National College of Ireland

Text Summarization of Customer Reviews Using Natural Language Processing

MSc Research Project
Data Analytics

Ridwan Atanda
Student ID: X19142366

School of Computing
National College of Ireland

Supervisor: Hicham Rifai

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Ridwan Atanda
Student ID:	X19142366
Programme:	Data Analytics
Year:	2019/2020
Module:	MSc Research Project
Supervisor:	Hicham Rifai
Submission Due Date:	17/08/2020
Project Title:	Text summarization of customer Reviews using Natural Language Processing
Word Count:	8997
Page Count	24

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ……

Date: ……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Text Summarization of Customer Reviews Using Natural Language Processing

Ridwan Atanda
x19142366

Abstract

Humans have a strong capability to summarize complex and lengthy documents in a simple and concise format. However, in processing and summarizing large volumes of documents within a fraction of seconds, machines outperform the humans. In this particular work, a novel text summarization model was developed by combining extractive and abstractive summarization methods to summarize the large volumes of customer reviews extracted from Amazon data set. The extractive method was used to capture a summary that selects the top-ranking sentences in the corpus using a graph-based TextRank algorithm while these summaries are further fed into a neural network of Long short-term memory (LSTM) to produce the final abstractive summary. The effectiveness of this approach has been measured using the most popularly adopted ROUGE metrics for Natural Language Processing Task. Among multiple models tested, Bi-LSTM is shown to effectively capture the salient information present in the reviews achieving high accuracy and resulted in a concise summary without losing the factual meaning of the reviews.

1 Introduction

With the advent of open markets and online marketing, there has been a tremendous amount of growth in the e-commerce sector led to unprecedented engagement both in financial and non-financial matters between the consumers and businesses on the web. To enhance the transparency and customer loyalty along with shopping experience, most of the online retailers such as Amazon encourages consumers to share their experiences, opinions on the products or services that have been purchased online. This particular feature has given significant power to consumers to express their views openly on the web, which led to a significant increase in the number of customer reviews. As the customers are free to express their opinions, these reviews tend to be lengthy and have only a few sentences that carry significant information about a product. Besides, the lengthy and usually bombastic sentences make it difficult for a prospective buyer to read and understand the complex jargon used by the customers and seldom helps the potential buyer in deciding whether to buy a particular product or not.

In the age of the digital world, customer reviews play a vital role for businesses to prosper as these views impact the decision making of the potential buyers. Hence, to provide a simple, clear, and easy understand summary of customer reviews which further influences the decision making of the buyer, a text summarization tool can be used to help the potential buyers. A text summarization tool is capable of converting long documents into short summaries while preserving the semantic richness of the document. The application of this technique provides an effective solution to summarize a large number of documents in various fields, including news articles, blogs, and research papers. Depending on the final output, conventionally, there

have been two main approaches used in the text summarization process and they are extractive summarization and abstractive summarization method which are used interchangeably as per the requirement. The extractive summarization takes into account the important words or sentences in the input document using statistical features to produce a summary of the document (Christian et al., 2016). In contrast, abstractive summarizes the document in a way similar to the human style of summarization by paraphrasing the document and produces novel words using language generation models. The extractive method is easier to implement because it copies the words or sentences from the input document, which results in a more grammatically correct summary and has been popularly adopted by many researchers (Joshi et al. 2019). On the other hand, the abstractive summarization has more sophisticated features that are relevant in producing a high-quality summary as opposed to the extractive approach, by incorporating a real-world knowledge, generalization and paraphrasing in its framework. The abstractive summarization has several key features in its framework which make an efficient tool to produce better summaries however, it is quite complex and difficult to enforce, hence a limited work is available in the literature (Rush et al., 2015).

To overcome the challenges faced by the abstractive method during implementation, Gupta's research team (Gupta and Gupta, 2019) have employed a deep learning approach using an encoder-decoder framework such as a recurrent neural network (RNN). In this method, the input from the encoder passes through an internal representation or hidden states that the decoder uses to construct the output sequence (LeCun et al., 2015). This approach has been successfully applied to several deep learning tasks in various fields, such as image captioning, speech recognition, machine translation, and video captioning for its reliable results. As a result, recent work (Niu et al., 2019) on text summarization are now directed towards the abstractive summarization method using the encoder-decoder deep learning technique. Several authors (Nallapati et al. 2016), (Zhou et al. 2018), (Shi et al. 2019) have employed this approach over time and have made significant contributions to the field. In the similar lines, the Weston group (Rush et al. 2015) implemented the long short-term memory (LSTM) to solve the problem of exploding and vanishing gradient of the regular RNN framework. In contrast, several authors (See et al. 2017), (Li et al. 2017) also discussed the key drawbacks of these techniques. One such drawback is the inability of the model to obtain a good representation of the input document in its framework during training, which results into incorrect factual information and repetition of summaries produced (See et al., 2017). This limitation of the abstractive approach is a major challenge when summarizing not only the Amazon reviews also other datasets due to repetition of the comments and complexity of the human language. Thus, this notion has prepared us for the computational issues and misinterpretation that may arise in text summarization when dealing with a large amount of text data, hence the key questions which are aimed to be addressed in this project are as follows:

I. *Can the abstractive summarization model produce a concise and human-readable summary when paired with the traditional extractive summarization model?*
II. *Can the Abstractive learning approach grasp the meaning of vocabulary in a raw, unstructured text to generate a concise and non-repetitive summary?*

To address these challenges, we propose a summary model that incorporates the extractive and abstractive framework for summarization tasks. The combination of these frameworks would enable the models to grasp the semantic meaning of the raw input text and yields a better representation to facilitate the creation of a concise, non-repetitive summary. The contribution of this paper can, therefore, be defined as follows:

- Explore the state-of-the-art framework for identifying the appropriate model for text summarization.
- Implement the models that best captures the semantic context and represents a better input document.
- Investigate different data mining techniques that can produce cohesive and non-repetitive summaries.
- Build a model that can understand the text documents on various topics and results in better summaries.

Considering the research objectives outlined above, the major contribution of this project is at improvisation of capturing a good representation of the input documents and producing a non-repetitive summary. Subsequently, in this project we review the state-of-the-art in text summarization domain in section 2, laying a foundation for appropriate methodology in section 3, followed by describing the design specifications in section 4. Likewise, the implementation procedure is explained in section 5 and section 6 describes the experimentation and evaluation of the results. The conclusion and future works are presented in section 7.

2 Related Work

This section gives a concrete overview of the state-of-the-art literature available in text summarization and demonstrates the evolution and improvisation of different methodologies over time. As previously mentioned, a summary model is required as it condenses all the text available in the documents and produces a summary that encompasses all the relevant details contained in the document. Generally, it can be done in two ways; extractive summarization and abstractive summarization, most of the research work in the past focused on extractive summaries since it essentially identifies important sentences in the document. In contrast, abstractive summarization does not solely depend on a simple extraction method to generate summaries, but rather generates new sentences intelligently from the given document(s). These methods are reviewed in this section to identify the best practices appropriate for this research and also to elaborate on the potential gaps in state-of-the-art papers literature.

