A NOVEL E-MAIL REPLY APPROACH FOR E-MAIL MANAGEMENT SYSTEM

A thesis submitted to Auckland University of Technology
in partial fulfilment of the requirements for the degree of
Master of Computer and Information Sciences

Supervisors

Muhammad Asif Naeem

Farhaan Mirza

15.07.2019

By

Yiwei Feng

School of Engineering, Computer and Mathematical Sciences

Copyright

Declaration

I hereby declare that the research paper titled __A NOVEL E-MAIL REPLY APPROACH FOR E-MAIL MANAGEMENT SYSTEM submitted by me is based on actual and original work carried out be me. Any reference to work done by any other person or institution or any material obtained from other sources have been duly cited and referenced. I further certify that the research paper has not been published or submitted for publication anywhere else.

Signature of candidate

Acknowledgements

This research was supported by Auckland University of Technology. I express my great appreciation to Dr. Muhammad Asif Naeem and Dr. Farhaan Mirza, my research supervisors, for their valuable and constructive suggestions, enthusiastic encouragement and useful critiques of this research work.

I thank Mr. Bumjun Kim for his help in offering me the resources in running the program, and Dr. William Liu for advice and assistance that kept my progress on schedule.

I would also like to show my gratitude to the fabulous colleagues at Chainport Labs, Dr. Gewei Zhang, Zhantao Feng, Erik Wu, Jamie P. Smillie, Kyle Wong and Matt Grant, for their sharing of wisdom and knowledge which paved its way for my research from an idea into a solid implementation.

I am immensely grateful to Caleb Millen and Loong Chek Jen for their insights and comments on an earlier version of the manuscript that greatly assisted the research.

Finally, I have to extend my thanks to my family including my parents, classmates and friends, because they are always there for me. Without them, this research would not have been possible.

Abstract

This project describes a novel intelligent E-mail reply system through information retrieval and information generation techniques. There are several difficulties to realise different kinds of functions using machine learning and deep learning algorithms. For example, the publicly available raw training datasets cannot meet the functional requirements of the model, and the information generation class models cannot satisfy the long text-based predictions due to limitations of the algorithm. It is well known that the Term Frequency-Inverse Document Frequency (TF-IDF) model is one of the most widely used feature extraction methods in information retrieval because of its simple algorithm and excellent performance. Meanwhile, The Document to Vector (Doc2Vec) model is an extension algorithm of Word to Vector (Word2Vec), which can train the index of documents together based on turning words into vectors. Good results have been achieved in determining the relationship between words within a document, as well as the correlation between different documents. Recently, the Gated Recurrent Unit (GRU) model is playing an increasingly important role in natural language processing (NLP) as an advanced method of applying a recurrent neural network (RNN). Also, the GRU model utilises deep neural networks to predict and generate information instead of extracting the original existing information. Specifically, we use these three algorithms to train and implement our models after heavily processing our training data. Experimental

results show that a hybrid model combining the GRU information generation model as the base with the method of sentence to vector embedding (Sent2Vec) is a practicable method for long-text prediction. In the end, an intelligent E-mail reply system is implemented in our experiment. Three models are compared through subjective human evaluation.

Contents

Copyright **2**

Declaration **3**

Acknowledgements **4**

Abstract **5**

1 Introduction **11**
1.1 Research Motivation 12
1.2 Research Questions 13
1.3 Contributions 14
1.4 Thesis Organisation 15

2 Related Work **17**
2.1 Recognition of E-mail System Problem 18
2.1.1 Background of E-mail System Development 18
2.1.2 E-mail Overload 19
2.2 New E-mail Response Approaches 21
2.2.1 Predicting response behaviour 22
2.2.2 Reusing Previous Reply E-mails 24
2.2.3 Automatically generating E-mail Response 25
2.3 Related Techniques 28
2.3.1 Traditional Feature Extraction Methods 29
2.3.2 Word Embedding 31
2.3.3 Similarity Measurement 34
2.3.4 Deep Learning and Neural Network 35
2.4 Research Gap 38
2.5 Summary 39

3 Design and Methodology **41**
3.1 System Design 43
3.2 Model Design 44
3.2.1 TF-IDF Based Model 45
3.2.2 Doc2Vec Based Model 49
3.2.3 GRU-Sent2Vec Hybrid Model 55

3.3 Summary 61

4 Implementation **63**
4.1 Data Preparing 63
4.1.1 Data Collection 64
4.1.2 Data Pre-processing 65
4.2 TF-IDF Based Model 69
4.2.1 Data Processing for TF-IDF 69
4.2.2 TF-IDF Modelling 72
4.2.3 The Main Parameters of TF-IDF 73
4.3 Doc2Vec Based Model 74
4.3.1 Doc2Vec Based Modelling 75
4.3.2 The Main Parameters of Doc2Vec 75
4.4 GRU-Sent2Vec Hybrid Model 76
4.4.1 Data Processing for GRU-Sent2Vec Hybrid Model 77
4.4.2 Building Up Sent2Vec Model 79
4.4.3 Mapping Sentences and Preparing Embedding Layers . . . 80
4.4.4 Training Model 80
4.4.5 Model Implementation 81
4.5 Intelligent E-mail Client Implementation 82
4.6 Summary 84

5 Evaluation and Discussion **85**
5.1 Parameter Tuning 85
5.1.1 TF-IDF 86
5.1.2 Doc2Vec 87
5.2 Setup for Human Evaluation 89
5.2.1 Test Data Generation 89
5.2.2 Measures 90
5.3 Results and Discussion 91
5.3.1 Comparison of the Three Models 91
5.3.2 Comparison of the Two Information Retrieval Models . . . 95
5.4 Summary 98

6 Conclusion **99**
6.1 Conclusion 99
6.2 Challenges and Limitations 101
6.3 Future Work 102

References **104**

Appendices **110**

List of Tables

5.1 Comparison of TF-IDF Parameters 87
5.2 Comparison of Doc2Vec Parameters 88

List of Figures

Figure. 2.1 Gmail Smart Prediction Model 28
Figure. 2.2 One RNN Cell Unit 37
Figure. 2.3 One LSTM Cell Unit 37
Figure. 2.4 One GRU Cell Unit 38

Figure. 3.1 Intelligent E-mail System Overview 42
Figure. 3.2 Retrieval-based Model 46
Figure. 3.3 Generative-based Model 46
Figure. 3.4 PV-DM Mode 50
Figure. 3.5 PV-DBOW Mode 52
Figure. 3.6 Basic One GRU Cell Unit 56
Figure. 3.7 Sigmode Activation 60
Figure. 3.8 Tanh Activation 61

Figure. 4.1 Data Processing Flowchart 65
Figure. 4.2 Dataset Overview 66
Figure. 4.3 Raw E-mail Content 66
Figure. 4.4 E-mail Content Length Statistics 67
Figure. 4.5 Received-Response E-mail Pairs 68
Figure. 4.6 Processed E-mail Content Length Statistics 69
Figure. 4.7 TF-IDF Based Model 70
Figure. 4.8 Doc2Vec Based Model 74
Figure. 4.9 GRU-Sent2Vec Hybrid Model 78
Figure. 4.10 Intelligent E-mail Client 82
Figure. 4.11 Intelligent E-mail Client - Default Response Mode 83
Figure. 4.12 Intelligent E-mail client - Doc2Vec Mode 83

Figure. 5.1 Test Results of TF-IDF Model 87
Figure. 5.2 Test Results of Doc2Vec Model 88
Figure. 5.3 M3 VS M9 Results 89
Figure. 5.4 Three Models Result 92
Figure. 5.5 Human Evaluation Results 93
Figure. 5.6 Sentence Repeat Statistics 94
Figure. 5.7 Ideal GRU-Sent2Vec Hybrid Model 94
Figure. 5.8 Learning Ability of TF-IDF and Doc2Vec 97
Figure. 5.9 Effect Comparison Between TF-IDF and Doc2Vec 98

Chapter 1

Introduction

Currently, there is a strongly increasing trend of social media use, such as using social networking sites and instant messaging (WhatsApp and Facebook etc.), dominating our communications (Tsay-Vogel, Shanahan & Signorielli, 2018; Batra, Sidhu & Sharma, 2018). However, Electronic mail (E-mail) is still the most common form of online business correspondence and is still a growing and effective communication tool for most enterprises and individuals (Tsay-Vogel et al., 2018). Meanwhile, E-mail is also an integral part of related personal Internet experience (Coussement & Van den Poel, 2008). For example, E-mail accounts (or E-mail addresses) are almost always required for registering on website accounts, including social networking sites, instant messaging and any other types of Internet services. Therefore, E-mail has become fully integrated into our daily lives and business activities.

According to the Radicati Group's statistics and projections (2018), more than 281 billion E-mails are sent and received worldwide every day, and this number is expected to increase by 18.5 per cent over the next four years. In 2018, more than half of the world's population used E-mail, with more than 3.8 billion users (GROUP et al., 2018). Based on the above statement, it can be

determined that an average user sends and receives an average of 74 E-mails per day, which also reveals the problem of E-mail overload.

1.1 Research Motivation

The problem of E-mail overload has not been solved in nearly half a century, and it is continually becoming worse. Enterprises still maintain and manage customer resources using E-mails because of handling users' feedback and consultation (Coussement & Van den Poel, 2008). To be specific, some customer service centres of various organisations receive hundreds of thousands of E-mails from customers every day. Although the staff in the customer service centres have high-level training, there is striking similarity among huge E-mail data. They need to spend lots of time replying to these E-mails, which results in high labour cost during this process for enterprises. Every E-mail user also suffers from E-mail overload. The development of the information age brings various benefits to our daily life, however, alongside its convenience, attendant problems are equally persistent.

For E-mail systems, E-mail overload was proclaimed as a 'universal problem' (Whittaker & Sidner, 1996). As a formal means of communication, E-mail is 'central' (L. A. Dabbish & Kraut, 2006), 'ubiquitous' (Pazos, Chung & Micari, 2013) and 'indispensable' (Hair, Renaud & Ramsay, 2007). Whittaker and Sidner (1996) presented that E-mail users tend to leave an increasing volume of unread or non-replied messages every day. Most users have numerous E-mails that they do not ever read or get to reply to in time, which leads to E-mail management issues, predominantly a messy and overwhelming mailbox.

There is a lot of time and labour wasted due to repeated E-mail responses;

thus, a rapid and efficient method is highly needed. We believe that NLP techniques using machine learning and deep learning algorithms have an incredible role in solving the above issues.

The current research mostly focuses on developing an E-mail system with an intelligent response function. However, there are still enormous research gaps either in reusing old E-mails based on an information retrieval method or the research of predictive generation-based responses based on neural networks. Therefore, the chance to solve these issues and find solutions sparked my motivation for investigating a novel E-mail reply system.

1.2 Research Questions

The main research question (MRQ) is: how can one use the most advanced text processing, machine learning and deep learning technology to design and develop an intelligent E-mail management system that sequentially implements intelligent response solutions to improve E-mail response efficiency and reduce the E-mail burden? This MRQ can be further divided into three sub-questions:

SRQ1: How can one improve the quality and trainability of the existing dataset by removing noise and marking data, so as to provide a good foundation for subsequent model training?

SRQ2: How can one design an intelligent reply function of an E-mail system with practical use value, which methods and algorithms should be chosen and how should the structure of the models and the architecture of the system should be designed in the experiment?

SRQ3: How can one evaluate the model quality and implementation results to verify the effect of this research project?

1.3 Contributions

This study contributes to the improvement of the existing model and relevant practical application. The holistic analysis of this study added to existing research by identifying a group of three essential models that should be considered in reducing E-mail overload, compared to existing models that are less innovative. For example, although the TF-IDF model was applied in intelligent E-mail systems in 2017 (Linggawa, 2017), there exist some limitations such as not marking E-mail labels and too much noise in the datasets. Holistic analysis of the benefits of the Doc2Vec model has not been done before. Certainly, for the GRU-Sent2Vec hybrid model, it uses a combination of information generation and retrieval, which is an innovative model and is proposed for the first time. Therefore, the study confirmed some results of innovative research that also emphasised the contributions of three aspects.

1. Current popular methods are limited to short text prediction, at the word-level. Like Chatbots, it can predict the next sentence based on the last sentence. However, E-mails are long-text, and if the content of the reply can be predicted from the whole of the received E-mail, work efficiency of E-mail users will be greatly improved. In this study, Sent2Vec is introduced into GRU and a novel hybrid model is constructed to make predictions based at the sentence-level rather than word-level. However, it should be noted that the scenarios applied by the models in this research are not limited to long text prediction but are suited for short-text prediction scenarios, such as Chatbots used in automatic reply in a chat room.

2. Using tags to improve the training corpus. The E-mail dataset is processed

using a logical matching method, which matches sent and received E-mails and uses them as a reference for answering new E-mails. This study expands the sample diversity of the public E-mail dataset, and the processed dataset will be published on GitHub[1] for further research.

3. By creating user interfaces to implement core functions, our research is truly applicable for many enterprise customer service departments. This intelligent E-mail reply suggestion system allows users to choose an intelligent reply function or direct reply. The system provides an opportunity for users to review automatically generated replies before they are sent.

Overall, our research achieves the study purpose from model design to algorithm improvement to a functional reality.

1.4 Thesis Organisation

In order to achieve the aim of this research, this paper is structured as follows:

Chapter 2: Chapter 2 studies the background of the development of E-mail technology and determines the most serious problems existing in the current E-mail management system. By searching and studying other researchers' methods of intelligently replying to a new E-mail and technologies in related fields, we find the idea of designing a new smart E-mail management system and determine the technical direction to be applied in this experiment.

Chapter 3: Chapter 3 demonstrates the research design which includes system design and the design of our three models. This chapter

[1]https://github.com/fxyfeier

shows all the techniques and methods used in the research experiments as well as including cosine similarity, machine learning algorithms and deep learning algorithms.

Chapter 4: Based on the system and model design of the previous chapter, Chapter 4 discusses the implementation of the entire system, involving the collection and pre-processing of data, modelling and training of the three models, parameter optimisation, and E-mail system client design for visual presentations.

Chapter 5: Chapter 5 mainly describes the evaluation of the models and discusses the results. Firstly, we carry out self-evaluation of the first two models that are based on an information retrieval method and determine the final three models based on their optimal parameters. Through human evaluation, we subjectively evaluate the effects of the three models. Finally, we have an in-depth discussion of the results.

Chapter 6: Chapter 6 summarises the effective implementation process and significant results of this research experiment. The constraints and challenges in our research are discussed and future research directions are considered.

Chapter 2

Related Work

In this chapter, our work covers several research areas. To promote the excellent performance of our research result, we reviewed recent relevant studies on intelligent E-mail management system solutions, information retrieval methods, and applications on machine learning and deep learning algorithms. We sought to understand current problems and investigated related technologies and existing solutions so that the more optimal models and methods could be integrated and implemented.

This chapter is organised as follows: Section 2.1 identifies the most significant problem that current users still face with E-mail. Our goal is also to mitigate this problem. In Section 2.2, we explore various approaches to designing an E-mail management system with intelligent responsiveness, which can be grouped into three categories. Section 2.3 describes novel technologies related to our research and their corresponding application areas and achievements. Section 2.4 illustrates the research gaps in this field by analysing the results of other researchers. Finally, in section 2.5, we summarise all the techniques of design and implementation in the relevant literature.

2.1 Recognition of E-mail System Problem

E-mail has been around for nearly fifty years, during which time technology has changed dramatically. It evolved from a simple communication system to support various management functions, including task management, personal archiving (Whittaker & Sidner, 1996), time management (Bellotti, Ducheneaut, Howard, Smith & Grinter, 2005), task coordination (Martin, Van Durme, Raulas & Merisavo, 2003) and information management (Whittaker, Bellotti & Gwizdka, 2006).

2.1.1 Background of E-mail System Development

According to Tomlinson (2009), E-mail was born in the fall of 1971, when there were two types of computer programs that could transmit files and raw information. However, both had significant usage limitations. For example, a person using a messaging program could only send notifications to recipients whose computers were matched with the senders. Ray Tomlinson studied these computer programs and developed a new one that could send and receive information over the Internet.

Although E-mail was invented in the 1970s (Tomlinson, 2009), it did not flourish until the 1980s. E-mail was not widely adopted in the 1970s, mainly because few people used ARPANET, a fundamental network system used before the Internet as we know it now. ARPANET was created in the late 1960s, and the network was slow. Limited by the network speed, users could only send very short messages, and could not send as much data as they do now. By the mid-1980s, with the rise of personal computers, E-mail began to be widely used among computer enthusiasts and college students. By the mid-1990s, thanks to the birth of the Internet browser, the number of Internet users around the world

had surged, and E-mail began to boom (Partridge, 2008).

E-mail increased more than sixfold from 1995 to 2001, according to a 2001 survey by Rogen International (Kirkgöz, 2010). In addition, it took about two hours a day for users to receive, check, prepare, and send the required E-mails. The rise of E-mail as a form of communication has also attracted more people to develop and improve its functions. With many unique features, it has become more user-friendly.

Thomas et al. (2006) compared E-mail with five forms of workplace media (face-to-face conversations, telephone, voice mail, postal mail, and faxes) and found that E-mail has four particular characteristics: it is asynchronous, text-based, multiple-recipient addressable and has built-in memory that allows messages to be stored, retrieved and forwarded.

The expansion and optimisation of these functions provided more convenience and benefits to users and contributed to the prosperity of E-mail. However, there is a problem that researchers and developers have been working on for more than two decades: E-mail overload (Whittaker & Sidner, 1996).

2.1.2 E-mail Overload

Whittaker and Sidner (1996) introduced the term 'E-mail overload'. They analysed the chaotic use of E-mail and concluded that the overload of E-mail was caused by the lack of capacity of the E-mail system to support asynchronous communication. It is embodied in two aspects: First, how to manage whether historical E-mail is easy to retrieve; second, how to track the current E-mail session state, and then provide and display the current task information.

Over the next decade, the problem of E-mail overload has worsened, and many researchers have tried to analyse the causes of the problem and actively

sought solutions. Thomas et al. (2006) addressed E-mail-related social processes using three data-set sources with E-mail log and textual analysis. This paper revealed the reasons for the E-mail overload to be unstable request and response pressures, task delegation, and shifting interactants.

Dabbish and Kraut (2006) proposed an important quantitative examination of E-mail overload. A nested model was built for regression analysis using the standard least squares method, and the model features were identified by predicting the importance and quantity of E-mail overload.

Stross (2008) made an interesting description 'E-mail has become the bane of some people's careers'. Many 'knowledge workers', like office workers, have experienced a deluge of E-mails waiting to be answered in their inboxes. Commonly, some crucial E-mails were ignored among these E-mails because they were not replied to or even noticed. A wave of high-profile Internet companies focused on eliminating E-mail overload.

NBC News (Tahmincioglu, 2011) reported that more than 100 trillion E-mails were sent worldwide in 2010, about 294 billion sent every day, 16.7% more than the previous year (from a technology research company Radicati Group). Faced with constant E-mail flow, many users often failed to read their E-mails on the same day, leaving many unread or unresolved messages in their E-mail inboxes.

Szóstek (2011) used several methods related to E-mail organisation and retrieval and proposed some methods of E-mail management. He focused on reducing E-mail stress for latent users and believed that archiving, filtering, regularly checking and continuously monitoring could help employees effectively manage their work E-mails.

Unfortunately, solutions from these studies still do not fundamentally solve the problem of E-mail overload. The main reason for E-mail overload has always been difficulty in keeping up with the speed of receiving E-mails (Grevet, Choi,

Kumar & Gilbert, 2014). Dabbish and Kraut (2006) showed that workers could control E-mail overload by using software designed to make E-mail easier to use or by adopting effective strategies.

It has long been expected that developing software can significantly reduce the administrative burden of a large amount of E-mail (Stross, 2008). Stross hypothesised that this technical solution could help deal with users' public E-mail accounts by preparing automatic replies.

2.2 New E-mail Response Approaches

In the previous section, by reviewing the technological development of the E-mail system over the past half-century, we found that the problem that remained unresolved was E-mail overload. Although the researchers were provided with tons of related suggestions and solutions, there has been still no substantial improvement.

E-mail users, especially those working in customer-related areas such as customer service departments and help desks, use E-mails to answer customer questions (Coussement & Van den Poel, 2008). Most of the time, the customer service person has probably responded to a lot of similar inquiries, however, they may still need to spend much time searching past replies to provide a similar solution (Coussement & Van den Poel, 2008). Users are eager to adopt a software application that can automatically identify the content of E-mails and generate suggestions for replies, reducing the daily pressure of responding to a large number of E-mails (Linggawa, 2017).

Some researchers have come up with several techniques for automatic reply to try to ease this tedious and frustrating process. Three of the major auto-response E-mail solutions are predicting response behaviour, reusing Previous

Reply E-mails, and automatically generating E-mail Response.

2.2.1 Predicting response behaviour

If an inbox is filled with a large collection of unread E-mails, users may have a hard time finding messages they need to read and respond to (Whittaker & Sidner, 1997; Bellotti, Ducheneaut, Howard & Smith, 2003).

Dabbish et al. (2005), used an organisational survey to analyse people's behaviour when they receive and respond to new E-mails. The survey was conducted by sending E-mails to 1,100 E-mail addresses of professors, faculty and students at Carnegie Mellon University. By collecting the participants' work environment (such as the nature of the job), the status of their E-mail use (including the frequency of E-mails sent and received), the habit of replying to E-mails and the details of behaviour they provided to handle five new non-spam E-mails with the category of the E-mail content, Dabbish et al. found that the use of E-mail reflects personal orientation, job requirements and interpersonal differences. This study provided a necessary reference for further research on predicting E-mail reply behaviour.

Dredze et al. (Dredze et al., 2008) developed a prototype of an intelligent E-mail system that could predict the response behaviour to an E-mail intelligently. It can be used to predict whether an incoming E-mail needs to be replied to and manages which E-mails need to be replied to in a time sensitive manner. At the same time, the alert system will also issue an alarm when an expected attachment is missing from the reply. In the whole research process, logistic regression was used as a classifier algorithm, and the sender-to-recipient relationship was extracted as the main feature for feature learning, while text features (such as the marking feature of the problem), keyword features, and word frequency-inverse

document frequency were also used in the training process. The prototype has been implemented as an extension of Mozilla Thunderbird E-mail client, which is an initial intelligent messaging system.

In a recent study, Yang et al. (2017) found various features that affect E-mail response behaviour, such as E-mail content, meta-data factors, and time series features. Considering binary dialogue and group discussions, they used the Avocado Research E-mail collection, a public dataset that is used to build models to predict recipients' response behaviours (response time and reply behaviour), as a training dataset. They conducted a detailed experiment at a technology company, and the results represented an understanding of E-mail response behaviour in an enterprise environment.

In many predictive response behaviour models, the primary method is to classify E-mails based on their subject or extracted content. Feature extraction is done by performing tag checking. First, the question keywords in the E-mail content, the question marks, the E-mail addresses, some particular keywords (such as attach, attachments, attached) in the E-mail are marked as special identifying information (Dredze et al., 2008). Then the word frequency from the training dataset is calculated. The next step is to generate a prediction of whether the received E-mail needs to be replied to, and then to mark each E-mail 'reply required' or 'no reply required'. The last step is to display it or provide a warning to the user (Ayodele & Zhou, 2009; L. Dabbish, Kraut, Fussell & Kiesler, 2004).

Most of these models adopt rule-based classification technology. However, the necessary condition of classification optimisation is a large amount of labelled training data, so it is an essential and challenging task to collect or process enough training datasets to construct a classifier knowledge base. Therefore, based on the limitations of the training corpus we can find, we will not use this

approach in our research.

2.2.2 Reusing Previous Reply E-mails

The second primary approach to designing an intelligent E-mail system is to reuse previous reply E-mails. This is also known as textual case-based reasoning (CBR) (Lapalme & Kosseim, 2003). The method is inspired by a scientific study of human memory cognition (Schank, 1982), using previous experience to solve similar new problems.

Lapalme and Kosseim (2003) divided CBR processing into three stages: case retrieval, case reuse and answer penalisation. Based on TF-IDF and mutual information measurement algorithms, they collected all basic word pairs from received E-mails and their replies and selected the most crucial word pairs.

