answering, etc., this code snippet sets up the required components (model and tokenizer).
- The loaded model would be used to carry out operations like producing predictions, embeddings, or features from the text inputs, and the loaded tokenizer would be used to preprocess text data in subsequent code.

```python
# We'll create a dictionary to convert option names (A, B, C, D, E) into indices and back again
options = 'ABCDE'
indices = list(range(5))

option_to_index = {option: index for option, index in zip(options, indices)}
index_to_option = {index: option for option, index in zip(options, indices)}


# Define a preprocessing function that prepares the data for model input.
def preprocess(example):
    # The AutoModelForMultipleChoice class expects a set of question/answer pairs
    # so we'll copy our question 5 times before tokenizing
    first_sentence = [example['prompt']] * 5
    second_sentence = [example[option] for option in 'ABCDE']

    # Our tokenizer will turn our text into token IDs BERT can understand
    tokenized_example = tokenizer(first_sentence, second_sentence, truncation=True)
    tokenized_example['label'] = option_to_index[example['answer']]

    return tokenized_example
```

```python
[ ] print(index_to_option)
```

```
{0: 'A', 1: 'B', 2: 'C', 3: 'D', 4: 'E'}
```

```python
# Following datacollator (adapted from https://huggingface.co/docs/transformers/tasks/multiple_choice)
# will dynamically pad our questions at batch-time so we don't have to make every question the length
# of our longest question.

#  Define a data collator class for multiple choice tasks using @dataclass.
@dataclass
class DataCollatorForMultipleChoice:
    tokenizer: PreTrainedTokenizerBase
    padding: Union[bool, str, PaddingStrategy] = True
    max_length: Optional[int] = None
    pad_to_multiple_of: Optional[int] = None

    def __call__(self, features):
        # Determine the label name based on whether 'label' or 'labels' is present in the features.
        label_name = "label" if 'label' in features[0].keys() else 'labels'

        # Extract labels from features and compute batch size and the number of choices.
        labels = [feature.pop(label_name) for feature in features]
        batch_size = len(features)
        num_choices = len(features[0]['input_ids'])

        # Flatten the features to prepare for padding.
        flattened_features = [
            [{k: v[i] for k, v in feature.items()} for i in range(num_choices)] for feature in features
        ]
        flattened_features = sum(flattened_features, [])

        # Pad the flattened features using the tokenizer.
        batch = self.tokenizer.pad(
            flattened_features,
            padding=self.padding,
            max_length=self.max_length,
            pad_to_multiple_of=self.pad_to_multiple_of,
            return_tensors='pt',
        )

         # Reshape the padded batch into the desired format.
        batch = {k: v.view(batch_size, num_choices, -1) for k, v in batch.items()}

        # Add the labels to the batch as a tensor.
        batch['labels'] = torch.tensor(labels, dtype=torch.int64)

        return batch
```

```
# Evaluation metric: MAP@3

K = 3

def apk(y_i_true, y_i_pred):

    assert(len(y_i_pred) <= K)
    assert(len(np.unique(y_i_pred)) == len(y_i_pred))

    sum_precision = 0.0
    num_hits = 0.0

    for i, p in enumerate(y_i_pred):
        #checks whether it is valid prediction
        if p in y_i_true:
            num_hits += 1
            precision = num_hits / (i + 1)
            sum_precision += precision

    return sum_precision / min(len(y_i_true), K)

def mapk(y_true, y_pred):
    return np.mean([apk(y_i_true, y_i_pred) for y_i_true, y_i_pred in zip(y_true, y_pred)])
```

```
[ ] test_df = pd.read_csv('/content/drive/MyDrive/kaggle-llm-science-exam/test.csv')
test_df['answer'] = 'A' # dummy answer that allows us to preprocess the test datataset using functionality that works for the train set

# Create a tokenized test dataset from the Pandas DataFrame and apply the 'preprocess' function to prepare the data.
test_ds = Dataset.from_pandas(test_df)
tokenized_test_ds = test_ds.map(preprocess, batched=False, remove_columns=['prompt', 'A', 'B', 'C', 'D', 'E', 'answer'])
```

```
  0%|          | 0/200 [00:00<?, ?ex/s]
Asking to truncate to max_length but no maximum length is provided and the model has no predefined maximum length. Default to no truncation.
```

```
# Set up K-Fold Cross-Validation

final_dfs = pd.DataFrame()
kf = KFold(n_splits=5, shuffle=True, random_state=71)
cv_list = []

for fold, (tr_idx, va_idx) in enumerate(kf.split(train_df)):
    # Create train/validation subsets.
    train_set = train_df.loc[tr_idx, ['prompt', 'A', 'B', 'C', 'D', 'E', 'answer']]
    valid_set = train_df.loc[va_idx, ['prompt', 'A', 'B', 'C', 'D', 'E', 'answer']]

    # Convert to Hugging Face's 'Dataset' format and preprocess.
    train_set = Dataset.from_pandas(train_set)
    tokenized_train = train_set.map(preprocess, remove_columns=['prompt', 'A', 'B', 'C', 'D', 'E', 'answer'])
    valid_set = Dataset.from_pandas(valid_set)
    tokenized_valid = valid_set.map(preprocess, remove_columns=['prompt', 'A', 'B', 'C', 'D', 'E', 'answer'])

    valid_label = train_df.loc[va_idx, 'answer'].values

    # Training arguments.
    training_args = TrainingArguments(
        output_dir='./',
        overwrite_output_dir=True,
        load_best_model_at_end=True,
        save_total_limit=1,
        evaluation_strategy="epoch",
        save_strategy="epoch",
        warmup_ratio=0.8,
        learning_rate=1e-6,

        per_device_train_batch_size=1,
        per_device_eval_batch_size=2,

        num_train_epochs=10,

        report_to='none',
        seed=422
    )

     # Initialize the model and trainer
    model = AutoModelForMultipleChoice.from_pretrained(deberta_v3_large)
    trainer = Trainer(
        model=model,
        args=training_args,
        tokenizer=tokenizer,
        data_collator=DataCollatorForMultipleChoice(tokenizer=tokenizer),
        train_dataset=tokenized_train,
        eval_dataset=tokenized_valid,
        callbacks=[EarlyStoppingCallback(early_stopping_patience=2)]
    )

    # training
    trainer.train()
```

```python
# training
trainer.train()

# validation
valid_pred = trainer.predict(tokenized_valid).predictions
valid_pred_ids = np.argsort(-valid_pred, axis=1)
valid_pred_letters = np.array(list('ABCDE'))[valid_pred_ids][:, :3]

# Compute MAP@3 score
valid_map3 = mapk(valid_label, valid_pred_letters)

print(f"{Fore.RED}{Style.BRIGHT}Fold {fold}: MAP@3 = {valid_map3:.5f}{Style.RESET_ALL}")
cv_list.append(valid_map3)

test_predictions = trainer.predict(tokenized_test_ds).predictions
fold_predict_df = pd.DataFrame(test_predictions, columns=[f'{x}{fold}' for x in ['A','B','C','D','E']])
final_dfs = pd.concat([final_dfs, fold_predict_df], axis=1)

# Clean up to avoid running out of memory.
del model, trainer, tokenized_train, tokenized_valid, train_set, valid_set
gc.collect()
```