2.1 Statistical Approach for Extractive Summarization

The 'Topic Sentence' was the first seminal work on text summarization which appeared in the year 1958 aimed at summarizing the scientific documents (Baxendale 1958). This method considered the first and last paragraphs as a basis for summarizing the documents. Interestingly, this simple yet effective method worked perfectly on scientific publications and

became a foundation for other methods to evolve. In the same year, Luhn (1958) put forward the theory called “frequency of terms of content”, which considers the most frequently used words in the document as significant and words that occur least as less significant. As a result, a frequency-based approach was used to score key words in the document to generate a summary. This method was popular until Edmundson (1968) introduced a new method that combined topic sentence and word frequency and adds to its cue words. These were the words that strongly related to the meaning of sentences and were further used to calculate the weight of each sentence in a document to generate human-readable summaries. Another entirely different strategy was that of DeJong (1979), he created the first knowledge-based summarization system called Fast Reading Understanding Memory Program (FRUMP) that uses a template filling method. This method used to obtain predefined text information that covers all topics in a news article and incorporates appropriate information to produce the final summary.

In 1995, automatic text summarization progressed as authors began using machine learning techniques to extract information from textual data, the very first work using a trainable method was a Naïve Bayes algorithm used in a supervised manner to classify important text in a document (Kupiec et al. 1995). This approach produced similarity of about 44% when compared to the human-generated summary. Similarly, the Shetty’ s group (Shetty et al. 2018) also proposed a DOCUSUM technique using K-Means to construct lexical clusters and selected topic keywords that generated summaries. This approach was based on using word features (contents, title, cue words), sentence-level features (location, length, cohesion), and clustering methods (Naïve Bayes, K-Means) to determine the output of the summary. However, apart from the methods mentioned above, there were other summarization methods such as graph-based (Yu et al. 2016) and neural networks (Khan et al. 2019) also contributed to the improvement of the extractive approach.

2.2 Graph-based approach for Extractive Summarization

Graph-based ranking algorithms have also been used in the summarization task, it works similar to the architecture of the PageRank algorithm introduced by Google (Brin and Page 1998). In 2004, the Radev research group (Erkan and Radev 2004) proposed a new method called LexRank algorithm which worked based on 'Lexical Centrality'. This basic idea was to construct a graph representing the phrases in the document therefore, similar phrases can be linked via the vertex and then used to construct the summary. In another work (Mihalcea et al 2004), unsupervised method for extracting keywords and sentence-level features using the TextRank algorithm was proposed. In this method, the keywords and sentences were treated as nodes in the graph followed by assigning an arbitrary value to each node. The computation continues to iterate until its convergences to a value below a certain threshold. In the end each vertex in the graph associates with a ranking and the vertices with the highest score, which further uses to generate the summary. In 2013, Ferreira et al. (2013) extended the work of (Mihalcea et al 2004) by considering four main features (similarity, semantic similarity, co-reference, discourse information) to achieve similarities between sentences. The introduction

of these features helped in selecting the salient sentences that represent the output summary. In another work (Yu et al. 2016), authors developed a novel approach named iTextRank which considers statistical and linguistic features such as similarities in titles, paragraph structures, special sentences, sentence positions and lengths when building sentence graph for the TextRank algorithm.

Following the developments in this domain, the Term Frequency and Inverse Document Frequency (tf-idf) has been used mainly to evaluate the context or significance of a word to a document given a larger body of a document. And this method has been widely integrated into the graph-based approach as a pre-processing measure to produce a high-quality summary. Khatri et al., (2018) computed term frequency-inverse sentence frequency (tf-isf) an adaptation of the tf-idf of a document to determine the sentences that should be included in a summary. As described in (Christian et al. 2016) the tf-idf scores increase with respect to the number of times a word appears across several documents, hence frequent words that appear in the document were included in the summary for news articles. It was also used as a weighting factor to determine words that will be relevant in a summary (Khan et al. 2019). After evaluating these extractive methods, we find that these methods consider only top-k relevant sentences from the input document and in most cases results in summaries that are almost equal in length to the original document which is a serious limitation. Therefore, there is a need for a novel model that can provide a more condensed summary while preserving the relevant details in the corpus. Hence, to address this limitation in this particular project work we implemented the deep learning encoder-decoder method which is described below.

2.3 Encoder-Decoder for Abstractive Summarization

A sequence to sequence framework has an encoder that reads a source article, transforms through its hidden states followed by a decoder that takes the hidden state as an input to produce an output. This model has been successfully applied to various NLP tasks and more recently has achieved the state-of-the-art abstractive summary result (Gupta and Gupta, 2019). A neural sequence-to-sequence model was first implemented by Rush et al., (2015) using the Recurrent Neural Network (RNN) to capture key phrases from the input document in the encoder and pass the resulting sentences to the decoder to generate a short concise human-readable summary. Similar encoder system was used by Xiang group (Nallapati et al., 2016) to capture relevant keywords and Out of Vocabulary (OOV) words in a source document and pass the corresponding sequences to a GRU-RNN decoder. This was carried out to solve the issues in modelling, hierarchical phrase-to-word, problems in keyword matching and to substantially boost summarization result over traditional methods. In another work, Li et al. (2017) put into consideration the latent semantic structure of the input document using RNN generative encoder to improve the quality of the summary produced. Most of these prevalent models have employed the RNN framework however, RNN frameworks are difficult to train due to the problems of vanishing and exploding gradient. It was later found that the LSTM could be a possible solution to these problems and was further implemented by several authors (Zhou et al., 2018), (Han et al., 2019). The LSTM based encoder-decoder framework was introduced by

Zhou et al., (2018) for abstractive summarization task. They further introduced an information filter system using a selective gate network that controls the flow of information from the encoder to the decoder.

Subsequently, Rekabdar et al. (2019) also proposed a complete Gated Recurrent Unit (GRU) for both encoder and decoder to solve the problem of vanishing gradient. Even though these approaches solve the problem of long output summary associated with the traditional approach, they have limitations in producing salient information from the input document and the inability to handle repetitions in summaries. To this end, Han et al. (2019) introduced a read again mechanism using double LSTM layers to improve the quality of the representation of the input document. This method was inspirited by the repetitive reading habit of humans before writing an article summary. Similarly, See et al. (2017) also addressed this problem by proposing a pointer generator network, which copies word from the input document via a pointer and generates novel words from a vocabulary via a generator. With this approach, factual information can be reproduced and summary's repetition can be properly handled when generating a final summary.

2.4 Attention-based Abstractive Summarization

The pointer generator network has addressed the problem of repetition and readability; thus, the attention mechanism has been further introduced to improve on readability, uncommon words, and repetition handling problem (Gupta and Gupta, 2019). In the attention-based encoder-decoder architecture, the decoder does not only receive the input representations from the encoder but also selectively focuses on some part of the input sequence at each decoding step. In later developments, Wang's research group (Shi et al., 2019) proposed a system called NEUSUM which encompasses the LSTM with an attention mechanism. The attention mechanism used in this was to enable the hidden layers to focus on a particular sequence on the input document, thus producing a non-repetitive readable summary. Moreover, Niu et al. (2019) presented a feedforward neural network to work on sentence-level summarization and used the attention-based mechanism similar to Shi et al. (2018) for encoding the input and a beam search mechanism for decoding the output to produce an accurate summary. Since the attention mechanism prevents the model from attending to the same part of the document by tracking past attention weights, it has been considered as the state-of-the-art approach to improving abstractive summarization results.

In the recent past, researchers incorporated the strength of the extractive and abstractive approach towards summarization tasks. This approach effectively represents the salient sentences from the source document by using the traditional extractive approach prior to subjecting the encoder-decoder framework that generates the abstractive summary. The Sun group (Hsu et al. 2018) proposed a unified model by combining the strength of extractor (Khan et al. 2019) and abstracter (See et al. 2017) models and introduced an inconsistency loss function that ensures the model to benefit from both the extractive and abstractive models. This approach led to improvements in ROUGE score when evaluated on benchmark summarization

datasets and outperforms past work on summarization tasks. Moreover, it was shown that the capturing synthetic and semantic features of the input document using word vectors and paragraph vectors before feeding the encoder-decoder model could improve the state-of-the-art results. Therefore, word vectors and PageRank were implemented to obtain extractive summaries while LSTM framework was trained for each sentence present in the corresponding extractive summary (Monalisa and Dipankar, 2020).