Lapalme and Kosseim's experimental process was based on the substitution of entities in E-mails, a classification model determined the extraction of words representing role information. Specifically, entities such as the sender name, company name, and specific business need to be tagged with the corresponding reply content using senders, subsidiaries, or financial institutions provided in the prepared repository (Lapalme & Kosseim, 2003). However, the limitation of this study is that its application field is very narrow, and this substitution mainly depends on specific enterprises. It is impossible to predict various relational areas and perform tag matching, and it is hard to adjust the numeric information (such as price, date and age). Their research was limited mainly to technologies of fifteen years ago. However, this research idea still provides us with a valuable reference.

Hewlett and Freed (2008) present a much better process of responding to recommendations. Their research focused on receiving a large number of similar

inquiry E-mails over a limited period time by engaging in user participation feedback, designing the system, and selecting the most relevant quick learning answers. The machine learning method they chose is Margin Infused Relaxed Algorithm (MIRA), which was proposed by Crammer and Singer (2003) used for dealing with multi-class problems. Also, all the existing messages with triggering properties and queries are simultaneously converted into TF-IDF vectors. In each round of testing, eight stimuli messages that are closest to the query message will be selected and displayed to users participating in the test, and machine learning algorithms are executed on the response selected by these users. Their research has achieved better performance than previous work.

The latest research using case-based reasoning to design an intelligent E-mail function is in Linggawa's (2017) master's thesis. He stored historical responses in a case base and resolved new mail issues by reusing previous solutions. The process combined text processing, semantic analysis (such as lexical analysis and synonym expansion) and a TF-IDF retrieved-based method to find similar exact matching cases. The experimental results showed that synonym expansion could improve the accuracy of retrieval matching. The training corpus used is the Enron E-mail dataset, which is a free online data resource.

It is a good design idea to reuse a similar previous E-mails as a new response. Although there is not much research that can be used for a reference regarding this method, our research will draw on the ideas of previous studies. Moreover, we will introduce the latest cutting-edge technologies and some new ideas.

2.2.3 Automatically generating E-mail Response

In some E-mail management systems, an E-mail response program is integrated, which can automatically predict response suggestions based on new

received E-mail.

In the early years, leading automatically generated responsive E-mail systems used three steps to form a response: 1, Identify the issues contained in the E-mail content; 2, Search the predefined solutions in the knowledge base; 3, Provide a reformulated solution (Busemann, Schmeier & Arens, 2000). In the study by Busemann et al. (2000), they used shallow text-based methods and machine learning techniques (STP and SML) to classify E-mail request categories.

The simplest and most common applications used pre-written response templates and filled in the blanks to assist the creation of new E-mails. Some systems were even able to extract templates automatically. For example, some call centres, such as Kana and RightNow, saved response time by tracking customer E-mails and designing answer templates for the common questions and answers (Lapalme & Kosseim, 2003).

Kosseim et al. (2001) used the information retrieval method and NLP technology in their research. Based on the Lexico-syntactic extraction methods, specific information (such as date, structure and organisation name) in a specific discourse domain of a message is presented in a structured template format. Then by combining semantic validation and discourse reasoning, a template of the answers is selected, the selected normative replies are filled into the template correspondingly, and the answers are organised into semantic responsive answers.

In recent years, artificial intelligence has made rapid progress in NLP. Based on previous research, a large number of studies have adopted new algorithms and models and made many breakthroughs. Among them, Google Gmail team research on intelligent E-mail response is very prominent.

In the paper, 'Smart Reply: Automated Response Suggestion for E-mail' (Kannan et al., 2016), published by the Google team in August 2016, a new deep

learning algorithm, RNN (Giles, Kuhn & Williams, 1994), was implemented for the design of an auto-reply E-mail function. They used the Long Short-Term Memory (LSTM) neural network (an improved model of the RNN) (Hochreiter & Schmidhuber, 1997) to process received messages and predict responses. Considering the high training cost of the LSTM neural network model, in order to improve the response quality, they combined semi-supervised graph learning (Ravi & Diao, 2016) and a semantic intention clustering method to generate offline response space, and then reduced selection to provide the best response suggestion.

Referring to the techniques proposed in the previous article in combination with machine learning classification, weighted keywords and similarity measurement techniques, Parameswaran et al. (2018) designed a function of automatically generating and suggesting short E-mails. The system was designed to respond to the various types of queries submitted by university staff and students. Their research has also been applied to the functional services of their university, which sends and receives large numbers of inquiry E-mails every day.

In May 2018, Google's research team improved their auto-reply model on Gmail's smart prediction function[1]. This impressive function has dramatically improved the Gmail user experience. The model they used is shown in Figure 2.1. It can predict the next word in a sentence that a user might want to type, based on the order of the preceding words. The core technology they used is still the LSTM neural network, which combines the natural Bag-of-Words (BOW) (Bengio, Ducharme, Vincent & Jauvin, 2003) method to balance delay constraints. At the same time, their research results benefit from the development of hardware technology. Most of their computing is performed using the TPUv2

[1]https://ai.googleblog.com/2018/05/smart-compose-using-neural-networks-to.html

Pod[2]. Practically real-time prediction is achieved.

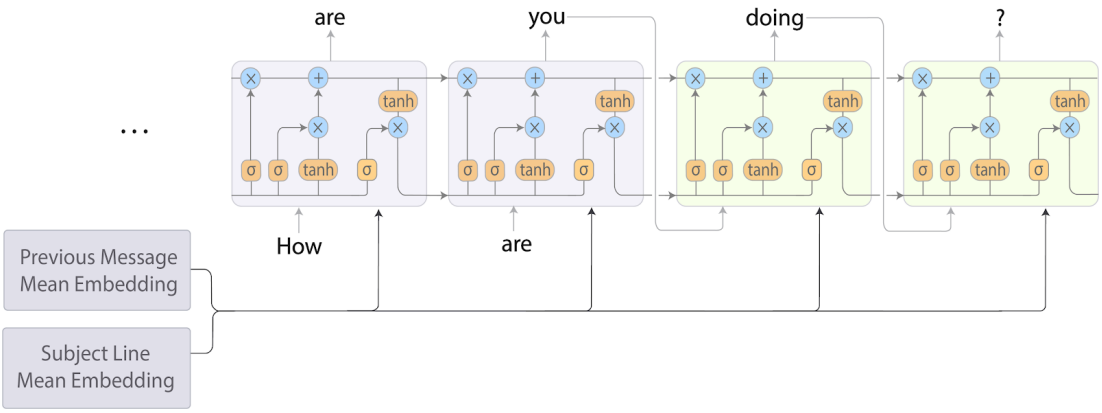


Figure 2.1: Gmail Smart Prediction Model[3]

It is a very challenging approach because deep neural networks have higher requirements in all aspects, such as the quality of training data. If the dataset is not large enough, then this high-level neural network may not learn relevance properly. Meanwhile, since it is an end-to-end unsupervised learning method, we cannot control the learning process and results. Nevertheless, although the dataset we can use is small, we still wanted to try the most advanced approach in this study. In order to make up the deficiency of objective conditions, we will introduce some new ideas to optimise the model structure.

2.3 Related Techniques

In the field of NPL (Manning, Manning & Schütze, 1999), technology is advancing rapidly. In the previous section, we analysed and discussed the three main types of E-mail responses in the existing literature. In this section, we mainly study related technologies that may be involved in this research. The

[2]https://cloud.google.com/tpu/

[3]https://2.bp.blogspot.com/-ilOCekdQP0Y/WvxdAt6fPZI/AAAAAAAACvE/2_bZTVZt2D8iwSeiKx1rB2rpTVbr_v9KQCLcBGAs/s1600/model3.png

main research interests include the application of TF-IDF in the field of information retrieval, similarity calculation, prevalent word embedding technology, neural network technology in the deep learning, and the design idea of Chatbot in the field of similar technology.

2.3.1 Traditional Feature Extraction Methods

The information retrieval method (Mooers, 1950), which requires users to access pre-stored information, is a technology closely integrated into this study. The core of the information retrieval process is retrieving the close correlation between query questions and query content (Baeza-Yates, Ribeiro et al., 2011), to extract the characteristics that reflect information content. The traditional feature extraction methods are based on the vector space model, including TF-IDF and One-hot vectors. Salton and Buckley (1988) proposed the concept of TF-IDF, which became the most widely used solution in the information retrieval field.

TF-IDF Weighting

Ramos et al. (2003) demonstrated the excellent performance of TF-IDF in determining critical information in the documents of corpus. The corpus they used was a random collection of 1,400 documents from an extensive United Nations database in 1988. In order to test the stability of TF-IDF in noisy environments, their experiment retained the original format tags and simulated more noise. They calculated the weight of queries based on the formula

$$\sum_{i} W_{i,d}$$

and selected the top 100 relevant documents in descending order. Compared with naÔve, another query retrieval method, the results showed that this simple algorithm could effectively enhance the query retrieval function through classification.

Chum et al. (2008) combined the weighting algorithm of TF-IDF with the minimum hash algorithm in their study and proposed a method to calculate the similarity between images and video, realising a fast index. Their experimental results confirmed that the TF-IDF algorithm improved the efficiency and quality of similarity calculation between the image and video field.

The feature extraction process of Dredze et al. (2008) used TF-IDF scores to find the words that best represent questions. For instance, the TF score is used to indicate the number of times a word appears in the question, the IDF value is the total number of sentences containing the word, and the words expressing the question are defined as the top 30 words with the highest TF-IDF score.

In a very recent paper, Kim et al. (2019) proposed a multi-co-training (MCT) approach in the field of document classification, combining TF-IDF, latent Dirichlet allocation (LDA) (Blei, Ng & Jordan, 2003) and Doc2Vec (Le & Mikolov, 2014) based on shallow neural networks. The combination of these three methods increased the diversity of feature sets used for classification. They also analysed and compared the characteristics of each method in their research.

Through a study of the above literature, we found that TF-IDF is the most straightforward algorithm, and it still has good performance in the field of information retrieval and information classification. One of our goals is to try using a simple algorithm to get excellent results.

2.3.2 Word Embedding

Traditional methods of information retrieval and classification mainly adopt algorithms based on Continuous Bag-of-Words (CBOW) or word frequency statistics. Whether generating one-hot vectors or TF-IDF weights, its disadvantages are that it ignores the position information of the words and the relationship between them, so the generated features are discrete and sparse, which may eventually lead to too many dimensions. In the process of NLP, grammar, word order, and semantics between words are important and cannot be underestimated (Lilleberg, Zhu & Zhang, 2015; Tang et al., 2014). Thus, the concept of word distribution in vector space was proposed (Rumelhart, Hinton, Williams et al., 1988). This way represents a word vector as a dense vector by reducing the number of dimensions used to represent a word, namely, word embedding or distributed representation of a word (Mikolov, Sutskever, Chen, Corrado & Dean, 2013a). Embedding word vectors enables models to capture the contextual semantics and syntactic similarity of words in a document, as well as their relationships to other words (Mikolov et al., 2013a; Tang et al., 2014).

Word to Vector

Currently, the most popular word embedding methods are Word2Vec (Mikolov et al., 2013a) and Glove (Pennington, Socher & Manning, 2014). In our research, the techniques used are derivative algorithms based on Word2Vec: Doc2Vec and Sent2Vec. Although the performance of Glove is better than Word2Vec in some cases (Pennington et al., 2014), we did not select Glove in our study because we chose to focus on applying the extended algorithms of Word2Vec.

The literal meaning of Word2Vec is to convert words into vector representations. Developed by the Google R&D team in 2013, the algorithm has become

one of the most popular techniques for word embedding. By using a shallow neural network, a well-trained model can learn the expressive relationships between semantics. It currently performs well in text classification, machine translation, subject recognition, and various NLP tasks.

Lilleberg, Yun, and Yanqing (2015) introduced the application of Word2Vec in text mining in their paper. The assumption in this research is that Word2Vec will bring additional semantic features to help with text categorisation. After deleting the stop words, they combined TF-IDF and Word2Vec to create a weighted sum of word vectors for the words in the corpus. Comparing the experimental results of the above method with those of TF-IDF and Word2Vec alone, showed that the performance of the former is the best in most cases. It also proved that TF-IDF and Word2Vec algorithms are reliable in text mining.

Zhou et al. (2015) applied the Skip-gram method (Mikolov, Chen, Corrado & Dean, 2013b) in Word2Vec to the research of community question and answer retrieval. In this experiment, the framework of the Fisher Kernel (Clinchant & Perronnin, 2013) was used to aggregate variable-size words into embedding fixed-length vectors. They showed that the advantage of this model is that it can find the semantic relations in the context, which can improve the performance of question-and-answer retrieval of community files. The effect of this model is significantly better than that of the latent topics model (Cai, Zhou, Liu & Zhao, 2011).

In the field of machine translation, many studies have shown that context within the scope of discourse can help achieve a smooth translation (Hardmeier, Stymne, Tiedemann & Nivre, 2013). When used to predict the semantic relationship between words across languages, Garcia et al. (Garcia, Tiedemann, España-Bonet & Màrquez, 2014) demonstrated that these models have powerful capabilities. They used the CBOW mode (Mikolov et al., 2013b) in Word2Vec for

bilingual cases and evaluated the model by predicting semantic-related words and cross-linguistic vocabulary substitution. The biggest challenge, however, is that some words in the translation pairs are lost during the training process.

With the rise of deep learning and various neural networks in the field of NLP, the Word2Vec algorithm plays an important role. Specifically, input from the source language is often inconsistent in length, and Word2Vec serves to align the length of the vector in the input layer (Singh et al., 2017).

Paragraph to vector

From the above literature research, we know that the Word2Vec algorithm is a distributed semantic representation of word construction. Training can be divided into two different models: CBOW and Skip-gram. The former model uses the context words to predict the centre word, while the latter model focuses on the use of the current word to predict the context words (Mikolov et al., 2013b). These ideas can also be extended to sentences and full documents.

Mikolov, Le and Sutskever (2013c) mentioned the limitations of the Word2Vec algorithm. That is because when faced with words with multiple meanings, the process of Word2Vec is to mix them into a common representation. Therefore it cannot resolve lexical ambiguity. Additionally, many complex language phenomena such as sarcasm, cannot be recognised (Le & Mikolov, 2014). In this case, the method of paragraph vector is factored out. In their research, Le and Mikolov (2014) introduced a more advanced algorithm based on Word2Vec and stated that it is superior to the Bag-of-words model and other techniques. Para2Vec, which uses vectors to represent paragraphs, can also be extended to a model of Sent2Vec or Doc2Vec, which means to use vectors to represent sentences or documents.

Le and Mikolov (2014) not only proposed two models of paragraph vector but also used the paragraph vector as a method of text comprehension, which is applied to emotion analysis and information retrieval. The training datasets used were Stanford sentiment treebank dataset (Socher et al., 2013) and IMDB dataset (Maas et al., 2011) respectively. The experimental results showed that the unsupervised paragraph vector algorithm could learn vector representations of texts of different lengths. The good performance in capturing paragraph semantics proves that this method is more competitive in the NLP field.

Zhu, Li and Melo (2018) proposed a framework for generating experimental sentence triplets, by comparing three methods of sentence embedding, this paper explored whether and how the similarity of sentence embedding is affected by the syntactic structure or semantic changes of a given sentence. They concluded that the method of sentence embedding could distinguish between negative and synonymous information in a sentence. In this case, the performance of the sentence embedding method is much better than that of the word embedding method.

The method of sentence embedding is also common in the related research of sentiment analysis and problem classification (Kiros et al., 2015). Although the method of sentence embedding performs well in analysing the relationships between sentences in a document, we cannot deny that the word embedding method performs well in some related fields. They all play an essential role in the research of NLP.

2.3.3 Similarity Measurement

In information retrieval and text mining, many algorithms use vectors to represent the features of the document. Theoretically, terms can be assigned to

vectors in different dimensions, then the Cosine similarity becomes a powerful metric when measuring the similarity between two documents using these vector relations (Singhal et al., 2001).

Salton et al. (Salton, Wong & Yang, 1975) proposed a model, namely the vector space model, for an automatic indexing system. The model can represent a document in a multi-dimensional vector space, and the terms in the document can be assigned to different dimensions by their weights, depending on their importance in the document. In this way, measuring the similarity between documents in the vector space can be converted into the method of measuring the distance between the two vectors by their cosine angle in that space. Therefore, the distance between vectors can reflect the degree of similarity between documents.

This model is very beneficial and can be extended to calculate similarity that can be represented by any vector. For example, it can be combined with the weights generated by TF-IDF, or with vectors generated by Word2Vec, Sent2Vec or Doc2Vec model, so it will be applied in our research.

2.3.4 Deep Learning and Neural Network

Hinton, Osindero and Teh (2006) put forward the concept of deep learning in their article. Based on the Directed Belief Networks (DBN), an unsupervised greedy layer-by-layer training algorithm was proposed to bring about the hope of solving the deep structure-related optimisation problems. Then the deep structure of the multi-layer automatic encoder was proposed. In addition, the Convolutional Neural Network (CNN) was presented by Lecun et al. (1989). It is the first real multi-layer structure learning algorithm to improve training performance by using spatial correlation and reducing the number of parameters.

This theory has become a solid foundation for modern deep learning research.

RNN

In many studies, we can see that deep learning has been successfully applied in many fields, such as computer vision, speech recognition, memory networks and NLP. One algorithm for processing language sequences is RNN (Rumelhart et al., 1988). The structure can process input sequences using its internal storage state, which is very suitable for serialisation processing, such as continuous handwriting recognition (Graves et al., 2009) or speech recognition (X. Li & Wu, 2015).

Vinyals and Le (2015) applied the sequence-to-sequence characteristics of the RNN to the task of conversation modelling to predict responses in question-and-answer conversations. Surprisingly, the experimental results were good in terms of fluency and accuracy. Their experiment used two training datasets: one from the IT help desk chat service and the other from movie transcripts. They found that this conversational model could extract knowledge from noisy but open domain datasets and generate simple and basic conversations. At the same time, the model could capture more important long-range correlations than the N-gram model, which became an important idea for designing Chatbots. However, they pointed out in their study that lack of consistency in the dialogue was the biggest problem that still needed to be addressed.

Following this research, one year later Jiwei et al. (2016) proposed an advanced Neural Conversation Model based on persona information.

RNN also extended out of many variants, including Bidirectional Associative Memory (BAM) Network (Kosko, 1988), Echo State Network (ESN) (Jaeger &

[4]https://en.wikipedia.org/wiki/Recurrent_neural_network/media/File:Recurrent_neural_network_unfold.svg

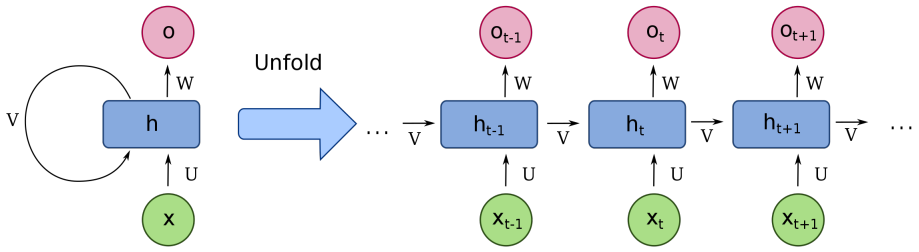


Figure 2.2: One RNN Cell Unit[4]

Haas, 2004), Neural History Compressor(Schmidhuber, 1992), as well as the most widely used LSTM (Hochreiter & Schmidhuber, 1997) and GRUs (Cho et al., 2014), which use gates to control their unit cells. A basic RNN cell unit is shown in Figure 2.2.

LSTM

A typical LSTM unit (as shown in Figure 2.3) consists of an input gate, an output gate and a forgotten gate. The unit can remember values at any time interval, and the three gates jointly determine the flow of information in and out of the unit (Hochreiter & Schmidhuber, 1997).

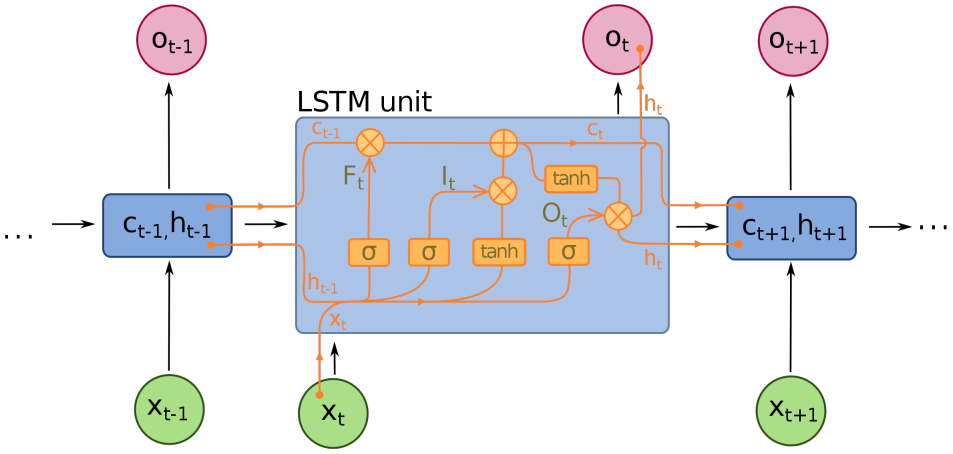


Figure 2.3: One LSTM Cell Unit [5]

GRU

The GRU algorithm is an improved LSTM algorithm proposed by Chung et al. (2014). Based on LSTM, it combines the forgotten gate and input gate into an update gate and combines the data unit state and hidden state, which makes the model structure simpler than LSTM. Its basic unit is shown in Figure 2.4. Since it is designed as two gates, its parameters are smaller than LSTM's, which is an advantage but also has limitations compared with LSTM.

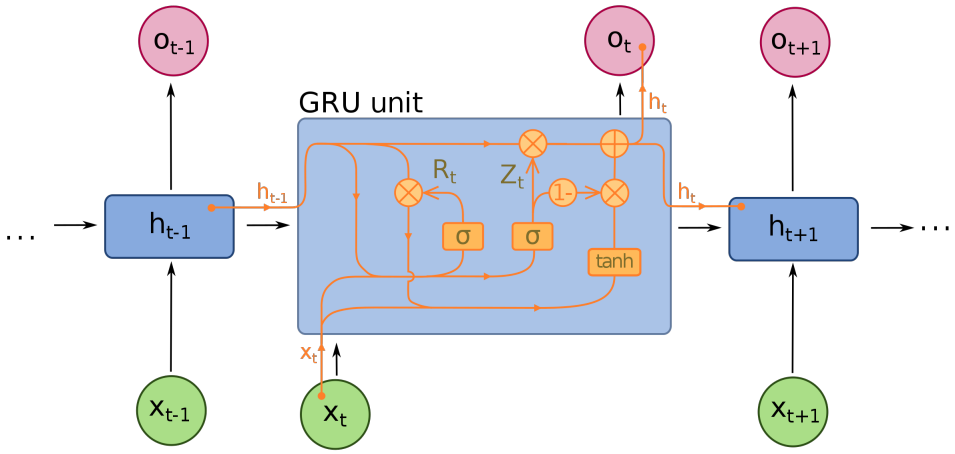


Figure 2.4: One GRU Cell Unit[6]

2.4 Research Gap

Developing an E-mail system with an intelligent reply function is still the direction that researchers are working on to solve the E-mail overload problem. In many studies, there is a big gap in both the reuse of past E-mails based on the information retrieval method and the prediction generated responses based on neural networks.

[5]https://en.wikipedia.org/wiki/Recurrent_neural_network/media/File:Long_Short-Term_Memory.svg

[6]https://en.wikipedia.org/wiki/Recurrent_neural_network/media/File:Gated_Recurrent_Unit.svg

There is not much research on reusing previous reply E-mails because this method needs ideal E-mail datasets in order to get good performance. But the E-mail datasets that can be used for research are limited. Some studies used small datasets collected by themselves or after simply processing the data, directly adopted the method of information retrieval. In this case, it was difficult to achieve the desired effect.

As for automatically generating E-mail response, many researchers have introduced word embedding, deep learning algorithms and various neural networks. Even Gmail, now the best at responding intelligently to E-mails, uses technology based on word-level predictions. This means using words to predict the following possible word in a sentence. However, most E-mails are long text; and even short E-mails usually contain more than two sentences. Research and exploration in this area are still lacking regarding how to predict and generate an E-mail consisting of multiple sentences. Part of our research is to design a model that combines multiple deep learning algorithms to explore the way to generate E-mail responses at the sentence or paragraph level.

