**DEVELOPMENT OF AN INTELLIGENT EMAIL AUTO-RESPONSE SYSTEM USING NATURAL LANGUAGE PROCESSING TECHNIQUES**

**BY**

**SOGUNLE,** OLUBUSOLAMI ROSEMARY **(19CG026487)**

**A PROJECT SUBMITED TO THE DEPARTMENT OF COMPUTER AND INFORMATION SCIENCES, COLLEGE OF SCIENCE AND TECHNOLOGY, COVENANT UNIVERSITY OTA, OGUN STATE.**

**IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF THE BACHELOR OF SCIENCE (HONOURS) DEGREE IN COMPUTER SCIENCE.**

**JULY, 2023**

CERTIFICATION

I hereby certify that this project, was carried out by Olubusolami Rosemary SOGUNLE in the Department of Computer and Information Sciences, College of Science and Technology, Covenant University, Ogun State, Nigeria, under my supervision.

|  |  |
| --- | --- |
| **Dr. Aderonke A. Oni** |  |
| *Supervisor* | **Signature and Date** |
|  |  |
| **Prof. Olufunke O. Oladipupo** |  |
| *Head of Department* | **Signature and Date** |

DEDICATION

I dedicate this work to God, who has been my very present help in times of need during my 4-year journey in this institution. I also dedicate this work to my friends and family, who support me.

ACKNOWLEDGEMENTS

My most profound gratitude goes to Almighty God, who has kept me and sustained me right from the very start of my degree up until this point. To Him be all the glory.

I want to express my sincere gratitude to my elder brother, Segun Sogunle, who encouraged me to study computer science and continually encouraged me all through my four-year journey to never give up in the face of challenges, especially during the course of this project.

I am extremely grateful to my parents, Dr Taiwo Sogunle and Mrs Ibironke Sogunle for their sacrifices, contributions, and prayers towards the completion of this degree. I also want to appreciate my amazing sisters, Eniola Sogunle and Omowunmi Sogunle and my cousin, Lawal Jomiloju, for their pieces of advice, support and encouragements during the course of this project and degree. I am extremely privileged to have them in my life.

My sincere gratitude also goes to my hardworking supervisor, Dr Aderonke Oni, who provided her support even when she was on a work break. Thank you for your excellent supervision and guidance ma.

I also appreciate the contributions of Dr Oluranti Jonathan, Dr Azubuike Ezenwoke, Mr Oshofuye Damilola, Mr Femi Odusote and Mrs Theresa Abiodun towards shaping this project work into a much better form. I am very grateful for all your insights, and I do not take them for granted.

TABLE OF CONTENTS

**CONTENT PAGE**

TITLE PAGE i

[CERTIFICATION ii](#_Toc144989225)

[DEDICATION iii](#_Toc144989226)

[ACKNOWLEDGEMENTS iv](#_Toc144989227)

[TABLE OF CONTENTS v](#_Toc144989228)

[LIST OF FIGURES ix](#_Toc144989229)

[LIST OF TABLES x](#_Toc144989230)

[ABSTRACT xi](#_Toc144989231)

[CHAPTER ONE: INTRODUCTION 1](#_Toc144989232)

[1.1 BACKGROUND INFORMATION 1](#_Toc144989234)

[1.2 Statement Of The Problem 2](#_Toc144989235)

[1.3 Aim And Objectives Of The Study 2](#_Toc144989236)

[1.4 Methodology 3](#_Toc144989237)

[1.5 Significance Of The Study 5](#_Toc144989238)

[1.6 Limitations Of The Study 5](#_Toc144989239)

[1.7 Project Organisation 6](#_Toc144989240)

[CHAPTER TWO: LITERATURE REVIEW 7](#_Toc144989241)

[2.1 Preamble 7](#_Toc144989243)

[2.2 Review Of Email Communication 7](#_Toc144989244)

[2.3 Review Of Email Overload 8](#_Toc144989245)

[2.4 Review Of Relevant Concepts 9](#_Toc144989246)

[2.4.1 Artificial Intelligence 9](#_Toc144989247)

[2.4.2 Machine Learning 10](#_Toc144989248)

[2.4.3 Natural Language Processing (NLP) 10](#_Toc144989249)

[2.4.4 Deep Learning 11](#_Toc144989250)

[2.4.5 Transfer Learning 12](#_Toc144989251)

[2.4.6 Transformers 13](#_Toc144989252)

[2.5 Review Of Related Methods 16](#_Toc144989253)

[2.5.1 Pattern Matching 16](#_Toc144989254)

[2.5.2 Semantic Textual Similarity (STS) 18](#_Toc144989255)

[2.5.3 Case-Based Reasoning (CBR) 18](#_Toc144989256)

[2.5.4 Neural Networks 19](#_Toc144989257)

[2.6 Review Of Existing Systems 22](#_Toc144989258)

[2.6.1 Gmail 22](#_Toc144989259)

[2.6.2 Microsoft Outlook 24](#_Toc144989260)

[2.7 Summary Of Literature Review 25](#_Toc144989261)

[CHAPTER THREE](#_Toc144989262)[: SYSTEM ANALYSIS AND DESIGN 26](#_Toc144989263)

[3.1 PREAMBLE 26](#_Toc144989264)

[3.2 The Proposed Sytem 26](#_Toc144989265)

[3.3 Requirements Analysis 26](#_Toc144989266)

[3.3.1 Functional Requirements 26](#_Toc144989267)

[3.3.2 Non-Functional Requirements 27](#_Toc144989268)

[3.4 Data Collection 27](#_Toc144989269)

[3.4.1 Description of the Enron Email Dataset 27](#_Toc144989270)

[3.4.2 Description of the Contrived Email Dataset 28](#_Toc144989271)

[3.5 Physical Design 32](#_Toc144989272)

[3.5.1 GPT Model Architecture 33](#_Toc144989273)

[3.5.2 The Proposed System Architecture 34](#_Toc144989274)

[3.6 Logical Design 37](#_Toc144989275)

[3.6.1 Use Case Diagram 37](#_Toc144989276)

[3.6.2 Activity Diagram 38](#_Toc144989277)

[3.6.3 Sequence Diagram 39](#_Toc144989278)

[3.7 Conceptual Design 41](#_Toc144989279)

[CHAPTER FOUR: SYSTEM IMPLEMENTATION AND EVALUATION](#_Toc144989280)

[43](#_Toc144989281)

[4.1 PREAMBLE 43](#_Toc144989282)

[4.2 System Requirements 43](#_Toc144989283)

[4.2.1 Hardware Requirements 43](#_Toc144989284)

[4.2.2 Software Requirements 44](#_Toc144989285)

[4.3 Implementation Tools 44](#_Toc144989286)

[4.3.1 Python 44](#_Toc144989287)

[4.3.2 JavaScript 45](#_Toc144989288)

[4.3.3 Jupyter Notebooks 45](#_Toc144989289)

[4.3.4 Visual Studio Code 45](#_Toc144989290)

[4.4 Development Methodology 45](#_Toc144989291)

[4.5 Evaluation Of The System 46](#_Toc144989292)

[4.5.1 Evaluation Criteria 46](#_Toc144989293)

[4.5.2 Evaluation Results 47](#_Toc144989294)

[4.5.3 Discussion 49](#_Toc144989295)

[4.6 Program Modules And Interfaces 50](#_Toc144989296)

[4.6.1 The Login Module 50](#_Toc144989297)

[4.6.2 The Inbox Module 51](#_Toc144989298)

[4.6.3 The Settings Module 52](#_Toc144989299)

[CHAPTER FIVE: SUMMARY, RECOMMENDATIONS AND CONCLUSION 53](#_Toc144989300)

[5.1 Summary 53](#_Toc144989302)

[5.2 Recommendations 54](#_Toc144989303)

[5.3 Conclusion 54](#_Toc144989304)

[REFERENCES 55](#_Toc144989305)

LIST OF FIGURES

[Figure 2.1: Transformer Architecture 14](#_Toc144988634)

[Figure 2.2: Gmail filter-templates Auto-response System (1) 23](#_Toc144988635)

[Figure 2.3: Gmail filter-templates Auto-response System (2) 23](#_Toc144988636)

[Figure 2.4: Gmail Smart Reply 24](#_Toc144988637)

[Figure 2.5: Microsoft Outlook Auto-response feature 25](#_Toc144988638)

[Figure 3.1: A sample of the rows and columns of the pre-processed Enron dataset 28](#_Toc144988639)

[Figure 3.2: A sample of the rows and columns of the contrived dataset 32](#_Toc144988640)

[Figure 3.3: GPT Architecture 34](#_Toc144988641)

[Figure 3.4: Architecture of the Proposed System 36](#_Toc144988642)

[Figure 3.5: Use case diagram for the proposed system. 38](#_Toc144988643)

[Figure 3.6: Activity Diagram for the proposed system 39](#_Toc144988644)

[Figure 3.7: Sequence Diagram for the Proposed System 40](#_Toc144988645)

[Figure 3.8: Entity Relationship Diagram for the Proposed System 41](#_Toc144988646)

[Figure 4.1: Precision-Recall Plot of the GPT2 models fine-tuned with the Enron and Contrived datasets 49](#_Toc144988647)

[Figure 4.2: Responses generated by the GPT2 model fine-tuned with the contrived dataset 50](#_Toc144988648)

[Figure 4.3: Sign in module/page 51](#_Toc144988649)

[Figure 4.4: Inbox module/page 52](#_Toc144988650)

[Figure 4.5: Settings Page 52](#_Toc144988651)

LIST OF TABLES

[Table 1.1: Objectives-Methodology Mapping Table 4](#_Toc144988823)

[Table 2.1: Comparison of the 3 major transformers considered 15](#_Toc144988824)

[Table 2.2: Comparison of the 4 considered methods with the Transformers method 21](#_Toc144988825)

[Table 3.1: Descriptions of the classes of the Contrived Dataset 29](#_Toc144988826)

[Table 4.1: Hardware Requirements Table 43](#_Toc144988827)

[Table 4.2: Software Requirements Table 44](#_Toc144988828)

[Table 4.3: BLEU Scores of GPT2 Instances 47](#_Toc144988829)

[Table 4.4: ROUGE scores for GPT2 re-trained and tested with Enron emails 48](#_Toc144988830)

[Table 4.5: ROUGE scores for GPT2 re-trained and tested with contrived emails 48](#_Toc144988831)

[Table 4.6: ROUGE scores for Base GPT2 tested against sample Enron emails 48](#_Toc144988832)

[Table 4.7: ROUGE scores for Base GPT2 tested against sample contrived emails 49](#_Toc144988833)

ABSTRACT

Email Communication is one of the most ubiquitous forms of online communications in the world today. It has many benefits such as allowing easy communication over a long distance, use in official settings and so on. However, with the benefits comes a significant problem called Email Overload, and several research works have been geared towards solving this problem. This project aims to mitigate the effects of email overload by developing an intelligent auto-response system using Natural Language Processing techniques. This will help to reduce the amount of time a user spends responding to repetitive emails. During the project, requirements were gathered, data was collected, the system was analysed and designed, and finally implementation and evaluation were carried out. In this project, the technique of transfer learning is carried out by fine-tuning Generative Pre-trained Transformer 2 (GPT2) with a pre-processed form of the Enron email dataset. A contrived email dataset is also created to test the effect of structure on the results of the fine-tuning and positive results were achieved. The GPT2 model trained with Enron email dataset achieved a better BLEU score, while the one trained with the contrived dataset achieved significantly better ROUGE scores (recall, precision and F1 score).