2.5 Summary of Related Work

We discussed various methods such as the traditional word and sentence level method (Shetty et al. 2018), the graph-based and tf-idf method (Khatri et al. 2018), and the abstractive methods (Hsu et al. 2018) which were employed to achieve a good summarization model. Therefore, based on the literature, we can see that no single model yielded the desired result of making a good summary of the text or articles. Hence, a novel model which is a combination of extractive and abstractive summarization models together would be appropriate to achieve arguably state-of-the-art results. The key merits of this combination model are due to its capability to handle uncommon and repetitive words and construct a concise human-readable summary. To the best our knowledge we found a limited work in the literature. This is the key motivation bending this project work to implement a summarization model by combining the features of the extractive and abstractive method on a novel dataset. To demonstrate the capabilities of this novel approach the Amazon reviews were adopted. Amazon is a popular e-commerce website and has a public data repository available for researchers to collect data for research purpose. Though the model built in this project was specifically tested on Amazon reviews, it is a generic model and usually applicable to other e-commerce websites as well.

In this particular work, we followed the extractive model described in (Christian et al. 2016) to achieve an extractive summary, which was further presented as an input to the abstractive model. In this case, we implemented a three-layer LSTM encoder and a single layer LSTM decoder with an attention mechanism. The encoder was stacked in three layers to make it easier for the model to get a deeper understanding of the reviews before passing corresponding sequences to the decoder which produces the final output. The important sentences retrieved from the extractive summary were contributed to generate a concise human-readable summary. While the stacked LSTM layer was used to effectively obtain a better representation of the extracted summary before constructing the abstractive summary. In Table 1, a detailed summary of the cited literature is presented for a better overview of the various methods used in text summarization domain.

Table 1: Overview of the literature review

Year	Reference	Framework	Dataset	Metrics
Extractive Summarization				
1958	Luhn	Word frequency and phrase frequency	Technical Articles	Human
1995	Kupiec et al.	Cluster Base and Naïve Bayes Classifier	Scientific Journals	Human
2004	Erkan & Radev	Graph-Based LexRank	DUC 2002, DUC 2003, DUC 2004	ROUGE
2013	Ferreira et al.	Graph-Based TextRank	CNN/DailyMail	ROUGE
2016	Christian et al.	Term Frequency-Inverse Sentence Frequency (tf-isf)	Online eBay Reviews	BLEU
2018	Khatri et al.	Term Frequency-Inverse Document Frequency (tf-idf)	DUC 2007	ROUGE, Human
Abstractive Summarization				
2015	Rush et al.	Complete RNN encoder-decoder	DUC 2003, DUC 2004, DUC 2007	ROUGE
2016	Nallapati et al.	RNN encoder with GRU decoder	CNN/DailyMail, DUC 2007	ROUGE, Human
2017	Li et al.	RNN with an Attention mechanism	DUC 2004	ROUGE
2017	See et al.	LSTM with pointer generator network	CNN/DailyMail	ROUGE, METEOR
2018	Zhou et al.	LSTM with selective gate network	CNN/DailyMail	ROUGE
2018	Niu et al.	LSTM with variational Autoencoders	Gigaword, DUC 2004	ROUGE
Extractive and Abstractive Summarization				
2018	Hsu et al.	Extractor= GRU, Abstractor= Pointer Generator Network	CNN/DailyMail	ROUGE, Human
2020	Monalisa Dey and Dipankar Das	Extractor = PageRank, Abstractor= Double LSTM encoder and single LSTM decoder	DUC/Gigaword	ROUGE, METEOR
Our Approach				
2020		Extractor = TextRank, Abstractor = Triple-layered LSTM with Attention mechanism	Online Amazon Reviews	ROUGE, Human

3 Research Methodology

This section discusses the research methodology as well as the type of assessment used in this particular study. To create a model that effectively generates reliable and succinct summaries of the Amazon reviews, the choice of the right data mining technique is crucial. After careful review of the popular data mining techniques such as Knowledge Discovery in Databases (KDD), Cross Industry Standard Process for Data Mining (CRISP-DM), and Sample, Explore, Modify, Model, Assess (SEMMA) (Azevedo et al. 2008). We have chosen the KDD method for this study, beginning with data collection, pre-processing, and model building, which aims to derive useful information from large corpus following due process. A detailed schematic of the proposed model is shown in Figure. 1.

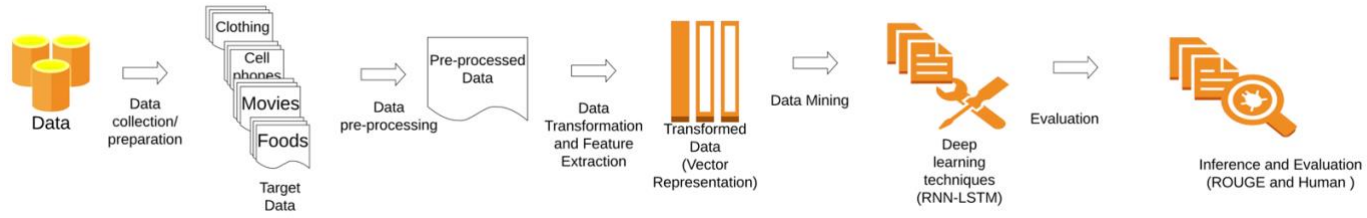

Figure 1: Proposed Methodology

3.1 Data Collection

First and foremost, step in the data summarization project was to collect the data, here we used online Amazon review dataset which is available to the general public (Ni et al. 2019). The entire corpus consists of approximately 233 million customer reviews covering all kinds of products, ranging from books, accessories, food, software, and movies. As the specific aim of the project was to build a robust model that could generalize online reviews; hence to demonstrate the capabilities of data summarization tools developed here, we selected four categories of the review sections. And they were from clothing, shoes and jewellery; cell phone and accessories; movies and TV; along with foods. The reviews consist of various headings such as ReviewerID, ReviewerName, ReviewText, Reference Summary, Ratings and Product Information.

3.2 Data Preparation and Pre-processing

We discover that in the online Amazon portal every product entry has millions of customer reviews. And it was beyond the capacity of the computational resources to handle all the data when fed for processing. Therefore, we decided to extract only 200,000 customer reviews for each category of the product which ended up to a total of 800,000 reviews for all four categories. Preliminary work was done to prepare the dataset, we first cleaned out the noise to reduce the inconsistencies in the dataset, as this may have negative effects in training the model and consequently affects the output results. The necessary columns (ReviewText and Reference Summary) were selected from the raw text file and transformed into a data frame. We subjected the model to delete missing values, duplicate reviews, stop words, special

characters, punctuations, HTML tags, and numbers because they were very common and do not contribute to the contextual meaning of the reviews. In the process, we performed tokenization and transformed the abbreviated words to their original format using contraction mapping and eventually, each text was subjected to convert into a lower case.

3.3 Feature Extraction

During the operation, the preprocessing stage was needed to enhance the extraction of relevant features from the raw text. These features were fed into various machine learning models as mentioned above for implementing the work. Here, the tf-idf features were extracted to obtain the frequent words and phrases that were relevant in the reviews using the *tf-idfVectorizer*[1] from the *Scikit learn* library in Python.