2.5 Summary

E-mail overload is still the biggest problem facing users today. In our review of E-mail management system technology, we found that there is still a lot of research potential in this space. Many researchers continue to contribute and believe that intelligent E-mail management systems will play an important role in alleviating E-mail overload.

After reviewing three main design methods of intelligent E-mail management systems, we adopted two main methods: E-mail reuse based on the information retrieval method and predictive response generation based on neural networks.

Relevant machine learning and deep learning algorithms will be involved in this study, including TF-IDF, Doc2Vec, Sent2Vec and GRU.

We found that the TF-IDF algorithm is the simplest and most direct algorithm, and it has good performance in the field of information retrieval and information classification. One of our goals is to try to use a simple algorithm to get excellent results. Moreover, although Glove performance is better than Word2Vec in some cases, in this study, in order to find semantic correlation, we adopted the derivative algorithms based on Word2Vec, namely Doc2Vec and Sent2Vec. In addition, due to the small number of training sets used in the experiment, we will adopt a lightweight GRU relative to LSTM for the neural network-based method of generating predicted responses.

Chapter 3

Design and Methodology

In Chapter 2, we explored previous research and techniques often used to develop intelligent E-mail response functions. In this chapter, we describe the system architecture design of the project and the architecture design of the three models. Also, we explain the related algorithms and principles used in these models in detail.

The following details the organisation of this chapter. Section 3.1 presents the design thoughts and system framework of the project. Section 3.2 introduces three models and presents relevant algorithms research. The models include TF-IDF, Doc2Vec and GRU-Sent2Vec hybrid model. Since the algorithms applied in each model are very different, the research on the algorithms is to fully understand and prepare for parameter adjustment in the process of model implementation. Section 3.3 gives a brief summary of this design and methodology.

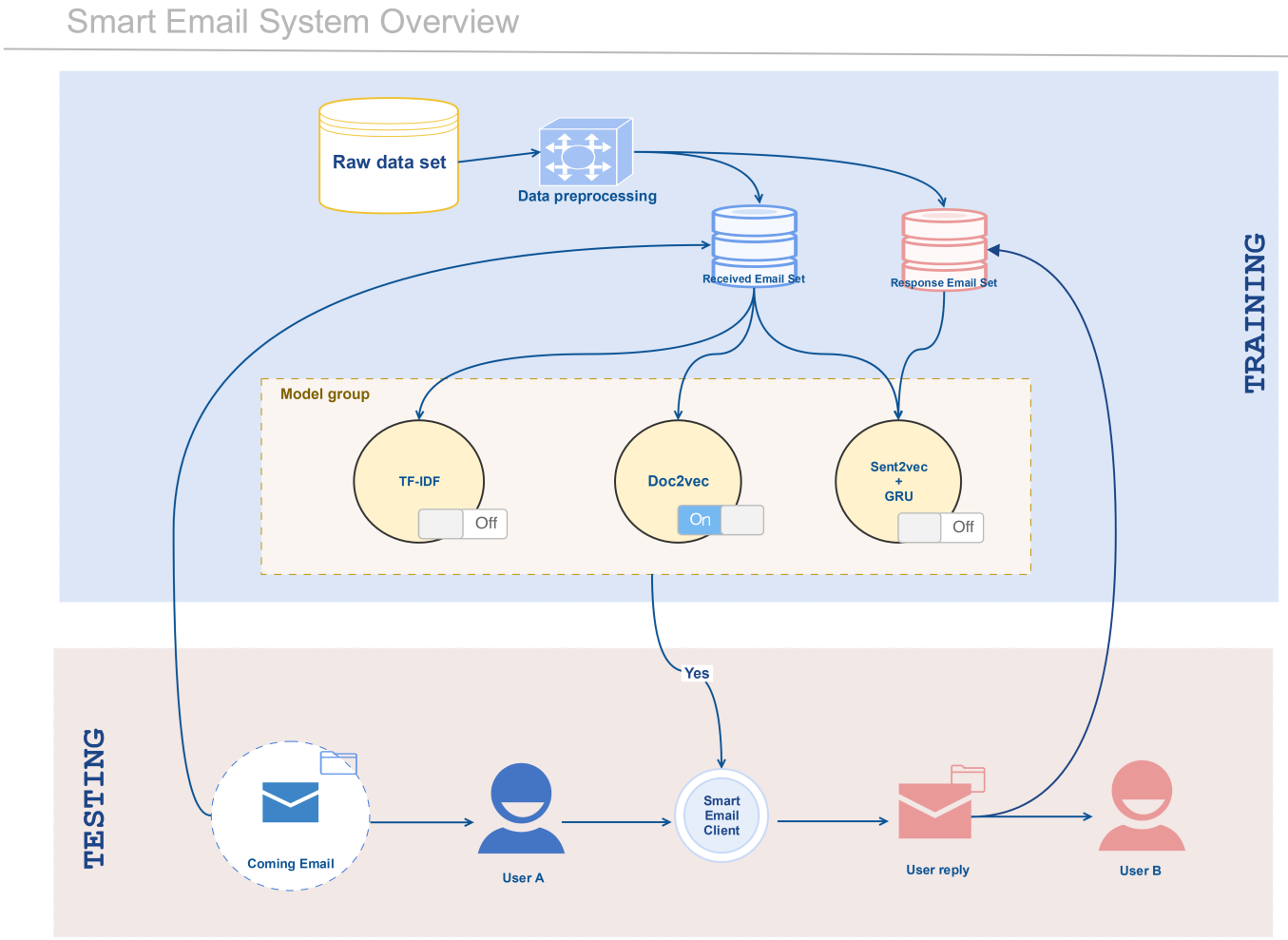


Figure 3.1: Intelligent E-mail System Overview

3.1 System Design

Our research aims to provide an intelligent response solution for individuals or corporate departments (such as service centres or help desks) that receive a large number of similar E-mails every day. This kind of solution can significantly save time, improve work efficiency and promote social productivity.

For this purpose, we developed an intelligent E-mail client with core functionality to extract information and learn new information based on learning from previous E-mails and then give intelligent response suggestions to new E-mails.

Three models were trained in this experiment in order to explore better performance: TF-IDF model, Doc2Vec model and GRU-Sent2Vec hybrid model. Their operating principles are different. The first two models belong to the information retrieval method, and the last model is a combination of the information retrieval method and the information generated method, which is a hybrid model specially designed for this experiment. The methods we use are also widely used in popular applications such as Chatbot. We have improved our method on the basis of relevant work and adapted it to the long-text E-mail format.

The system framework of this experiment is shown in Figure 3.1. Initially, we needed to execute a series of text prepossessing steps on the original E-mail dataset. For the processed E-mails, we only kept the matching E-mail pairs (namely received E-mail - reply E-mail), so that we could obtain the relationships between the received E-mails and reply E-mails and put them into two databases. The next step is to train our three models using the results of text processing.

Upon completion of the model training, when a new E-mail is received, the intelligent E-mail management system client will offer users two options: one is to reply directly, and the other is to use the smart reply function. In terms

of the smart reply function, users can choose their preferred reply suggestion generated by the three models.

Regardless of whether users directly reply to an E-mail or modify the response suggestions generated by the models, after replying, the newly received E-mail will be paired with the sent E-mail, and then the results will be stored in both databases in preparation for the models' further learning.

In this study, we chose not only a traditional machine learning method, but also a neural network based on a word embedding algorithm and deep learning algorithm. Our development vision is not limited to a rules-based specific field, nor is it intended to produce only brief, common, yet ineffective feedback suggestions (e.g. Thank you, Yes, please, No, thank you, Best regards, and so on) — our aim is to design a long-text response suggestion that is more suitable for the E-mail format. The application scenario is more suitable for a company's service centre because there, a large number of similar E-mails need to be sent and received manually. The next section will introduce the design ideas of the three models.

3.2 Model Design

In reviewing relevant techniques from previous studies, we identified two main approaches to designing our models: information retrieval (Figure 3.2) and information generation (Figure 3.3). For the retrieval-based model, this project adopted TF-IDF and Doc2Vec algorithms for experiments. TF-IDF is a classical algorithm that is applied in the fields of information retrieval and text classification. Due to its good performance, Doc2Vec has become more and more popular in the industry in recent years. In addition, Linggawa (2017) also used the TF-IDF algorithm to design an intelligent E-mail client sharing a similar

motivation.

In terms of the second approach mentioned above, we selected a deep learning algorithm to design our generative model. Since the sequence-to-sequence (seq2seq) model was successfully introduced from the field of machine translation into the Chatbot dialogue system, we tried to apply a seq2seq model, GRU, to our intelligent E-mail management system. In order to obtain better experimental performance, we improved some algorithm aspects, such as nested Sent2Vec into GRU, and designed a hybrid model of information generation combined with information retrieval. Next, we will illustrate the design thoughts of these three models.

3.2.1 TF-IDF Based Model

TF-IDF is a popular term weighting scheme widely used in information retrieval and text mining (Trstenjak, Mikac & Donko, 2014). In this section, we will introduce the TF-IDF model's core content, the TF-IDF algorithm and the Cosine Similarity algorithm of the vector space model. In the modelling process, we used the Python programming language library, Scikit-learn library[1].

Term Frequency (TF): According to the statement by Luhn (1957), the TF weight of the terms for a document is proportional to the frequency of terms appearing. In the traditional formula, the number of occurrences of terms indicates the frequency of occurrence of terms, as shown in Equation 3.1. The modelling algorithm of TF-IDF is modified in the Scikit-learn library (such as TF for word frequency), which is slightly different from the traditional formula.

$$TF_{(t,d)} = n_{i,j} \tag{3.1}$$

[1]https://scikit-learn.org/stable/

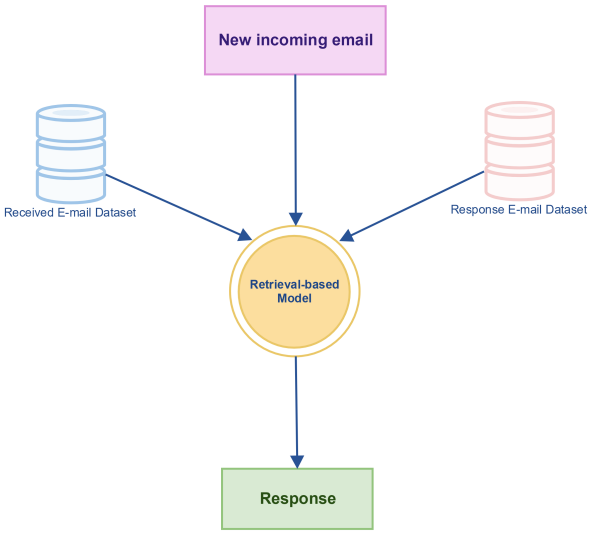


Figure 3.2: Retrieval-based Model

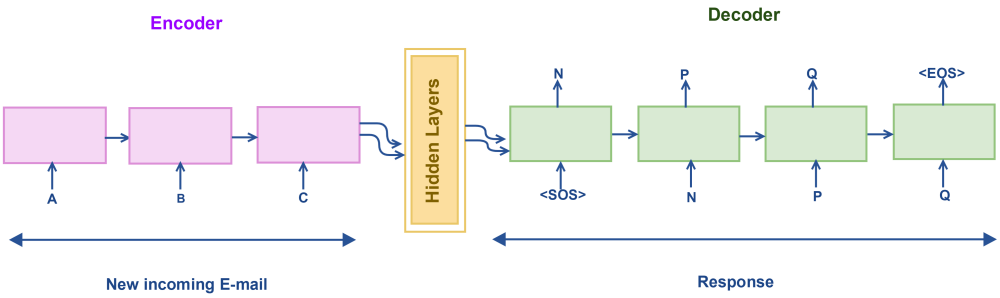


Figure 3.3: Generative-based Model

with:

- $n_{i,j}$: The number of times the word t_i appears in the document d_j.

Inverse Document Frequency (IDF): Inverse Document Frequency, also known as IDF (Equation 3.2[2]), measures the importance of terms in the document or its particularity in the whole corpus. Actually, certain terms (such as 'the', 'of', 'is') have a very high frequency in most documents, but they contribute very little to the importance of the content in the document. Sometimes less frequent terms, however, are more relevant to the topic of the document.

$$IDF_{(d,t)} = \log \frac{N_{(D)} + 1}{N_{(t,D)} + 1} + 1 \tag{3.2}$$

with:

- $N_{(D)}$: Total number of documents in Corpus D.
- $N_{(t,D)}$: The number of documents containing the word t.
- $IDF_{(d,t)}$: The word t's IDF value in the document d_j.

TF * IDF: The value of TF-IDF (Equation 3.3) is the multiplication of TF and IDF (Ramos et al., 2003). This numerical result is intended to reflect the importance of words to documents in collections or corpus.

$$TF - IDF_{(n,d)} = TF * IDF \tag{3.3}$$

The vector of TF-IDF is normalised by the Euclidean norm (Equation 3.4):

$$v_{norn} = \frac{\boldsymbol{v}}{||\boldsymbol{v}||} \tag{3.4}$$

[2]https://scikit-learn.org/stable/modules/feature_extraction.html

TF-IDF is a method based on the Bag-of-words statistics (J. Yang, Jiang, Hauptmann & Ngo, 2007). Assuming the document is just a collection of words, the document can be vectorised by calculating the $TF - IDF_{(n,d)}$ value of each word. The vectorised document refers to the vector space model, which allows calculation of the similarity between all documents in a corpus by using the Cosine theorem.

Cosine Similarity: Using the Euclidean dot product formula, the magnitude and angle of two non-zero vectors (**A** and **B**) can be expressed. Moreover, the included angles between vectors provide a basis for calculating the similarity of documents, which is called Cosine Similarity (as shown in Equation3.5).

$$\boldsymbol{A} \cdot \boldsymbol{B} = ||\boldsymbol{A}|| \cdot ||\boldsymbol{B}|| * cos(\theta) \tag{3.5}$$

The Cosine Similarity method is suited to any number of dimensions; hence, documents are represented by vectors in information retrieval and text mining. Also, Cosine Similarity is treated as a high-dimensional positive space for each different term that is assigned to a different dimension. Therefore, we can measure the similarity between the two documents in terms of the subject (Jeon, Croft & Lee, 2005).

In this case, we use one of the simplest examples to show how to calculate the semantic similarity between two documents. Suppose we have two documents, A and B, each consisting of four words. The two document vectors can be represented as $\boldsymbol{A} = [\boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{a}_3, \boldsymbol{a}_4]$ and $\boldsymbol{B} = [\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3, \boldsymbol{b}_4]$. The similarity between document A and B can be calculated by Equation 3.6.

$$
\begin{aligned}
similarity_{(A,B)} = cos(\boldsymbol{A}, \boldsymbol{B}) &= \frac{\boldsymbol{A} \cdot \boldsymbol{B}}{||\boldsymbol{A}|| \cdot ||\boldsymbol{B}||} \\
&= \frac{\sum_{i=1}^{n}(A_i * B_i)}{\sqrt{\sum_{i=1}^{n}(A_i^2)} * \sqrt{\sum_{i=1}^{n}(B_i^2)}} \\
&= \frac{a_1 * b_1 + a_2 * b_2 + a_3 * b_3 + a_4 * b_4}{\sqrt{a_1^2 + a_2^2 + a_3^2 + a_4^2} * \sqrt{b_1^2 + b_2^2 + b_3^2 + b_4^2}}
\end{aligned}
\tag{3.6}
$$

This method can help us calculate the similarity of all the documents in the corpus. The implementation of this model will be demonstrated in the following chapter.

3.2.2 Doc2Vec Based Model

Doc2Vec is an extension of Word2Vec (Le & Mikolov, 2014), similar to Word2Vec, except that it uses a fixed-length vector to represent an entire document. This unsupervised learning algorithm can express paragraphs or documents as vectors that are well suited for document processing tasks. For example, it can be used to compare similarities between paragraphs or documents. In this section, we will explain the related algorithms for the implementation of the four modes in the Doc2Vec model.

PV-DM: In the Distributed Memory Model of Paragraph Vectors (PV-DM) architecture (Figure 3.4), the algorithm is similar to CBOW in Word2Vec, but with a new document's ID added during the training process. Combined with the document's ID, this mode can predict the next word from the previous word in the specified window range. The ID, like a common word, is mapped to a vector that has the same dimensions as other word vectors, but it comes from

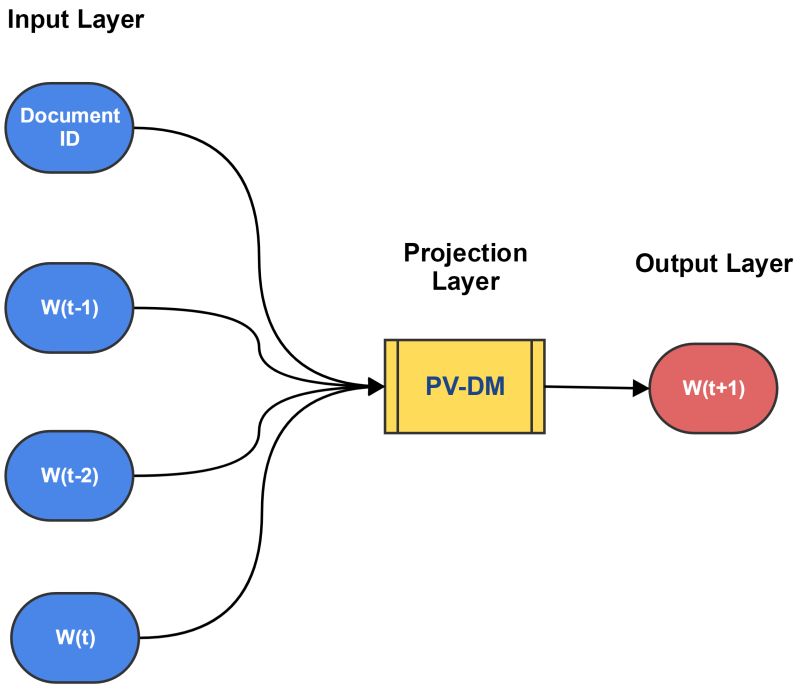


Figure 3.4: PV-DM Mode

the other different vector spaces. It can be expressed as Equation 3.7.

$$\mathcal{L} = \sum_{w \in C} \log p(DocID, w_{(t-2)}, w_{(t-1)}, w_{(t)} | w_{(t+1)}) \tag{3.7}$$

with:

- $DocID$: The ID of one document.
- w: The words in corpus C.
- $w_{(t)}$: The word in the current time t in the time series.

In the training process of the document, the words in the same document share the same document vector, which is equivalent to using the semantics of the entire document to predict the probability of each word.

PV-DBOW: PV-DBOW is an abbreviation for Paragraph Vector - Distributed Bag of Words (Mikolov et al., 2013b). As can be seen from Figure 3.5, its architectural design is similar to the Skip-gram in Word2Vec. Nevertheless, this mode differs in that it predicts the probability of current words based on document's ID, and the output is a randomly sampled word in the paragraph. It can be expressed as a maximum likelihood function, as shown in Equation 3.8.

$$\mathcal{L} = \sum_{w \epsilon C} \log p(w_{(t-2)}, w_{(t-1)}, w_{(t)}, w_{(t+1)} | DocID) \tag{3.8}$$

with:

- $DocID$: The ID of one document.
- w: The words in corpus C.
- $w_{(t)}$: The word in the current time t in the time series.

According to some related experiments, PV-DBOW is faster than PV-DM in computing speed, because it stores less data in the operation process. However, PV-DM works better for less frequent word prediction. We will conduct specific experimental operations in Chapter 4, and evaluate the effects of experimental results in Chapter 5.

Furthermore, based on the above two different architectures, there are two different modes in the training stage: Hierarchical Softmax and Negative Sampling.

Hierarchical Softmax: The core idea of the Hierarchical Softmax method is to use the Huffman Tree, which is the optimal binary tree with the shortest weighted path length. By putting the information with high weight in front, the maximum value of approximate conditional likelihood is calculated to reduce the computational burden. It can be expressed as Equation 3.9.

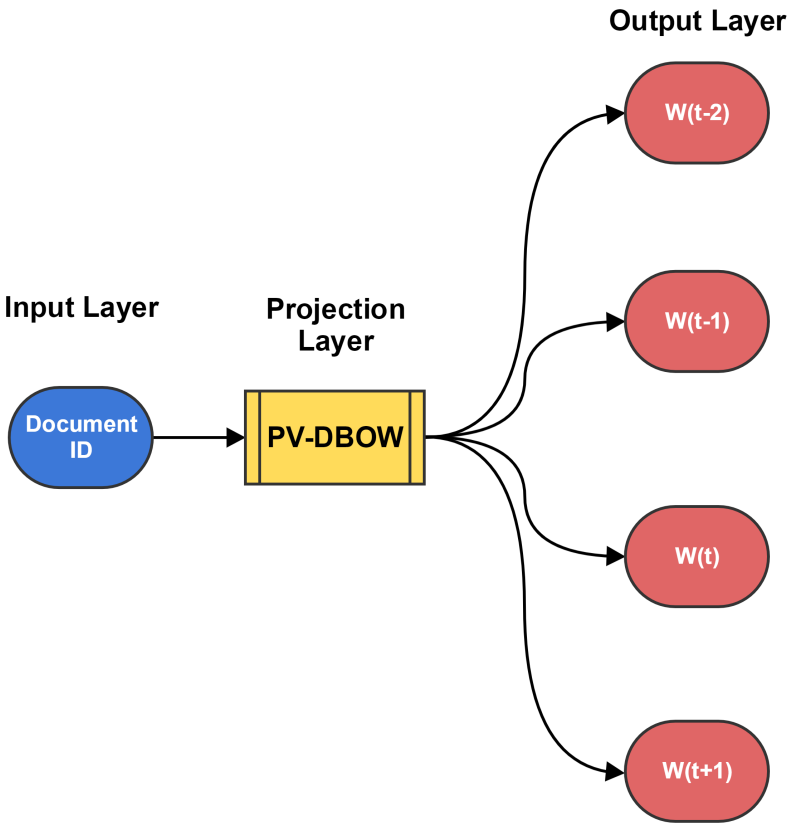


Figure 3.5: PV-DBOW Mode

$$p(d_j^w|X_w, \theta_{j-1}^w) = [\sigma(X_w^T \theta_{j-1}^w)]^{1-d_j^w} * [1 - \sigma(X_w^T \theta_{j-1}^w)]^{d_j^w} \tag{3.9}$$

with:

- d_j^w: The encoding of the word w on the jth node.
- X_w: The sum of the context vectors. In the projection layer, the vectors of the input layer are combined to form a matrix X_w.
- θ_{j-1}^w: The parameter vector corresponding to the non-leaf node in the path.
- X_w^T: X_w's transpose.

For example, we have a training sample with the centre word w, and the

context around it is written as $context(w)$. We can obtain the probability of a predicted word expression as Equation 3.10.

$$\begin{aligned}\mathcal{L} &= \sum_{w \in C} \log \prod_{j=2}^{l^w} [\sigma(X_w^T \theta_{j-1}^w)]^{1-d_j^w} * [1 - \sigma(X_w^T \theta_{j-1}^w)]^{d_j^w} \\ &= \sum_{w \in C} \sum_{j=2}^{l^w} (1 - d_j^w) * \log[\sigma(X_w^T \theta_{j-1}^w)] + d_j^w * \log[1 - \sigma(X_w^T \theta_{j-1}^w)]\end{aligned} \tag{3.10}$$

To solve the maximum value, we can use the gradient ascending algorithm to calculate the derivatives of θ and X, and the operation process is as follows Equation 3.11.

$$\begin{aligned}\frac{\partial \mathcal{L}(w,j)}{\partial \theta_{j-1}^w} &= [1 - d_j^w - \sigma(X_w^T \theta_{j-1}^w)] X_w \\ \frac{\partial \mathcal{L}(w,j)}{\partial X_w} &= [1 - d_j^w - \sigma(X_w^T \theta_{j-1}^w)] \theta_{j-1}^w\end{aligned} \tag{3.11}$$

Parameter θ and the word vector updated expression formulas are shown in Equation 3.12.

$$\begin{aligned}\theta_{j-1}^w &= \theta_{j-1}^w + \eta[1 - d_j^w - \sigma(X_w^T \theta_{j-1}^w)] X_w \\ v(\widetilde{w}) &= v(\widetilde{w}) + \eta[1 - d_j^w - \sigma(X_w^T \theta_{j-1}^w)] \theta_{j-1}^w\end{aligned} \tag{3.12}$$

with:

- η: learning rate.
- $v(\widetilde{w})$: The word w's vector.