FOLD 0

| Epoch | Training Loss | Validation Loss |
|---|---|---|
| 1 | 1.611300 | 1.609449 |
| 2 | 1.620600 | 1.610504 |
| 3 | 1.616700 | 1.593135 |
| 4 | 1.608300 | 1.600539 |
| 5 | 1.574000 | 1.555467 |
| 6 | 1.533200 | 1.446381 |
| 7 | 1.374400 | 1.307687 |
| 8 | 1.204800 | 1.223704 |
| 9 | 0.924200 | 1.296124 |
| 10 | 0.782300 | 1.347473 |

Fold 0: MAP@3 = 0.67833

FOLD 1

| Epoch | Training Loss | Validation Loss |
|---|---|---|
| 1 | 1.613100 | 1.608853 |
| 2 | 1.609800 | 1.608821 |
| 3 | 1.610700 | 1.609304 |
| 4 | 1.604000 | 1.586339 |
| 5 | 1.569600 | 1.525717 |
| 6 | 1.524600 | 1.435965 |
| 7 | 1.333400 | 1.353580 |
| 8 | 1.225600 | 1.298170 |
| 9 | 1.013700 | 1.371726 |
| 10 | 0.861300 | 1.509892 |

Fold 1: MAP@3 = 0.67333

**Initialization of the Model:**

The DebertaV2ForMultipleChoice model is the one being utilized. Certain layers' weights, such as "pooler.dense.weight," "classifier.weight," "pooler.dense.bias," and "classifier.bias,".
To effectively use this model for predictions and inference, it is advised to train it on a downstream task.

**Instruction Procedure:**

Each fold of the training process consists of 10 epochs, for a total of five cross-validation folds.
Training loss and validation loss are tracked epoch-by-epoch.

**Metrics of Performance:**

The evaluation metric is the Mean Average Precision at 3 (MAP@3), which is calculated after each fold.
The model's performance on the multiple-choice task for each validation set is indicated by the MAP@3 values, which are reported for each fold.

**Training Advancement:**

The training time per epoch is given, illustrating both the length of time needed to finish each epoch and the training progress.

FOLD 2

| Epoch | Training Loss | Validation Loss |
|---|---|---|
| 1 | 1.609300 | 1.609239 |
| 2 | 1.610400 | 1.608782 |
| 3 | 1.619500 | 1.605416 |
| 4 | 1.604600 | 1.585162 |
| 5 | 1.554500 | 1.506296 |
| 6 | 1.481800 | 1.421090 |
| 7 | 1.288600 | 1.365332 |
| 8 | 1.169500 | 1.439716 |
| 9 | 0.990500 | 1.578342 |

Fold 2: MAP@3 = 0.63417

FOLD 3

| Epoch | Training Loss | Validation Loss |
|---|---|---|
| 1 | 1.614100 | 1.608989 |
| 2 | 1.610500 | 1.608855 |
| 3 | 1.613400 | 1.599948 |
| 4 | 1.603000 | 1.595540 |
| 5 | 1.555300 | 1.511646 |
| 6 | 1.485800 | 1.292380 |
| 7 | 1.143000 | 1.190665 |
| 8 | 0.952600 | 1.274834 |
| 9 | 0.708400 | 1.466390 |

Fold 3: MAP@3 = 0.65833

FOLD 4

| Epoch | Training Loss | Validation Loss |
|---|---|---|
| 1 | 1.616700 | 1.608765 |
| 2 | 1.615600 | 1.609131 |
| 3 | 1.616000 | 1.602130 |
| 4 | 1.608000 | 1.581030 |
| 5 | 1.584700 | 1.543132 |
| 6 | 1.545800 | 1.432844 |
| 7 | 1.245700 | 1.279293 |
| 8 | 1.100300 | 1.337485 |
| 9 | 0.763700 | 1.591478 |

Fold 4: MAP@3 = 0.64917

**Notes:**

Based on the reported values, it appears that the validation loss varies over epochs while the training loss appears to decrease, potentially indicating overfitting or underfitting.

The model's capacity to place right answers among the top three options can be understood from the reported MAP@3 values.