CHAPTER ONE

# INTRODUCTION

## BACKGROUND INFORMATION

The 21st century has seen the development of new disruptive fields like Artificial Intelligence (AI), Block Chain, Internet of Things (IoT), ICT and many others as a result of the rapid advancements in technology (Bongomin, Gilibrays, Oyondi, Musinguzi, & Omara, 2020). Several communication technologies have been introduced through ICT to help bridge the communication gap even more. In recent years, the development of internet-based communication technologies (CT) like Email, has transformed fields such as education, allowing for improved collaboration between instructors and learners, therefore ushering in a digital transformation (Santos*,* Batista, & Marques, 2019).

Electronic mail (Email) is one of the most ubiquitous communication technologies used today, and its impact can be seen across most sectors of the world economy. The total number of emails sent and received daily is estimated to reach 376.4 billion by 2025 (Statista, 2021). The global number of email users is also estimated to reach 4.7 billion by 2026 (Radicati Team, 2022).

Despite the positive impact of email on global communication, there are some drawbacks. Because of the high amount of information exchanged daily, one prevalent concern related to emails is information overload. Information Overload is simply a phenomenon whereby an individual receives much more information than they can process (Roetzel, 2019). This can lead to emails going unanswered for extended periods of time, leaving senders feeling abandoned.

Implementing an intelligent automatic response mechanism or system that works in tandem with existing email management solutions (such as spam management) will significantly improve email users’ response times. Such improvement in response time will have a beneficial ripple effect across the economy, as procedures involving information transmission via email in education, business, healthcare, and many other areas will be carried out faster and easier.

There are already automatic response mechanisms available from Google, Microsoft and other organizations that help to generate and send automatic responses to users. A succinct example is the canned response in Gmail, implemented using its filter and template features. However, a downside to the canned response is that it only checks for exact words, and if the words in a received email message do not directly match those in the predefined template, nothing will be sent.

The benefits of having an intelligent automated response system are numerous. In education, lecturers can now more thoughtful responses generated and sent to students who send repetitive emails. In healthcare, patients can get quick answers, preserving their lives and health in the long run. In everyday life, the mental effort required to develop a response and the time involved are minimized, resulting in increased efficiency and productivity. Therefore, this study aims to reduce time spent on responding to repetitive emails and the mental exertion involved.

## Statement Of The Problem

Over the years, there has been a steady increase in the amount of information that individuals all over the world have to process, especially in very limited time frames. One of the many sources of this large amount of information is emails, resulting in the problem of email overload. The mental exertion and time required to respond to emails sometimes take a toll on the email users or recipients, either physically or mentally. This highlights the need to channel technological advancements towards addressing the issue of email overload.

Researchers have made several attempts to develop methods to minimize email overload. Some are designed to help with spam management, while others are intended to help with email composition and response generation. Currently, the bulk of email response solutions offered by prominent email client businesses either involve excessive manual setup or do not deliver responses that capture the complete essence and context of a received email. As a result, an intelligent automatic response system that is simple to use and can offer sensible responses to incoming emails is required.

## Aim And Objectives Of The Study

The aim of this study is to develop an intelligent auto-response system using Natural Language Processing techniques. The objectives of this study are:

1. To identify requirements for intelligent auto-response email.
2. To gather and pre-process the dataset to be used for training the response model.
3. To model and design the components of the intelligent auto-response system.
4. To implement a working prototype of the intelligent auto-response system.
5. To evaluate the performance of the auto-response system.

## Methodology

1. Review of existing scholarly literature and direct observation of existing systems.
2. Cleaning and processing of the dataset using Python libraries such as Pandas, Natural Language Toolkit (NLTK) and Spacy.
3. Modelling and designing the components of the auto-response system using UML diagrams such as activity diagram, sequence diagram, use case diagram etc. as well as the GPT2 architecture.
4. Implementation of a working prototype of the system using Figma for the interface design, React framework for the frontend, Python Flask for the backend and transfer learning with GPT2 for the response model.
5. Evaluation of the performance of the auto-response system using standard metrics like BLEU and ROUGE.

Table 1.1 gives a mapping of the objectives and methodology in a tabular format.

Table 1.1: Objectives-Methodology Mapping Table

|  |  |  |
| --- | --- | --- |
| **S/N** | **OBJECTIVES** | **METHODOLOGY** |
| 1. | To identify requirements for intelligent auto-response email. | **Literature Review and Existing Systems**  1. Review of scholarly literature and peer reviewed articles.  2. Direct observation of existing systems. |
| 2. | To acquire and pre-process the dataset to be used for training the response model. | **Data Gathering and Pre-processing**  Cleaning and processing of the dataset using Python libraries such as Pandas, Natural Language Toolkit (NLTK) and Spacy. |
| 3. | To model and design the intelligent auto-response system. | **System Design and Modelling**  1. Modelling of requirements using UML diagrams: Use-case, Activity and Sequence Diagrams.  2. System Architecture design using Draw.io  3. Generative Pretrained Transformer (GPT) Architecture design.  4. Database Design modelling using Entity Relationship Diagrams. |
| 4. | To implement a working prototype of the intelligent auto-response system. | **Implementation and Prototype Development**  1. Interface Design using Figma and react  2. Middleware engine development using Python Flask  3. Response model development using transfer learning with GPT-2.  4. Database creation using PostgreSQL |
| 5. | To evaluate the performance of the auto-response system. | **Evaluation of the System**  Model evaluation using BLEU score and ROUGE score. |

## Significance Of The Study

The approach or tool proposed in this study improves on already existing auto-response solutions in the ICT industry by generating content-specific and intelligent responses by applying Machine Learning and Natural Language Processing Techniques. This study is of the most significance in corporate work environments where individuals receive emails daily in the following ways:

1. In academic environments, lecturers can better deal with students’ emails which are usually repetitive.
2. Large commercial organizations in which information is transmitted and required at very high speeds, and email requests are usually repetitive.

## Limitations Of The Study

The major limitations of the study include:

1. The intelligent auto-response system can only process emails in English Language.
2. Lack of adequate and structured email response generation datasets for training the response model. The dataset used was the Enron email dataset which had a huge bias towards Enron activities.

## Project Organisation

Chapter One of the project contains an explanation of the project, problems with existing solutions, the need for an improved solution, the method of implementation, the significance of the study, and the limitations. Chapter Two describes the existing systems related to the project topic, the methodology, algorithm, and techniques used in related systems. Chapter Three describes the analysis and system design. Chapter Four shows the implementation of the system in detail and the results obtained. Chapter Five summarises the project and gives recommendations, suggestions, conclusions, and references.

CHAPTER TWO

# LITERATURE REVIEW

## Preamble

This study aims to develop an auto-response system that reduces the effects of email overload by utilizing modern technologies like Machine Learning and Natural Language Processing to improve the basic features of sending and receiving emails provided by existing email clients. This literature review begins with a review of email communication, email overload and its effects. It then proceeds to identify four major methods that have been employed by researchers to generate intelligent email responses.

## Review Of Email Communication

The Electronic mail, popularly called Email, can be described as a type of digital communication that allows individuals or users to exchange messages and files over the Internet. Email first appeared 1960s when researchers were experimenting with using computers to communicate with one another. It is important to note that Email came into existence before the Internet. The first version of Email was created in 1965 at the Massachusetts Institute of Technology (MIT) as part of the institution’s Compatible Time-Sharing System. The major purpose was to allow users to exchange documents and texts on a central disk while logging in from remote terminals (Gibbs, 2016).

Ray Tomlinson invented Email as we know it today in 1971. He came up with the idea of using the @ symbol to separate the username from the domain name in an email address while working on the Advanced Research Projects Agency Network (ARPANET) project, which was a forerunner to the Internet. Tomlinson was prompted to come up with this idea due to an Internet Request for Comments published by Dick Watson (Watson, 1971). The publication described a method for sending emails to teletype printers, each with its own mailbox. Tomlinson thought the approach was flawed because users should be the ones with mailboxes and decide whether or not to print (Partridge, 2008).

The 1970s and 1980s saw a very significant and promising increase in the use of email. Individuals, especially scholars and researchers, started to use the email more. With the growth in traction of the World Wide Web in the 1990s, came the growth of email. This was because there were more people enlightened and willing to use the Internet at this point.

Electronic mails experienced a very impressive increase in usage with an enormous amount (billions) of emails being sent and received every single day by the 1990s and 2000s. This widespread adoption can be attributed to the emergence of popular providers of email services including Hotmail, Yahoo Mail and Gmail that provided free email services to users.

A lot of effort has also been put towards making the email more secure and reliable. For instance, functionalities like encrypted messaging, automated filters, and even junk mail (spam) protection have been introduced in recent times. Even with the rise of social media and other digital communication channels, email continues to be a mainstay of internet communication.

## Review Of Email Overload

Over the years, independent researchers and even researchers at technological companies have attempted to develop techniques and features to mitigate the effects of email overload. The term “email overload” was first described in 1996 as utilizing Email for tasks other than asynchronous communication which is its original purpose (Whittaker & Sidner, 1996).

The use of emails over the years has evolved to include much more functions than just asynchronous communication. New functions for the email include task management and personal archiving (Whittaker & Sidner, 1996), email marketing – which is a subset of digital marketing (Bala & Verma, 2018) and many others. It should be emphasized that email marketing accounts for a significant portion of what is known as “Spam,” which further contributes to email overload (Fariborzi & Zahedifard, 2012).

Research works on email overload frequently refer to, or reference works on information overload because email overload is one of several types of information overload. Information overload is a phenomenon where the amount of information available exceeds the processing capacity of the recipients (Roetzel, 2019).(Roetzel, 2019) The adverse effects of information overload can both be psychological and economic. Some psychological repercussions of email information overload include stress and blurring work-life boundaries. (Lanctot & Duxbury, 2021), while the economic repercussions can be calculated in the millions of dollars lost due to reduced productivity and innovation (Hemp, 2009; Spira & Burke, 2009).

(*Information Overload: Now $900 Billion – What Is Your Organization’s Exposure? - Basex Blog »*, n.d.)(Roetzel, 2019)(Roetzel, 2019)Several research works have identified the best approaches to reduce the effects of information overload. Some approaches are focused on reducing the amount of incoming information or information that users have to process, while others are focused on helping the recipients increase their information processing capacity (Soucek & Moser, 2010). This research work focuses primarily on the approach of managing and reducing the amount of information that users must process, specifically in their emails.

## Review Of Relevant Concepts

This section details the various concepts that are relevant to this project. They include:

### Artificial Intelligence

Artificial intelligence is the ability of machines or computer programs to perform tasks that typically require human intelligence. In artificial intelligence, computers are used to simulate or mimic human intelligence. It is a vast field with contributions from numerous disciplines, such as philosophy, mathematics, economics, neuroscience etc. It is also made up of 6 significant sub-disciplines, namely Natural Language Processing (NLP), Machine Learning (ML), Knowledge Representation, Computer Vision, Robotics and Automated Reasoning (Russell & Norvig, 2020).

Artificial intelligence has radically altered the way things are done across the world. It has improved the efficiency of carrying out otherwise monotonous and repetitive operations. Personalized learning experiences for learners are now available in schools, enhancing learning ability. It has also found use in the automobile and healthcare industries (Zhang & Lu, 2021).

### Machine Learning

Machine learning is a branch of Artificial Intelligence (AI) that focuses on creating algorithms and statistical models that allow computer systems to learn and improve without being programmed explicitly. It can be described as the study of how computers can learn without being expressly coded.

There are several machine learning algorithms that both researchers and machine learning engineers have used over the years. Some examples of algorithms used frequently in machine learning include Linear Regression, Logistic Regression, Gradient Descent, Support Vector Machines, K-Nearest Neighbour and so on (Ray, 2019).