Negative Sampling: Because Hierarchical Softmax has some disadvantages, for instance, once the corpus is very large or the central word w in a training sample is a very uncommon word, it will take a long time to find the word in the Hoffman Tree. Negative Sampling is a valid solution because it can eliminate the

complex computation of the Huffman Tree and make the model much simpler.

In the Negative Sampling method, the central word w is related to $context(w)$. If the prediction is correct, it is defined as a positive sample; if the prediction is wrong, it is defined as a negative sample. For a given sample, the probability is expressed as Equation 3.13.

$$p(u|context(w)) = \begin{cases} \sigma(X_w^T \theta^u) & \text{Positive sample} \\ 1 - \sigma(X_w^T \theta^u) & \text{Negative sample} \end{cases} \tag{3.13}$$

with:

- u: The central word u to be predicted
- $p(u|context(w))$: The probability of being correct for a given sample u.
- θ^u: The parameter vector corresponding to the given sample.
- X_w: The sum of the context vectors. In the projection layer, the vectors of the input layer are combined to form a matrix X_w.
- $\sigma(X_w^T \theta^u)$: The probability when the context is $context(w)$ and the predicted central word is u.

Then a negative sample function can be expressed as Equation 3.14.

$$g(w) = \prod_{u \in \{w\} \cup NEG(w)} p(u|context(w)) \tag{3.14}$$

with:

- $u \in \cup NEG(w)$: When the context is $context(w)$, the probability that the predicted central word is u.

Using logistic regression, we can obtain the corresponding log-likelihood function as Equation 3.15.

$$\begin{aligned}\mathcal{L} &= \log \prod_{w \in c} g(w) \\ &= \sum_{w \in C} \log \prod_{u \in \{w\} \cup NEG(w)} [\sigma(X_w^T \theta^u)]^{L^w(u)} * [1 - \sigma(X_w^T \theta^u)]^{1 - L^w(u)}\end{aligned} \tag{3.15}$$

with:

- $L^w(u)$: The frequency of a word u appearing in a context is represented by a random length.

Similar to Hierarchical Softmax, we use the stochastic gradient ascent method to update the gradient with just one sample at a time for an iterative update, and then get the Equation 3.16.

$$\begin{aligned}\theta^u &= \theta^u + \eta [L^w(u) - \sigma(X_w^T \theta^u)] X_w \\ v(\widetilde{w}) &= v(\widetilde{w}) + \eta \sum_{u \in \{w\} \cup NEG(w)} \frac{\partial \mathcal{L}(w, u)}{\partial X_w}\end{aligned} \tag{3.16}$$

3.2.3 GRU-Sent2Vec Hybrid Model

With the rapid improvement of hardware that supports extensive capabilities and advances in deep learning technology, many researchers proposed various novel neural network algorithms. Especially after the concept of distributed representations of word vector (Mikolov et al., 2013b) was widely accepted, Mikolov et al. (2011) found that many results have verified those language models trained by using a neural network based on big datasets are significantly superior to traditional language models in terms of performance. They also stated that deep learning methods had achieved very high performance across different NLP tasks. Different from Doc2Vec, sequence neural network models

such as GRU are related to the order of words or sentences, which determines the whole process of model training. The structure of one single GRU cell is shown in Figure 3.6.

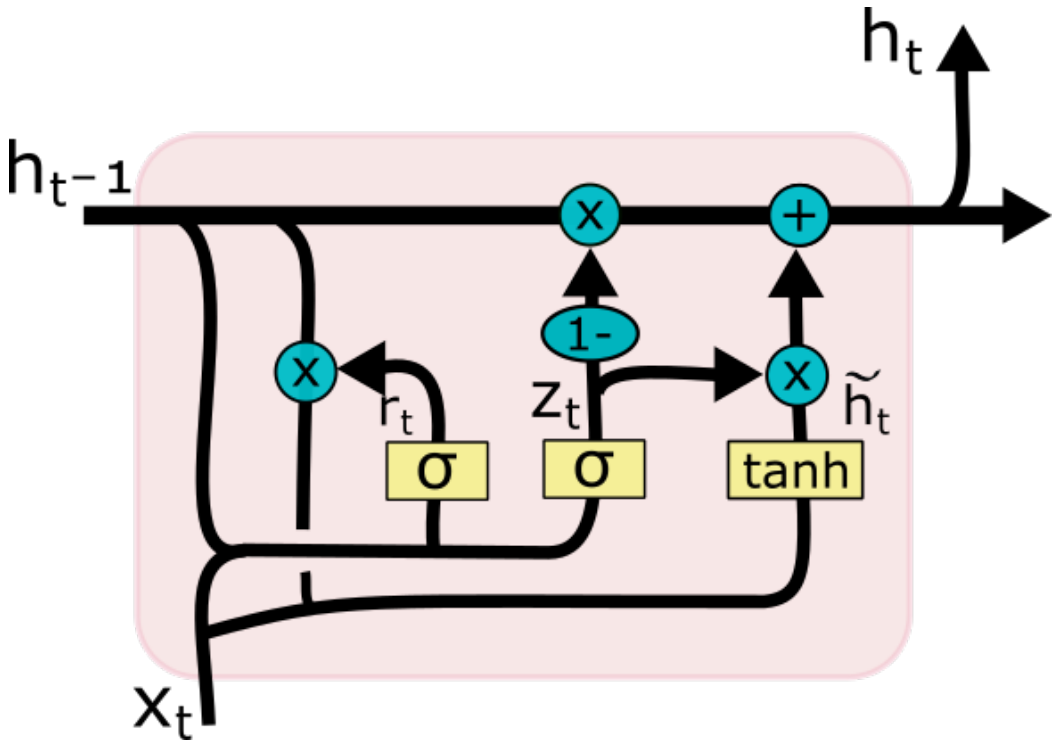


Figure 3.6: Basic One GRU Cell Unit

As we mentioned in the last chapter, Chung et al. (2014) proposed the structure of the GRU, which was designed as a reset gate that enables data learning and updating. It is much easier to implement than LSTM because GRU removes the cell state, then uses a hidden state to transfer information, and relatively reduces the amount of parameter tuning tasks during data modelling and training tasks (Chung et al., 2014). As far as we know, the original RNN suffers from short-term memory and the vanishing gradient issues in the process of backpropagation. If the sequence is long enough, it is difficult to pass information from the earlier time step to the end step. However, if the gradient value becomes extremely small, then the neural network learns nothing.

Therefore, the core thought of GRU is to allow the model to transmit information completely to the cells in order to avoid the disappearance of gradient chains caused by the length of the sequence.

Moreover, Sent2Vec whose algorithm mechanism is the same as Doc2Vec, will be implemented to the embedded layer of GRU to realise a new algorithm mechanism in this experiment. To study the GRU algorithm mechanism (refer to (Chung et al., 2014)), we split the model structure into four parts.

The Structure of One GRU Cell

This part is the process of implementing data encoding.

Reset Gate: We start with the calculation method of the reset gate r_t (seeing Equation 3.17). Basically, at time step t, it is used to decide how much past information for the model should be dropped.

$$r_t = \sigma(W_t \cdot [x_t, h_{t-1}]) \tag{3.17}$$

with:

- x_t: The input at the time t.
- h_{t-1}: At the last time $t - 1$, the information that is remembered in the hidden layer.
- W_t: The weight matrices of Reset Gate.

Update Gate: The update gate z_t for time step t using Equation 3.18.

$$z_t = \sigma(W_z \cdot [x_t, h_{t-1}]) \tag{3.18}$$

with:

- W_z: The weight matrices of Update Gate.

The purpose of the update gate is to decide what information needs to be discarded or added. When the input x_t of the current time t enters the network cell, it is with the output h_{t-1} of the previous time t – 1, multiplied by its weight $W(z)$, and then applies the Sigmoid Activation function to compress the results between 0 and 1, indicating how much of the previous information is discarded. Through copying all previous information, the update gate helps the model eliminate the risk of vanishing gradient issues.

Current Memory Content: The formula of $r_t * h_{t-1}$ in this step determines what to remove from the previous time step. The calculation process is an element-wise multiplication between the previous output h_{t-1} and the value of reset gate function r_t with a weight W, and then the nonlinear activation function Tanh (Equation 3.19) is applied to the scope convergence of the input x_t.

$$\tilde{h_t} = tanh(W \cdot [r_t * h_{t-1}, x_t]) \tag{3.19}$$

with:

- W: The weight matrices of Current Memory Content.

Final Memory at Current Time Step: In the final step (Equation 3.20), the neural network adds the information $(1 - z_t) * h_{t-1}$ retained from the output at the previous moment, and the new information learned from the current moment $z_t * \tilde{h_t}$ to memory. In this way, the output value h_t at time t can be obtained and passed into the next network cell.

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h_t} \tag{3.20}$$

After a series of calculation steps, we can discover how the GRU uses the update and reset gates to store and filter information. This process acceptably eliminates the problem of disappearing gradients because the model knows the new input each time instead of retaining the relevant information and passing it to the next time step of the network, which also solves the problem of short-term memory. Therefore, after training appropriately, the model can execute well even in a complicated situation.

Decoding Process

GRU decoding is essentially a multi-classification prediction process. The formula can be expressed as Equation 3.21.

$$\begin{aligned} h_t &= f(W * h_{t-1}) \\ y_t &= softmax(W_h \cdot h_t) \end{aligned} \tag{3.21}$$

with:

- y_t: The output at time t.

The Related Functions

Sigmoid Activation (σ)**:** The Sigmoid Activation (Figure 3.7) is to compress the value range between 0 and 1, which implements updating or forgetting data (Yonaba, Anctil & Fortin, 2010). Specifically, any number times 0 equals 0, which means that the value of information disappears or is 'forgotten'. Also, any number times 1 is the original value, so it stays the same or called 'holds'. Values of 0 to 1 characterise how much information is selected to be forgotten or retained. Through the neural networks, GRU cells learn which unimportant data will be forgotten, which data is important and will be retained.

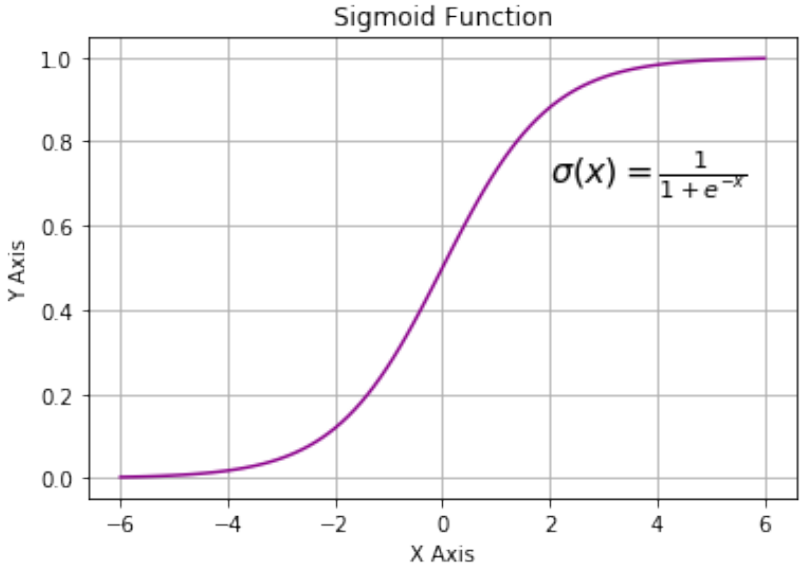


Figure 3.7: Sigmode Activation

Tanh Activation(tanh): A Tanh Activation function[3] (Figure 3.8) is similar to a Sigmoid Activation function. The difference is that it compresses values between -1 and 1. To be specific, when vectors go through neural networks because of various mathematical operations, they may undergo many transformations, that may result in some values exploding. In this case, the Tanh Activation is used to help ensure the values stay between -1 and 1 to go through the networks.

Loss Function: In order to avoid the problem of the learning rate falling in the process caused by the dispersion of the gradient, we used Cross-entropy (Equation 3.22) as the loss function to solve it. Cross-entropy loss is usually used to measure the performance of the classification model. The output of the classification model is a probability value between 0 and 1. When the prediction probability deviates from the actual value, the Cross-entropy loss increases. Therefore, the loss of a perfect model is zero (Murphy, 2012).

[3]https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

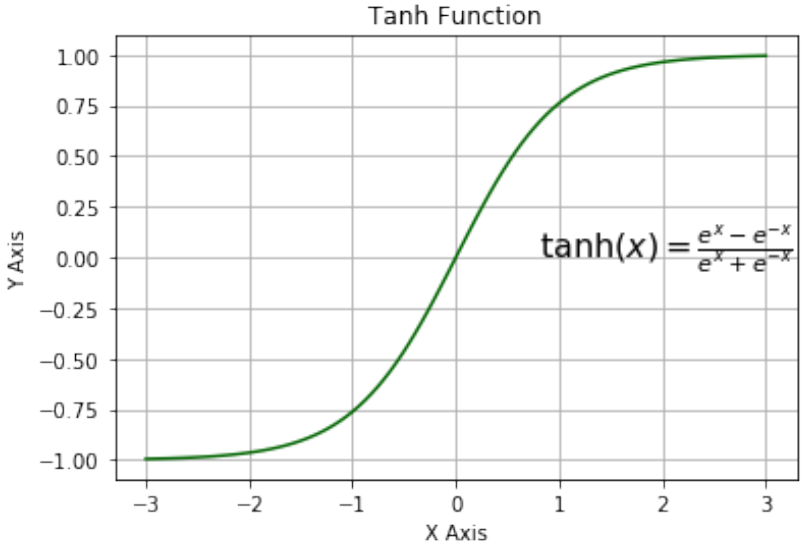


Figure 3.8: Tanh Activation

$$max_\theta \frac{1}{N} \sum_{n=1}^{N} \log p\theta(y^{(n)}|x^{(n)}) \tag{3.22}$$

Attention Mechanism: When Bahdanau, Cho and Bengio (2014) introduced a Soft Attention Model, the attention mechanism began to play an increasingly important role in the NLP field, for it allows the neural network to know how to ignore the noise and focus on related things (as shown in Equation 3.23). It uses the probability of the output as a weight reference.

$$c_i = \sum_{j=1}^{T_x} y_{ij} h_j \tag{3.23}$$

3.3 Summary

This chapter presented the design ideas and system framework of an intelligent E-mail management system based on three methods, which belong to the information retrieval method and the information generation method. In addition, relevant algorithms and calculation principles related to TF-IDF,

Doc2Vec and GRU models are studied in depth. Understanding these state-of-the-art algorithms can help us successfully implement the core functions of this experiment in the next chapter.

Chapter 4

Implementation

In this chapter, we present the details of the implementation process. The following details the organisation of this chapter. Section 4.1 illustrates the data preparation process and operational details. Data processing is a vital part of the whole project implementation process. From Section 4.2 to Section 4.4, we mostly describe the implementation process of these three models and adjust the different parameters to determine the performance effect. In Section 4.5, a simple client is designed to display and visualise the effect of our project, which is of great significance for us to understand and analyse the experimental results more intuitively. Section 4.6 is a short summary of the chapter.

4.1 Data Preparing

The preparation and processing of training data is the first step in project development. As far as we know, the quality of the data will directly affect the results of the project. In order to obtain the best results based on the limited resources, we have invested significant time and effort to prepare ideal data for this experiment. In this section, we specifically focus on data collection and data

pre-processing.

4.1.1 Data Collection

In Chapter 2, we looked at training corpora from other studies that could be used by us. We found that most of the E-mail datasets are small-scale datasets collected by researchers, but there are also some large E-mail datasets, such as Avocado Research E-mail (L. Yang et al., 2017), Yahoo! E-Mail dataset (Kooti, Aiello, Grbovic, Lerman & Mantrach, 2015; Di Castro, Karnin, Lewin-Eytan & Maarek, 2016), and Gmail dataset (Kannan et al., 2016) that are used for their own companies' research. Due to privacy or copyright restrictions, most datasets are not publicly available, or some datasets need to be purchased.

Fortunately, there is also a 'real' public free E-mail dataset, the Enron E-mail Dataset[1]. It provides us with the ability to perform research work on E-mail. This E-mail dataset was collected and prepared by the CALO Project from approximately 150 executives of Enron, which was published online by the Federal Energy Regulatory Commission during the investigation of Enron's economic problems. This dataset contains no attachments, some sensitive messages have been deleted, and some recipients or addresses have been replaced with other information (such as user@enron.com). However, this is the best training dataset available to us. After selecting the dataset, we need to fully understand the data, including the content and the structure of the data, and then deal with it to meet our model requirements.

[1]https://www.cs.cmu.edu/ enron/

4.1.2 Data Pre-processing

Data Pre-processing Flowchart: Figure 4.1 shows the workflow of the data pre-processing. The purpose of this process is to produce a generic dataset for our three models.

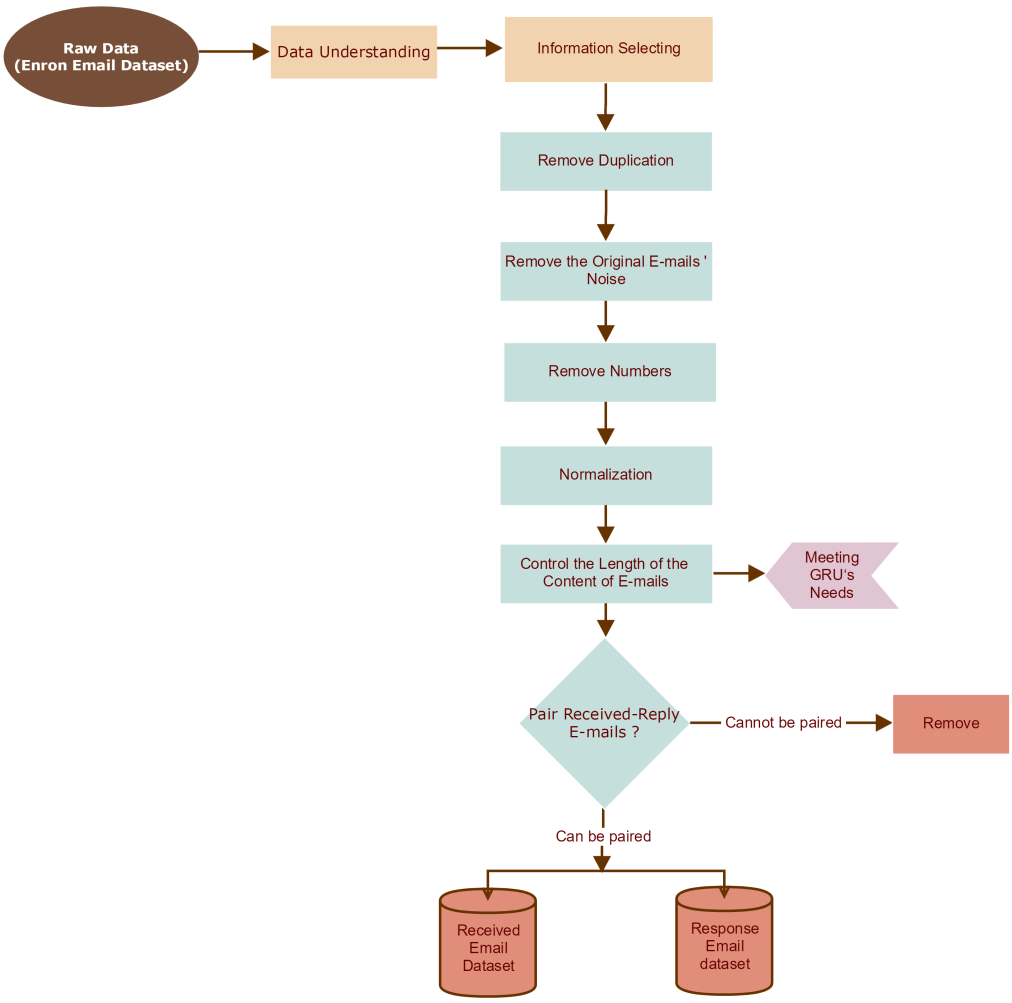


Figure 4.1: Data Processing Flowchart

Data Understanding and Information selection: The understanding of the dataset is our experimental foundation for data processing. The dataset that we downloaded was composed of a large number of TXT files, which means that every letter in their inbox and outbox of these 150 employees was one TXT file.

The raw data sample is shown in appendix A.

The first step was to transfer all the E-mails into a CSV file with the information we needed. Each column contains the sending date, sender E-mail address, receiver E-mail address, E-mail subject and E-mail content. It is easy to experiment using a unified data processing method. The sorted data is shown in Figure 4.2.

	Date	From	To	Subject	Content
0	2001-05-14 23:39:00	frozenset({'phillip.allen@enron.com'})	frozenset({'tim.belden@enron.com'})	NaN	Here is our forecast
1	2001-05-04 20:51:00	frozenset({'phillip.allen@enron.com'})	frozenset({'john.lavorato@enron.com'})	Re:	Traveling to have a business meeting takes the...
2	2000-10-18 10:00:00	frozenset({'phillip.allen@enron.com'})	frozenset({'leah.arsdall@enron.com'})	Re: test	test successful. way to go!!!
3	2000-10-23 13:13:00	frozenset({'phillip.allen@enron.com'})	frozenset({'randall.gay@enron.com'})	NaN	Randy, Can you send me a schedule of the salar...
4	2000-08-31 12:07:00	frozenset({'phillip.allen@enron.com'})	frozenset({'greg.piper@enron.com'})	Re: Hello	Let's shoot for Tuesday at 11:45.
5	2000-08-31 11:17:00	frozenset({'phillip.allen@enron.com'})	frozenset({'greg.piper@enron.com'})	Re: Hello	Greg, How about either next Tuesday or Thursda...

	Date	From	To	Subject	Content
count	517401	517401	495554	498214	516245
unique	224122	20328	55385	159289	241859
top	2001-06-27 23:02:00	frozenset({'kay.mann@enron.com'})	frozenset({'pete.davis@enron.com'})	RE:	The request has been completed with all resour...
freq	1118	16735	9155	6477	148

Figure 4.2: Dataset Overview

```
df2['Content'][9]
```

'---------- Forwarded by Phillip K Allen/HOU/ECT on 10/16/2000 01:42 PM ---------- "Buckner, Buck" <buck.buckner@honeywell.com> on 10/12/2000 01:12:21 PM To: "\'Pallen@Enron.com\'" <Pallen@Enron.com> cc: Subject: FW: fixed forward or other Collar floor gas price terms Phillip, > As discussed during our phone conversation, In a Parallon 75 microturbine > power generation deal for a national accounts customer, I am developing a > proposal to sell power to customer at fixed or collar/floor price. To do > so I need a corresponding term gas price for same. Microturbine is an > onsite generation product developed by Honeywell to generate electricity > on customer site (degen). using natural gas. In doing so, I need your > best fixed price forward gas price deal for 1, 3, 5, 7 and 10 years for > annual/seasonal supply to microturbines to generate fixed kWh for > customer. We have the opportunity to sell customer kWh \'s using > microturbine or sell them turbines themselves. kWh deal must have limited/ > no risk forward gas price to make deal work. Therein comes Sempra energy > gas trading, truly you. > > We are proposing installing 180 - 240 units across a large number of > stores (60-100) in San Diego. > Store number varies because of installation hurdles face at small percent. > > For 6-8 hours a day Microturbine run time: > Gas requirement for 180 microturbines 227 - 302 MMcf per year > Gas requirement for 240 microturbines 302 - 403 MMcf per year > > Gas will likely be consumed from May through September, during peak > electric period. > Gas price required: Burnertip price behind (LDC) San Diego Gas & Electric > Need detail breakout of commodity and transport cost (firm or > interruptible). > > Should you have additional questions, give me a call. > Let me assure you, this is real deal!! > > Buck Buckner, P.E., MBA > Manager, Business Development and Planning > Big Box Retail Sales > Honeywell Power Systems, Inc. > 8725 Pan American Frwy > Albuquerque, NM 87113 > 505-798-6424 > 505-798-6050x > 505-220-4129 > 888/501-3145 >'

```
df2['Content'][8]
```

"1. login: pallen pw: ke9davis I don't think these are required by the ISP 2. static IP address IP: 64.216.90.105 Sub: 255.255.255.248 gate: 64.216.90.110 DNS: 151.164.1.8 3. Company: 0413 RC: 105891"

```
df2['Content'][112]
```

'Ina, I keep getting these security requests that I cannot approve. Please take care of this. Phillip ---------- Forwarded by Phillip K Allen/HOU/ECT on 08/08/2000 04:28 PM ---------- ARSystem@ect.enron.com on 08/08/2000 07:17:38 AM To: phillip.k.allen@enron.com cc: Subject: Request Submitted: Access Request for frank.ermis@enron.com Please review and act upon this request. You have received this email because the requester specified you as their Manager. Please click =phillip.k.allen@enron.com to approve th Request ID : 000000000001282 Request Create Date : 8/8/00 9:15:59 AM Requested For : frank.ermis@enron.com Resource Name : Market Data Telerate Basic Energy Resource Type : Applications'

Figure 4.3: Raw E-mail Content

Removing Duplication: From the data overview (Figure 4.2), it can be

seen that the total number rows in the dataset is 517,401, among which there are 498,214 messages with titles, and the highest frequency of occurrence is 'RE:', appearing 6,477 times. Meanwhile, in the content column of the E-mail dataset, we found that nearly half of the content is duplicate. The reason is that if employee A sends an E-mail to employee B, the E-mail in A's outbox will be the same as the E-mail received in B's inbox. Therefore, we realised that duplicate messages need to be removed entirely leaving unique messages for further processing. The process of this step is to compare the title and content at the same time to avoid the possibility that the content is the same but not sent by the same person.