**Suggestion:**

The output highlights that additional training on the particular multiple-choice task would help the model perform better when making predictions.

```python
final_dfs['A'] = final_dfs[['A1','A2','A3','A0']].mean(axis=1)
final_dfs['B'] = final_dfs[['B1','B2','B3','B0']].mean(axis=1)
final_dfs['C'] = final_dfs[['C1','C2','C3','C0']].mean(axis=1)
final_dfs['D'] = final_dfs[['D1','D2','D3','D0']].mean(axis=1)
final_dfs['E'] = final_dfs[['E1','E2','E3','E0']].mean(axis=1)

final_dfs[['A', 'B', 'C', 'D', 'E']].head()
```

**Calculating the Mean Column-wise:**

In order to determine the mean for each of the options "A," "B," "C," "D," and "E," the code measures the data across four columns that are labeled with the suffixes "1, "2, "3, and "0."

To calculate the mean of columns 'A1', 'A2', 'A3', and 'A0', for choice 'A', for instance.

The new columns 'A', 'B', 'C', 'D', and 'E' represent the mean values, respectively.

**Determining Average Values:**

For the given columns, the row-wise mean is determined using the mean(axis=1) method.

**Making New Columns:**

The DataFrame final_dfs contains the computed mean values in columns 'A', 'B', 'C', 'D', and 'E'.

**Presenting the Outcome:**

Lastly, the code snippet uses the head() method to show the first few rows of the Data-Frame with the newly created columns "A," "B," "C," "D," and "E."

By computing their mean values and storing these summarized values in new columns for each choice ('A', 'B', 'C', 'D', 'E'), you can aggregate or summarize the data found in columns 'A1', 'A2', 'A3', 'A0', 'B1', 'B2', 'B3', 'B0', 'C1', 'C2', 'C3', 'C0', 'D1', 'D2', 'D3', 'D0', 'E1', 'E2', 'E3', and 'E0'.

|   | A | B | C | D | E |
|---|---|---|---|---|---|
| 0 | -2.553663 | -1.515964 | -2.539781 | 0.134061 | -2.477152 |
| 1 | -2.774576 | -2.955051 | -3.052081 | -3.093159 | -3.282637 |
| 2 | 0.600896 | -2.402701 | -0.271390 | -3.179917 | -1.798219 |
| 3 | -2.107654 | -2.341429 | -1.240922 | -2.639822 | -3.033341 |
| 4 | -0.814792 | -1.063014 | -1.505676 | 0.037207 | -1.512424 |

```
submission = test_df[['id', 'prediction']]
submission.to_csv('LLM_Output.csv', index=False)

display(pd.read_csv('Output1.csv').head())
display(pd.read_csv('Output2.csv').tail())
```

| id | prediction |
|---|---|
| 0 | D B E |
| 1 | A D C |
| 2 | A C E |
| 3 | C A B |
| 4 | D A B |
| 5 | B C D |
| 6 | A C B |
| 7 | D B E |
| 8 | C A B |
| 9 | A B E |
| 10 | E B A |
| 11 | E A C |
| 12 | C E A |
| 13 | E D A |
| 14 | B D C |
| 15 | B C E |
| 16 | C A B |
| 17 | E B D |
| 18 | A D C |
| 19 | E D B |
| 20 | D C B |
| 21 | D C E |
| 22 | C D B |
| 23 | B D E |
| 24 | A E D |
| 25 | E D B |
| 26 | A C E |
| 27 | D B C |
| 28 | E B C |
| 29 | C B D |
| 30 | B D E |
| 31 | E D B |
| 32 | A E B |
| 33 | D B E |
| 34 | E C D |

LLM_Output

# CHAPTER 6

# CONCLUSION AND FUTURE ENHANCEMENTS

## 6.1 SUMMARY OF FINDINGS

Large-scale transformer-based models, like BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pre-trained Transformer), are called LLMs. Training and inference are resource-intensive processes because of their large number of parameters, which can vary from millions to billions. This makes them computationally intensive. Specialized hardware, like GPUs or TPUs, is frequently needed for LLMs to efficiently handle their computational demands.

Recurrent neural networks (RNNs) of the long-range dependency type (LSTM) are used to identify long-range dependencies in sequential data.

comparatively less computationally intensive than large-scale LLMs because of their more straightforward designs and smaller number of parameters.

With CPUs or GPUs, LSTMs can be trained effectively without the need for extremely specialized hardware. In a variety of NLP tasks, such as question answering, text classification, translation, and language generation, LLMs have demonstrated outstanding performance. Because of their transformer architecture and extensive pre-training on large corpora, they are exceptional at comprehending semantics, contextual information, and intricate patterns in natural language. Since LSTMs are good at modeling sequential data, they are frequently used for tasks like language modeling, speech recognition, and time series prediction. Compared to large-scale LLMs, LSTMs may have trouble handling very long sequences or capturing extremely complex patterns, even though they are capable of capturing short- and moderate-range dependencies. Transformer-based architectures are used by LLMs, which allow for attention mechanisms and parallel processing for context capture. Long Short-Term Memory Networks (LSTMs) are recurrent networks that use memory cells to store data across sequences. For a wide variety of NLP tasks involving text generation and contextual understanding, LLMs are well suited. LSTM’s are frequently used for sequential data modeling and other tasks where knowing long-term dependencies is important, even though they might not always need the deep context knowledge that LLMs offer.