### Natural Language Processing (NLP)

Natural Language Processing (NLP) is a branch of Artificial Intelligence (AI) that studies the3 interface between human language and computer systems. It entails the creation of algorithms and models that allow computers to comprehend, interpret, and generate or synthesize human language. Russell and Norvig (2020) outlined three major reasons for computers to perform Natural Language Processing. The first is to communicate, the second is to learn, and the third is to enhance scientific comprehension of languages.

NLP is widely used for tasks such as text classification, indexing and searching enormous texts, data extraction, automatic summary generation, question answering, knowledge acquisition, and text generation (Chowdhary, 2020).

There are two major subsets of Natural Language Processing, namely Natural Language Understanding (NLU) and Natural Language Generation (NLG):

1. Natural Language Understanding (NLU)

This simply refers to the capacity of machines to perceive, interpret, and process human language in a manner comparable to that of humans.

It is a component of Natural Language Processing (NLP) that focuses on comprehending and extracting insights from text or audio (speech) inputs. Many applications rely on it, including AI-powered assistants, chatbots, search engines, and language translation services. It enables machines to understand and respond to user questions and demands easily and instinctively (Navigli, 2018).

Language identification, tokenization, part-of-speech tagging, syntactic and semantic parsing, named entity recognition, sentiment analysis, and machine translation are some of the activities involved in Natural Language Understanding (NLU). These tasks enable machines to understand and respond to unstructured textual information.

1. Natural Language Generation (NLG)

This branch of Natural Language Processing (NLP) involves creating natural language text or voice from structured or unstructured input.

Without human interaction, Natural Language Generation (NLG) algorithms evaluate data and convert it into human-readable formats such as reports, summary data, writings, emails, and chatbot responses. The output of a Natural Language Generation (NLG) task is often textual; however, the input might be in the form of images, sounds, or text (Gatt & Krahmer, 2017).

Natural Language Generation (NLG) systems generate correctly constructed, logical, and semantically relevant language using statistical techniques, rule-based approaches, deep learning, and neural networks. By training on massive volumes of data, these computers can learn to emulate human writing styles and preferences.

NLG offers a wide range of applications, including corporate intelligence, marketing, customer service, education, journalism, and medicine. By providing fast and accurate information, NLG systems can automate repetitive writing jobs, increase interaction between stakeholders and clients, and aid in decision-making.

### Deep Learning

Deep learning, often known as deep neural networks, is a branch of machine learning that focuses on developing artificial neural networks with several layers. These neural networks are designed to learn from enormous amounts of data in a manner identical to how human beings learn. They are patterned after the layout of the human brain.

The advent of big data and cloud computing has aided in ushering in the era of deep learning. The amount of information available has grown tremendously over time, raising the issue of Information Overload. Deep learning fills a demand for an efficient and effective technique to extract relevant information from the massive amount of data accessible. It has made considerable progress in processing and utilizing these massive volumes of data (Mu & Zeng, 2019).

Through supervised learning, the neural network is trained to recognise patterns in the input data in deep learning. The more data the network is trained on, the better it becomes at spotting patterns and forecasting. Unsupervised and semi-supervised learning approaches can also be utilised. Deep learning has been used successfully in a variety of applications such as image recognition, speech recognition, natural language processing, medical imaging and autonomous driving (Shinde & Shah, 2018).

Natural Language Processing (NLP) has benefited greatly from deep learning. Techniques such as Neural Language Modelling, Word Embeddings, Parsing, and others have improved accuracy and efficiency in information retrieval, machine translation, information extraction, text categorization, summarization, and text generation (Otter, Medina, & Kalita, 2021).

Some recent trends in Deep Learning based Natural Language Processing (NLP) are distributed representations for feature engineering, Convolutional Neural Networks (CNNs) for information retrieval and Recurrent Neural Networks (RNNs) for language generation (Young, Hazarika, Poria, & Cambria, 2018). Others include Long Short-Term Memory (LSTM) as an improvement over plain RNNs for language generations (Yu, Si, Hu, & Zhang, 2019) and the introduction of attention mechanisms in transformers (Vaswani *et al.*, 2017). The adoption of the transfer learning/fine-tuning pair, which permits the reuse of powerful pre-trained models for a variety of use cases, is the most recent of these breakthroughs.

### Transfer Learning

Transfer learning is a machine learning strategy that includes exploiting knowledge learnt from a task to improve performance on a separate but connected task. In other words, transfer learning entails applying pre-trained models or knowledge from one activity to improve learning in another.

Transfer learning is often used in deep learning to improve a model's performance on a new task by applying knowledge learnt from a previously trained model. The pre-trained model can be trained on a huge dataset and utilized as a place to begin for a new, related task. The learnt parameters of the pre-trained model can be fine-tuned or updated with less data tailored to the current job, resulting in a model that can learn faster and with a greater degree of precision (Zhuang *et al.*, 2021).

Transfer learning can be described as a technique that is extremely useful in NLP tasks when there is a shortage of training data for a certain task. A pre-trained language model trained on a big corpus, for example, can be changed or fine-tuned to suit to a specific language-generation task using a comparatively smaller dataset. Researchers have also utilized transfer learning to improve performance on similar tasks, such as using a pre-trained sentiment analysis model to improve efficiency on a text classification task (Ruder, Peters, Swayadimpta, & Wolf, 2019).

Nowadays, transfer learning is an effective strategy for increasing the efficiency and accuracy of machine learning models, especially when significant amounts of training data are unavailable for use.Top of Form

### Transformers

A transformer is a type of deep learning model which makes use of a mechanism called self-attention. Self-attention, also known as scaled dot-product attention or intra-attention is simply a mechanism used by a transformer to determine the importance of each word in a sentence and the relationships between the words in the sentence. This is done by creating representations of each word that include context from other words (Vaswani *et al.*, 2017).

The use of self-attention helps the transformer to capture long-range dependencies i.e., determine relationships between words that are distant from each other in the same sequence, which is a major advantage when performing Natural Language Processing tasks such as summarization, question answering and so on.

Central to the transformer architecture are two core components: The encoder and the decoder. The encoder works by simultaneously assigning attention weights to every word in an input sequence. The process is then repeated for every word thus broadening the context and understanding of the model about the sequence. The result is a transformed representation of the input sequence that is context rich.

The decoder, on the other hand, takes the transformed representation and gives a word-by-word generation of the output sequence. The transformer model also utilises self-attention to understand the relationships between words generated at every point while encoder-decoder attention is used to focus on the relevant portion of the encoded input sequence. Both the encoder and decoder work together to ensure proper and contextual generation of output. Figure 2.1 shows the detailed architecture of a transformer (Vaswani *et al.*, 2017).

A picture containing text, diagram, screenshot, plan

Description automatically generated

Figure 2.1: Transformer Architecture  
Source: Attention is All You Need by Vaswani *et al.* (2017)

The introduction of the transformer architecture brought about a revolution in the way Natural Language Processing tasks are performed. This has led to the development of several types of transformers in recent years and three major ones are:

1. Bi-directional Encoder Representations from Transformers (BERT): This is a type of transformer which was pre-trained on a large corpus and can capture the context of an input sequence in bidirectional manner. It uses Masked Language Modelling (MLM) and can be used for Natural Language Processing tasks such as Question Answering and Sentiment Analysis (Devlin, Chang, Lee, & Toutanova, 2018).
2. Generative Pre-trained Transformers (GPT): This is a type of transformer which uses only the decoder part of the transformer architecture and is tailored towards language generation tasks. It uses an auto-regressive approach to generate text word by word (Radford, Narasimhan, Salimans, & Sutskever, 2018).
3. Text-to-Text Transformer (T5): This is a type of transformer that utilises a unified text-to-text approach and converts all Natural Language Processing tasks into language generation tasks (Raffel *et al.*, 2020).

Table 2.1 compares the 3 types of transformers relevant to this study in terms of several features.

Table 2.1: Comparison of the 3 major transformers considered

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S/N** | **FEATURE** | **GPT** | **BERT** | **T5** |
| 1. | **Method of pre-training** | Autoregressive Language Modelling | Masked Language Modelling (MLM) | Masked Language Modelling (MLM) |
| 2. | **Text generation capabilities** | Primarily designed for text generation | Not designed for text generation tasks. | Designed for text generation tasks. |
| 3. | **Architecture** | Decoder-only | Encoder-only (BERT base), Encoder-Decoder (BERT large) | Encoder-Decoder |
| 4. | **Usage** | Language generation tasks | Broad range of Natural Language Processing tasks like Named Entity Recognition (NER) and Sentiment Analysis. | Wide range of Natural Language Processing tasks |
| 5. | **Model size** | Extremely large | Relatively large | Relatively large |

From the comparison, it can be deduced that the Generative Pretrained Transformer (GPT) is well suited for the intelligent email response generation approach carried out in this study as it was primarily designed for language generation tasks.

## Review Of Related Methods

Following a review of existing literature, it was discovered that several email response generation methods have been employed by both organisations and researchers alike to tackle the problem of email overload. The methods include Pattern Matching, Textual Similarity Matching, Case-Based Reasoning, and Neural Networks.

### Pattern Matching

Pattern matching in Natural Language Processing (NLP) simply refers to the process of identifying and matching a pre-defined sequence of words (patterns) in a target sequence of words i.e., a sentence. It is the process of searching for patterns in a given sequence of words according to some pre-defined rule, majorly for the purpose of extracting useful information.

In pattern matching, linguistic techniques like Regular Expressions are often used to capture the patterns of interest. Such patterns could be simple word sequences, syntactic structures, semantic relationships, or language patterns that are complex to a degree. Through pattern matching, these sequences can easily be detected for retrieval.

Pattern matching can either be rule-based or machine-learning-based. Rule-based pattern matching systems are systems in which the experts manually define the patterns to be used for the pattern matching. This technique was utilised in a study conducted by Sneiders, Sjöbergh, & Alfalahi (2016). The authors attempted to use textual pattern matching by regular expressions to categorize incoming emails, which would then receive a specific response. They manually created text patterns that specify the request as well as the context of email messages and used them to classify the emails in their dataset. They also identified 12 reasons why the approach might fail, including vagueness in the Email, poorly structured messages, unique wordings, and others.

During the study, the authors discovered that inquiry emails typically have two parts: the context or description of the Email and the response trigger, which can be a complaint, a question, or a statement requiring action. In simple cases, it was noted that the context and response trigger would appear in the same sentence. However, in most cases, the context or description of the Email and the response trigger are usually separated in individual sentences.

Rule-based pattern matching is advantageous in that it gives room for precision, obtaining interpretable results and efficiency. However, it can be subject to limited coverage, poor context awareness and inability to handle vagueness or ambiguity. It also largely depends on the structure of the incoming text among other factors (Sneiders *et al.*, 2016).

Machine-Learning-based pattern matching involves the use of machine learning algorithms to learn and identify patterns in textual data intelligently and automatically. The patterns are often learnt by feeding the model with labelled data, and the model is then expected to learn the patterns and generalise to unseen data. It typically involves data preparation, feature extraction, training and finally using the model to identify patterns.

Pattern matching can be either used to extract information or classify data. As rule-based pattern matching systems often require a lot of domain knowledge and expertise, machine learning models like Convolutional Neural Networks (CNNs) are now trusted to perform exceedingly better at identifying patterns in labelled data and classifying text (Minaee *et al.*, 2021).

### Semantic Textual Similarity (STS)

Semantic Textual Similarity is a technique which measures the degree of relatedness of two samples or sequences of text. The output of the technique is usually a kind of score which depicts the level of similarity between the two texts.