3.4 Data Mining

The data mining phase involved two main stages, namely the extractive and the abstractive stage. The results of the first stage were consequently fed as an input to the second phase, which further produces the final output. In the extractive stage, resulting vectors were fed from the extracted features of the *tf-idf* and were fed into an unsupervised graph-based ranking algorithm using TextRank (Christian et al. 2016). This upon selects the candidate sentences in the reviews that would eventually produce an extractive summary. While in the abstractive phase, a sequence to sequence RNN-LSTM model was implemented using *TensorFlow* and *Keras*. The extractive summary served as the input to this phase and the data was divided into Train and Test using the *sklearn* library. The training was performed by fine-tuning hyperparameters to achieve a model that yields a better representation of the reviews and also capable of making accurate predictions.

3.5 Inference and Evaluation

After training the model, the next activity was to utilize this trained model to make predictions on unseen data and also measure its accuracy. Furthermore, this model produces a probability distribution for each token in the output sequence for all possible characters. That means for each summary the model predicted, it would produce an array of the maximum amount of words that can occur in the summary, in addition to this, the probability would show how likely a particular word be included in the summary. To make sense of this probability distribution there was a need to use a decoding algorithm such as Greedy search or Beam search decoder for prediction (Wilt et al. 2010). For any prediction, the Greedy search decoder essentially considers the words with the highest likelihood and concatenates all the predicted words to get the final output sequence. While on the other hand, the Beam search decoder does not only

[1] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

consider the most likely word for each prediction, it takes into account top-k words with high probabilities (k is called beam size). Hence, it does not give one output sequence like the Greedy search but gives k-different outputs along with their probabilities. The Greedy search has mostly been implemented due to its performance in achieving better results with lesser computational resources as opposed to the Beam search method.

Upon making inferences from the predictions, we moved on to calculating the model's accuracy. The model's accuracy was achieved by measuring overlapping words between the reference summary and the model's generated summary. Historically, precision and recall were used to obtain model's accuracy, however, these metrics do not measure how much of the model's predicted summary was, in fact, relevant or needed. Thus, these methods were insufficient, hence we further proceeded to measure the accuracy of the model using Recall-Oriented Understudy for Gisting Evaluation (ROUGE) metrics. In this case, we compare these summaries on different levels of granularities using ROUGE-1, ROUGE-2 and ROUGE-L.

4 Design Specification

Here, we discuss the workflow of the implementation process carried out in this particular work. Throughout this section, we define the structure underlying the implementation of the proposed models. The model was a combination of extractive and abstract summarization techniques. In the next process step, the reviews were passed on to the TextRank algorithm, since this was an unsupervised learning technique, there were no necessary steps needed to pass the reference summaries along with. The TextRank generates an extractive summary of the reviews which is further concatenated with the reference summary and serves as an input to the RNN-LSTM model. The schematic of the working models is presented in Figure. 2 and described in detail below.

4.1 Stage 1: Extractive Approach

This was the first experiment to generate an extractive summary. This was performed by using an unsupervised learning strategy, which picks up key sentences in the reviews. The detailed procedure is presented in the subsequent sections below.

4.1.1 Preprocessing

For any machine learning task, it is important to pre-process the dataset in an acceptable format, which is possibly one of the most critical stages as its impact will be seen in the remaining phases of the model system. Thus, in the proposed workflow, the data was prepared by removing inconsistencies and irregularities which may affect the output result. Four of the review categories listed above were gathered and concatenated into a data frame. Because of limited computing resources, we pick 200,000 rows for each review category. As mentioned previously rows with missing reviews or summaries, along with stop words and punctuations,

are removed from the data frame because they were common and do not contribute much to the character of the text. Finally, contraction mapping was performed to convert abbreviated words to their base format and converted to lowercase words.

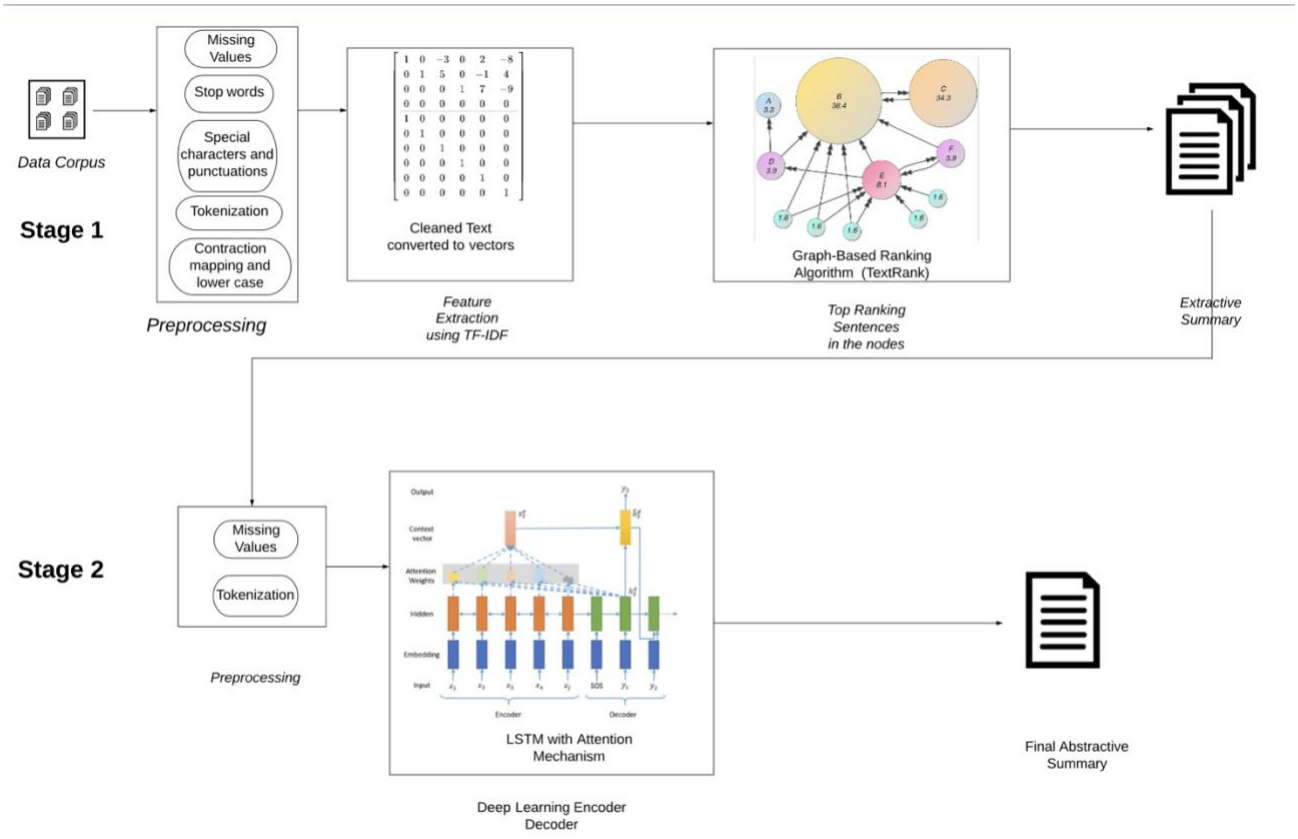

Figure 2: Model Flow

4.1.2 Feature Engineering

As shown in Figure. 2, the next stage was to extract the features from the prepared clean text and recall that for any machine learning algorithm, data fed into it must be represented in numerical values. Thus, it was a necessary step to extract word tokens from the document and compute the frequency of word tokens using *tf-idf* (Christian et al., 2016), then further construct word vectors out of these frequencies. The tf-idf has been widely adopted in NLP tasks to extract relevant features from textual data. It is a weighting mechanism that shows the importance of a word in a document given a larger body of documents. For each word in the ReviewText, we may simply state that it assigns a tf-idf value, and such values differ depending on the significance of that word in the review. It is not feasible to store these strings of words and it corresponding tf-idf values into the computer memory. Therefore, it saves space and maps every word to a numerical hash function in a fairly distributed manner given that the space of the hash value is sufficiently large. This way the extracted features consumes less memory when converted to vectors.