## 6.2 FUTURE DIRECTIONS AND POTENTIAL ENHANCEMENTS

## USER INTERFACE ENHANCEMENTS:

Improvements to user interfaces are crucial in determining how users interact with different digital platforms. These updates cover a wide range of enhancements intended to make interactions easier, increase user engagement, and offer clear navigation. By adopting responsive design principles, interfaces are evolving to better fit a variety of devices, guaranteeing consistent user experiences on tablets, smartphones, and PCs. Clearer communication and easier comprehension are facilitated by visual enhancements like intuitive iconography, harmonized color schemes, and refined typography. Incorporating interactive elements such as animations, transitions, and micro-interactions enhances user engagement and facilitates the navigation of intricate processes. By utilizing user data and preferences, personalization features adjust interfaces to each user's unique needs, promoting a sense of community and raising user satisfaction. Improvements in accessibility, such as better keyboard navigation and screen readers, guarantee inclusivity and meet a range of user requirements. Artificial intelligence (AI)-powered chatbots and voice interfaces are transforming interactions by providing more conversational and natural experiences. User-centric design, seamless functionality, and accessibility are still crucial aspects of interface evolution that will drive innovation and improve user experiences in all digital environments.

**Scale and Capacity:** Building even bigger language models with more parameters will improve the model's ability to comprehend context and produce text that is more logical and pertinent to the context.

**Efficiency:** Techniques for optimization and efficiency improvements to lower the memory and computational costs associated with large-scale model training and inference.

**Multimodal Understanding:** Combining multimodal features with language models to allow them to understand and produce text from a variety of inputs, including images, videos, and audio.

# APPENDIX

## ➢ Hyper-parameters:

During model training, the learning rate establishes the step size.
**Batch Size:** The quantity of samples processed prior to the model's parameters being updated.
**Epochs:** The total number of times the model is trained by passing the entire dataset both forward and backward.
Ratio of input units dropped during training to avoid overfitting is known as the "dropout rate."
**The quantity of units and layers:** parameters unique to each architecture that specify the breadth and depth of neural networks.
**Loss function and optimizer**: These are used to calculate and optimize the loss during training.

### ➢ Parameters for Text Processing:

Tokenization parameters are methods and setups that divide text into words or tokens.
**Sequence Length:** The maximum length of the tokenized input sequences.
**Vocabulary Size:** The quantity of distinct tokens taken into account by the model.

### ➢ Parameters Particular to the Model:

Activation functions, recurrent dropout, and hidden units are examples of LSTM/GRU parameters in recurrent neural networks.
**Transformer Parameters:** In transformer-based architectures such as BERT or GPT, the quantity of attention heads, layers, and hidden dimensions.

### ➢ Metrics for Evaluation:

Metrics Configuration: Set of parameters that specify the evaluation metrics (such as accuracy, precision, recall, and F1-score) that are used to assess the performance of the model.

## EVALUATION PARAMETERS:

### 1) MEAN ABSOLUTE PERCENTAGE ERROR

A popular metric for assessing forecasting model or prediction accuracy is Mean Absolute Percentage Error (MAPE), particularly when dealing with regression or time series analysis issues. It calculates the average absolute percentage difference, expressed in percentage terms, between the actual and predicted values.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n} |1 \frac{Ai - Fi}{Ai} 1| * 100$$

n is the number of observations.
Ai is the actual value of the observation.
Fi is the forecasted or predicted value.

By expressing the average error as a percentage of the real values, MAPE offers a relative accuracy metric. Since it's a relative error measure, it's useful for comparing how well models perform across various datasets or time series with different scales. Nevertheless, MAPE has certain drawbacks, including its sensitivity to actual values that are zero or almost zero (division by zero) and its susceptibility to data outliers.

Even though MAPE is frequently used, it's important to take other metrics into account in addition to it, particularly when working with datasets that have notable variations or particular characteristics that could affect how errors are interpreted.

## 2) Root Mean Square Error (RMSE):

Root Mean Square Error quantifies the deviation between predicted values by a model or estimator and the actual observed values. It's derived from the variance of residuals, showcasing how closely the observed data aligns with the model's predictions. Smaller RMSE values indicate a closer fit of the model to the data, signifying a stronger alignment between predicted and observed values. RMSE is calculated as the square root of the average of squared errors.

The formula for RMSE is represented as:

RMSE=2RMSE=n1∑i=1n(yi−xi)2

Where:

- n represents the total number of data points.
- yi stands for the predicted value.
- xi signifies the actual or observed value.

RMSE is a pivotal metric in assessing model performance, as it gives a comprehensive view of how well the model's predictions align with the observed data. Larger errors have a proportionately greater impact on RMSE, making it sensitive to significant deviations between predicted and actual values.

Root Mean Square Error (RMSE) serves as a pivotal metric in evaluating the accuracy of predictive models by quantifying the disparity between predicted and observed values. Essentially, it measures the average magnitude of errors between predicted and actual values, offering a comprehensive insight into the model's performance.

The calculation of RMSE revolves around the variance of residuals, showcasing how closely the observed data points align with the predicted values generated by the model. The formula for RMSE involves taking the square root of the average of squared errors, thereby giving higher weightage to larger errors:

RMSE = $\sqrt{(\Sigma(yi - xi)^2 / n)}$ Here,

'n' signifies the total number of data points, 'yi' represents the predicted value generated by the model, and 'xi' denotes the actual or observed value.

The significance of RMSE lies in its ability to represent the degree of alignment between.

the model's predictions and the observed data. Smaller RMSE values indicate a tighter fit of the model to the dataset, suggesting a more accurate representation of the observed values. Conversely, larger RMSE values reflect greater discrepancies between the predicted and actual values, signifying a weaker fit of the model to the data.

One crucial aspect of RMSE is its sensitivity to larger errors. Larger discrepancies between predicted and observed values have a disproportionate impact on the RMSE, influencing its value more significantly. This sensitivity enables RMSE to provide a robust evaluation of model performance by highlighting substantial deviations that might affect the overall accuracy of predictions.

Overall, RMSE acts as a fundamental metric in model assessment, offering a comprehensive understanding of the model's predictive capabilities. Its capacity to account for both small and large errors makes it a valuable tool for assessing the accuracy and reliability of predictive models across various domains and applications.