Over the years, the methods for carrying out Semantic Textual Similarity have gone through a lot of evolution ranging from non-context aware methods like Jaccard Similarity Coefficient, Word Movers Distance (WMD) and Cosine Similarity with Bag of Words to context aware methods like using Cosine Similarity with Word Embeddings and Cross Encoders (Chandrasekaran & Mago, 2022).

Semantic similarity has found a lot of use in the intelligent email response generation field. A study conducted by Mittal and Ramamurthy (2016) introduced an automatic response system for e-learning usage. They presented an approach for generating automatic email responses by using semantic textual similarity. They also utilized Word Sense Disambiguation (WSD) to fully understand the meanings of the words. Lesk algorithm was utilized to implement the word sense disambiguation after which a semantic score was computed using Path algorithm.

Itakura, Kenmotsu, Oka, & Akiyoshi (2010) attempted matching query emails to pre-existing FAQs in a database. The authors utilized Support Vector Machines (SVM) and a modified Jaccard coefficient to perform the matching. The system worked by first breaking down the sentences into words, then computing the feature vectors using the modified Jaccard coefficient. Using the feature vectors, the Support Vector Machine model was able to check the similarity between the query email and the FAQs. Once it found a match, it sent an appropriate FAQ answer; otherwise, it sent the query email to the operators.

### Case-Based Reasoning (CBR)

Case-Based Reasoning is a problem-solving approach in which past experiences called cases, are used to provide solutions to new and arising problems. It works by searching for comparable cases to a new problem and their solutions, modifying those solutions as needed and using them for the problem (Smyth, 2019).

CBR involves several steps in solving a problem. Starting out, the system retrieves past cases like the current problem based on various factors. Once similar cases are found, the system reuses the solutions and knowledge from those cases, adapting them to fit the current problem. The adapted solution is then applied to the problem, with possible revisions and modifications to meet specific requirements. Finally, the solution and corresponding case are stored in a library, forming a knowledge base for future use. This iterative process allows the system to accumulate knowledge and effectively solve subsequent problems.

Naeem, Linggawa, Mughal, Lutteroth, & Weber (2018) created a novel strategy that entails using a case-based reasoning approach to retrieve old responses to respond to a new email. The previously written and sent replies are stored in the case base and are retrieved based on their level of similarity to the current email. To retrieve cases, the system uses similarity scores which are generated by calculating the dot product of the document vectors. The document vectors contain the vector of the terms that are contained in the documents. After sending a reply, the original email and the reply are stored as a case to be reused in the future.

Case-Based Reasoning is particularly useful in situations where explicit rules or formulas for problem-solving may be inadequate or unavailable. It leverages the similarity between problems and the knowledge stored in past cases to provide effective and context-specific solutions. CBR has been applied in various domains, including medicine (Bentaiba-Lagrid, Bouzar-Benlabiod, Rubin, Bouabana-Tebibel & Hanini, 2020), recommendation systems (Hernández-Nieves, Hernández, Gil-González, Rodríguez-González, & Corchado, 2021) and so on.

### Neural Networks

Neural Networks can be referred to as machine learning models modelled after the human brain that can identify patterns and relationships in data and learn from them. A neural network is a type of machine learning process called Deep Learning. It typically consists of 3 types of layers namely the input layer, the hidden layer(s), and the output layer. Neural Networks have found a lot of application in Natural language Processing in recent years.

In intelligent email response generation, several neural network techniques have been applied. In 2016, researchers at Google developed “Smart Reply” which is a Gmail inbox feature that suggests short email responses to users by leveraging Long Short Term Memory networks (LSTMs) which is a special and improved type of Recurrent Neural Networks (RNNs) (Kannan *et al.*, 2016). The model was trained on a large dataset of emails and their corresponding replies. It learnt to predict the most appropriate response given the email's content and context.

Feng, Naeem, Mirza, & Tahir (2020) described a novel deep learning technique in a smart email client that employed the Document2Vector (Doc2Vec) algorithm, and a hybrid model called GRU-Sent2Vec which combined the Gated Recurrent Unit (GRU) and Sentence to Vector (Sent2Vec) algorithms. The authors aimed to develop a system that could generate sentence-level predictions or responses as compared to the word-level responses generated in earlier research (Kannan *et al.*, 2016). The Doc2Vec algorithm was evaluated to be better than the TF-IDF algorithm in predicting intelligent responses. However, the researchers lacked a large enough dataset to properly evaluate the GRU-Sent2Vec approach.

Arsovski, Oladele, Cheok, Premcevski, & Markoski (2022) introduced a semi-automatic responder which differentiates between query and non-query emails and then generates automatic responses to the query emails. The query email dataset was divided into polar questions and factual questions. A polar question is a question whose answer can either be a “yes” or a “no”, while a factual question is a question that can only have a fact as its answer and usually consists of question words (who, why, where, what, when etc.). The approach employed the use of Natural Language Processing, Neural Networks and Artificial Intelligence Markup Language. Table 2.2 gives a comparison of the four intelligent response generation methods.

Table 2.2: Comparison of the 4 considered methods with the Transformers method

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S/N** | **METHOD** | **DESCRIPTION** | **ADVANTAGE** | **DISADVANTAGE** |
| 1. | Pattern Matching | Identifies predefined patterns or rules in text to generate responses. | Simple to implement compared to other methods | Limited flexibility and generalisation. |
| 2. | Semantic Textual Similarity (STS) | Measures the similarity between texts based on their semantic meaning. | Ability to capture semantic meaning and context. | Requires labelled training data for accurate similarity |
| 3. | Case-Based Reasoning | Uses past cases as a basis for generating responses to new problems. | Efficiently utilizes past experiences for relevant responses. | Requires extensive case library and retrieval mechanisms. |
| 4. | Neural Networks e.g. RNNs, LSTM etc. | Learn patterns and relationships from data to generate responses. | Can capture complex patterns and dependencies. | Requires large amounts of data and computational resources. |
| 5. | Transformers e.g. GPT2, BERT etc. | Advanced neural network models specifically designed for natural language processing tasks. | Can efficiently capture long-range dependencies and context in text.  Can also be fine-tuned for specific tasks with less computational power. | Requires significant computational resources for training or pre-training. |

From the table, it can be deduced that transformers like GPT2 are well suited for language generation tasks that do not involve labelled data and require generation of coherent and long sequences of texts. Thus, a transformer would be good fit for the intelligent email auto-response system.

## Review Of Existing Systems

Two major systems which utilise some of the previously mentioned methods will be considered in this section.

### Gmail

Gmail, the email client owned by the tech giant Google, provides many automation features for its users. An example is the sending of pre-written emails with “templates” which allows users to create generic replies to frequent and similar emails. Another feature in the Gmail client is the “filters” feature which helps to manage and sort incoming email messages through the creation of rules which apply to them.

The aforementioned Gmail features, templates and filters, can be coupled to build a system in which generic templates are automatically sent based on a set of filter criteria, emulating automatic response (Sanhz, 2022).

A disadvantage of this simple system is the lack of intelligence in the automatic response as replies will only be sent if incoming email messages contain the exact words pre-written in the email templates. This indicates the use of the pattern matching technique discussed in the previous section. The process of setting up this simple auto-response system is also complex and time consuming.

Another disadvantage of this system is that involves a lot of manual intervention and set up from the user. There is no visible option to set up the auto-response system easily and quickly. To get the system up and running, a user might have to go online to read about it and learn how to use the separate “template” and “filter” features, before learning how to combine them. Figures 2.2 and 2.3 show the set-up of the auto-response system in Gmail.

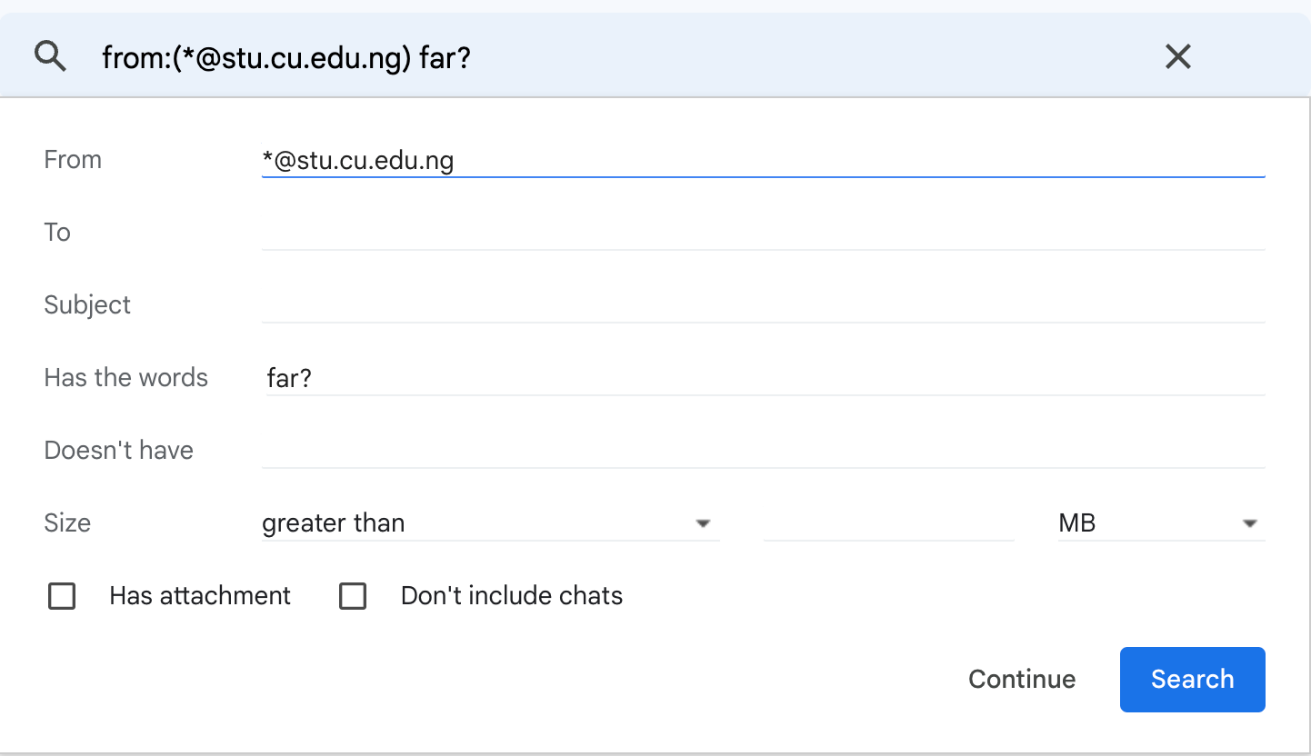


Figure 2.2: Gmail filter-templates Auto-response System (1)  
Source: https://mail.google.com/

A screenshot of a computer screen

Description automatically generated with medium confidence

Figure 2.3: Gmail filter-templates Auto-response System (2)  
Source: https://mail.google.com/

Gmail, however, possesses a “Smart Reply” feature that suggests short email responses to users by through a type of Recurrent Neural Network (RNN) called the Long Short Term Memory network (Kannan *et al.*, 2016). Figure 2.4 shows the smart reply feature in action. The major issue with the replies generated is that although they are contextually accurate, they are usually short.

Graphical user interface, text, application, chat or text message

Description automatically generated

Figure 2.4: Gmail Smart Reply  
Source: https://mail.google.com/

### Microsoft Outlook

Microsoft Outlook has an “automatic replies” feature which allows users to automatically reply to emails when they are unavailable or on vacation, as well as a “rules” feature which serves the same function as the Gmail “filters” feature. Figure 2.5 shows this feature.