Removing Noise: We randomly read the contents of three E-mails (Figure 4.3) and found much noise in our dataset. For example, many numbers, symbols, and E-mails with past responses, and such useless information will lead to poor model training results. Finally, we obtained a relatively clean dataset by removing this noise.

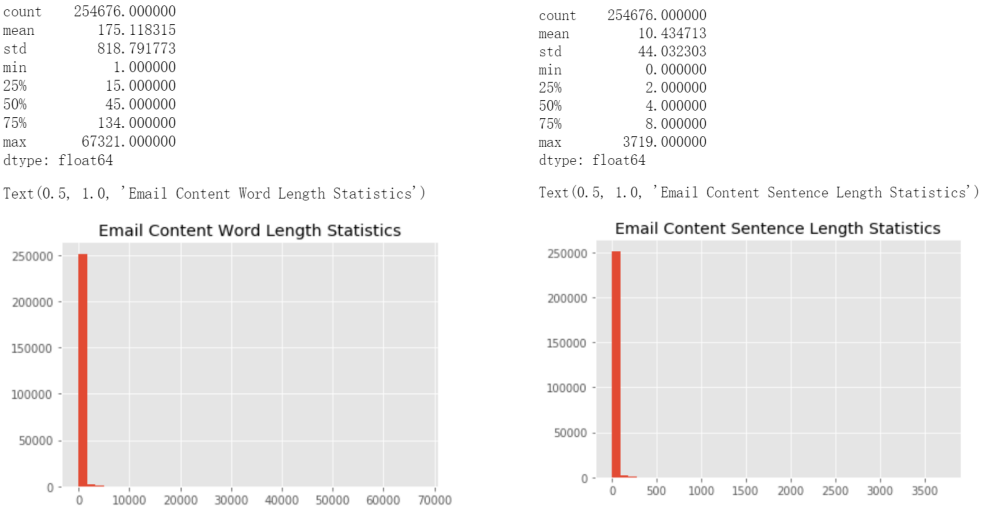


Figure 4.4: E-mail Content Length Statistics

Controlling The Length of E-mail Contents: After cleaning up the useless information in the E-mail content, our dataset still contained a large number

of empty E-mails and excessively long E-mails, which is not conducive to our training of the sequence model (GRU model). According to the statistics of E-mail content length (shown in Figure 4.4), approximately 75% of the 254,676 processed E-mails contain less than 150 words or 10 sentences. Therefore, considering the requirements of the models we chose and to avoid significantly affecting the size of the E-mail dataset, we decided to remove E-mails that were empty or longer than 30 sentences.

Pairing Received - Response E-mails: The vital role of this step is self-evident, as it is the basic form for three models training. The method used in the E-mail pairing process is that we logically filtered messages using the sending time, title, and the name of the senders and receivers, the code sample is shown in appendix B.1. We then selected the E-mails that most likely relate to each other and placed them into two databases (Received E-mail Dataset and Response E-mail Dataset) separately. The result after processing is shown in Figure 4.5.

	Received	Response
0	Randy Can you send me a schedule of the salary...	Phillip I m working on getting the official li...
1	resumes of whom ?	The commercial support people that you and Hun...
2	Christy I read these points and they definitel...	Phillip To the extent that we can give Chair H...
3	Phillip I have a meeting tomorrow morning with...	Yes you can use this chart . Does it make sense ?
4	Phillip Which one should I do the x is half th...	I like the cedar t version better . Why don t ...
5	Phillip The Social Security and Medicare tax h...	Thanks .
6	Phillip This is the candidate I spoke with you...	Adrianne I cannot download his resume . Please...
7	Phillip Lets try this one . .Thanks ! Adrianne .	Left message and sent email . I will let you k...
8	Phillip I need to get the contract for Galaxy ...	Greg I would rather sign and fax copies of the...
9	Phillip I am waiting to get info . on two more...	Jeff Can you resend the info on the three prop...
10	Bernie Good Morning . I hope all is well . I h...	Kirk I ve added my comments . See attached Giv...

Figure 4.5: Received-Response E-mail Pairs

The raw data was successfully processed for our three models after a series

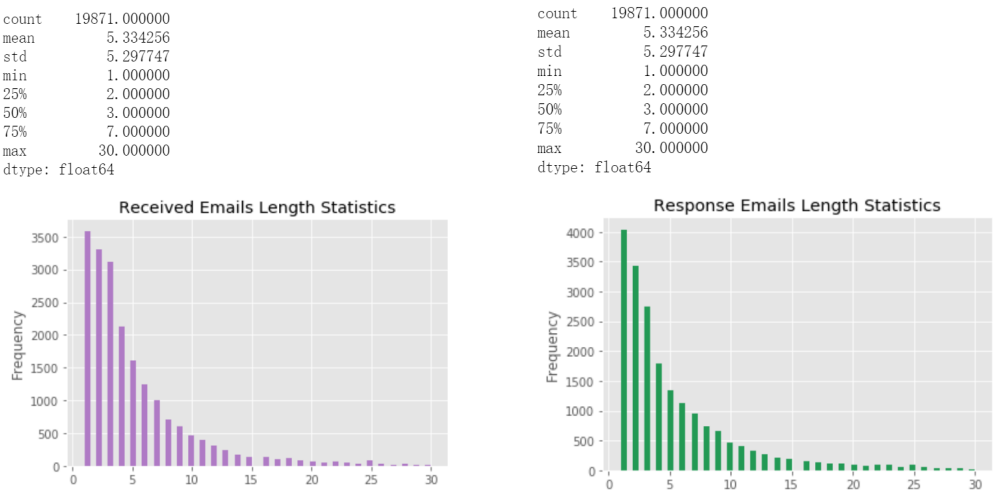


Figure 4.6: Processed E-mail Content Length Statistics

of processing. Finally, we got a total of 19,871 pairs of E-mails, each with a content-length range of 1 to 30 sentences (as shown in Figure 4.6). Next, we can entered the model training stage.

4.2 TF-IDF Based Model

Figure 4.7 presents the workflow of specialised text processing and model training for TF-IDF modelling.

4.2.1 Data Processing for TF-IDF

Based on the general dataset processing in the previous section, we needed to process further the data to adapt to the different models.

Word Lowercasing: In text processing, if the first case of the same word is different, it is treated as two different words. To avoid this, we expressed all the letters in lowercase. The NLTK[2] library helped us achieve this process outstandingly.

[2]https://www.nltk.org/

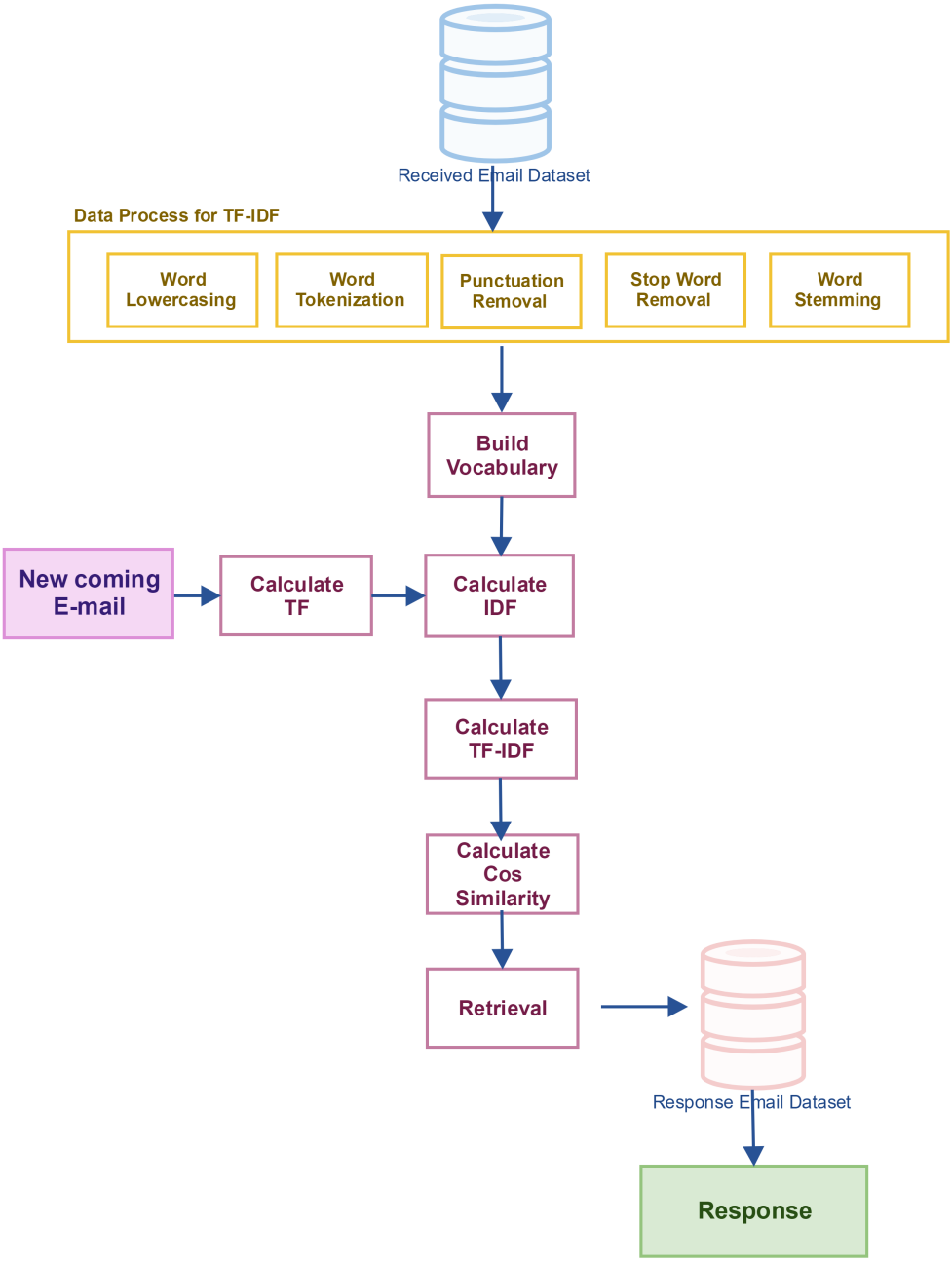


Figure 4.7: TF-IDF Based Model

Word Tokenization: The next step was to split the document into a word list and remove the punctuation without real semantics. We used the word_tokenize() method, which is the key to stemming words and stop-word removing and to

facilitate the vocabulary for corpus.

Stop Word Removal: The term 'stop words' or related term 'stop list' was created by Hans Peter Luhn (1960), the father of information retrieval. It usually refers to the commonly used words or phrases in a document without any information that expresses important meanings, such as auxiliary words, relative pronouns, letters and punctuation marks.

In data or text processing, it is usually necessary to remove stop words to improve efficiency and save space and time for processing big data. In practice, the list of stop words is dynamic and needs to be selected based on the specific situation and the functions to be implemented. In this experiment, the stop word list we designed for the three models contains 422 words.

Word Stemming: The introduction of stem analysis in linguistic morphology into information retrieval techniques is a milestone in NLP. The core content of word stemming is to convert words into their corresponding similar forms like their stems or roots and then to merge all the related words as long as sufficient (Porter, 1980). Dr. Porter won the Tony Kent Strix award in 2000 for this algorithm.

In text processing, there are many words from the same root word, such as 'love, loved, loving, lovely, lover, lovingly'. They have similar meanings, but these different forms unnecessarily increase the dimensions of the word vector space. The stemming process was done by converting these words into the unified source of 'love', aiming to help reduce the dimensions of the term matrix and improve the efficiency of the classifier.

After deep processing, the data we obtained was more suitable for modelling TF-IDF.

4.2.2 TF-IDF Modelling

In the modelling process, we used a Python programming language library, Scikit-learn library[3], which is an efficient tool for machine learning and data analysis, as it encapsulates the TF-IDF algorithm and Cosine Similarity algorithm in the modelling process.

Building Vocabulary: First of all, we used the function of feature extraction, CountVectorizer in Scikit-learn to count the number of times each word appears. It is an easy way to tag a collection of text documents, index each known word, and encode new documents using the index set as well.

The fit() function can index one or more documents, and each document can be encoded as a vector by calling the transform() function. Eventually, it converts the document into a coding vector whose length is the number of indexes carrying the information about each word in the document.

The shape of the vector we obtained is (19871, 20300), which means that there are 19,871 documents in training dataset, and there are 20,300 unique words without stop words in the vocabulary of these documents.

Calculating IDF and TF-IDF Scores: After building corpus vocabulary and a sparse matrix by CountVectorizer, TfidfTransformer was used to count the IDF value and TF-IDF value of each word in the given corpus as well as a new document. It should be noted that if some specific words do not appear in the training corpus, their TF-IDF values can be 0.

Calculating Similarity Scores: Euclidean Normalisation is also applied in the Scikit-learn library. For a new document, we only needed to calculate its TF-IDF vector related to the entire corpus. By dot product with all other document vectors, we could calculate the cosine distance between a new document and all

[3]https://scikit-learn.org/stable/

documents in the corpus, so that we could sort similar documents for retrieval.

4.2.3 The Main Parameters of TF-IDF

Each model involves many parameters and understanding the meaning of each parameter helped us optimise the model. Our TF-IDF model was mainly affected by the following three parameters.

Max_df: The value of max_df is a given threshold that words will be ignored if their frequency occurrence is higher than this value. For example, setting max_df to 75% means that a word will be dropped if it appears in more than 75% of the documents during modelling, and such words are considered stop words that contribute very little to the meaning of the document.

Ngram_range: This is the window range size that was set to study the relationship between words in the window range. From the lower(min_n) to the upper(max_n), all values of 'n' will be used. The larger the 'n' value range is, the more computing time and storage space are required, but the resulting quality may be improved. Therefore, we needed to consider the setting of this parameter reasonably.

Input: In terms of the input passed to the model, we redesigned the training dataset into three different forms: 'E-mail Subject', 'E-mail Content' and 'E-mail Subject plus Content'. Based on preliminary experimental results, it made no sense in practice to extract similar information using the subjects of the E-mails. Therefore, we only selected the content of E-mail as the input of TF-IDF model training.

4.3 Doc2Vec Based Model

Figure 4.8 interprets the training workflow based on the Doc2Vec model. As for data processing, it is similar to the process of TF-IDF. Hence, we demonstrated it directly from the modelling section.

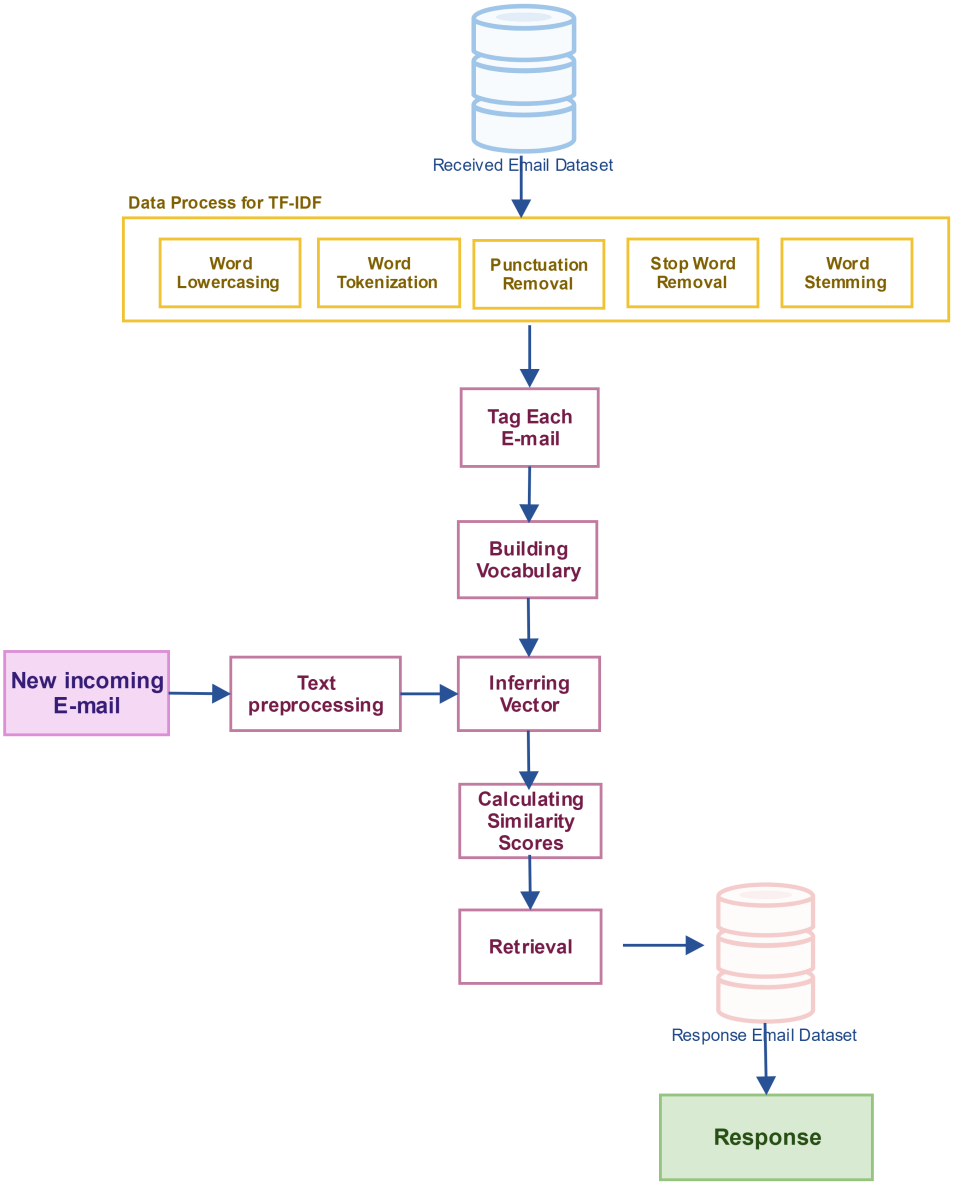


Figure 4.8: Doc2Vec Based Model

4.3.1 Doc2Vec Based Modelling

Gensim[4] is an open source Python library for automatically extracting semantic topics from documents. It supports a variety of theme-model algorithms, including TF-IDF, LSA, LDA, and Word2Vec. It also supports stream training and provides API interfaces for common tasks such as similarity calculation and information retrieval. Therefore, we chose the Gensim platform to train our Doc2Vec model.

Tag Each E-mail: Before training the model, we needed to tag each document (E-mail) in the corpus. To get the correlations between words in the document, we embedded the document's ID with the words in the document.

Building Vocabulary: In the Gensim library, the function Word2VecVocab can create a vocabulary for the model. In addition to recording all unique words, this object provides additional functionality such as creating a Huffman tree (the more frequent the words, the closer they are to the roots of the tree), to eliminate uncommon words.

Inferring Vector: After training the model, the infer_vector function can infer vectors for new documents.

Calculating Similarity Scores: A built-in most_similar module in Gensim is used to calculate the similarity of document vectors.

4.3.2 The Main Parameters of Doc2Vec

Among the numerous parameters, our experimental Doc2Vec model mainly depended on the following five parameters.

Vector_size: This is the dimension of the eigenvector we specified, which is determined by the size of the training corpus. Our training dataset contains

[4]https://radimrehurek.com/gensim/

19,871 E-mails (not large), so we set the parameter values to 100, 150, and 200, and the results of the comparison will be shown in the next chapter.

Dm: Dm is the choice of training mode. 'dm = 0' means we chose PV-DBOW, while 'dm = 1' means a choice of PV-DM.

Hs: This is the choice of the classification algorithm. If the value is 1, the Historical Softmax method is adopted, else if the value is 0 (the default value), Negative Sampling method is used.

Window: This represents the window size of the maximum distance between the current word and the predicted word in the sentence when we train the model, that is, the context range that affects the prediction.

Epochs: Epochs are also called iterations. In theory, more iterations can get better training results, but to some extent, more iterations may also lead to over-fitting problems. The iteration value was matched to the size of the dataset so that we could confirm the value within an optimal range.

The above parameters are the most critical factors affecting the results of the Doc2Vec model training. We will evaluate the optimal parameters in the next chapter.

4.4 GRU-Sent2Vec Hybrid Model

Figure 4.9 illustrates the training workflow for GRU-Sent2Vec hybrid model. The design idea of this model is to combine the information generation model with the information retrieval model, using both GRU and Sent2Vec technologies. In this design, on the one hand, we expected to generate long-text reply E-mails, on the other side, we considered that our training data set was too small for the deep neural network. It is a novel attempt for an intelligent generated E-mail system.

As for selecting a suitable platform, frameworks for deep learning have sprung up in universities and companies such as TensorFlow[5], Caffe[6], Theano[7] and Keras[8]. Among these technologies, PyTorch[9] is a simple, elegant and efficient framework. The design of PyTorch follows three levels of low-to-high abstraction from tensor to variable (autograd) to nn.Module, which represents high-dimensional arrays (tensor), auto-derivatives (variables), and neural networks (layers/modules). These three abstractions are closely related and can be modified and manipulated simultaneously. Therefore, our design of the GRU model used PyTorch as the implementation framework.

4.4.1 Data Processing for GRU-Sent2Vec Hybrid Model

We had to put more effort into the data processing used for the GRU-Sent2Vec hybrid model training than the TF-IDF and Doc2Vec models because we intended to input sentences as sequences instead of words. Before modelling, we needed to separate each sentence in order to generate a sentence dictionary that was related to the unique index, and then apply the Sent2Vec model to train the sentence vectors, which would be used as an embedding layer of the GRU model.

Text Standardisation: In the initial data processing phase, we have done much work on data cleansing. This step is to ensure the standardisation for the next process, and the received and response E-mail dataset will be used as input to train the model at the same time.