### 3) Mean Absolute Error (MAE):

The Mean Absolute Error assesses the average magnitude of errors within a set of predictions, disregarding their direction. As a linear score, MAE treats all individual differences equally in the average computation, providing insight into the typical size of errors expected from the prediction model. Unlike RMSE, MAE lacks sensitivity to the squared differences, making it more robust against extreme outliers.

The formula to calculate MAE is expressed as:

MAE= ∣MAE=n1∑i=1n∣yi−xi∣

Where:

- n represents the total number of data points.
- yi signifies the predicted value.
- xi denotes the actual or observed value.

MAE always yields a value lesser than or equal to RMSE. If all errors possess equal magnitudes, MAE equals RMSE. A lower MAE value signifies higher accuracy in the predictive model, reflecting smaller average errors between predicted and actual values. Its equal consideration of all errors makes MAE a reliable metric for overall model performance evaluation.

The Mean Absolute Error (MAE) serves as a fundamental metric for evaluating the performance of predictive models by measuring the average magnitude of errors without considering their direction. Unlike other evaluation metrics such as Root Mean Square Error (RMSE), MAE provides an indication of the typical size of errors expected from a prediction 56 model in an absolute sense.

The calculation of MAE involves computing the absolute differences between predicted values generated by the model and the actual observed values. It quantifies the average of these absolute differences across the entire dataset.

The formula for MAE is represented as:

$$MAE = (1 / n) \sum_{i=1}^{n} |y_i - x_i|$$

In this equation: -

- 'n' denotes the total number of data points within the dataset.
- 'yi' represents the predicted value obtained from the model.
- 'xi' signifies the actual or observed value in the dataset.

The MAE metric offers a linear score, treating each individual difference equally in the computation of the average. Unlike RMSE, which emphasizes larger errors due to squaring the differences, MAE considers the absolute differences directly. This characteristic makes MAE more robust against extreme outliers or unusually large errors within the dataset.

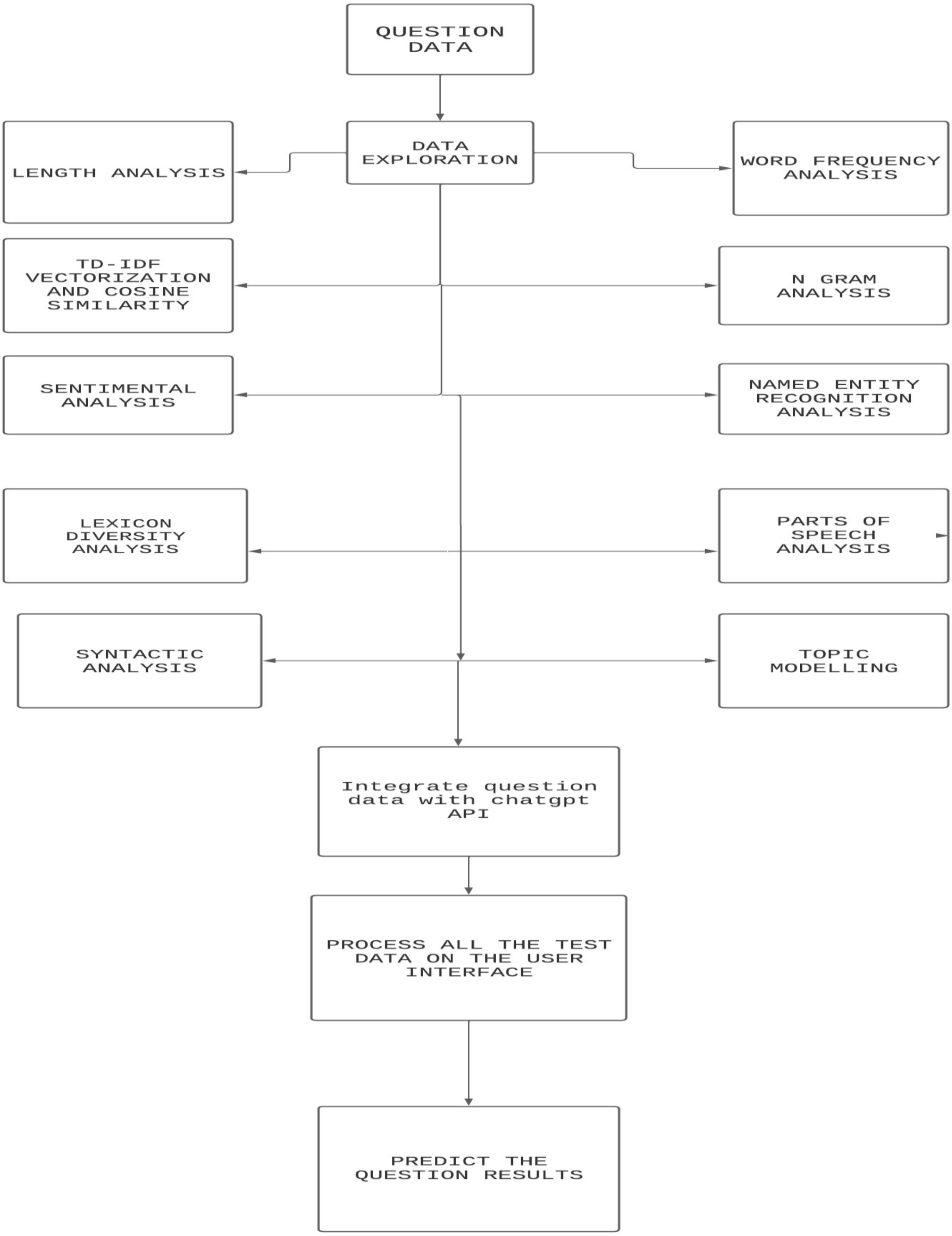
MAE's robustness against outliers stems from its direct focus on the absolute differences between predicted and observed values. It offers a comprehensive understanding of the average magnitude of errors without the influence of directional information. This property makes MAE particularly useful in scenarios where extreme values might disproportionately affect the evaluation metric's interpretation.