This system has a similar disadvantage of a manual set up by the user like the Gmail client. Users must go to the settings to configure the “automatic replies” feature. It is also very limited in functionality as only one customized message can be sent to all incoming emails during a set time interval. It is most suited for quick out-of-office responses.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 2.5: Microsoft Outlook Auto-response feature  
Source: https://outlook.office365.com/

## Summary Of Literature Review

An extensive review of different methods shows that combining the transfer learning approach with transformers might produce better results and efficiency than using pattern matching or building a neural language model from scratch. The proposed intelligent email auto-response system will employ the transfer learning approach by fine-tuning the GPT-2 language model with a smaller email dataset.

CHAPTER THREE

# SYSTEM ANALYSIS AND DESIGN

## PREAMBLE

This section details the analysis and design of the proposed system. It presents the requirements for the system, the techniques taken to achieve the system as well as diagrams to model the proposed system.

## The Proposed Sytem

The proposed system will be a web-based email client running on the Microsoft Outlook server. The application will allow users to receive new emails and send replies. The replies sent will either be generated by them or the email client. For intelligently generated email responses, they can choose to allow the system automatically generate responses and send them or vet the automatically generated responses before it is sent out.

## Requirements Analysis

Requirements analysis refers to the process of gathering, analyzing, documenting and refining features or requirements for a software. It is also known as requirements engineering. Software requirements refer to the descriptions of functionalities provided by the software system as well as the constraints under which it operates. The requirements for this study are detailed in this section.

### Functional Requirements

This refers to the core functionality or services to be provided by the proposed software. These requirements are as follows:

1. Users shall be able to receive emails and reply to emails on the auto-response system.
2. Users shall be able to select between automatic reply or manual reply.
3. The system should be able to analyze incoming emails, extract relevant information such as sender, subject, and content, and understand the context of the message.
4. The system should be able to generate appropriate and contextually relevant responses based on the content and intent of the incoming email.

### Non-Functional Requirements

This refers to the descriptions of the system’s operational capabilities and constraints. They are as follows:

1. The system should ensure prompt handling of incoming emails and generation of replies within an acceptable timeframe.
2. The system should have a simple and intuitive interface for end-users.
3. The system should display new emails within an acceptable timeframe.
4. The system should seamlessly engage the Microsoft Outlook server for easy email retrieval and reply.

## Data Collection

Two major datasets were used during this project for re-training/fine-tuning the GPT2 model. The first dataset is the processed form of the Enron email dataset created by [Feng *et al.* (2020)](https://www.zotero.org/google-docs/?xc8rb3) while the second is a contrived email dataset created for the purpose of improving performance.

### Description of the Enron Email Dataset

The original dataset of Enron emails comprises around 500,000 emails that were authored by Enron Corporation employees. This dataset was acquired by the United States Federal Energy Regulatory Commission as part of their investigation into the company's collapse. The processed version utilised in this particular study contains approximately 19871 email pairs arranged in rows, featuring two columns: the "received" column and the "response" column. It should be noted that further cleaning and pre-processing were carried out. Figure 3.1 shows the first 20 rows of the dataset.

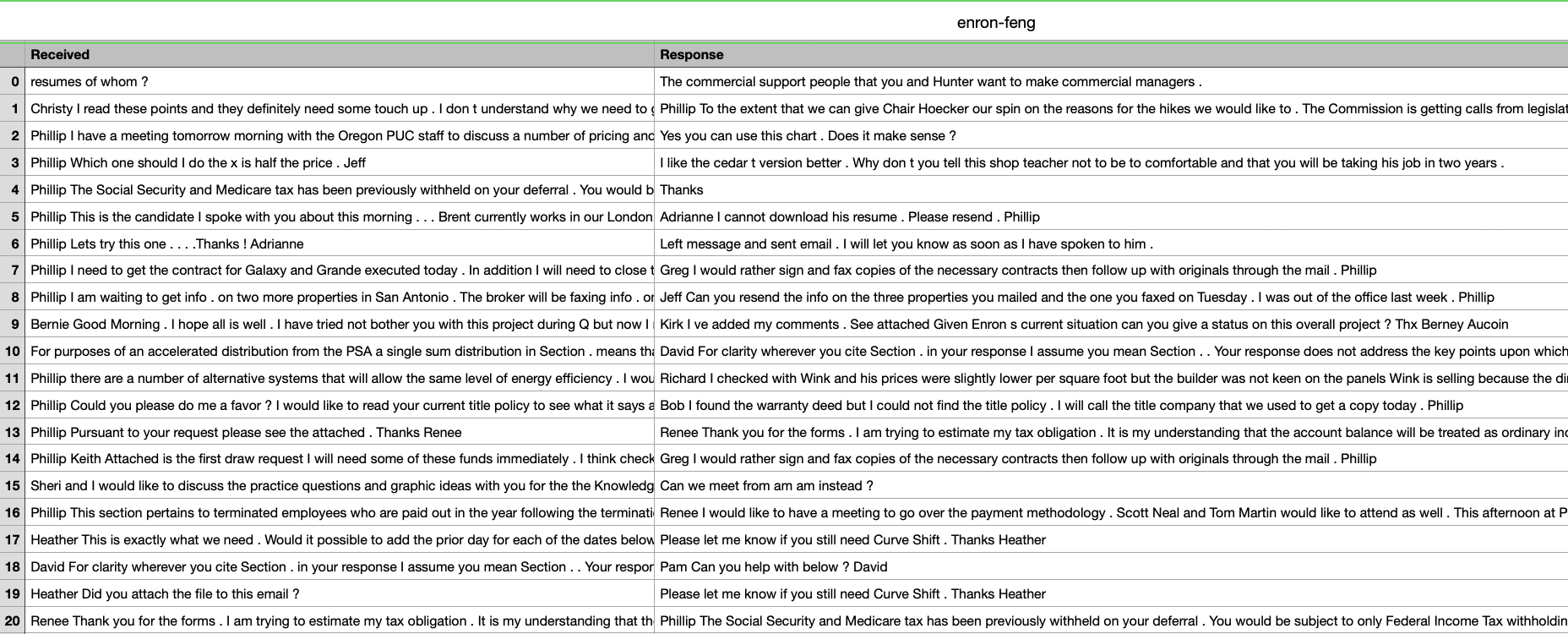


Figure 3.1: A sample of the rows and columns of the pre-processed Enron dataset

### Description of the Contrived Email Dataset

The contrived email dataset consists of two columns, the “received” column and the “response” column, just like the pre-processed Enron email dataset. The dataset was created by carrying out a many-to-one mapping between 6 classes of query emails and one specific response for each class. It should be noted that each class contains multiple emails that are essentially asking the same thing. The classes were sampled from emails typically sent in a university setting. Descriptions of the classes are shown in Table 4.8.

Table 3.1: Descriptions of the classes of the Contrived Dataset

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S/N** | **CLASS** | **DESCRIPTION** | **SAMPLE EMAIL** | **SAMPLE**  **RESPONSE** |
| 1. | A | This category contains emails asking a lecturer for suggestions on the list of textbooks required for a particular course. | “Dear Sir/Ma, can you suggest the essential textbooks for the course 'CSC121: Introduction to Computer Science'? Best regards.” | “Certainly! The essential textbooks for the course are [list of textbooks]. Make sure to have these textbooks to enhance your understanding of the subject. If you have any further questions, feel free to ask. Best regards.” |
| 2. | B | This category contains emails regarding student issues and can be written to the dean of student affairs. | “Good morning, sir/ma! I'm seeking information on resources or workshops available for job interview preparation. Thank you.” | “Thank you for your email. You can visit the Student Council office for information concerning this. They will attend to you. If you have any further queries, feel free to ask.” |
| 3. | C | This category contains various emails requesting exeat for various purposes. | "Dear Sir/Ma, I hope this email finds you well. I am writing to request an exeat for 23-07-2023 as I have been invited to participate in a prestigious academic competition. This opportunity holds great value for my educational development, and I kindly request your permission to represent our institution. Thank you for considering my request. Sincerely." | "Thank you for your exeat request. We acknowledge your commitment to academics and recognize the significance of personal obligations. Your request is approved, and the residency administrator will be informed. Kindly visit their office to fulfill the required procedures for your exeat. Best regards." |
| 4. | D |  | "Dear Sir/Ma, I am writing to inquire about the payment schedule for the upcoming semester's school fees. Could you please provide me with the necessary details? Thank you. Regards." | "Thank you for your email regarding school fees. Please visit the administrative office during working hours for assistance with your specific situation. Best regards." |
| 5. | E | This category contains emails regarding course registration and structure. | "Dear Sir/Ma, I encountered an error while registering for a course online. Could you please assist me in resolving this issue? Thank you. Regards." | "Thank you for your inquiry. Information about the registration process for the semester can be found on the student portal. Be sure to meet the provided deadlines to secure your desired courses. If you have any difficulties or specific cases, feel free to visit the departmental office for assistance. Best regards." |
| 6. | F | This category contains emails that are unrelated to university activities (out of the domain). | "Dear Customer Support, I recently purchased a product online, but it arrived damaged. Can you please guide me through the return and refund process? Thank you. Regards." | "Hello, I am AR (Auto-Responder) and I assist the owner of this email. I am unable to understand your email. Please note that I can only reply to some kind of emails for now. Please resend your email or wait for the owner to come around. Cheers." |

Based on the 6 classes, a primary dataset of 121 rows was obtained. Data augmentation techniques were then utilised to generate a dataset of about 9500+ rows for retraining the GPT2 model. Although the best ROUGE scores were obtained from the GPT2 model that was retrained/fine-tuned with this augmented data, the data augmentation techniques did not provide extremely high-quality text. Hence, there is the need for a larger and high-quality primary dataset.

It should also be noted that the augmented dataset was created with the consistency of the length of strings (email or response) in mind. Hence, a consistent length of elements in the “email” and “response” column was a key consideration in creating the dataset. This is in contrast with the pre-processed Enron email dataset which had column elements ranging from one word to multiple sentences. Figure 3.2 shows some rows of the created dataset.

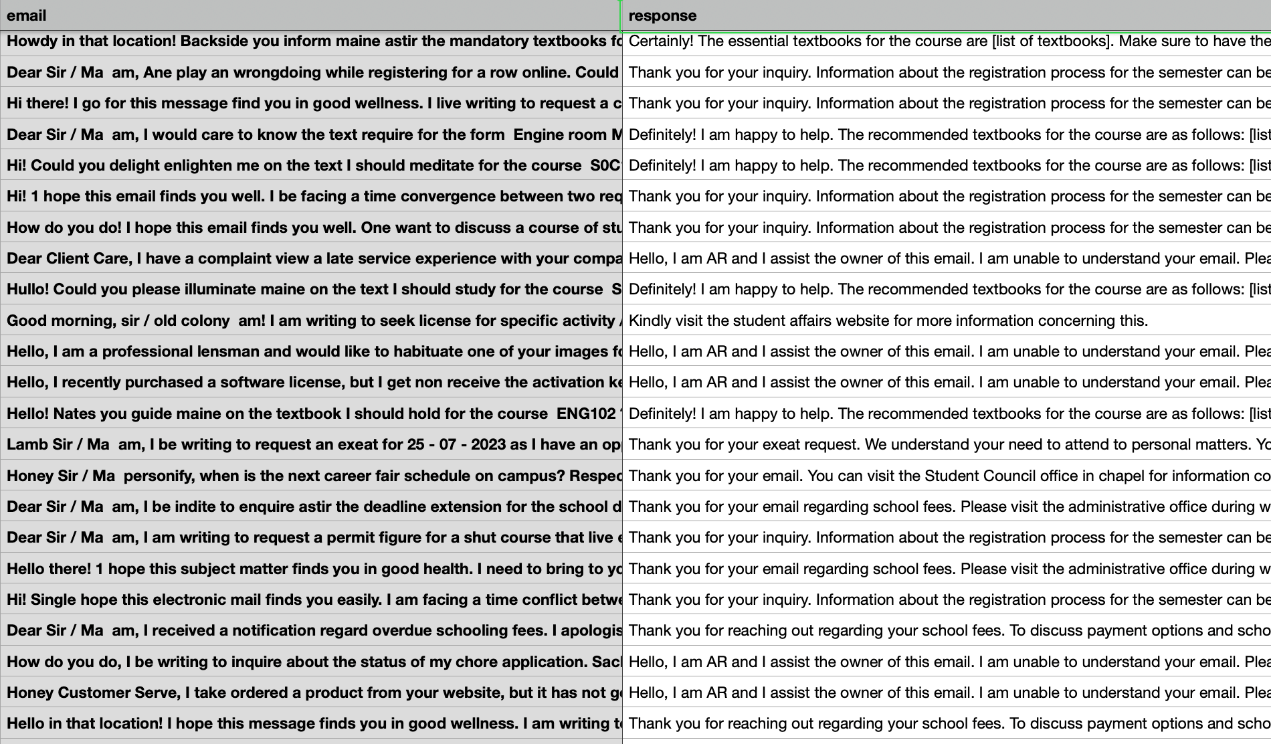


Figure 3.2: A sample of the rows and columns of the contrived dataset

## Physical Design

In physical design, the major design decisions include defining the input and output mechanisms of the software. It has two major categories namely the input design and the output design. For the intelligent auto-response system, it is concerned with how information is passed in and passed out of the system. Both input and output design are essential to ensure that the users or other systems can interact with the proposed system effectively.