4.1.3 TextRank and Extractive Summary

As shown in Figure. 2, the resulting vectors from the previous stage were fed into a ranking algorithm named TextRank that works based on the PageRank algorithm introduced by Google (Brin and Page 1998). This innovative unsupervised graph-based ranking algorithm has been widely adopted in NLP task, it converts the resulting vectors from the previous phase into a web graph. In the graph, there were nodes and edges, and it assumes that each node has equal weights and the importance of a node would be determined by the number of edges that points to it. This looks like a voting mechanism (Yu et al., 2016), whereby the more edges that point to a node, that means the node becomes significant. Therefore, the word vectors with the highest nodes are considered relevant to the text document and are a good candidate for the extractive summary. Base on a threshold these word vectors were picked and combined to form the extractive summary.

4.2 Step 2: Abstractive Approach

The extractive summary obtained from the previous step is concatenated with the reference summary in the dataset and serves as an input to the deep learning model which produces the final output summary. To this end, we employed the artificial neural network since they learn the best way to make sense of unstructured data. Many data mining models implement the Convolutional Neural Networks (CNN) or the Recurrent Neural Networks (RNN) depending on the task at hand. For instance, researchers mostly implement CNN for deep learning tasks that involve images and videos, such as image captioning or facial recognition. While the RNN is more suitable for tasks that take a sequence of words as the input and produces sequences of words as the output. Based on the merits, we consider this and implement the RNN for this research work and the schematic of the encoder-decoder model is shown in Figure 3. The RNN is an encoder-decoder framework, the encoder extracts the text of equal length from the raw text while the decoder generates translation from this representation. It passes through hidden layers to compute weight and biases that help in generating a better representation of the input text. However, the RNN is only effective for short sequences of words, as it suffers from the problem of vanishing and exploding gradients for longer sequences. Thus, we implemented an extension of the RNN which was the LSTM capable of handling long sequences of data and was developed to handle the problem of vanishing gradient that can be encountered when training RNN. We also integrated the attention mechanism inspired by Bahdanau et al. (2015) into the RNN-LSTM model text. This helped the decoder to determine the source words to concentrate on when generating the next word. Hence, the abstractive approach was the RNN-LSTM with the attention-based mechanism.

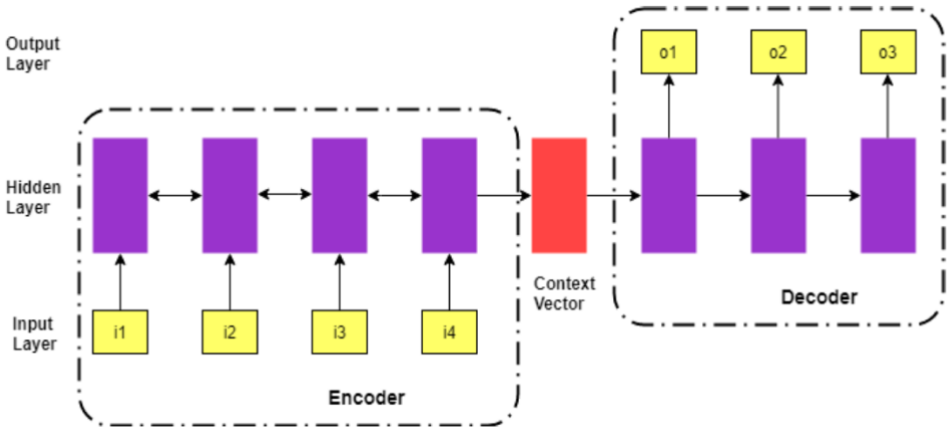

Figure 3: Encoder-Decoder Model

5 Implementation

This section discusses the procedures taken to achieve the summarization task set out for this research.

5.1 System configuration

The Python (version 3.6.9) programming language was used to perform the implementation for this work due to the availability of enough library packages which can be readily imported. It was deployed both on our local machine and as well on the Google cloud services. The local machine was an Intel *Core i5, 3.1GHz processor, 8GB Ram,* and *64 bits MacOS Catalina*. The first experimentation for step 1 was carried out on a local machine, and due to more computational power and requirement for a Graphics Processing Unit (GPU), the second stage of the experiment was moved to Google cloud platform popularly known as '*Google Colab'*. Google cloud platform is an Infrastructure as a service (IaaS) provided by Google[2] that runs entirely on the Google cloud platform and uses the Google compute engine backend for all computing purposes. The execution runtime was set to utilize the free GPU of 1xTesla K80, 2496 CUDA cores, and a RAM of 12GB. Because the cloud service was a free account, the GPU service can only be run for up to 12 hours per day, due to this constraint, it took about a week for the model training.

5.2 Dataset Description

For this research work, we collected the Amazon review dataset (Ni et al., 2019) which is available in a public data repository[3], the reviews span over multiple products for four years.

[2] https://cloud.google.com/compute/docs/resources
[3] http://deepyeti.ucsd.edu/jianmo/amazon/index.html

Since the collected dataset was so huge, it was unlikely that would feed into the models. Therefore, to iterate quickly we selected some specific categories of the product reviews for testing and debugging the model. As stated above, the selected categories were clothing, shoes and jewellery, cell phone and accessories, movies and TV, and food, and end up with some 800,000 reviews. The dataset attributes are listed in Table 2 below.

Table 2: Dataset Description

Attributes	Description
ID	Row numbers
Overall	Ratings of the Customer reviews
ReviewTime	Time of Review
ReviewID	Unique Customers ID
ReviewerName	Customer Name
ReviewText	Product Review or comment
Summary	Reference summary

5.3 Implementation Flow

First of all, we collected the product reviews of the four categories as mentioned earlier. The reviews which were in JSON format parsed into the Python Jupyter notebook using the *Json* library and converted into a *pandas*[4] data frame. A sample of 200,000 rows was considered for each product reviews and ended up with a total of 800,000 reviews which further used for preprocessing. The Dataframe consists of various attributes mentioned in Table 1, and the necessary columns 'ReviewText' and 'Summary' fields were selected from the data frame for further manipulation. From here, rows with missing values were dropped from the data frame and resulted in approximately 783,000 rows for preprocessing. Following this preprocessing task such as text cleaning by removing duplicate reviews, stop words, special characters, punctuations, HTML tags and numbers were removed using the Python *NLTK* library, *Beautiful soup*, and *Regex* libraries. Also, performed a contraction mapping to convert abbreviated words to their base form and the resulting text are converted into the lower case using the *lower()* function from NLTK library[5].

After cleaning the data, the next step was to extract relevant features from it. In this case, the *tf-idf* vectors were extracted for the reviews. It extracted the words that were relevant to the subject matter in the reviews and assigns a hash value to these words. After that, the extracted features were converted into an adjacency matrix and later transformed into a normalized *tf-idf* vector. All these process steps were achieved by using the *tf-idfVectorizer* and *CountVexctorizer* classes from the Scikit learn library. These vectors were further converted into a graph whereby word vectors (sentences of the summary) were nodes and the relationship between these vectors were the edges. From the resulting graph, we implemented the TextRank algorithm which selects top ranking sentences for the extractive summary. Finally, the extractive summaries resulted in about 80,000 rows, because while creating the

[4] https://pandas.pydata.org
[5] https://www.nltk.org

graph, the threshold was set to pick reviews that were longer than 25 words in length. Following this, the extractive summary from the TextRank was concatenated with the ReviewText and reference summary in the data set and converted into a data frame, this marks the end of step 1 of the implementation process.