By providing a straightforward measure of the average error magnitude, MAE complements other evaluation metrics, offering a different perspective on the performance of predictive models. Its robustness against extreme values makes it a valuable tool for assessing model accuracy, especially in situations where the dataset contains outliers or instances with significantly large errors.

In summary, MAE serves as an essential evaluation metric, offering insights into the typical size of errors generated by prediction models. Its emphasis on the absolute differences between predicted and observed values without considering their direction makes it a reliable and robust metric for assessing model performance across various domains and applications.

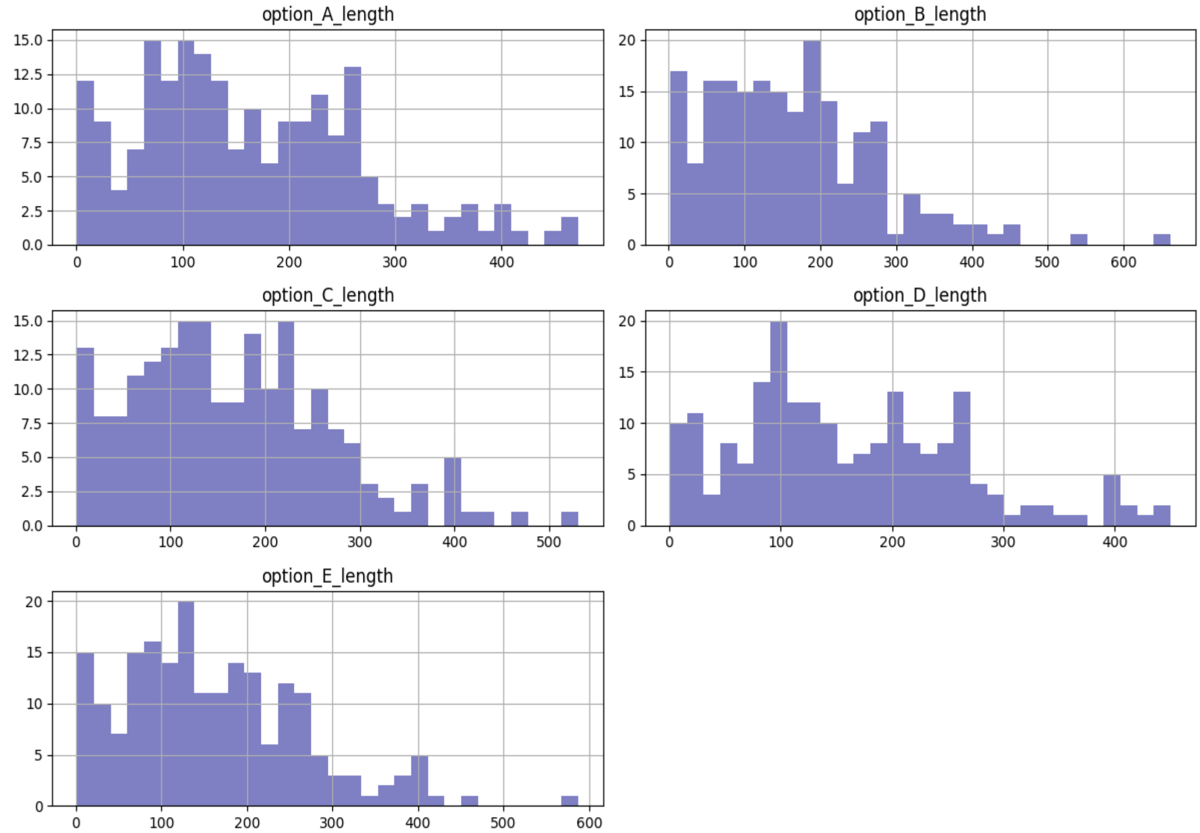
# LIST OF FIGURES

## DATA ANALYSIS FLOW CHART

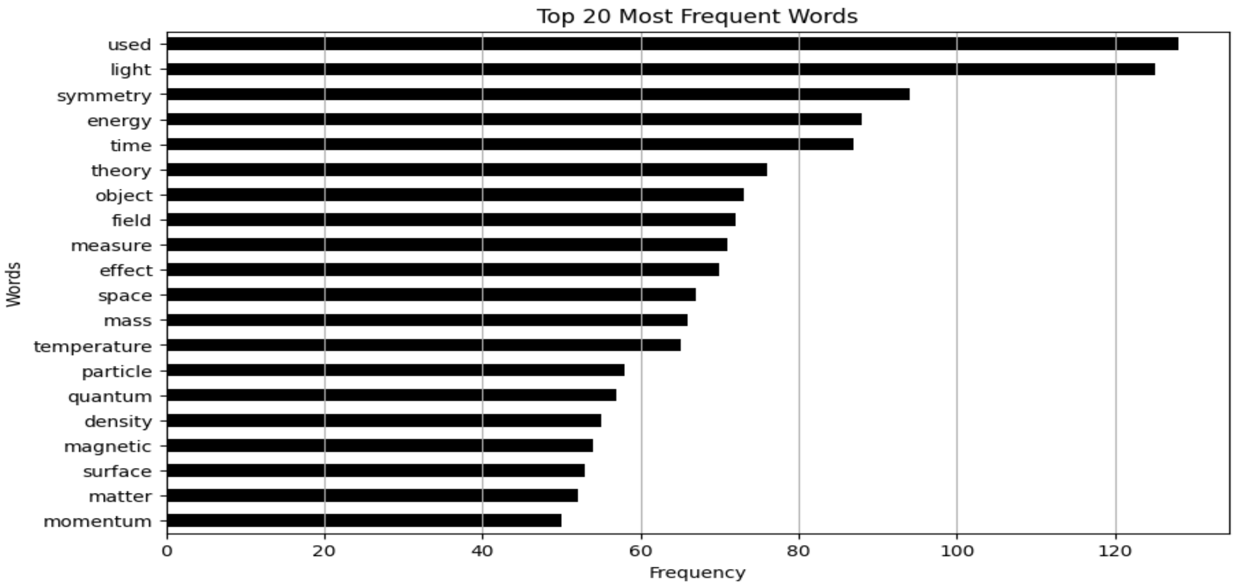


## DATA VISULIZATION PLOTS:

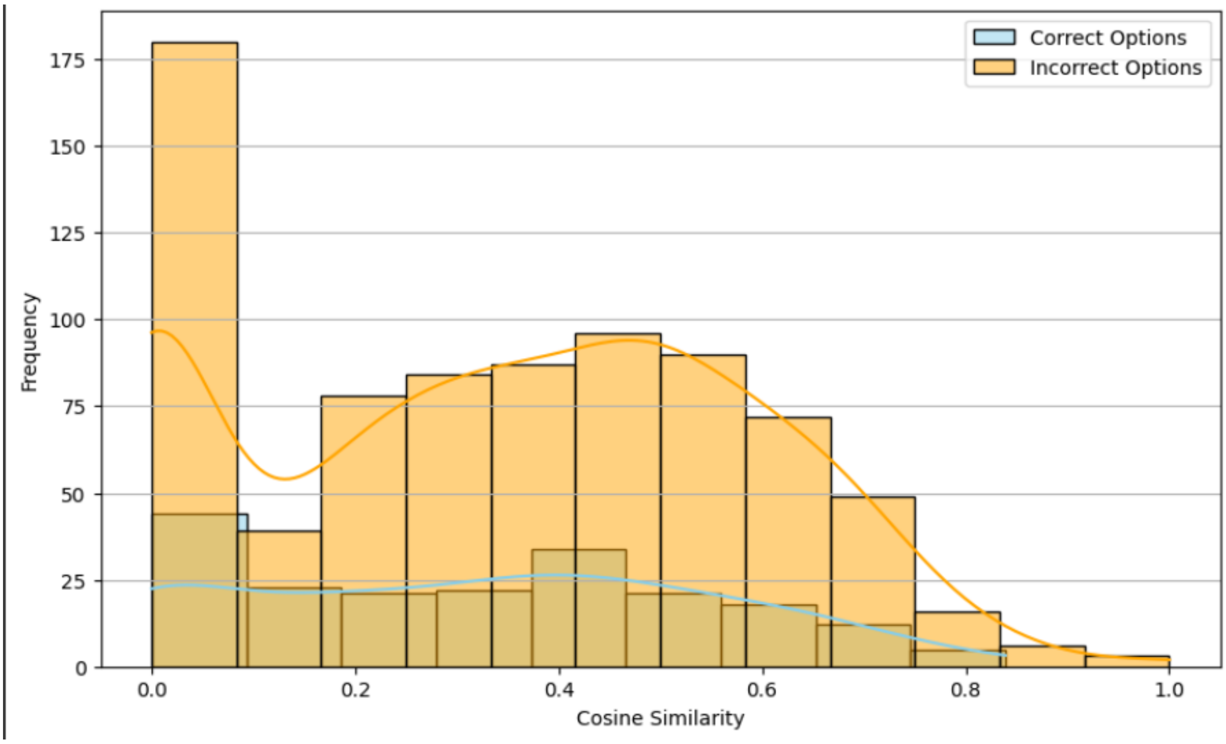
### Histogram for each option Length:



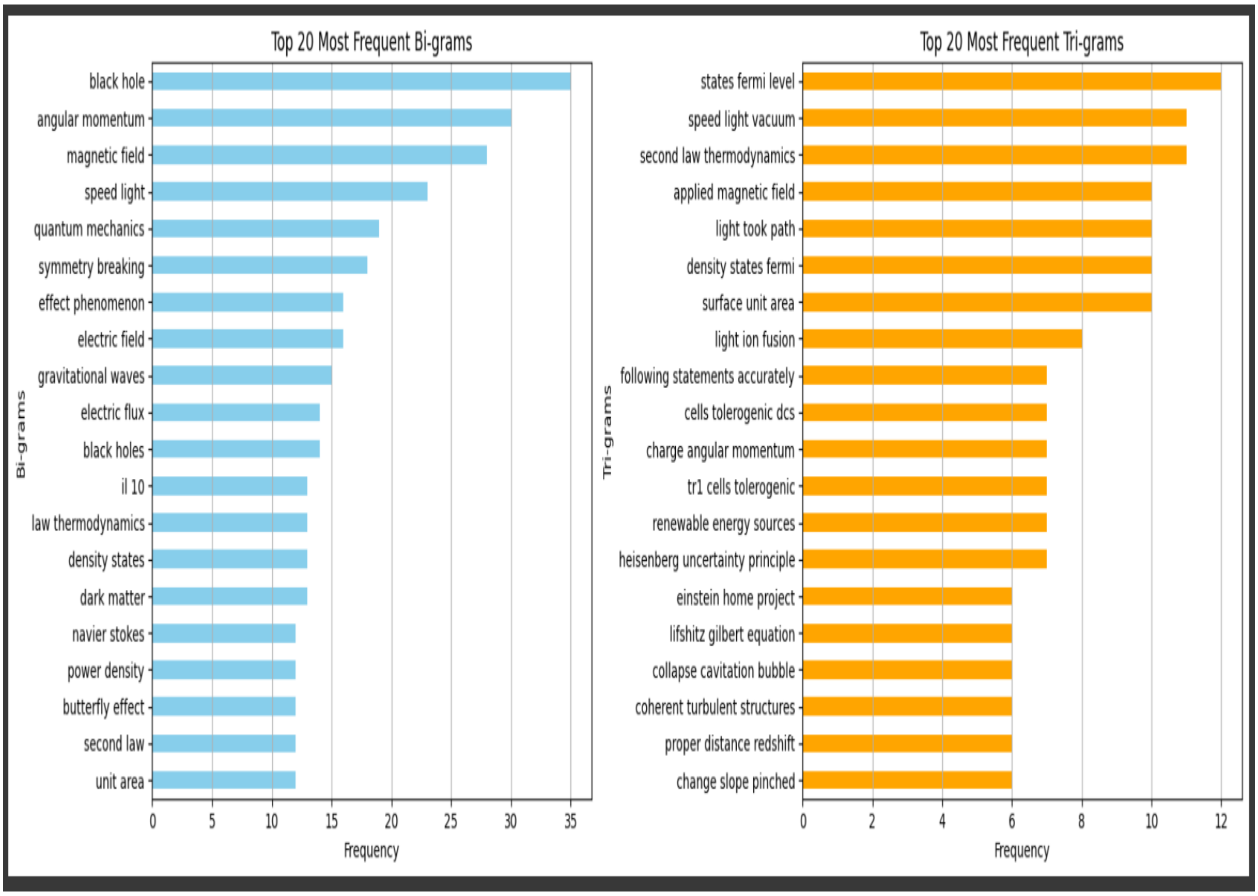

## WORD FREQUENCY:



## DISTRIBUTION OF COSINE SIMILARITIES:




## N – GRAM ANALYSIS



# REFERENCE


[1] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever. *Language Models are Unsupervised Multitask Learners*, [20 MARCH 2023]

[2] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray Benjamin, Chess Jack, Clark Christopher, Berner Sam, McCandlish AlecRadford, Ilya Sutskever, Dario Amodei. *Language Models are Few-Shot Learners*, [22 July 2020]

[3] Bishal Lamichhane. *Evaluation of ChatGPT for NLP-based Mental Health Applications*,
[28 Mar 2020]

[4] DONGJIE WANG, University of Central Florida, USA CHANG-TIEN LU, Virginia Tech, USA YANJIE FU, University of Central Florida, USA. *Towards Automated Urban Planning: When Generative and ChatGPT-like AI Meets Urban Planning*, [8 April 2023]

[5] YIHAN CAO∗ , Lehigh University & Carnegie Mellon University, USA SIYU LI, Lehigh University, USA YIXIN LIU, Lehigh University, USA ZHILING YAN, Lehigh University, USA YUTONG DAI, Lehigh University, USA PHILIP S. YU, University of Illinois at Chicago, USA LICHAO SUN, Lehigh University, USA. *A Comprehensive Survey of AI-Generated Content (AIGC): A History of Generative AI from GAN to ChatGPT*, [7 March 2023]

[6] Ding-Qiao Wang, Long-Yu Feng, Jin-Guo Ye, Jin-Gen Zou, Ying-Feng Zheng. *Accelerating the integration of ChatGPT and other large- scale AI models into biomedical research and healthcare*,
[10 February 2023]

[7] Niful Islam1 , Debopom Sutradhar1 , Humaira Noor1 , Jarin Tasnim Raya2 , Monowara Tabassum Maisha2 , Dewan Md Farid1 1Department of CSE, United International University (UIU), Bangladesh 2Department of CSE, University of Asia Pacific (UAP), Bangladesh. *Distinguishing Human Generated Text From ChatGPT Generated Text Using Machine Learning*, [26 May 2023]

[8] Michael Agyemang Adarkwah, Samuel Amponsah, Micheal M van Wyk, Ronghuai Huang, Ahmed Tlili, Boulus Shehata, Ahmed Hosny Saleh Metwally, Huanhuan Wang. *Awareness and acceptance of ChatGPT as a generative conversational AI for transforming education by Ghanaian academics: A two-phase study,* [10 September 2023]

[9] Stephen Atlas, *ChatGPT for Higher E ChatGPT for Higher Education and Pr ducation and Professional De essional Development: A elopment: A Guide to Conversational AI,* [1st January 2023]

[10] Fei-Yue Wang, Juanjuan Li, Rui Qin, Jing Zhu, Hong Mo, Bin Hu, *ChatGPT for Computational Social Systems: From Conversational Applications to Human-Oriented Operating Systems,* [2 April 2023]

[11] Michal Kosinski, *Theory of Mind May Have Spontaneously Emerged in Large Language Models*, [4 Feb 2023]

[12] Brady D. Lund, Ting Wang, Nishith Reddy Mannuru, Bing Nie, Somipam Shimray, *Ziang Wang. ChatGPT and a New Academic Reality: AI-Written Research Papers and the Ethics of the Large Language Models in Scholarly Publishing,* [10 March 2023]

[13] Brady D. Lund and Ting Wang, *Chatting about ChatGPT: How may AI and GPT impact academia and libraries?,* [30 July 2023]

[14] Dinesh Kalla (Doctoral Candidate) Colorado Technical University Microsoft (Big Data Support Escalation Engineer) Charlotte, North Carolina, Nathan Smith (Doctoral Candidate) Colorado Technical University Collins (Aerospace Principle Technical Publications Specialist) San Diego, California, *Study and Analysis of Chat GPT and its Impact on Different Fields of Study,* [ 3 March 2023]

[15] Aakash Ahmad1 , Muhammad Waseem2 , Peng Liang3 , Mahdi Fehmideh4 , Mst Shamima Aktar3 and Tommi Mikkonen2. *Towards Human-Bot Collaborative Software Architecting with ChatGPT,* [26 February 2023]

*************************************************************************