### GPT Model Architecture

The Generative Pretrained Transformer (GPT) architecture is based on the Transformer architecture by Vaswani *et al.* (2017). The architecture consists of only the decoder part of the transformer architecture. The original GPT architecture forms the base for GPT2, GPT3 and GPT4. The version of GPT used in this study is GPT2. Figure 3.3 shows the architecture of the GPT model.

The components of the GPT decoder-only architecture include:

1. Token and Positional Embeddings: These are used to capture both the semantic meaning and positional information of words in the input text sequence. Token embeddings help to represent the meaning of individual words or sub-words, enabling the model to understand the contextual semantics of the text. On the other hand, positional embeddings help to encode the relative position or order of tokens in the sequence, helping the model capture the sequential nature and long-range dependencies in the sequence.
2. Feed-Forward Neural Network: This a component of the decoder block of the architecture. It is simply a neural network in which the links or connections between the nodes of the network do not form a cycle. This helps the GPT architecture to process the information from the self-attention mechanism.
3. Masked Self-Attention: This ensures that the GPT model attends only to the preceding words when generating each word, rather than peeking ahead to the future words that are yet to be generated. This encourages the model to learn to predict the masked or missing words based solely on the context provided by the preceding words. This is also the other major component of the decoder block.
4. Decoder Block: This is the unit component of the decoder-only architecture of the GPT model which is made up of the Feed-Forward Neural Network and the Masked Self-attention mechanism.

A picture containing text, screenshot, line, parallel

Description automatically generated

Figure 3.3: GPT Architecture  
Source: https://cameronrwolfe.substack.com/p/understanding-the-open-pre-trained-transformers-opt-library-193a29c14a15

The GPT model differs from the original transformer model in the following ways:

1. Masked Language Modelling: In the original Transformer, both the encoder and decoder have access to the entire input sequence during training. However, in GPT the model uses a masked language modelling (MLM) objective during pre-training. This involves randomly masking certain tokens in the input sequence and training the model to predict the masked tokens based on the context. This helps the model learn to generate coherent and contextually appropriate responses.
2. Decoder-Only Architecture: Unlike the original Transformer, which uses both an encoder and a decoder, GPT adopts a decoder-only architecture. This means that it focuses on autoregressive language generation tasks and does not have a separate encoder component. The decoder attends to all previous positions in the input sequence, allowing it to generate text in a sequential manner.

### The Proposed System Architecture

System architecture refers to the abstract representation that depicts the arrangement and functioning of multiple components and subsystems within a system. It aids in the representation of the organization of a system and offers an understanding of how it acts or behaves. It also focuses on the high-level representation of the components of a software system.

The proposed system architecture for the intelligent auto-response system involves a number of components including the user interface, the external email server, the email transfer component (sending and retrieval), the database, the engine and the response generator (the GPT2 model). It is shown in Figure 3.4. The functions of the components of the architecture are as follows:

1. User Interface: A user can interact with the system using the user interface to read received emails from and send replies to the email server.
2. Email Transfer Mechanism: This handles the process of sending and and retrieval of emails.
3. The Engine: The engine oversees processing emails for sending, retrieval or response generation.
4. Response Generator: This is the intelligent part of the system which utilises transfer learning with GPT2 to generate intelligent responses to received emails.
5. Database: The purpose of this is to store a small number of recent emails for an email user for the display on the user interface.
6. Email Server: This refers to the external email server from which emails are retrieved and are sent to.

A diagram of email processing

Description automatically generated with medium confidence

Figure 3.4: Architecture of the Proposed System

In this architecture, a user can interact with the system using the user interface to read received emails from and send emails to the email server. The email transfer component handles this process of sending and retrieval of emails. The engine is in charge of processing emails for sending, retrieval or response generation. The responses generator is the intelligent part of the system which utilises transfer learning with GPT2 to generate intelligent responses to received emails. A database is also available to store a small amount of recent emails for an email user.

Figure 3.2 shows a representation of the system architecture for the proposed intelligent auto-response system.

## Logical Design

In software engineering, logical design refers to the stage of the software development life cycle in which high-level design choices are made to define the system's organization functionality and behaviour. For the proposed system, the goal of logical design is to provide an abstract representation of the software system that is independent of specific implementation characteristics.

### Use Case Diagram

A use-case diagram is a category of UML diagram employed to identify and graphically illustrate the involved users within different use-case scenarios. In the case of the proposed intelligent auto-response system, the use-case diagram will help to represent the possible interactions between users of the system and the system. The use-case diagram for the proposed auto-response system is shown in Figure 3.5.

The use-case diagram shows that the user can login to the system, send emails, receive emails, view those emails in the inbox and generate intelligent responses, which is the core of the system.

A picture containing line, diagram, screenshot, parallel

Description automatically generated

Figure 3.5: Use case diagram for the proposed system.

### Activity Diagram

This is an advanced flowchart which shows the flow of activities or actions in a system. It can be described as a visual representation of the workflow inside a given system, and this means it depicts the more dynamic parts of a system. This depiction is done by displaying the arrangement and interdependence of actions, as well as decision points and branching routes.

Figure 3.6 shows the activity diagram for the intelligent auto-response system.

A picture containing text, diagram, sketch, drawing

Description automatically generated

Figure 3.6: Activity Diagram for the proposed system

### Sequence Diagram

A sequence diagram is a type of UML diagram that depicts interactions between users and the proposed system as well as between the parts of the system. For the intelligent auto-response system, the diagram shows how the users interact with the system and how the system interacts with the email server. Figure 3.7 shows the sequence diagram for the proposed system.

A picture containing text, diagram, receipt, parallel

Description automatically generated

Figure 3.7: Sequence Diagram for the Proposed System

## Conceptual Design

This section shows the Entity Relationship (ER) diagram of the proposed system as well as the database tables. The Entity Relationship diagram is shown in Figure 3.8.

**A picture containing text, screenshot, font, diagram

Description automatically generated**

Figure 3.8: Entity Relationship Diagram for the Proposed System

The normalised table representation are as follows:

1. Table: User

This table stores user information, including their email, password, and preference for email response method.

Normalized Representation: User(id, email, password, preference) with id as the primary key.

1. Table: Email

This table represents individual emails and contains details such as the server ID, subject, content, sender, thread ID, date, preview, and unread status.

Normalized Representation: Email(id, server\_id, subject, content, sender, thread\_id, date, unread)

Here, id is the primary key and thread\_id is a foreign key to the EmailThread table.

1. Table: Recipient

This table maintains the recipients of each email. It associates each recipient with the corresponding email ID.

Normalized Representation: Recipient(id, email\_id, recipient) with id as the primary key and email\_id as the foreign key to the email table.

1. Table: EmailThread

This table represents email threads, which are groups of related emails. It helps organize and group emails together based on their thread ID.

Normalized Representation: EmailThread(id)

CHAPTER FOUR

# SYSTEM IMPLEMENTATION AND EVALUATION

## PREAMBLE

This section details the implementation of the proposed intelligent auto-response system. It presents the hardware and software requirements required to make use of the system, the development methodology, program modules and interfaces and the evaluation of the system.

## System Requirements

This refers to the required standards that the system must have to perform as desired. It includes both the software and hardware requirements.

### Hardware Requirements

These requirements are the necessary hardware specifications for running the software system. It also includes the minimum required hardware resources to train the model used in the system. The hardware requirements for the project are detailed in Table 4.1.

Table 4.1: Hardware Requirements Table

|  |  |  |
| --- | --- | --- |
| **S/N** | **REQUIREMENT** | **HARDWARE** |
| 1. | Processor | Intel core i5 or higher/ Apple M1 chip |
| 2. | Primary Memory | 16 GB RAM or higher |
| 3. | Architecture | 64Bit (X64) for windows / ARM X64 |
| 4. | Secondary Storage | 32GB HDD or higher |

### Software Requirements

These requirements are necessary for running the software system successfully. They must be pre-installed and set for the intelligent auto-response system to run smoothly. Table 4.2 details the software requirements.

Table 4.2: Software Requirements Table

|  |  |  |
| --- | --- | --- |
| **S/N** | **REQUIREMENT** | **SOFTWARE** |
| 1. | Operating System | **Mac OS:** v11.0.0 or higher  **Windows:**10 or higher  **Linux:** Ubuntu |
| 2. | Programming Language | Python3, JavaScript |
| 3. | Database Management System | PostgreSQL |
| 4. | Development Tool | Visual Studio Code, Jupyter Notebooks |
| 5. | Supported browser | Google Chrome, Safari |

## Implementation Tools

These are the tools utilised while building the software project. They helped to ensure the successful development of the intelligent email auto-response project.

### Python

Python is the core programming language utilised for this project. It is an object-oriented general purpose programming language with strong applications in the field of Data Science and Machine Learning as well as Web Development. Python was used to build the backend and the handle the training and fine-tuning of the GPT2 language model. As a result of its wide acceptance and large community support, Python has several useful libraries and frameworks. Some of the Python libraries utilised in this project include Pandas, Celery, NLTK, Spacy, Flask (a web development framework) and SQLAlchemy (for database management).

### JavaScript

JavaScript is a powerful scripting programming language that serves as one of the core technological tools of the World Wide Web, alongside HTML and CSS. Using JavaScript allows the easy manipulation of HTML and CSS giving a more dynamic approach to web applications. A JavaScript library, React, was used to build the frontend of the auto-response system.

### Jupyter Notebooks

The Jupyter Notebook application is an application that allows the editing of a notebook via a web browser. Jupyter notebooks are useful in that they allow the easy sharing of program codes and other needed coding information. Jupyter notebook was used in this project to train the GPT2 language model.

### Visual Studio Code

Visual Studio Code is an open-source code editor that provides a range of useful plugins and functionalities when building software. It was utilised in this project to build and integrate the backend and the frontend.

## Development Methodology

The software development methodology employed in this study is the Iterative Waterfall methodology. The iterative waterfall methodology combines features of the Waterfall model and the Agile model. This means that sequential processes common to the Waterfall model such as Requirements Gathering, Analysis, Design and so on still take place in a sequential manner, with each process starting only when the preceding one has ended. However, agile features come into this methodology or model by allowing ease of going back to previous phases to implement changes as needed.