Table 3: Hyper Parameters

Parameters	Description	Value
Neural Layer(s)	Three-stacked LSTM encoder, and a single layer LSTM decoder	3 encoder, 1 decoder
Hidden-Layers	All layers between the output and input	4
Seq_lenght_x	Length of sequence in Encoder	300
Seq_lenght_y	Length of sequence in the decoder	26
Embedding Dimension	The dimension of embedding in encoder and decoder	200
Attention	To remember the lengthy sequence and what the decoder will focus on when receiving text sequences	Bahdanau's Attention
Learning Rate	How quickly model will adapt to the problem	0.01
Optimizer	The algorithm that minimizes the loss function	rmsprop
Loss Function	Each text output in the decoder are mutually exclusive and converted to a one-hot vector	sparse categorical crossentropy
Drop_out	Reduces overfitting and improves performance only	0.4
Activation	Defines the output of each node given a text or sequences of text	SoftMax

In step 2, the abstractive summarization was implemented using the proposed RNN-LSTM framework. The resulting data frame from step 1 consists of an extractive summary and the reference summary and ReviewText, which would be used to train the neural network. The cleaning process in step 1 was repeated, in this case, inconsistences and irregularities were removed from the summaries and every word converted to lower case. We then fix the length of the extractive summary and the reference summary based on the maximum length of the sequence, the length varies. Thus, the time step can be made to be equal to the length of the longest summary in the data frame with shorter summaries padded with zeros. START and END special tokens were also inserted at the beginning and at the end of the summary to help determine where the sequence starts and finishes. Here, we chose *sostok* and *eostok* as START and END tokens. Also, words whose count is below 6 is considered as rare words and was also removed. The sentences are further tokenized into sequences to form the vocabularies and divided into the train, and test using the *Keras preprocessing* package. The model was built using the Keras library and TensorFlow backend, Table 3 above clearly outlines all the hyperparameters used for training.

6 Experimentation and Evaluation

This section discusses the experiments carried out in this work along with the inference and evaluation of the model result. The performance of the model described above was tested on the Amazon reviews dataset. Three experiments were performed, the first being the baseline

approach which follows a triple-layered LSTM encoder and a single Layer LSTM decoder, the second experiment maintained the first approach with an addition of the attention mechanism to the hidden layer. And the third experiment was a bidirectional LSTM for both encoder and decoder. After the implementation of the models, there was a need to measure how well it performs, to this end the ROUGE metrics were considered for evaluation metrics. This particular method considered as standard metrics for measuring the performance of NLP models. It works by direct comparison between the model generated summary and the reference summary (Human summary), so the ROUGE values were computed from the number of overlapping words between these summaries. There were different variations of the ROUGE scores such as ROUGE-N, ROUGE-S, ROUGE-L and ROUGE-W. The ROUGE-N and ROUGE-L have been implemented in this work as it was used in most of the state-of-the-art papers (Zhou et al., 2018), (Han et al., 2019).

The ROUGE-N measures unigram, bigram, trigram and higher-order n-gram overlap, thus we measure ROUGE-1 and ROUGE-2 which captures the unigram and bi-gram overlap between summaries. While ROUGE-L measures the longest matching sequence of words.

6.1 Experiment 1

In this experiment, the baseline approach performed in (Monalisa and Dipanka, 2020) was implemented by maintaining the encoder at three layers of LSTM encoder and a single layer decoder. This is concerning the research question discussed earlier, the additional layers in the baseline model would help in capturing enough salient representation from the input text (in this case the extracted summaries). To achieve this, the hyperparameters presented in Table 4 were followed excluding the attention mechanism. We also need to iterate over the full dataset several times to get the best result, thus the epoch number was set to 100, however, early stopping was used to measure the model performance while training to avoid overfitting. After building the model, it was subsequently compiled and fit using the *Keras* module. The dataset was divided into 90% training and 10% testing. For every 35 runs of epochs, the model performance was measured and the ROUGE score captured, this we used to measure the improvement of the model and to see how well it performs during the training.

Table 4: Experiment 1 ROUGE scores

Epochs	ROUGE-1	ROUGE-2	ROUGE-L
35	6.1125	1.8853	6.3194
70	7.0041	1.9201	6.8814
100	7.1413	1.9866	6.8019

It is evident from Table 4 that the ROUGE-1 score at 100 epochs achieved the best result. This could be due to the overlapping of unigrams words between the machine-generated summaries and the human reference summary. We also note that the model could not focus on important words from the input text rather produces a grammatically incorrect summary. In the subsequent section, the second experiment is presented which shows an improvement to this baseline model and helps to fix the problem in experiment 1.

6.2 Experiment 2

In this experiment, we consider the addition of attention mechanism to the previous experiment, so that the model could focus on the important part of the input sequences before producing the output. The same hyperparameters displayed in Table 4 were used. Early stopping was used to stop training the neural network at the right time by monitoring the validation loss. Table 5 below depicts the ROUGE result for this experiment.

Table 5: Experiment 2 ROUGE scores

Epochs	ROUGE-1	ROUGE-2	ROUGE-L
35	12.1138	4.6229	11.0021
70	13.0881	4.8324	12.0001
100	14.5211	4.8481	12.8821

It is evident from Table 5 that the model learnt well and shows improvement over the baseline model as shown in Table 4. It can be seen from Table 5 that the model learned as the number of epochs increases but the difference is not significantly large. The ROUGE-2 score is low because it captures bi-grams words. This means that in most cases the machine summary does not overlap with the human-generated summary when it comes to comparing two words at a time. This experiment captures the salient information from the input sequence and produces a better result than the previous experiment. A notable result worth mentioning in this experiment is that even though the model could summarize the reviews there are still repetitions in the summary result. This could be possible because the model could not handle uncommon words that means whenever an uncommon word exist the model replaces it with a common word.

6.3 Experiment 3

This experiment is an improvement to the second experiment. In this case, the encoder was a Bidirectional LSTM (Bi-LSTM) and the decoder was also Bi-LSTM to capture the sequences of words from both sides of the neural layer. For every single LSTM layer, there was another LSTM layer in the reverse direction and then both are combined to form a single bi-directional LSTM. Thus, this experiment was a complete Bi-LSTM layer for encoder and decoder with an attention mechanism. The hyperparameters in Table 4 still applies and the model result is shown in Table 6 below.

Table 6: Experiment 3 ROUGE scores

Epochs	ROUGE-1	ROUGE-2	ROUGE-L
35	13.4941	4.5213	12.0431
70	14.821	5.6119	12.9934
100	15.931	6.5530	13.8833

From Table 6, looking at the values it is evident that this experiment, however, shows a little improvement over the second experiment. In this case, the model was unable to capture the

uncommon words and fix them in the right position. However, it was also noticed in some instances the model failed to capture uncommon words. Overall this model performs better than the other two experiments performed earlier. This clearly shows that the inclusion of the Bi-LSTM improves the model and was able to accurately capture salient information from the input sequence.