Sentence Tokenisation: Tokenising sentences is the basis for the next step to establish matches with indexes and sentence vectors. We used three sentence

[5]https://www.tensorflow.org/
[6]https://caffe.berkeleyvision.org/
[7]http://deeplearning.net/software/theano/
[8]https://keras.io/
[9]https://pytorch.org/

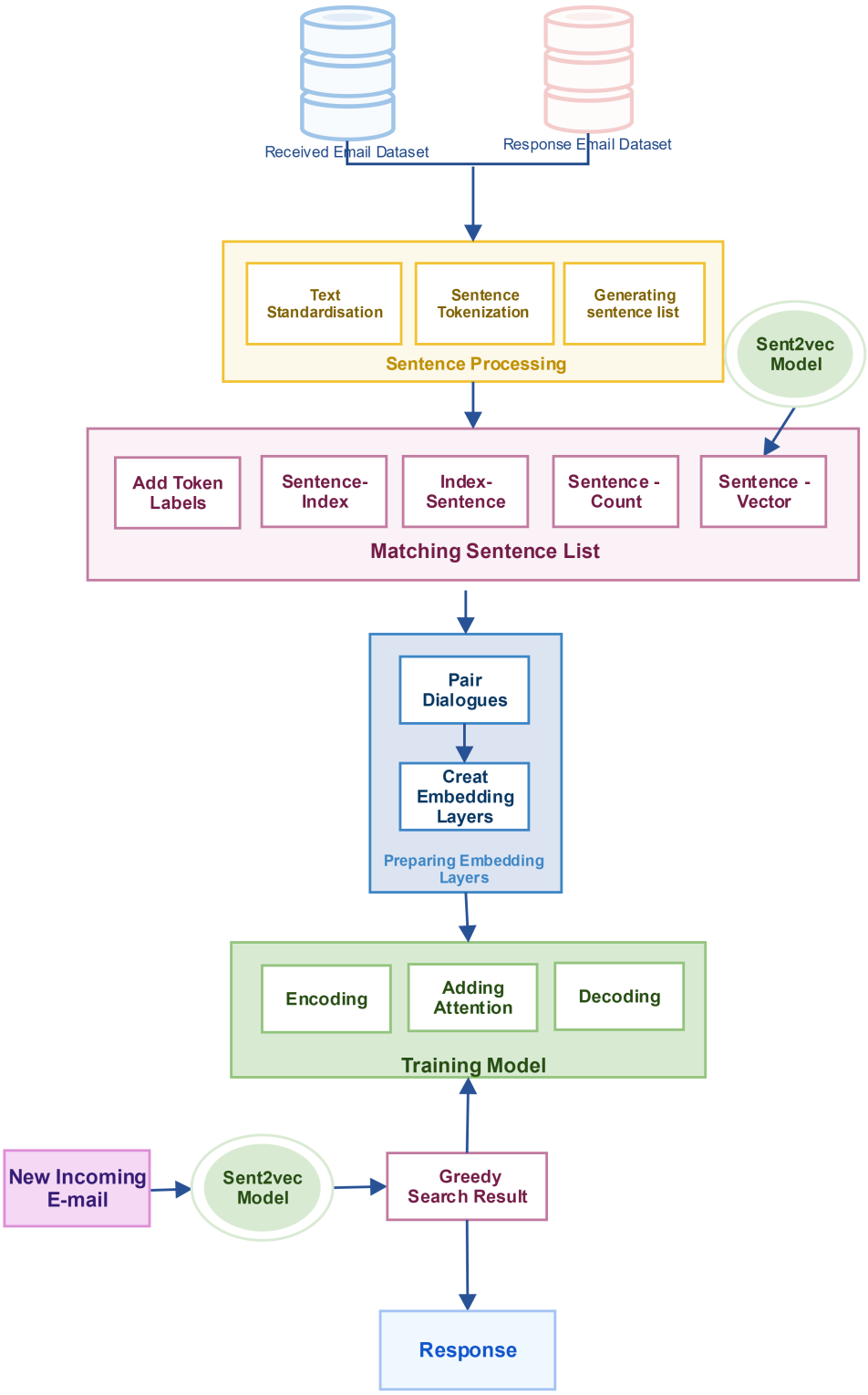


Figure 4.9: GRU-Sent2Vec Hybrid Model

separators, '. ? !', as references to split documents into sentences. This is the point of the previous step of standardisation, removing other symbols, and ensuring that the last sentence of the document ends in '.'. The function of sent_tokenize()[10] provided by NLTK can split the text into sentences.

Generating Sentence List: We created a list of all the sentences from the two datasets. There is a total number of 232,948 sentences in 39,742 E-mails. For these sentences, we built a sentence-list by removing duplicates. Each sentence in this list is unique, making a total of 99,336 sentences.

4.4.2 Building Up Sent2Vec Model

The model of Sent2Vec is built to train the vector of each sentence. The process and principles of training are similar to Doc2Vec whereas, the difference is that the ID of the document is replaced with the ID of the sentence for training. It references part of the workflow of Figure 4.8.

Before training the sentences' vectors, we first defined three sentence-markers and assigned a 0-2 index order. These three markers, PAD_token, SOS_token, and EOS_token, were used for padding short documents and marking the beginning and end of a document when training in the GRU sequence model. Then we performed sentence vector training on these three markers with all the sentences in the list together.

This Sent2Vec model was also applied in subsequent processes at the step of information retrieval and matching.

[10]https://www.nltk.org/api/nltk.tokenize.html

4.4.3 Mapping Sentences and Preparing Embedding Layers

Mapping sentences: Since there is no implicit mapping of a sentence sequence to a discrete numeric space, we need to create a mapping set by mapping each unique sentence in the sentence list to its index value and vector.

The contents of the mapping set are as follows:

- sentences to indexes
- indexes to sentences
- sentences to vector

Pairing Dialogues: After a series of processing steps, such as splitting statements and matching sets, we needed to re-establish the pairs of query and response and their corresponding mapping set. We converted the documents into tensors by converting sentences to their indexes and zero-padding. After determining the maximum sentence length in a batch, the corresponding positions of other short sentences were filled with zeroes. Therefore, the shape of the tensor was a matrix (batch_size, max_length).

Preparing the Embedding Layer: Using vectors of sentences (generated by the Sent2Vec model) as a weight, we created a hidden embedding layer to train along with the input data. This was done because we believed that the sentences' vectors in an embedded hidden layer carry semantic association information between sentences. A sample of the code is shown in appendix B.2.

4.4.4 Training Model

Encoding: After transforming the sentence index into a sentence embedding vector, we mapped each sentence into a 150-dimensional feature space. We

packed and unpacked padding using nn.utils.rnn.pack_padded_sequence and nn.utils.rnn.pad_packed_sequence to pass the sequencing batch to the GRU module and return the output and final hidden state.

Decoding and Adding Attention: The decoder uses the context vector and internal hidden state to generate the next sentence in the sequence, which continues to generate sentences until the output EOS_token, which indicates the end of the document. To improve the decoder's capability and prevent the loss of valuable information, we added an attention layer to allow the decoder to focus on certain parts of the input sequence.

Training procedure: In order to achieve better convergence, we used the teacher forcing method in iterative training. Specifically, we used the current target sentence as the next input to the decoder instead of the decoder's current guess. The benefit of doing this was to accelerate the convergence of the iterative process, but a new problem was the instability of model training. We set the learning rate at 0.0001 and the number of iterations at 4000 in this experiment.

4.4.5 Model Implementation

Considering the limitation of training data, we introduced the design thought of combining information generation and information retrieval methods in the result generation stage.

New Query Processing (with Sent2Vec): Unlike the previous two models, instead of a simple normalisation process, we continued to adopt the Sent2Vec model to match similar sentences in the sentence list of the original training dataset. In the case of a small sample set, the intention was to ensure that the results generated by the GRU model were always valid.

Greedy Search and Generating Response: In the prediction generation

phase, we used a greedy search method[11], that is, for each time series, we selected the sentence with the maximum Softmax value from decoder_output. In a single time-step series, this decoding method has high efficiency and excellent performance.

4.5 Intelligent E-mail Client Implementation

In order to display the results and analyse the experimental effect intuitively, we also designed a simple client for visualisation. Figure 4.10 illustrates the user interface when a new incoming E-mail is opened.

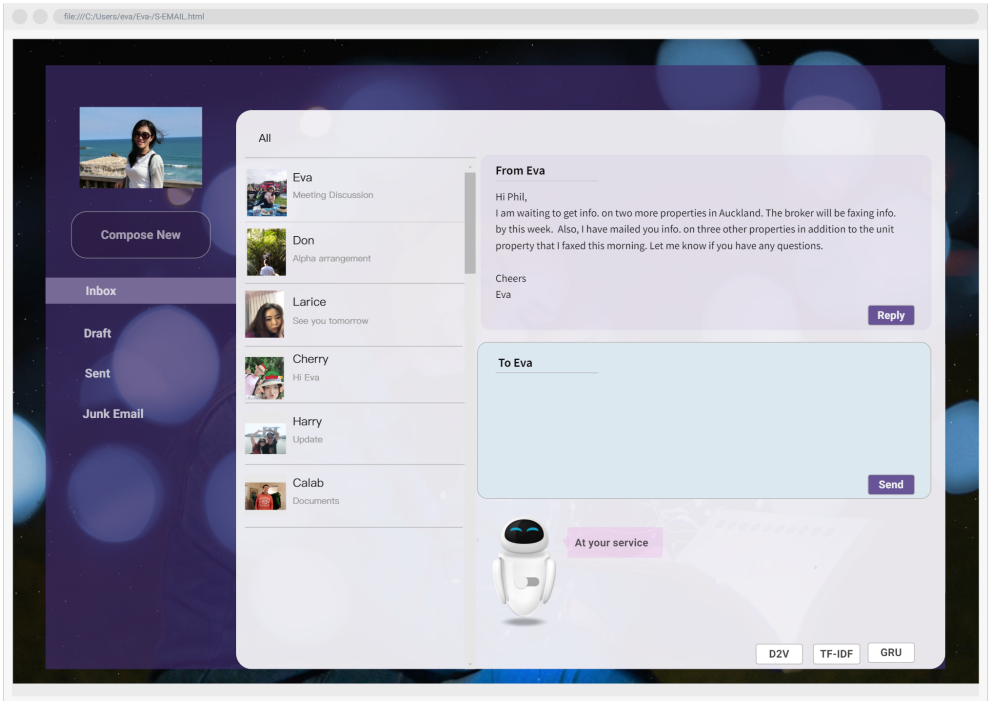


Figure 4.10: Intelligent E-mail Client[12]

Users can select either to reply directly or to use and modify intelligent suggestions. If the latter is chosen, by clicking the button on the robot in Figure

[11] https://pytorch.org/tutorials/beginner/chatbot_tutorial.html
[12] The image of the robot is adopted from the movie WALL-E

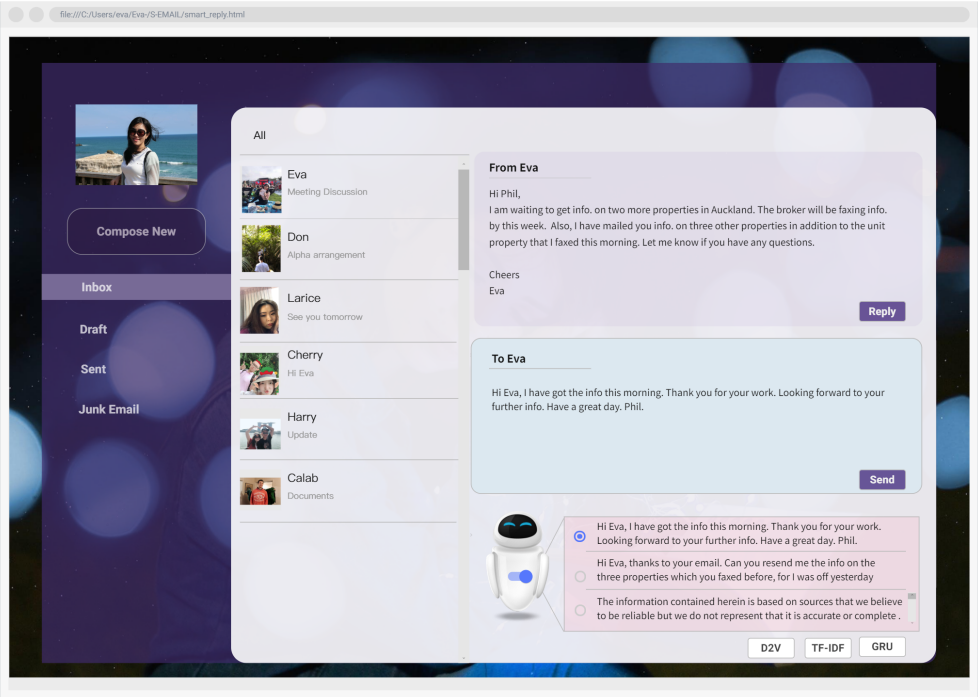


Figure 4.11: Intelligent E-mail Client - Default Response Mode

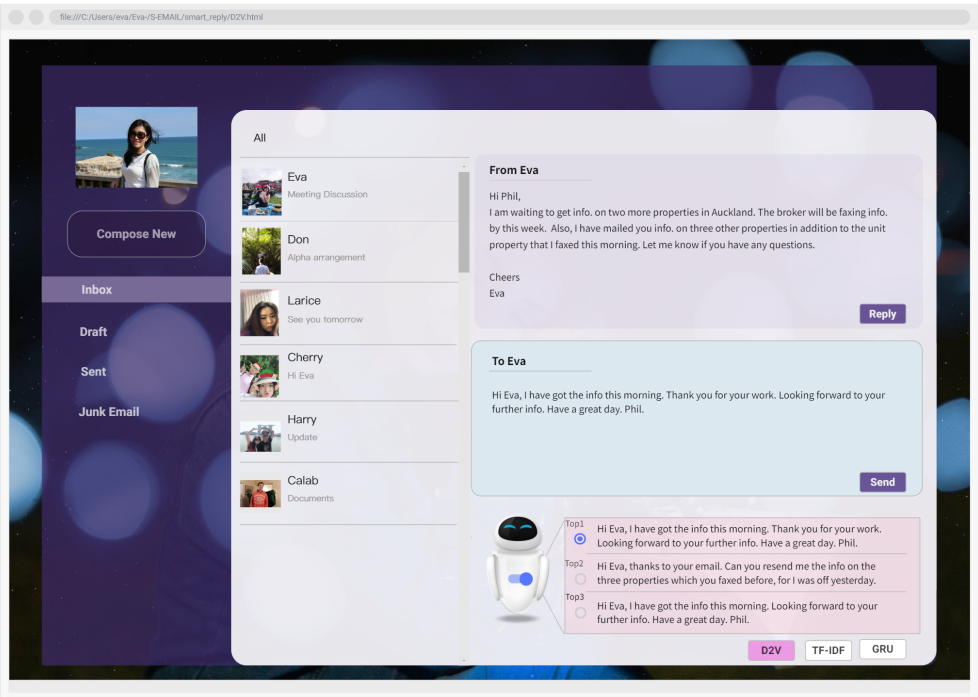


Figure 4.12: Intelligent E-mail client - Doc2Vec Mode

4.11 three suggestion responses from each of the three models are presented.

This intelligent E-mail client provides users with the functions to choose recommendations from each model. Figure 4.12 presents the top three suggestions from model Doc2Vec.

4.6 Summary

In this chapter, we introduced the whole implementation process of this experiment. First of all, the quality of experimental data is the decisive factor affecting the experimental results. The processing of training data in the early stage is the most important part before the establishment of the training models, and the processing method is based on a full understanding of the original data. Secondly, this chapter described the implementation process of the three models and the influence of their main parameters on the models. Finally, a simple, intelligent E-mail client with core functionality was designed and demonstrated.

Chapter 5

Evaluation and Discussion

In the previous chapter, we presented the design and implementation of the TF-IDF, Doc2Vec and GRU-Sent2Vec hybrid models. The main purpose of this chapter is to evaluate these three models and discuss the findings. In Section 5.1, we use the method of Recall@k (Malheiros, Moraes, Trindade & Meira, 2012) to optimise the parameters. Since the evaluation of language models has subjective factors, in order to evaluate the performance of these three models, Section 5.2 introduces the methods of designing a test dataset and a human evaluation. Then, Section 5.3 discusses the experimental results, including a comparison of the results of the effects of the three models, as well as a comparison of the results of the learning ability and subjective evaluation of the two models, TF-IDF and Doc2Vec. Section 5.4 briefly summarises the chapter.

5.1 Parameter Tuning

We use the method of call@K to self-evaluate the models. In this section, we only perform parameter tuning of the TF-IDF and Doc2Vec models. We do not consider the GRU-Sent2Vec model because of limitations, which will be

discussed later in this chapter. This process randomly selects 200 documents from the training corpus as queries, and then carries out vector inference on these documents and compares them with the vectors in the training corpus. This self-evaluation process is based on the similarity level between the same documents and the query.

We assume that the test dataset consisting of these 200 documents is some new data, and then evaluate them based on the models' response to them. The expected result is that the same document in the training set will be extracted in either the first or the first three positions for the 200 test queries. The formula is expressed as Equation 5.1:

$$\begin{aligned} \text{Recall rate@1} &= \frac{\text{Retrieved documents in top 1}}{200} \\ \text{Recall rate@3} &= \frac{\text{Retrieved documents in top 3}}{200} \end{aligned} \tag{5.1}$$

5.1.1 TF-IDF

We constructed a self-assessment of the TF-IDF model and found that the most influential parameter for this model was N-gram size (Table 5.1). The higher the value of N, the higher the accuracy (Recall rate@1). From the results, it seems not to have had much impact on the recall rate of the top three. However, in the meantime, the training time had an exponential increase.

After four rounds of training with different parameters, we also collected the similarity scores between 200 documents and the most similar documents in the training corpus (the majority are compared to themselves). As can be seen from Figure 5.1, the similarity scores are slightly different.

Considering the range of our training dataset was not large, we ignored the time consumption and selected Ngram = (1, 4) as the model parameter of TF-IDF. The results are shown in Table 5.1.

Table 5.1: Comparison of TF-IDF Parameters

TF-IDF model	Ngram	Training time	Retrieved numbers (200)			Recall rate@1	Recall rate@3
			0	1	2		
1	(1,1)	1.8910167s	173	12	5	86.5%	95%
2	(1,2)	5.3181312s	173	11	4	86.5%	94%
3	(1,3)	10.235551s	174	12	3	87%	94.5%
4	(1,4)	15.074302s	175	10	5	87.5%	95%

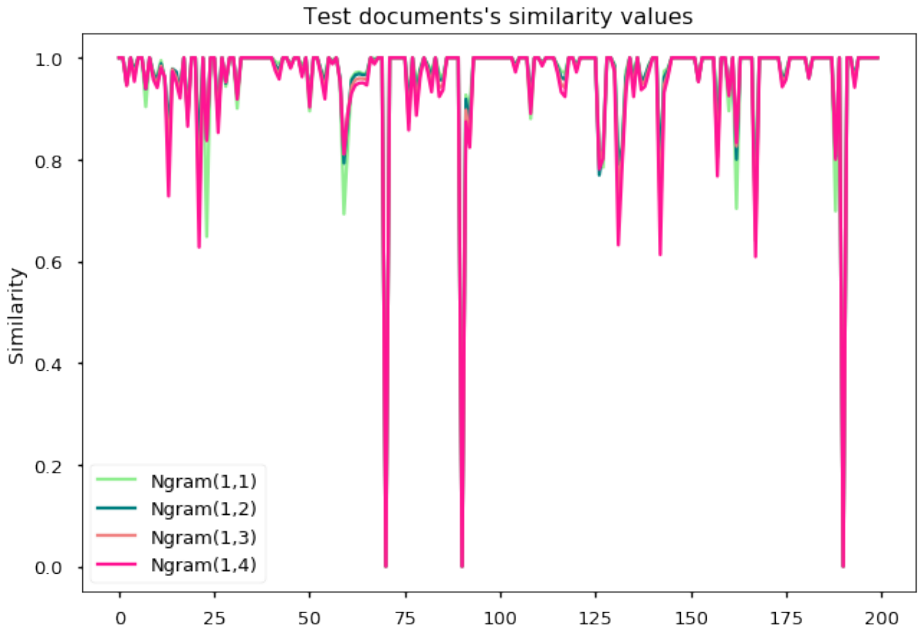


Figure 5.1: Test Results of TF-IDF Model

5.1.2 Doc2Vec

Compared with TF-IDF, the Doc2Vec model contains more parameters. As for these parameters, eleven combinations were selected to conduct eleven rounds of training for the model (M1 - M11). The results of the test are illustrated in Table 5.2 and Figure 5.2.

After all steps of the test, we found that Vector-size 150, Window value 5, and Epochs 1500, were most suitable for the training dataset. This indicated that the Negative Sampling training method was generally superior to the Hierarchical Softmax training method. In the same Negative Sampling training mode, M3 with Distributed Memory (DM) architecture had the highest top-3 recall rate,

Table 5.2: Comparison of Doc2Vec Parameters

NO.	Doc2Vec model	Vector_size	Windows	Epochs	Training time	Retrieved numbers (200)			Recall rate@1	Recall rate@3
						0	1	2		
1	NS+ DM	100	5	500	346.51621s	174	5	3	87%	91%
2	NS+ DM	100	5	1000	698.35794s	173	9	5	86.5%	93.5%
3	**NS+ DM**	100	5	1500	1013.3812s	173	11	5	86.5%	**94.5%**
4	NS+ DM	150	5	1500	1059.6635s	175	8	4	87.5%	93.5%
5	NS+ DM	200	5	1500	1019.1251s	176	7	2	88%	92.5%
6	NS+ DM	150	10	1500	1042.714s	175	9	3	87.5%	93.5%
7	NS+ DM	150	5	2000	1407.6521s	172	12	2	86%	93%
8	HS+DM	150	5	1500	1196.5312s	171	9	6	85.5%	93%
9	**NS+DBOW**	150	5	1500	850.749s	177	6	5	**88.5%**	94%
10	HS+DBOW	150	5	1500	1011.517s	174	10	4	87%	94%
11	HS +DBOW	150	10	1500	1048.288s	172	9	6	86%	93.5%

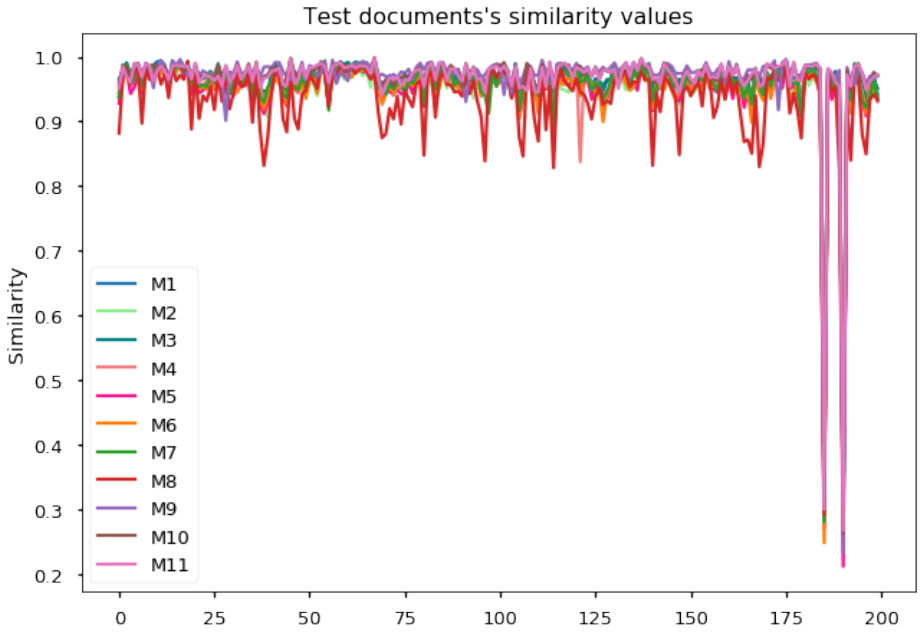


Figure 5.2: Test Results of Doc2Vec Model

and its top-1 recall rate performance was not bad, while M9 with Distributed Bag of Words (DBOW) architecture had the highest accuracy (Recall rate@1). To further confirm our parameter selection, we compared the similarity of the 200 documents tested by the two groups (Figure 5.3), and found that M9 showed a more stable level than M3. Meanwhile, the training time of M9 exhibited strong competitiveness.