This study involved initially following the waterfall model but over time, there had to be changes made in the previous stages like requirements gathering, data collection etc. Several changes had to be made while training the model as the process of fine-tuning the GPT2 model required a lot of experimentation and testing. Therefore, the iterative waterfall model was well suited for this project.

Some of the common processes under iterative waterfall methodology include Requirements Elicitation, Design, Implementation, and Integration.

## Evaluation Of The System

The metrics to be used for evaluating the intelligent email auto-response system are based on the responses generated by the response model. They include the BLEU score and the ROUGE score.

As a result of structural inconsistencies discovered while using the pre-processed Enron email dataset during this study, a contrived dataset was created to test the effect of proper dataset structure on the performance of the fine-tuned GPT2 model.

The contrived dataset has a limited scope of emails sent in a university environment. It consists of 6 categories of questions with each category representing one type of question asked in many ways and mapped to one or two responses. This means the model gets exposed to many different patterns of asking the same thing and determines what response to give based on its learning.

### Evaluation Criteria

There were two major evaluation criteria adapted in the evaluation of the system. They include the Bi-Lingual Evaluation Understudy Score (BLEU) Score and the Recall-Oriented Understudy for Gisting Evaluation (ROUGE) Scores.

The BLEU score is a standard metric for measuring the quality of text generated by a machine or computer. It ranges between zero and one and measures the similarity between the text generated by the model and actual references that have been deemed to be high quality.

The idea behind the ROUGE scores is to assess how well the generated text captures important information present in the reference text.

### Evaluation Results

The results of the evaluation using the BLEU and ROUGE metrics are detailed in this section.

1. Bi-Lingual Evaluation Understudy Score (BLEU Score)

For this study, the BLEU score was calculated for 4 instances of the fine-tuned GPT2 model. The instances and corresponding BLEU score are displayed in the table. Table 4.3 shows the BLEU scores obtained from evaluating different instances of GPT2.

Table 4.3: BLEU Scores of GPT2 Instances

|  |  |  |
| --- | --- | --- |
| **S/N** | **INSTANCE** | **BLEU SCORE** |
| 1. | GPT2 fine-tuned with Enron emails tested against unseen Enron emails | ~ 0.66 |
| 2. | GPT2 fine-tuned with contrived emails tested against unseen contrived emails | ~ 0.60 |
| 3. | Base GPT2 tested against unseen Enron emails | ~ 0.54 |
| 4. | Base GPT2 tested against unseen contrived emails | ~ 0.53 |

1. Recall-Oriented Understudy for Gisting Evaluation (ROUGE) Score

Three types of ROUGE scores namely the ROUGE-1, ROUGE-2 and ROUGE-L are calculated for the four instances of the GPT2 model. Each metric has three scores: 'r' (recall), 'p' (precision), and 'f' (F1 score).

ROUGE-1 measures the measures the overlap of unigram (single word) sequences between the generated text and the reference text. ROUGE-2 measures the overlap of bigram (two-word) sequences while ROUGE-L measures the longest common subsequence (LCS) between the generated and reference summaries, considering both unigrams and longer sequence matches. The ROUGE scores for the four instances of the model are shown in the Tables 4.4, 4.5, 4.6 and 4.7.

Table 4.4: ROUGE scores for GPT2 re-trained and tested with Enron emails

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S/N** | **ROUGE VARIANT** | **RECALL** | **PRECISION** | **F1-SCORE** |
| 1. | ROUGE-1 | 0.1278 | 0.120 | 0.103 |
| 2. | ROUGE-2 | 0.009 | 0.009 | 0.008 |
| 3. | ROUGE-3 | 0.091 | 0.090 | 0.074 |

Table 4.5: ROUGE scores for GPT2 re-trained and tested with contrived emails

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S/N** | **ROUGE VARIANT** | **RECALL** | **PRECISION** | **F1-SCORE** |
| 1. | ROUGE-1 | 0.625 | 0.555 | 0.580 |
| 2. | ROUGE-2 | 0.437 | 0.422 | 0.425 |
| 3. | ROUGE-3 | 0.612 | 0.545 | 0.569 |

Table 4.6: ROUGE scores for Base GPT2 tested against sample Enron emails

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S/N** | **ROUGE VARIANT** | **RECALL** | **PRECISION** | **F1-SCORE** |
| 1. | ROUGE-1 | 0.212 | 0.035 | 0.057 |
| 2. | ROUGE-2 | 0.002 | 0.0003 | 0.006 |
| 3. | ROUGE-3 | 0.163 | 0.025 | 0.041 |

Table 4.7: ROUGE scores for Base GPT2 tested against sample contrived emails

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S/N** | **ROUGE VARIANT** | **RECALL** | **PRECISION** | **F1-SCORE** |
| 1. | ROUGE-1 | 0.127 | 0.079 | 0.192 |
| 2. | ROUGE-2 | 0.004 | 0.004 | 0.004 |
| 3. | ROUGE-3 | 0.108 | 0.070 | 0.080 |

A plot of precision values against recall values of the ROUGE instances of 2 models is shown in Figure 4.1 The first model with lower values is GPT2 re-trained and tested with Enron emails, while the second with higher values is GPT2 re-trained and tested with contrived emails.

A graph with a line and orange dot

Description automatically generated

Figure 4.1: Precision-Recall Plot of the GPT2 models fine-tuned with the Enron and Contrived datasets

### Discussion

From the BLEU scores, it can be deduced that the GPT2 model fine-tuned with Enron emails performed the best against unseen data. While the BLEU score is a good metric, the ROUGE score can also provide further insight into the quality of text generated by the model.

Based on Tables 4.4, 4.5, 4.6 and 4.7, it can be deduced that GPT2 re-trained with the contrived dataset seemed to have the best performance out of all the instances, especially better than the GPT2 instance re-trained with Enron emails. This further supports the idea that structured unlabelled data will tend to provide better performance when used for fine-tuning instead of unstructured and generalised unlabelled data.

Figure 4.2 shows some responses generated by the GPT2 model fine-tuned with the contrived dataset when tested with an unseen portion of the contrived dataset.

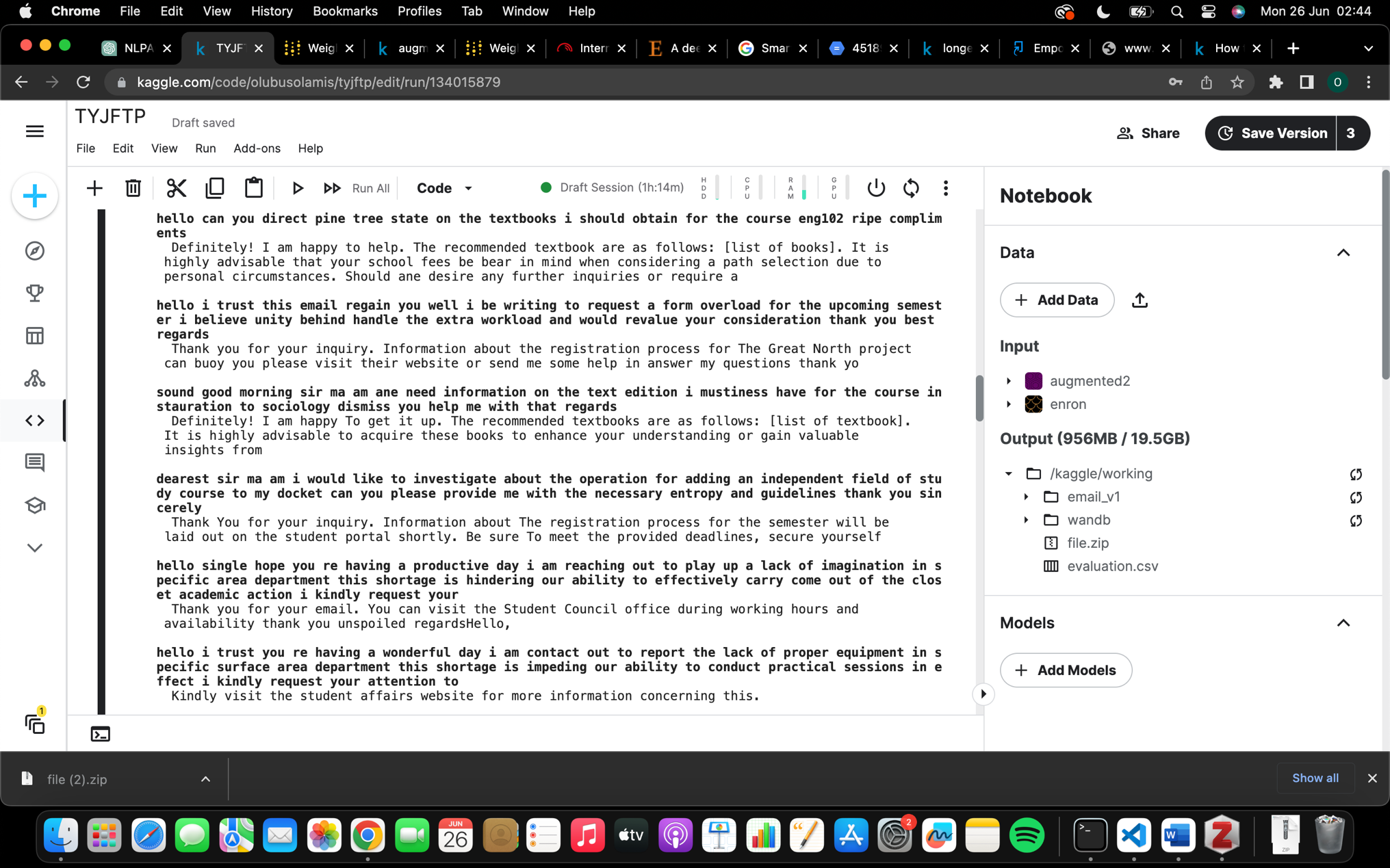


Figure 4.2: Responses generated by the GPT2 model fine-tuned with the contrived dataset

It should be noted that the data augmentation techniques used to create the contrived dataset is a major factor affecting the quality of the dataset as some sentences are not as high quality as they should be.

## Program Modules And Interfaces

This section details the sections of the user interface and modules of the intelligent automatic response system.

### The Login Module

This page is where the user gets to sign into the system using their Outlook email and password. Figure 4.3 shows the login module.

A screenshot of a computer

Description automatically generated

Figure 4.3: Sign in module/page

### The Inbox Module

This is where a user gets to see the emails in their inbox and interact with either read or unread emails. The inbox module is placed on the side navigation bar alongside settings and logout. Figure 4.4 shows the inbox module.

A screenshot of a computer

Description automatically generated

Figure 4.4: Inbox module/page

### The Settings Module

This is where the user can select between either enable automatic responses or disable it based on their preference. This is represented in Figure 4.5.

A screenshot of a computer

Description automatically generated

Figure 4.5: Settings Page

CHAPTER FIVE

# SUMMARY, RECOMMENDATIONS AND CONCLUSION

## Summary

The Electronic Mail (Email), as a communication technology, is one of the products of the rapid advancement and development of technology and technological tools. Email is an asynchronous means of communication that is largely used in the corporate world as well as for personal uses.

The introduction of the Email brought about both good and bad outcomes. A negative effect is the problem of Email Overload. Common approaches that have been employed to mitigate email overload include spam filtering, automatic response generation and AI-assisted writing. The automatic response generation approach which is the focus of this study can be implemented using various techniques such as case-based reasoning, neural networks, and pattern matching.

A technique called transfer learning was used in this study. Transfer learning is simply the process of re-using a pre-trained model on a relatively new but related task or domain. This means that the pre-trained model is re-trained with data specific to the new task or domain. A key benefit of transfer learning is that it requires less data, and the model may converge faster.