6.4 Discussion

In this particular work, different summarization techniques were implemented on the Amazon reviews dataset to obtain a concise summary which in return intended to saves reading time for online shoppers. This model comes in handy and provides a gist of the product reviews to the users when they want to purchase on the website without having to read through long reviews. The data summarization method demonstrated in this particular work, not only applicable to the Amazon dataset, it can also be tested on other online retail websites, news articles, and research papers to obtain a concise summary. While summarizing the reviews, the objective of this research was to achieve an excellent representation of the reviews before training, this helps the model to generate factual and grammatically correct summaries. The first experiment implemented was to capture salient sentences from the input sequence as it was done in (Monalisa and Dipankar, 2020), however, this model fails to produce grammatically correct summaries in most occurrences. Even though the model was able to identify salient information from the input sequence, the poor performance was as a result of the model's inability to generate long output sequence. To resolve this, the second experiment was extended with attention mechanism which was capable of focusing on only the important sentences from the input document and solving the problem of long sequences of words that will be passed to the decoding layer. In this case, the model learned to a significant level and could able to fit the uncommon words in the right places. However, this experiment was producing repetitive summaries. Hence, as a result, the third experiment was embedded with a Bi-LSTM model to resolve this issue. It performed better and achieved the best ROUGE scores.

Figure 5 clearly shows the distribution of the ROUGE scores for each experiment. It is evident from the bar graphs that the third experiment has the highest Rouge-1, Rouge-2 and Rouge-L scores. Even though the difference is not significantly large in numbers, the third experiment with the Bi-LSTM obtained the best summary result. This is concerning the research objectives set earlier, we have successfully explored the state-of-the-art methods in text summarization and was able to achieve a summary model that will capture salient information from the input text to produce a coherent non-repetitive summary.

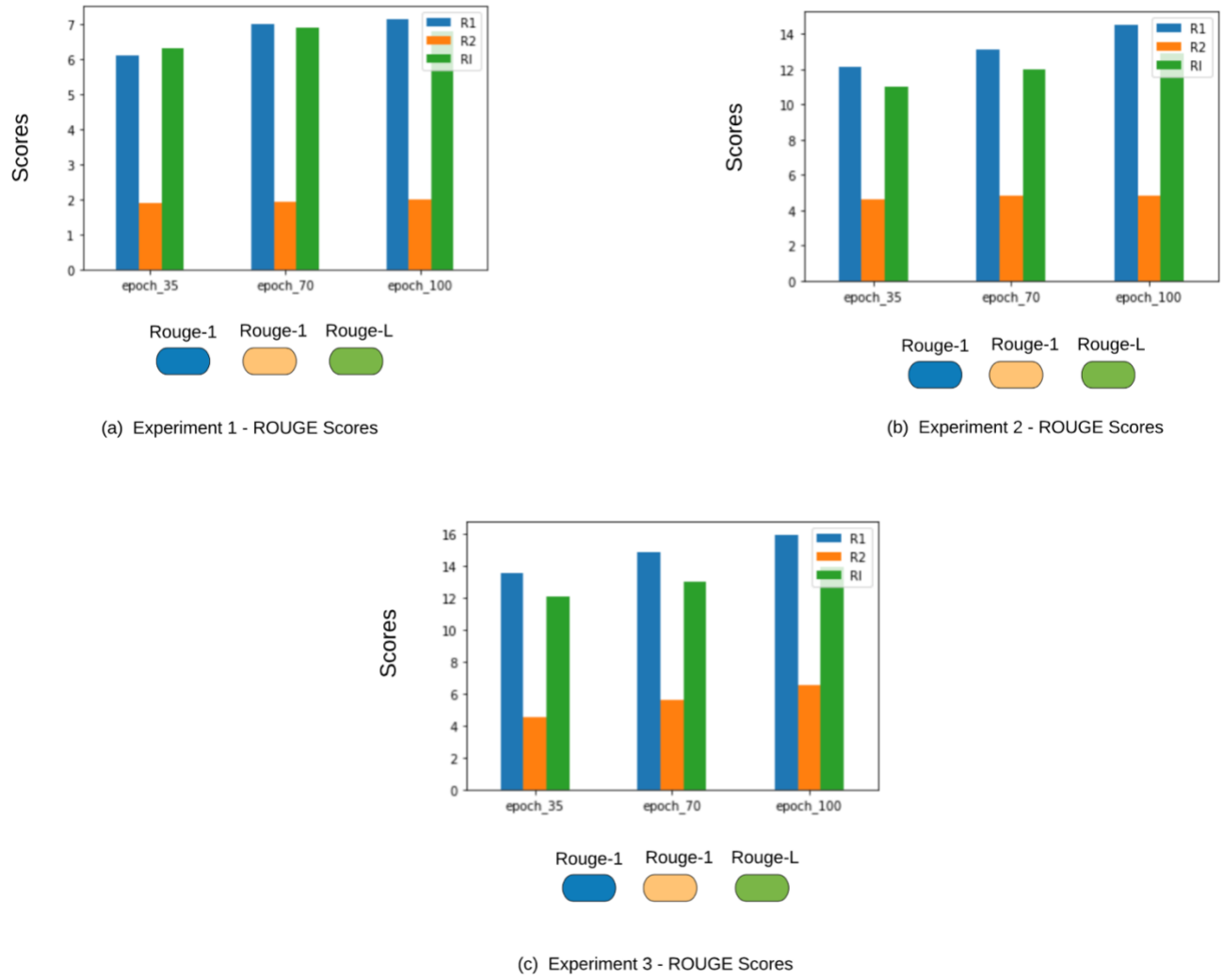

Figure 4: ROUGE scores Overview

Apart from quantitative analysis, we also performed a qualitative analysis by employing five different people to rate the summary generated by the best model. A random sample of 20 was selected from each of the product reviews. Then compared the ReviewText with the machine-generated summary and human summary based on coherence, grammatical correctness, repetitions, informative, and conciseness. The summaries were rated as *good*, *moderate*, and *poor*. The results show that nearly 80 per cent of the summaries were rated good while the remaining 20% was rated moderate, and poor. The results from the human evaluation show that the combination of the extractive and abstractive approach can generate a more informative and concise summary. A detailed summary of the machine-generated and human-generated summers for five example cases are presented in Table 7 which is self-explanatory.

Table 7: Qualitative Analysis

1	**Review Text**: So I purchased these on a whim. They are very quick to prepare (under 2 minutes). T his makes it a perfect quick, filling meal. These have a good amount of weight to them. The noodles are thick and flavorful. The sauce is very tasty as well. Little bit of a tang, slightly hot, but very flav orful. The crushed peanuts included is a nice touch and definitely make it more filling... Also worth noting is, you do not need a microwave to make these. It sure does make things easier, but you can p repare the noodles with just hot water as well by covering and letting sit for a few. Overall very satisfied, it tasted much better than the 2 minutes of preparation would have me expect. I see myself purchasing more of these and trying other Annie Chun foods. I recommend these. **Original summary**: Very tasty and filling **Predicted summary**: Easily prepared great taste (Good)
2	**ReviewText:** This has always been one of my favorites. It is down to Earth, sad, but not extremely harsh or overly scary to children like other similar movies, and heart-warming. To me and my famil y, this is and always will be one of our favorites of the holiday. **Original summary:** One of my favorite holiday movies! **Predicted summary:** Great movie (Good)
3	**ReviewText:** The color is dark Hot pink and the gems don't come down as far as shown on the pictu re!!!Then its all scratched! And broken. The gems fell off. It looks like they are glued on with Elmer 's glue. I totally do NOT recommend!!!! **Original summary:** Horrible!! **Predicted summary:** Not recommend (Good)
4	**ReviewText:** I rarely leave reviews but this dress is so perfect for me and I was very unsure about buying it so hopefully I can help others. I'm a size 18 plus and the XL fits me well. It is so comfortable, flattering, and easy to nurse in! It's easy to dress up or down. I'm so excited I found this dress! As soon as I tried it on I ordered it in another color. I may even get a third because it's just so perfect. **Original Summary**: I rarely leave reviews but this dress is so perfect for me and I was very unsure about buying ... **Predicted summary**: Great dress (Good)
5	**ReviewText:** The shipping was over the date shown on the shipping days. I waited over a month fo r this case to come. And when it did come the quality was very bad, I pay the same price for a simila r case and it came with a bubble wrap and glue and no scratch!!! This case is Not worth the buy. Th e case look Cheap and turtle shipping. :(**Original summary:** Not happy **Predicted summary:** Great case (Poor)