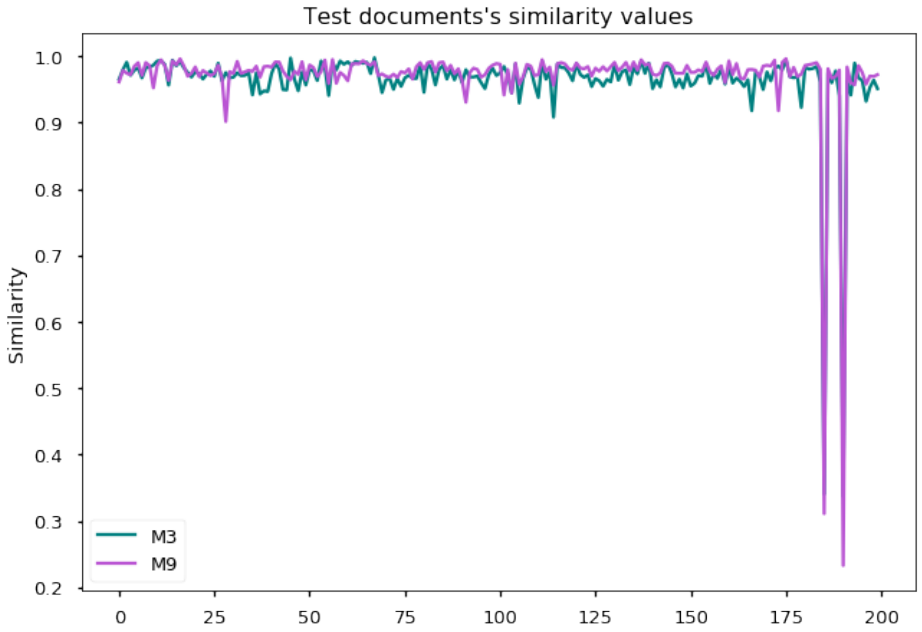


Figure 5.3: M3 VS M9 Results

5.2 Setup for Human Evaluation

Among the various methods for evaluating the effects of Natural Language Generation (NLG), there are some methods for automatic evaluation, such as NIST, BLEU, and ROUGE. Belz and Reiter (2006) believed that automated evaluation methods had great potential after comparing various assessment methods, but the best way to evaluate NLG Models is through human assessment. Meanwhile, our training dataset contained non tagged information, and we could not find a uniform standard or a unified model to evaluate these three models simultaneously. Therefore, the best way to compare our three language models was by using human evaluation.

5.2.1 Test Data Generation

Since there was not much similarity between E-mails in the dataset (Enron E-mail Dataset), it was difficult to evaluate the functional performance and

learning capabilities of the three models. In order to compare the effects of these models, we designed the test data according to the training data. Five E-mails were randomly selected as five independent topics from the received E-mail sub-dataset, and then five similar E-mails were designed according to each topic. After several rounds of training, we observed the responses of each model to similar E-mails and their ability to learn new information.

For the test E-mails we designed, we followed the rules:

- Change the entity noun (such as time, place and name)
- Change the sentence order of the paragraph
- Add some information to the E-mail
- Delete some information from the E-mail
- Change the expression of the sentence

5.2.2 Measures

We used two criteria to evaluate the performance of three models:

1. In the first case, we compared the final best responses given by the three models, respectively. In other words, we only selected the responses of the top1 related E-mails extracted by Doc2Vec based and TF-IDF based models, as well as the predicted response generated by the GRU-Sent2Vec hybrid model, to compare the final implementation results of our experiment. A sample is shown in appendix C.1.

2. In the second case, we only compared the two information retrieval models. After the first four rounds of training on new test E-mails, the fifth

designed E-mail was entered into the two models as a query. The experimental results of the two models may extract four similar E-mails with different performance. We listed the top five similar E-mails extracted by the two models with their corresponding similarity scores, and then five participants from different academic fields chose the model which gave the suggestion that matches the similarity most closely based on subjective judgement. A sample is shown in appendix C.2.

5.3 Results and Discussion

In this section, we discuss the results in two layers. At the first layer, we analyse and explain the last best suggestions of the three models given by the participants. As for the second layer, because GRU, a generative model based on deep neural network, has a strict limit on the amount of training data we constructed a comparative analysis of these two information retrieval models respectively from their learning ability of new information and the performance based on individual subjective evaluation.

5.3.1 Comparison of the Three Models

Figure 5.4 shows an example taken from topic 1 of the experimental results from the three models. As mentioned earlier, these three models use two different methods. Therefore, in the first stage of human evaluation, we made subjective selections on the final response suggestions, which were also the final results of our experiment. We selected the topic 1 response from two information retrieval models as their best response suggestions. Also, the one predictive response generated by the GRU-Sent2Vec hybrid model was treated as the object

of evaluation for this model.

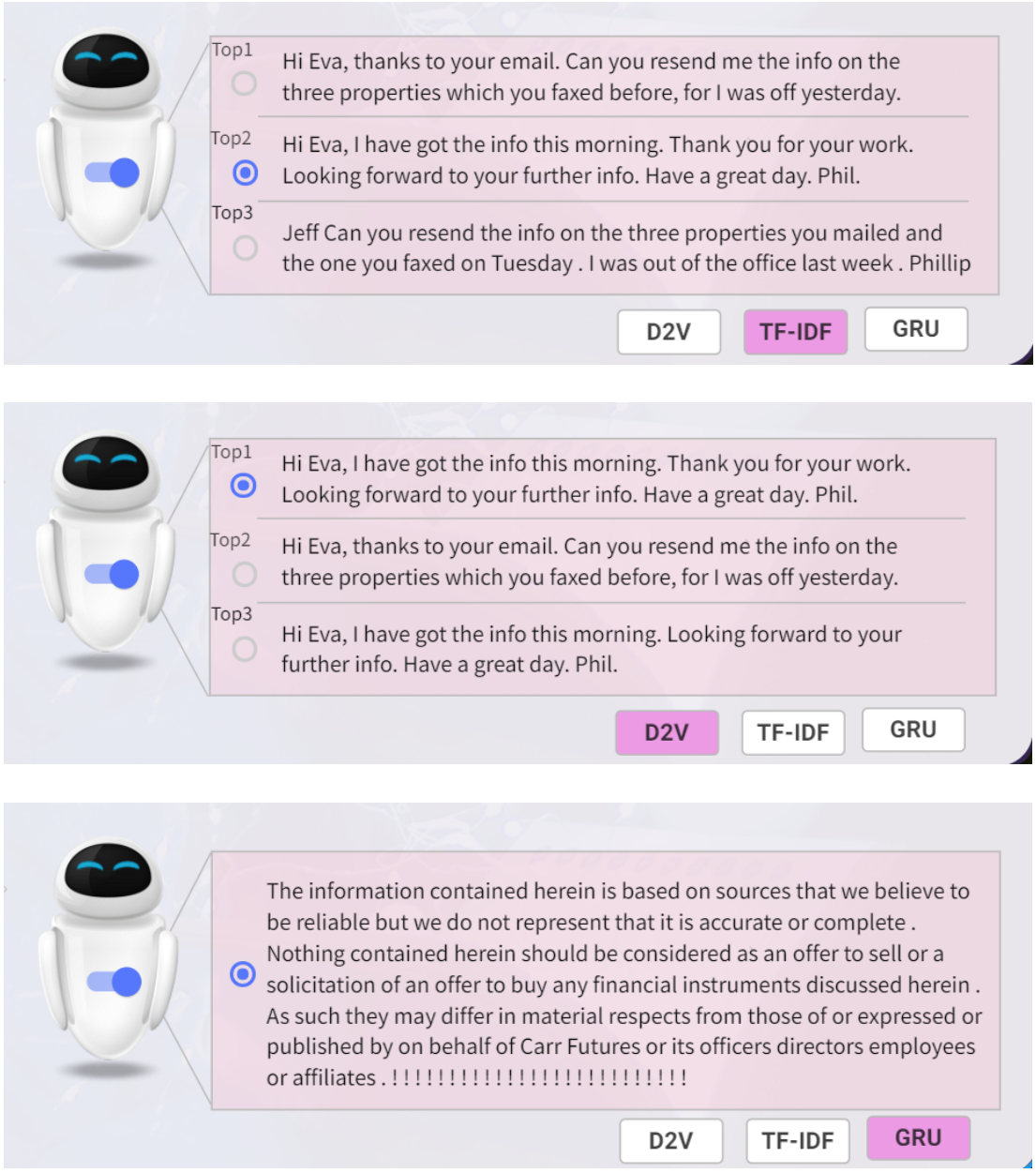


Figure 5.4: Three Models Results[1]

After several rounds of training, the three models each suggested responses to five new E-mails. The first round of the evaluation of the overall result is shown in Figure 5.5. According to the subjective evaluation, five participants chose

[1]The image of the robot is adopted from the movie WALL-E

relevant answers for each new E-mail, and this was a multiple-choice process. The results show that TF-IDF and Doc2Vec, the two information retrieval models, had better performance than the information generation model.

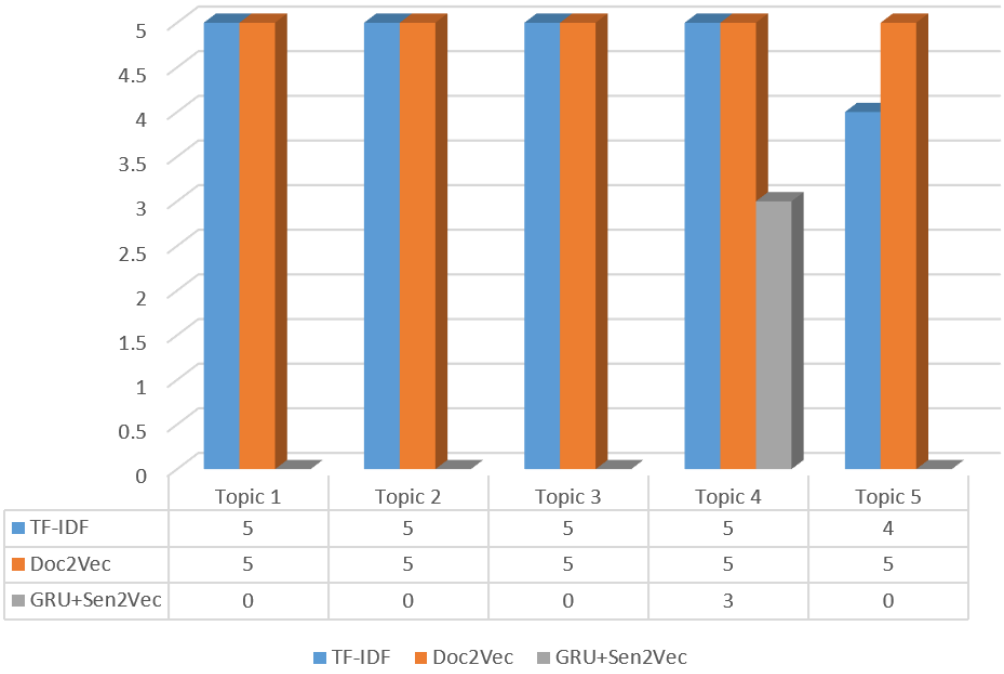


Figure 5.5: Human Evaluation Results

The GRU-Sent2Vec hybrid model is not ideal for several reasons. The main reason is the quality of our training corpus. For deep neural networks, learning relatively accurate feature rules first requires vast datasets. The total number of sentences in our training dataset was 99,431, which is not sufficient. Second, the average number of repeated sentences in the training dataset was only 2.3 (Figure 5.6), while the majority of sentences only appear once. Such extremely low probability distribution of repeated sentences can hardly provide adequate learning information for deep neural networks.

Although we considered this result at the beginning, the reason we experimented with the GRU-Sent2Vec hybrid model was to try this novel idea in order

```
count     99431.000000
mean          2.344822
std          59.809265
min           1.000000
25%           1.000000
50%           1.000000
75%           2.000000
max       16935.000000
dtype: float64
```

Figure 5.6: Sentence Repeat Statistics

to verify whether the generated model could be implemented in a series of sentences.

We expected that, given the right environment, the GRU-Sent2Vec hybrid model could predict and generate a series of sentences as the output according to the input sentences. Theoretically, this has potential value for future research. Moreover, the optimal implementation would look like Figure 5.7.

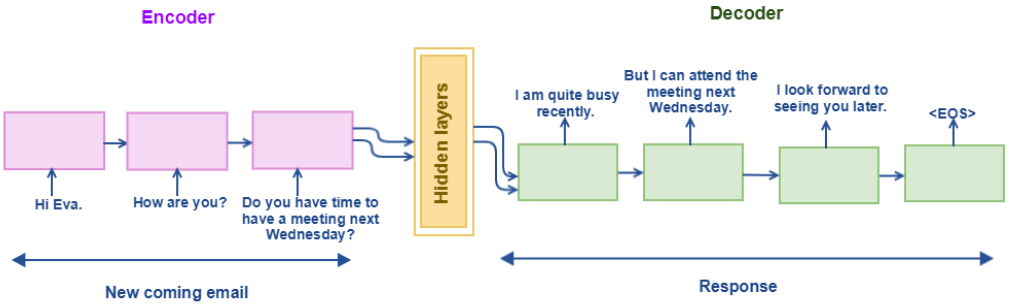


Figure 5.7: Ideal GRU-Sent2Vec Hybrid Model

5.3.2 Comparison of the Two Information Retrieval Models

1. Comparison of Learning Ability

First, we compared the learning ability of the Doc2Vec and TF-IDF Models. Since our training dataset initially did not contain many similar E-mails, we conducted five rounds of training on the models using five similar test E-mails designed and generated for each topic in the previous section. If at the end of each round of training, the model could always find the similar E-mails learned in previous rounds from the database for new E-mails, it meant that the model had the ability to learn new information. The specific process is as follows:

Round 1: We tested it by simply replacing the names of five randomly selected E-mails from the original training dataset. Both models found the five related original E-mails in the first most similar ranking.

Round 2: We put the first round of E-mails together with the responses of corresponding designs into the training set, and then we modified the five selected topic E-mails by changing the order of the sentences. In the second round of testing, the two models found 10 related E-mails in the first two most similar rankings including the original 5 E-mails and the 5 E-mails put into the training set after the first round.

Round 3: We put the E-mails from the second round of E-mail modification into the training set together with the replies from the corresponding designs, and then we modified the 5 selected topic E-mails by adding some information. The results of the third round of testing showed that in the first 3 most similar rankings, TF-IDF found 14 related E-mails while Doc2Vec found 15 related E-mails with the original 5

E-mails and 10 E-mails put into the training set after the first two rounds.

Round 4: We put the E-mails from the third round of E-mail modification into the training set together with the replies from the corresponding designs, and then we modified the 5 selected topic E-mails by deleting some information. In the fourth round of tests, TF-IDF found 18 related E-mails in the first 5 most similar rankings, while Doc2Vec found 19 related E-mails involving in the original 5 E-mails and 15 E-mails put into the training set after the first three rounds.

Round 5: We put the E-mails from the fourth round of design into the training set together with the replies from the corresponding designs, and then we modified the 5 selected topic E-mails by changing the expression of some information. The results of the fifth round of testing showed that in the first 4 most similar rankings, TF-IDF found 19 related E-mails and Doc2Vec found 23 related E-mails with the original 5 E-mails and the 20 E-mails put into the training set after the first four rounds.

After 5 rounds of training, the results of each round could reflect the models' learning abilities. The Doc2Vec model presented a more stable learning ability, especially in the fifth round. When the semantic expression was changed, it presented a better information resolution ability than the TF-IDF model. It was proven that Doc2Vec is capable of extracting semantic correlations between words in an article. The results are shown in Figure 5.8.

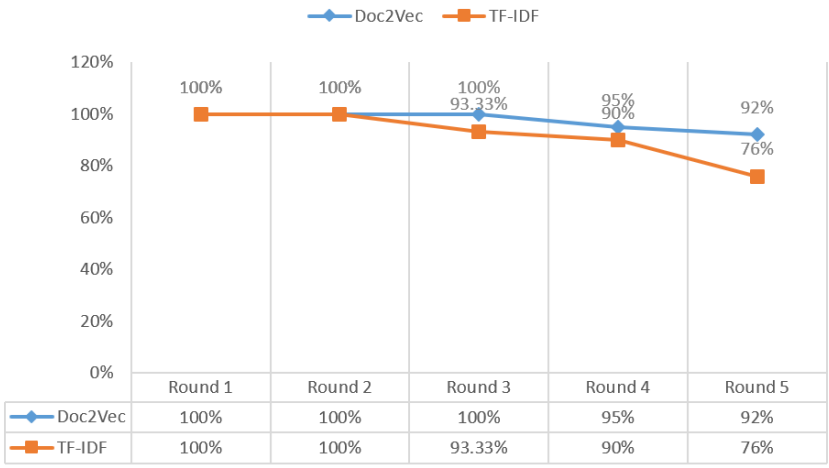


Figure 5.8: Learning Ability of TF-IDF and Doc2Vec

2. Comparison of Effect

After five rounds of training, five participants subjectively evaluated the accuracy of the two models based on the information retrieval mechanism and compared the similarity score. Figure 5.9 presents five new test E-mails on behalf of the five topics. The top five E-mails with the highest similarity in each topic were extracted from the training dataset by two models. Meanwhile, five participants compared and selected the results extracted from the two models under five topics, which means there are twenty-five results that should be selected for each topic, which was a single selection process.

From the results of the selection data given by the five participants, we concluded that the effect of the Doc2Vec model was significantly better than that of the TF-IDF model from the perspective of subjective evaluation.

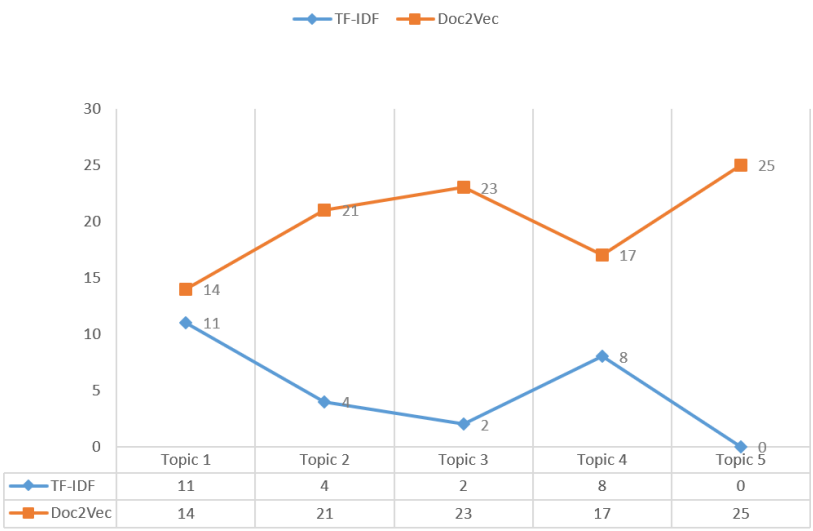


Figure 5.9: Effect Comparison Between TF-IDF and Doc2Vec

5.4 Summary

In this chapter, we mainly implemented the evaluation process of three models and discussed the related results. Firstly, we used the method of call@K to evaluate the TF-IDF and Doc2Vec models and optimised their parameters. Next, five participants subjectively evaluated the effects of the three models, and the experimental results showed that the two models based on the information retrieval were much better than the generative model. Thirdly, we compared the learning ability and performance of the two information retrieval models. By conducting self-assessment and human evaluation, the Doc2Vec based model performed outstandingly. In the end, the overall evaluation results demonstrated that Doc2Vec presented the most positive effect, but the GRU-Sent2Vec hybrid model showed huge potential for future development.

Chapter 6

Conclusion

6.1 Conclusion

Over the years, E-mail overload has not been alleviated but has become increasingly serious during the development of the information age. Therefore, we proposed a novel method on mitigating the issue of E-mail overload and applied three machine learning and deep learning algorithms to model the E-mail dataset. The experimental results showed that this novel method had potential, and we designed an application program with core functions for the experiment.

This paper presents a novel intelligent E-mail response approach, software applications and design solutions. This E-mail management system is based on machine learning and deep learning technologies. By learning similar reply rules, the three models trained in this project can conduct response predictions on newly received E-mails and provide reference reply suggestions. The results of the experiment have showed good performance in improving the effectiveness of people who are required to handle the practical problems of using E-mail in daily life and corporate business. The first thing to be emphasised is that we designed

a novel GRU-Sent2Vec hybrid model that can be used to predict responses based on sentence-level, which goes beyond the limits of word-level prediction. Although the quality of the training corpus limits the effect, the results have been informative and have the potential to guide further development for industrial applications in the future.

As for the evaluation of the effects, it has a subjective component. We designed a set of human evaluation questionnaires. The results of the questionnaire survey show that this research project has significant application value. At the same time, we obtained some improved conclusions and further design proposals for the project, which are conducive to the promotion of the efficiency of solving the practical work.

At the beginning of this project, we listed the main research question and three sub-questions of this research, and our study successfully answered these questions. We used the Enron E-mail dataset as the original data source. Based on the review of related research papers, a process of extracting the trainable E-mail set (SRQ 1) was designed. Combining analysis and research on a large number of machine learning and deep learning algorithms, we finally decided to adopt three methods that are simple: TF-IDF, Doc2Vec (which can mine the relationship between words in a document), and the combination of the production prediction algorithm, GRU (currently the most popular in the field of NLP) and Sent2Vec, jointly realising our experiment (SRQ 2). The performance effects, accuracy analysis, and evaluation were demonstrated. The results discussion answered SRQ 3.

We filled a gap in the field of research to some extent. First, for research teams (except for Gmail or Outlook technical teams), E-mail data for training corpus resources is very limited and the best open source dataset we could use is the Enron E-mail dataset. The data had to be processed and prepared to

ensure the received E-mails were matched with the sent E-mails. Although the processed dataset was still not perfect, the fruits of the research have reference value for this industry. Secondly, the methods we adopted have not been used in other studies (except for TF-IDF). Especially, in the designing of the hybrid model, we transformed sentences into vectors and embedded them into the GRU model to predict long-text content. This research demonstrated the feasibility of the current method, which opens up possibilities for more research in the future. We also hope that our research results can be translated into commercial applications as soon as possible integrating a combination of research, education and production.

6.2 Challenges and Limitations

As stated previously, the experimental models gave coherent and reasonable responses after analysing newly received E-mails. However, there are still many limitations and challenges in the process.

The first limitation is the training dataset. Unlike E-mail datasets from help desk or customer service centres, which contain many similar inquiries from customers, our experimental dataset has very little similarity between E-mails because the E-mails were collected from Enron employees. Besides, there was no marked data in the original training dataset, which made it extremely difficult in the test and evaluation stage. To solve this challenge, this research adopted self-evaluation by randomly selecting 200 samples from the training dataset as marked data and human evaluation. More importantly, the small size of the data samples limited the capabilities of the GRU.

The second limitation is the computing power of the hardware for achieving machine learning or deep learning algorithms. For in the training model stage,

especially in the training of GRU, the time cost of training was very high, and machine failure issues also occurred. Therefore, due to the dual limitations of data quality and hardware, the GRU-Sent2Vec hybrid model's parameter optimisation and self-evaluation were not implemented for this project.

6.3 Future Work

This experiment has made a breakthrough both in practical application and theoretical innovation. However, there is still plenty of scope to push that further.

The first area for potential development is in terms of functional expansion. In this experiment, we mainly focused on using various algorithms to solve the efficiency problem at the E-mail reply stage. However, the efficiency of users' reading information needs to be improved. There is a factor that in the receiving phase, a practical and feasible method will be used with TF-IDF. It will help people save time reading E-mails by highlighting the keywords extracted from the received E-mails.

The second area for potential development is to improve the model algorithm continuously. Jiwei et al. (2016) stated that a new embedding layer could be trained with users' information in the model training stage. In this way, this intelligent E-mail management system will be able to identify the senders or recipient's information and make personalised answer prediction with more sufficient data.

The last area for potential development is the improvement of computing power. It is feasible that the experimental platforms could be transferred from the computer to the cloud in order to accelerate model training and utilise cloud services, for example, a Google Cloud designed Tensor Processing Unit (TPU) for running cutting-edge machine learning models with AI services that can reach

about 30 times the graphics processing unit (GPU) computing power[1].

[1]https://cloud.google.com/tpu/

References

Ayodele, T. & Zhou, S. (2009). Applying machine learning techniques for e-mail management: solution with intelligent e-mail reply prediction. *Journal of Engineering and Technology Research*, *1*(7), 143–151.

Baeza-Yates, R., Ribeiro, B. d. A. N. et al. (2011). *Modern information retrieval*. New York: ACM Press; Harlow, England: Addison-Wesley,.

Bahdanau, D., Cho, K. & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Batra, A., Sidhu, K. & Sharma, S. (2018). Characteristics of women whatsapp users and use pattern. *Journal of Education, Society and Behavioural Science*, 1–7.

Bellotti, V., Ducheneaut, N., Howard, M. & Smith, I. (2003). Taking email to task: the design and evaluation of a task management centered email tool. In *Proceedings of the sigchi conference on human factors in computing systems* (pp. 345–352).

Bellotti, V., Ducheneaut, N., Howard, M., Smith, I. & Grinter, R. E. (2005). Quality versus quantity: E-mail-centric task management and its relation with overload. *Human–Computer Interaction*, *20*(1-2), 89–138.