In the context of this study, Generative Pre-trained Transformer 2 (GPT2) was used. The GPT2 model was fine-tuned using 2 datasets; a pre-processed version of the Enron email dataset with 2 columns called “received” and “response” and a contrived email dataset with 2 columns namely “email” and “response”. The former had about 19000+ rows of data while the latter had about 9500+ rows.

During evaluation, it was observed that GPT 2 models fine-tuned with the Enron dataset and contrived dataset had very close BLEU scores with the Enron model performing better with a score of 0.66 and the contrived dataset model having a score of 0.60. However, using the ROUGE score metric, the GPT2 model fine-tuned with the contrived dataset outperformed the GPT2 model fine-tuned with the Enron dataset.

These results showed that certain things considered while creating the contrived dataset such as the length and consistency of column elements, had a great influence on the performance of the GPT2 model, as shown by the ROUGE scores.

## Recommendations

The following are recommendations for the intelligent auto-response system:

1. Results from this study show that using a larger, higher-quality, and more structured dataset to fine-tune GPT2 can result in a high level of accuracy for response generation in various domains.
2. The system can be improved to incorporate intelligent email writing or composition using GPT leading to a more robust email client.
3. A more powerful Generative Pre-trained Transformer (GPT) such as GPT3.5 or GPT4 can also be utilised to get better results.

## Conclusion

Through the creation and implementation of an intelligent automatic response system, this final year project successfully tackles the dilemma of information overload in email communication. The built system displays great capabilities in creating intelligent and contextually relevant answers by leveraging the capability of fine-tuned GPT-2, together with a well-chosen pre-processed email dataset and a custom contrived dataset.

Throughout the project, extensive research and understanding of AI and NLP techniques were applied to fine-tune GPT-2 for the specific task of email response generation. The created system improves on existing systems by generating longer sequences of responses that are context relevant. It also provides an easy way for users to set up the auto-response mechanism without having to combine multiple features together.

For future applications, the system's practical benefits can improve education, healthcare, business, and ordinary communication, therefore the project's contributions go beyond academic limits. The created email response system is also a valuable addition to the field of natural language processing and communication technologies.

REFERENCES

Arsovski, S., Oladele, M., Cheok, A., Premcevski, V., & Markoski, B. (2022). An approach to email categorization and response generation. *Computer Science and Information Systems*, *19*(2), 913–934. https://doi.org/10.2298/CSIS211101009A

Bala, M., & Verma, D. (2018). A critical review of digital marketing. *M. Bala, D. Verma (2018). A Critical Review of Digital Marketing. International Journal of Management, IT & Engineering*, *8*(10), 321–339.

Bentaiba-Lagrid, M. B., Bouzar-Benlabiod, L., Rubin, S. H., Bouabana-Tebibel, T., & Hanini, M. R. (2020). A case-based reasoning system for supervised classification problems in the medical field. *Expert Systems with Applications*, *150*, 113335. https://doi.org/10.1016/j.eswa.2020.113335

Bongomin, O., Gilibrays Ocen, G., Oyondi Nganyi, E., Musinguzi, A., & Omara, T. (2020). Exponential Disruptive Technologies and the Required Skills of Industry 4.0. *Journal of Engineering*, *2020*, 1–17. https://doi.org/10.1155/2020/4280156

Chandrasekaran, D., & Mago, V. (2022). Evolution of Semantic Similarity—A Survey. *ACM Computing Surveys*, *54*(2), 1–37. https://doi.org/10.1145/3440755

Chowdhary, K. R. (2020). *Fundamentals of Artificial Intelligence*. Springer India. https://doi.org/10.1007/978-81-322-3972-7

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv Preprint ArXiv:1810.04805*.

Fariborzi, E., & Zahedifard, M. (2012). E-mail marketing: Advantages, disadvantages andimproving techniques. *International Journal of E-Education, e-Business, e-Management and e-Learning*, *2*(3), 232.

Feng, Y., Naeem, M. A., Mirza, F., & Tahir, A. (2020). Reply Using Past Replies—A Deep Learning-Based E-Mail Client. *Electronics*, *9*(9), 1353. https://doi.org/10.3390/electronics9091353

Gatt, A., & Krahmer, E. (2017). *Survey of the State of the Art in Natural Language Generation: Core tasks, applications and evaluation*. https://doi.org/10.48550/ARXIV.1703.09902

Gibbs, S. (2016, March 7). How did email grow from messages between academics to a global epidemic? *The Guardian*. https://www.theguardian.com/technology/2016/mar/07/email-ray-tomlinson-history

Hemp, P. (2009, September 1). Death by Information Overload. *Harvard Business Review*. https://hbr.org/2009/09/death-by-information-overload

Hernández-Nieves, E., Hernández, G., Gil-González, A. B., Rodríguez-González, S., & Corchado, J. M. (2021). CEBRA: A Case-Based Reasoning Application to recommend banking products. *Engineering Applications of Artificial Intelligence*, *104*, 104327. https://doi.org/10.1016/j.engappai.2021.104327

Itakura, K., Kenmotsu, M., Oka, H., & Akiyoshi, M. (2010). An Identification Method of Inquiry E-mails to the Matching FAQ for Automatic Question Answering. In A. P. de Leon F. de Carvalho, S. Rodríguez-González, J. F. De Paz Santana, & J. M. C. Rodríguez (Eds.), *Distributed Computing and Artificial Intelligence* (Vol. 79, pp. 213–219). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-14883-5\_28

Kannan, A., Kurach, K., Ravi, S., Kaufmann, T., Tomkins, A., Miklos, B., Corrado, G., Lukacs, L., Ganea, M., Young, P., & Ramavajjala, V. (2016). Smart Reply: Automated Response Suggestion for Email. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 955–964. https://doi.org/10.1145/2939672.2939801

Lanctot, A., & Duxbury, L. (2021). When everything is urgent! Mail use and employee well-being. *Computers in Human Behavior Reports*, *4*, 100152. https://doi.org/10.1016/j.chbr.2021.100152

Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., & Gao, J. (2021). Deep learning—Based text classification: A comprehensive review. *ACM Computing Surveys (CSUR)*, *54*(3), 1–40.

Mittal, A., & Ramamurthy, A. (2016). Automatic Email Response System in E-learning. *Proceedings of the International Conference on Advances in Information Communication Technology & Computing - AICTC ’16*, 1–5. https://doi.org/10.1145/2979779.2979868

Mu, R., & Zeng, X. (2019). A review of deep learning research. *KSII Transactions on Internet and Information Systems (TIIS)*, *13*(4), 1738–1764.

Naeem, M. A., Linggawa, I. W. S., Mughal, A. A., Lutteroth, C., & Weber, G. (2018). A Smart Email Client Prototype for Effective Reuse of Past Replies. *IEEE Access*, *6*, 69453–69471. https://doi.org/10.1109/ACCESS.2018.2878523

Navigli, R. (2018). *Natural Language Understanding: Instructions for (Present and Future) Use.* *18*, 5697–5702.

Otter, D. W., Medina, J. R., & Kalita, J. K. (2021). A Survey of the Usages of Deep Learning for Natural Language Processing. *IEEE Transactions on Neural Networks and Learning Systems*, *32*(2), 604–624. https://doi.org/10.1109/TNNLS.2020.2979670

Partridge, C. (2008). The Technical Development of Internet Email. *IEEE Annals of the History of Computing*, *30*(2), 3–29. https://doi.org/10.1109/MAHC.2008.32

Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). *Improving language understanding by generative pre-training*.

Radicati Team. (2022, November 14). *The Radicati Group, Inc. » New Announcements Reports » Email Statistics Report, 2022-2026*. https://www.radicati.com/?p=17936

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., & Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, *21*(1), 5485–5551.

Ray, S. (2019). A Quick Review of Machine Learning Algorithms. *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, 35–39. https://doi.org/10.1109/COMITCon.2019.8862451

Roetzel, P. G. (2019). Information overload in the information age: A review of the literature from business administration, business psychology, and related disciplines with a bibliometric approach and framework development. *Business Research*, *12*(2), 479–522. https://doi.org/10.1007/s40685-018-0069-z

Ruder, S., Peters, M. E., Swayamdipta, S., & Wolf, T. (2019). Transfer Learning in Natural Language Processing. *Proceedings of the 2019 Conference of the North*, 15–18. https://doi.org/10.18653/v1/N19-5004

Russell, S. J., & Norvig, P. (2020). *Artificial intelligence: A modern approach* (4th edition). Pearson.

Sanhz, J. (2022, May 3). *Gmail: How to Create and Add an Automatic Reply—Technipages*. Technipages. https://www.technipages.com/gmail-how-to-create-and-add-an-automatic-reply

Santos, H., Batista, J., & Marques, R. P. (2019). Digital transformation in higher education: The use of communication technologies by students. *Procedia Computer Science*, *164*, 123–130. https://doi.org/10.1016/j.procs.2019.12.163

Shinde, P. P., & Shah, S. (2018). A Review of Machine Learning and Deep Learning Applications. *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, 1–6. https://doi.org/10.1109/ICCUBEA.2018.8697857

Smyth, B. (2019). A Tale of Two Communities: An Analysis of Three Decades of Case-Based Reasoning Research. In K. Bach & C. Marling (Eds.), *Case-Based Reasoning Research and Development* (Vol. 11680, pp. 343–357). Springer International Publishing. https://doi.org/10.1007/978-3-030-29249-2\_23

Sneiders, E., Sjöbergh, J., & Alfalahi, A. (2016). Email Answering by Matching Question and Context-Specific Text Patterns: Performance and Error Analysis. In Á. Rocha, A. M. Correia, H. Adeli, L. P. Reis, & M. Mendonça Teixeira (Eds.), *New Advances in Information Systems and Technologies* (pp. 123–133). Springer International Publishing. https://doi.org/10.1007/978-3-319-31232-3\_12

Soucek, R., & Moser, K. (2010). Coping with information overload in email communication: Evaluation of a training intervention. *Computers in Human Behavior*, *26*(6), 1458–1466. https://doi.org/10.1016/j.chb.2010.04.024

Spira, J. B., & Burke, C. (2009). *Intel’s War on Information Overload: A Case Study*.

Statista. (2021, February). *Emails sent per day 2025 | Statista*. Statista. https://www.statista.com/statistics/456500/daily-number-of-e-mails-worldwide/

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). *Attention Is All You Need*. https://doi.org/10.48550/ARXIV.1706.03762

Watson, R. (1971). *Mail Box Protocol* (Request for Comments RFC 196). Internet Engineering Task Force. https://doi.org/10.17487/RFC0196

Whittaker, S., & Sidner, C. (1996). *Email overload: Exploring personal information management of email*. 276–283.

Young, T., Hazarika, D., Poria, S., & Cambria, E. (2018). Recent Trends in Deep Learning Based Natural Language Processing [Review Article]. *IEEE Computational Intelligence Magazine*, *13*(3), 55–75. https://doi.org/10.1109/MCI.2018.2840738

Yu, Y., Si, X., Hu, C., & Zhang, J. (2019). A review of recurrent neural networks: LSTM cells and network architectures. *Neural Computation*, *31*(7), 1235–1270.

Zhang, C., & Lu, Y. (2021). Study on artificial intelligence: The state of the art and future prospects. *Journal of Industrial Information Integration*, *23*, 100224. https://doi.org/10.1016/j.jii.2021.100224

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., & He, Q. (2021). A Comprehensive Survey on Transfer Learning. *Proceedings of the IEEE*, *109*(1), 43–76. https://doi.org/10.1109/JPROC.2020.3004555