Belz, A. & Reiter, E. (2006). Comparing automatic and human evaluation of nlg systems. In *11th conference of the european chapter of the association for computational linguistics*.

Bengio, Y., Ducharme, R., Vincent, P. & Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, *3*(Feb), 1137–1155.

Blei, D. M., Ng, A. Y. & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, *3*(Jan), 993–1022.

Busemann, S., Schmeier, S. & Arens, R. G. (2000). Message classification in the call center. In *Proceedings of the sixth conference on applied natural language processing* (pp. 158–165).

Cai, L., Zhou, G., Liu, K. & Zhao, J. (2011). Learning the latent topics for question retrieval in community qa. In *Proceedings of 5th international joint conference on natural language processing* (pp. 273–281).

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. & Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Chum, O., Philbin, J., Zisserman, A. et al. (2008). Near duplicate image detection:

min-hash and tf-idf weighting. In *Bmvc* (Vol. 810, pp. 812–815).

Chung, J., Gulcehre, C., Cho, K. & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Clinchant, S. & Perronnin, F. (2013). Aggregating continuous word embeddings for information retrieval. In *Proceedings of the workshop on continuous vector space models and their compositionality* (pp. 100–109).

Coussement, K. & Van den Poel, D. (2008). Improving customer complaint management by automatic email classification using linguistic style features as predictors. *Decision Support Systems*, *44*(4), 870–882.

Crammer, K. & Singer, Y. (2003). Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, *3*(Jan), 951–991.

Dabbish, L., Kraut, R., Fussell, S. & Kiesler, S. (2004). To reply or not to reply: Predicting action on an email message. In *Acm 2004 conference. citeseer.*

Dabbish, L. A. & Kraut, R. E. (2006). Email overload at work: an analysis of factors associated with email strain. In *Proceedings of the 2006 20th anniversary conference on computer supported cooperative work* (pp. 431–440).

Dabbish, L. A., Kraut, R. E., Fussell, S. & Kiesler, S. (2005). Understanding email use: predicting action on a message. In *Proceedings of the sigchi conference on human factors in computing systems* (pp. 691–700).

Di Castro, D., Karnin, Z., Lewin-Eytan, L. & Maarek, Y. (2016). You've got mail, and here is what you could do with it!: Analyzing and predicting actions on email messages. In *Proceedings of the ninth acm international conference on web search and data mining* (pp. 307–316).

Dredze, M., Brooks, T., Carroll, J., Magarick, J., Blitzer, J. & Pereira, F. (2008). Intelligent email: reply and attachment prediction. In *Proceedings of the 13th international conference on intelligent user interfaces* (pp. 321–324).

Garcia, E. M., Tiedemann, J., España-Bonet, C. & Màrquez, L. (2014). Word's vector representations meet machine translation. In *Proceedings of ssst-8, eighth workshop on syntax, semantics and structure in statistical translation* (pp. 132–134).

Giles, C. L., Kuhn, G. M. & Williams, R. J. (1994). Dynamic recurrent neural networks: Theory and applications. *IEEE Transactions on Neural Networks*, *5*(2), 153–156.

Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H. & Schmidhuber, J. (2009). A novel connectionist system for unconstrained handwriting recognition. *IEEE transactions on pattern analysis and machine intelligence*, *31*(5), 855–868.

Grevet, C., Choi, D., Kumar, D. & Gilbert, E. (2014). Overload is overloaded: email in the age of gmail. In *Proceedings of the sigchi conference on human factors in computing systems* (pp. 793–802).

GROUP, R. et al. (2018). Email statistics report, 2018-2022-executive summary. *Email Stat. Rep*, 3.

Hair, M., Renaud, K. V. & Ramsay, J. (2007). The influence of self-esteem and locus of control on perceived email-related stress. *Computers in Human Behavior*, *23*(6), 2791–2803.

Hardmeier, C., Stymne, S., Tiedemann, J. & Nivre, J. (2013). Docent: A document-level decoder for phrase-based statistical machine translation. In *Acl 2013 (51st annual meeting of the association for computational linguistics); 4-9 august 2013; sofia, bulgaria* (pp. 193–198).

Hewlett, W. R. & Freed, M. (2008). An email assistant that learns to suggest reusable replies. In *Aaai workshop, technical report ws-08-04* (pp. 28–35).

Hinton, G. E., Osindero, S. & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, *18*(7), 1527–1554.

Hochreiter, S. & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Jaeger, H. & Haas, H. (2004). Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *science*, *304*(5667), 78–80.

Jeon, J., Croft, W. B. & Lee, J. H. (2005). Finding similar questions in large question and answer archives. In *Proceedings of the 14th acm international conference on information and knowledge management* (pp. 84–90).

Kannan, A., Kurach, K., Ravi, S., Kaufmann, T., Tomkins, A., Miklos, B., ... others (2016). Smart reply: Automated response suggestion for email. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 955–964).

Kim, D., Seo, D., Cho, S. & Kang, P. (2019). Multi-co-training for document classification using various document representations: Tf–idf, lda, and doc2vec. *Information Sciences*, *477*, 15–29.

Kirkgöz, Y. (2010). Analyzing the discourse of e-mail communication. In *Handbook of research on discourse behavior and digital communication: Language structures and social interaction* (pp. 335–348). IGI Global.

Kiros, R., Zhu, Y., Salakhutdinov, R. R., Zemel, R., Urtasun, R., Torralba, A. & Fidler, S. (2015). Skip-thought vectors. In *Advances in neural information processing systems* (pp. 3294–3302).

Kooti, F., Aiello, L. M., Grbovic, M., Lerman, K. & Mantrach, A. (2015). Evolution of conversations in the age of email overload. In *Proceedings of the 24th international conference on world wide web* (pp. 603–613).

Kosko, B. (1988). Bidirectional associative memories. *IEEE Transactions on Systems, man, and Cybernetics*, *18*(1), 49–60.

Kosseim, L., Beauregard, S. & Lapalme, G. (2001). Using information extraction and natural language generation to answer e-mail. *Data & Knowledge Engineering*, *38*(1), 85–100.

Lapalme, G. & Kosseim, L. (2003). Mercure: Towards an automatic e-mail follow-up system. *IEEE Computational Intelligence Bulletin*, *2*(1), 14–18.

Le, Q. & Mikolov, T. (2014). Distributed representations of sentences and documents. In *International conference on machine learning* (pp. 1188–1196).

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, *1*(4), 541–551.

Li, J., Galley, M., Brockett, C., Spithourakis, G. P., Gao, J. & Dolan, B. (2016). A persona-based neural conversation model. *arXiv preprint arXiv:1603.06155*.

Li, X. & Wu, X. (2015). Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition. In *2015 ieee international conference on acoustics, speech and signal processing (icassp)* (pp. 4520–4524).

Lilleberg, J., Zhu, Y. & Zhang, Y. (2015). Support vector machines and word2vec for text classification with semantic features. In *2015 ieee 14th international conference on cognitive informatics & cognitive computing (icci* cc)* (pp. 136–140).

Linggawa, I. (2017). *Reusing past replies to respond to new email: A case-based reasoning approach* (Unpublished doctoral dissertation). Auckland University of Technology.

Luhn, H. P. (1957). A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of research and development*, *1*(4), 309–317.

Luhn, H. P. (1960). Key word-in-context index for technical literature (kwic index). *American Documentation*, *11*(4), 288–295.

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y. & Potts, C. (2011). Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1* (pp. 142–150).

Malheiros, Y., Moraes, A., Trindade, C. & Meira, S. (2012). A source code recommender system to support newcomers. In *2012 ieee 36th annual computer software and applications conference* (pp. 19–24).

Manning, C. D., Manning, C. D. & Schütze, H. (1999). *Foundations of statistical natural language processing*. MIT press.

Martin, B. A., Van Durme, J., Raulas, M. & Merisavo, M. (2003). Email advertising: Exploratory insights from finland. *Journal of Advertising Research*, *43*(3), 293–300.

Mikolov, T., Chen, K., Corrado, G. & Dean, J. (2013b). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T., Deoras, A., Kombrink, S., Burget, L. & Černockỳ, J. (2011). Empirical evaluation and combination of advanced language modeling techniques. In *Twelfth annual conference of the international speech communication association*.

Mikolov, T., Le, Q. V. & Sutskever, I. (2013c). Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. & Dean, J. (2013a). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (pp. 3111–3119).

Mooers, C. N. (1950). *The theory of digital handling of non-numerical information and its implications to machine economics* (No. 48). Zator Co.

Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. MIT press.

Parameswaran, A., Mishra, D., Bansal, S., Agarwal, V., Goyal, A. & Sureka, A. (2018). *Automatic email response suggestion for support departments within a university* (Tech. Rep.). PeerJ Preprints.

Partridge, C. (2008). The technical development of internet email. *IEEE Annals of the History of Computing*, *30*(2).

Pazos, P., Chung, J. M. & Micari, M. (2013). Instant messaging as a task-support tool in information technology organizations. *The Journal of Business Communication (1973)*, *50*(1), 68–86.

Pennington, J., Socher, R. & Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (emnlp)* (pp. 1532–1543).

Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, *14*(3), 130–137.

Ramos, J. et al. (2003). Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning* (Vol. 242, pp. 133–142).

Ravi, S. & Diao, Q. (2016). Large scale distributed semi-supervised learning using streaming approximation. In *Artificial intelligence and statistics* (pp. 519–528).

Rumelhart, D. E., Hinton, G. E., Williams, R. J. et al. (1988). Learning representations by back-propagating errors. *Cognitive modeling*, *5*(3), 1.

Salton, G. & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information processing & management*, *24*(5), 513–523.

Salton, G., Wong, A. & Yang, C.-S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, *18*(11), 613–620.

Schank, R. C. (1982). *Dynamic memory: A theory of reminding and learning in computers and people* (Vol. 240). Cambridge University Press Cambridge.

Schmidhuber, J. (1992). Learning complex, extended sequences using the principle of history compression. *Neural Computation*, *4*(2), 234–242.

Singh, S. P., Kumar, A., Darbari, H., Singh, L., Rastogi, A. & Jain, S. (2017). Machine translation using deep learning: An overview. In *2017 international conference on computer, communications and electronics (comptelix)* (pp. 162–167).

Singhal, A. et al. (2001). Modern information retrieval: A brief overview. *IEEE Data Eng. Bull.*, *24*(4), 35–43.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. & Potts, C. (2013). Recursive deep models for semantic compositionality over a

sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing* (pp. 1631–1642).

Stross, R. (2008). Struggling to evade the e-mail tsunami. *New York Times. Retrieved May*, *1*, 2010.

Szóstek, A. M. (2011). 'dealing with my emails': Latent user needs in email management. *Computers in Human Behavior*, *27*(2), 723–729.

Tahmincioglu, E. (2011, January). *It's time to deal with that overflowing inbox.* (www.nbcnews.com [Online; posted 01-January-2011])

Tang, D., Wei, F., Yang, N., Zhou, M., Liu, T. & Qin, B. (2014). Learning sentiment-specific word embedding for twitter sentiment classification. In *Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 1: Long papers)* (Vol. 1, pp. 1555–1565).

Thomas, G. F., King, C. L., Baroni, B., Cook, L., Keitelman, M., Miller, S. & Wardle, A. (2006). Reconceptualizing e-mail overload. *Journal of Business and Technical Communication*, *20*(3), 252–287.

Tomlinson, R. (2009). The first network email. *E-mail Home [cit. 2015-11-04]. Dostupné z: http://openmap. bbn. com/˜ tomlinso/ray/firste-mailframe. html*.

Trstenjak, B., Mikac, S. & Donko, D. (2014). Knn with tf-idf based framework for text categorization. *Procedia Engineering*, *69*, 1356–1364.

Tsay-Vogel, M., Shanahan, J. & Signorielli, N. (2018). Social media cultivating perceptions of privacy: A 5-year analysis of privacy attitudes and self-disclosure behaviors among facebook users. *new media & society*, *20*(1), 141–161.

Vinyals, O. & Le, Q. (2015). A neural conversational model. *arXiv preprint arXiv:1506.05869*.

Whittaker, S., Bellotti, V. & Gwizdka, J. (2006). Email in personal information management. *Communications of the ACM*, *49*(1), 68–73.

Whittaker, S. & Sidner, C. (1996). Email overload: exploring personal information management of email. In *Proceedings of the sigchi conference on human factors in computing systems* (pp. 276–283).

Whittaker, S. & Sidner, C. (1997). Email overload: exploring personal information management of email. *Culture of the Internet*, 277–295.

Yang, J., Jiang, Y.-G., Hauptmann, A. G. & Ngo, C.-W. (2007). Evaluating bag-of-visual-words representations in scene classification. In *Proceedings of the international workshop on workshop on multimedia information retrieval* (pp. 197–206).

Yang, L., Dumais, S. T., Bennett, P. N. & Awadallah, A. H. (2017). Characterizing and predicting enterprise email reply behavior. In *Proceedings of the 40th international acm sigir conference on research and development in information retrieval* (pp. 235–244).

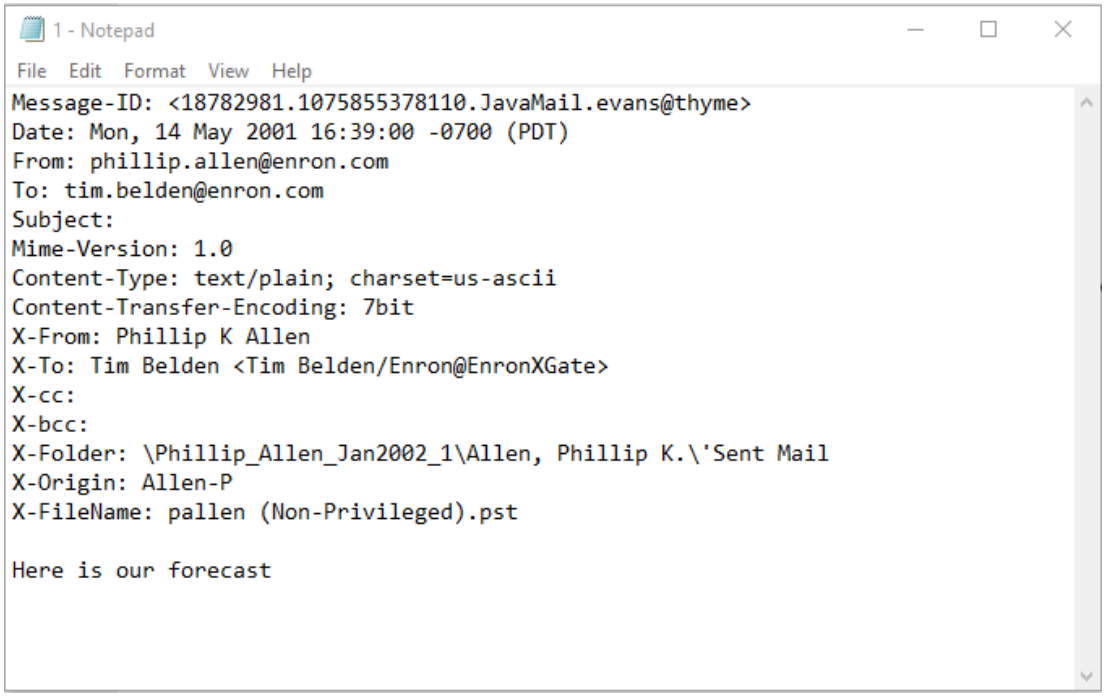
Yonaba, H., Anctil, F. & Fortin, V. (2010). Comparing sigmoid transfer functions for neural network multistep ahead streamflow forecasting. *Journal of Hydrologic Engineering*, *15*(4), 275–283.

Zhou, G., He, T., Zhao, J. & Hu, P. (2015). Learning continuous word embedding with metadata for question retrieval in community question answering. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)* (Vol. 1, pp. 250–259).

Zhu, X., Li, T. & de Melo, G. (2018). Exploring semantic properties of sentence embeddings. In *Proceedings of the 56th annual meeting of the association for computational linguistics (volume 2: Short papers)* (Vol. 2, pp. 632–637).

Appendix A

Sample of Raw Data



Appendix B

Example of Code Snippet

B.1 Data preprocessing

In [11]:

```
#based on Subject to finde related emails with replied email.(use two types to limit the send time)
def find_replied_emails_TitleBased(df):

    for i in range(len(df)):
        df[i]['ID'] = i
    founds1=[]
    for i in range(len(df)):
        best_index = -1
        best_date = datetime.datetime(2050, 10, 18, 10, 0)
        for j in range(len(df)):
            if df[j]['Date'] <= df[i]['Date']:
                continue

            if df[i]['Subject']=='' and df[j]['Subject']=='' :

                if df[i]['From']==df[j]['To'] and df[i]['To']==df[j]['From']:
                    if df[j]['Date']< best_date:
                        best_date = df[j]['Date']
                        limited_time=df[i]['Date']+ datetime.timedelta(days=7)
                        if best_date> limited_time:
                            continue
                        else:
                            best_index = j
            else:
                if str(df[i]['Subject']) in str(df[j]['Subject']) :
                    if df[i]['From']==df[j]['To'] and df[i]['To']==df[j]['From']:
                        if df[j]['Date']< best_date:
                            best_date = df[j]['Date']
                            limited_time=df[i]['Date']+ datetime.timedelta(days=30)
                            if best_date> limited_time:
                                continue
                            else:
                                best_index = j
                    else:
                        continue



        if best_index > -1:
            founds1.append(i)
            df[best_index]['reply_to'] = i
        df[i]['replied_by'] = best_index

    return df,founds1
```

B.2 Preparing the Embedding Layer

In [28]:
```
# FloatTensor containing pretrained Sent2Vec weights

matrix_len = len(voc.sentences_list)
weights_matrix = np.zeros((matrix_len, 150))
sents_found = 0

for i, sentence in enumerate(voc.sentences_list):
    try:
        weights_matrix[i] = voc.sent2vec[sentence]
        sents_found += 1
    except KeyError:
        weights_matrix[i] = np.random.normal(scale=0.6, size=(hidden_size ))


def create_emb_layer(weights, non_trainable=False):
    num_embeddings, hidden_size = weights.shape
    embedding = nn.Embedding.from_pretrained(weights)
    if non_trainable:
        embedding.requires_grad = False
    return embedding, num_embeddings, hidden_size

weights = torch.FloatTensor(weights_matrix)
embedding, num_embeddings, hidden_size = create_emb_layer(weights, non_trainable=False)
```

Appendix C

Sample of Human Evaluation

C.1 Sample of the First Human Evaluation

No.	New email	Model	Reply suggestions	Your choices
1	Phil, I am waiting to get info. on two more properties in Auckland. The broker will be faxing info. by this week. Also, I have mailed you info. on three other properties in addition to the unit property that I faxed this morning. Let me know if you have any questions. Eva	Doc2Vec	Hi Eva, I have got the info this morning. Thank you for your work. Looking forward to your further info. Have a great day. Phil.	
		TF-IDF	Hi Eva, thanks to your email. Can you resend me the info on the three properties which you faxed before, for I was off yesterday.	
		GRU	The information contained herein is based on sources that we believe to be reliable but we do not represent that it is accurate or complete . Nothing contained herein should be considered as an offer to sell or a solicitation of an offer to buy any financial instruments discussed herein . As such they may differ in material respects from those of or expressed or published by on behalf of Carr Futures or its officers directors employees or affiliates . !	
2	Good morning. Is things going well? Have you viewed the first draft of this book? Please make an effort to review this content and give me your comments by next week. You can add subtract correct etc.	Doc2Vec	Hi Eva. I've added my comments. See attached. Thank you. Cheers. Phil.	
		TF-IDF	Hi Eva. I have checked the relevant content and I think there are some details need to be discussed further. Can we arrange a meeting by this weekend?	
		GRU	COLUMBIA GAS TRANSMISSION CORPORATION NOTICE TO ALL INTERESTED PARTIES OCTOBER Notice ID NO RESPONSE REQUIRED SUBJECT CAPACITY ALLOCATION TIMELY NOMINATION CYCLE FOR OCTOBER PLEASE NOTE THAT ALL ALLOCATIONS ARE SUBJECT TO CHANGE BASED ON THE RECEIPT OF ANY NEW NOMINATIONS OR INTRADAY NOMINATIONS .	
3	Hi Phil. Can I read your current title policy to see what it says about easements? You should have received a copy during your closing. I'll be happy to make the copy or whatever makes it easy for you.	Doc2Vec	Eva, I send you the title policy in the attachment. Please check it. Best regards. Phil	
		TF-IDF	Hi, Eva. Around 300 pages and you can decided the copy style. The title policy in the attachment. Please check it. Best regards. Phil	
		GRU	Attached is PIRA s latest Electricity Daily Demand Forecast .	
4	Phil and I would like to discuss the practice questions with you. For the the Knowledge System, we wanted to get some feedback from you. And please send more details regarding last Friday's meeting.	Doc2Vec	Would you like to meet at AUT WZ building L3-3102 at the following Monday morning? Cheers.	
		TF-IDF	Would you like to meet at AUT WZ building L3-3102 at the following Monday morning? Cheers.	
		GRU	No problem .	
5	Hi, Phil, thank you for you work. We also need to verify the curve change of the date. Can you find the closing position for the previous few days and then add the day before each date to the pivot table?	Doc2Vec	I have added the new PivotTable, please see the attachment. Please let me know if there are other needs. Best regards.	
		TF-IDF	Got it, will send you latter.	
		GRU	Sounds good .	

C.2 Sample of the Second Human Evaluation

Topic	No.	Model	Content	Suggestion similarity score	Your sellection
Topic_1: Phil, I am waiting to get info. on two more properties in Auckland. The broker will be faxing info. by this week. Also, I have mailed you info. on three other properties in addition to the unit property that I faxed this morning. Let me know if you have any questions. Eva	1	Doc2Vec	Hi, Phil, the broker will be faxing info. by this week. Also, I have mailed you info. on three other properties in addition to the unit property that I faxed this morning. I am waiting to get info. on two more properties in Auckland. Let me know if you have any questions. Eva.	0.8479	
		TF-IDF	Phil, I am waiting to get info. on two more properties in Auckland. The broker will be faxing info. by this week. Also, I have mailed you info. on three other properties in addition to the unit property that I faxed this morning. Let me know if you have any questions. Eva.	0.7537	
	2	Doc2Vec	Phil, I am waiting to get info. on two more properties in Auckland. The broker will be faxing info. by this week. Also, I have mailed you info. on three other properties in addition to the unit property that I faxed this morning. Let me know if you have any questions. Eva.	0.8448	
		TF-IDF	Hi, Phil, the broker will be faxing info. by this week. Also, I have mailed you info. on three other properties in addition to the unit property that I faxed this morning. I am waiting to get info. on two more properties in Auckland. Let me know if you have any questions. Eva.	0.66294	
	3	Doc2Vec	Hi, Phil, I am waiting to get further info from the broker. I hope they can connect with me by this week. In addition to the unit property, I have mailed you info. on three other properties this morning. Let me know if you have any questions. Eva.	0.7541	
		TF-IDF	Phillip I am waiting to get info . on two more properties in San Antonio . The broker will be faxing info . on Monday . One is units for and the other is units for . Also I have mailed you info . on three other properties in addition to the unit property that I faxed this AM . Let me know if you have any questions . Jeff Smith The Smith Company Circle Drive Austin TX office fax mobile jsmith austintx .com	0.424	
	4	Doc2Vec	Hi, Phil, I am waiting to know your further decision. In addition to the house in San Antonio, I have mailed you info. on three other houses in Auckland. Let's keep in touch. Eva.	0.6355	
		TF-IDF	Hi, Phil, I am waiting to get further info from the broker. I hope they can connect with me by this week. In addition to the unit property, I have mailed you info. on three other properties this morning. Let me know if you have any questions. Eva.	0.3687	
	5	Doc2Vec	Phillip I am waiting to get info . on two more properties in San Antonio . The broker will be faxing info . on Monday . One is units for and the other is units for . Also I have mailed you info . on three other properties in addition to the unit property that I faxed this AM . Let me know if you have any questions . Jeff Smith The Smith Company Circle Drive Austin TX office fax mobile jsmith austintx .com	0.5768	
		TF-IDF	Hi, Phil, I am waiting to know your further decision. In addition to the house in San Antonio, I have mailed you info. on three other houses in Auckland. Let's keep in touch. Eva.	0.1375	