7 Conclusion and Future Work

The problem of abstractive summarization of generating a concise and non-repetitive summary a seldom investigated challenge in the recent literature was addressed in this particular work. Majority of the existing research work focused on solving the problem of generating a summary lesser than the length of the initial document but omit the point of generating factual, meaningful, non-repetitive summaries. Thus, a novel combination was built on the merits of extractive and abstractive text summarization models and subjected to rigorous testing and demonstrated its capabilities on a huge amount of real-time data. The current model based on strong theory effectively selects the salient information from the input document (i.e. Amazon reviews) in the first part its model whereas in the second stage it employs the deep learning approach to generate a concise non-repetitive summary.

It was found that the baseline model effectively summarized the customer reviews, however, failed to capture the factual details from the review comments. To improvise the results a second experiment was embedded with an attention mechanism to improve the ROUGE scores which however produced a summary that captured the factual meaning in the customer's comments. Finally, a Bi-LSTM was implemented to further improve the output, which resulted in effectively capturing salient information present in the reviews and it also outperformed the baseline model with its attention mechanism and achieved best ROUGE score. Though the present model performed to a significant level, still there is a room for improvement on the final experiment. One way is to embed it with a beam search decoder instead of the greedy search and in another way is to train on more data so the model can learn well on the dataset which is a subject of the future work.

8 Acknowledgement

I would like to thank my project supervisor, Hicham Rifai, for his encouragement, advice and guidance during this research. I would also like to acknowledge my family and friends for their love and support throughout my Master's degree.

References

Azevedo, A. I. R. L. and Santos, M. F. (2008) 'Kdd, semma and crisp-dm: a parallel overview', In *Multi Conference on Computer Science and Information Systems*, Amsterdam, Netherlands, pp. 182-185.

Baxendale, P.B. (1958) 'Machine-made index for technical literature—an experiment', *IBM Journal of research and development*, 2(4), pp.354-361.

Brin, S. and Page, L. (1998) 'The anatomy of a large-scale hypertextual web search engine', *Computer Networks and ISDN Systems*, 30(1-7), pp. 107-117.

Christian, H., Agus, M.P. and Suhartono, D. (2016) 'Single document automatic text summarization using term frequency-inverse document frequency (TF-IDF)', *ComTech: Computer, Mathematics and Engineering Applications*, *7*(4), pp.285-294.

DeJong, G. (1979) 'Prediction and substantiation: A new approach to natural language processing' *Cognitive Science*, 3(3), pp.251-273, doi: 10.1016/S0364-0213(79)80009-9

Dey, M. and Das, D. (2020) 'A Deep Dive into Supervised Extractive and Abstractive Summarization from Text', In *Data Visualization and Knowledge Engineering*, 5(9), pp. 109-132, doi: 10.1007/978-3-030-25797-2_5

Edmundson, H.P. (1969) 'New methods in automatic extracting', *Journal of the ACM (JACM)*, 16(2), pp.264-285, doi: 10.1145/321510.321519

Erkan, G. and Radev, D.R. (2004) 'Lexrank: Graph-based lexical centrality as salience in text summarization', *Journal of artificial intelligence research*, *22(8)*, pp.457-479, doi: 10.1613/jair.1523

Ferreira, R., Freitas, F., de Souza Cabral, L., Lins, R.D., Lima, R., França, G., Simskez, S.J. and Favaro, L. (2013) 'A four dimension graph model for automatic text summarization', In *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, Atlanta, GA, 2013, pp. 389-396, doi: 10.1109/WI-IAT.2013.55

Gupta, S. and Gupta, S.K. (2019) 'Abstractive summarization: An overview of the state of the art', *Expert Systems with Applications*, 121(3), pp.49-65, doi: 10.1016/j.eswa.2018.12.011

Han, X.W., Zheng, H.T., Chen, J.Y. and Zhao, C.Z. (2019) 'Diverse Decoding for Abstractive Document Summarization', *Applied sciences*, 9(3), p.386. doi: 10.3390/app9030386

Hsu, W.T., Lin, C.K., Lee, M.Y., Min, K., Tang, J. and Sun, M. (2018) 'A unified model for extractive and abstractive summarization using inconsistency loss', *56th Annual Meeting of the Association for Computational Linguistics, ACL 2018*. Melbourne, Australia, July 2018, pp.132-141, doi: 10.18653/v1/p18-1013

Hu, D. (2019) 'An introductory survey on attention mechanisms in NLP problems', In *Proceedings of SAI Intelligent Systems Conference (IntelliSys)*, London, United Kingdom, September 2019, pp. 432-448. doi: 10.1007/978-3-030-29513-4_31

K. Shetty and J. S. Kallimani, (2017) 'Automatic extractive text summarization using K-means clustering,' in 2017 *International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*, Mysuru, India, 2017, pp. 1-9, doi: 10.1109/ICEECCOT.2017.8284627.

Khatri, C., Singh, G. and Parikh, N. (2018) 'Abstractive and extractive text summarization using document context vector and recurrent neural networks', *the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data.* London, United Kingdom, August 2018, pp. 1150-1157.

Khan, R., Qian, Y. and Naeem, S. (2019) 'Extractive based Text Summarization Using K-Means and TF-IDF', *International Journal of Information Engineering & Electronic Business*, *11*(3), pp. 135-148.

Kupiec, J., Pedersen, J. and Chen, F. (1995) 'A trainable document summarizer', *In Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*. Seattle, USA, July 1995, pp. 68-73, doi: https://doi.org/10.1145/215206.215333

Luhn, H.P. (1958) 'The automatic creation of literature abstracts', *IBM Journal of research and development,* 2(2), pp.159-165.

Mihalcea, R. and Tarau, P. (2004) 'Textrank: Bringing order into text', *In Proceedings of the 2004 conference on empirical methods in natural language processing*. Barcelona, Spain, July 2004, pp. 404-411.

Nallapati, R., Zhou, B., Gulcehre, C. and Xiang, B. (2016) 'Abstractive text summarization using sequence-to-sequence rnns and beyond', *20th SIGNLL Conference on Computational Natural Language Learning (CoNLL)*. Berlin, Germany, August 2016, pp. 280-290 doi:10.18653/v1/k16-1028

Ni, J., Li, J. and McAuley, J. (2019) 'Justifying recommendations using distantly-labelled reviews and fine-grained aspects', In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China, November 2019 pp. 188-197.

Niu, J., Sun, M., Rodrigues, J.J. and Liu, X. (2019) 'A Novel Attention Mechanism Considering Decoder Input for Abstractive Text Summarization', In *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. Shanghai, China, May 2019, (pp. 1-7). doi:10.1109/ICC.2019.8762040

P. Li, W. Lam, L. Bing, and Z. Wang, (2017) 'Deep recurrent generative decoder for abstractive text summarization', in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark, September 2017, pp. 2091–2100 doi: 10.18653/v1/d17-1222

Rekabdar, B., Mousas, C. and Gupta, B. (2019) 'Generative adversarial network with policy gradient for text summarization', In *2019 IEEE 13th international conference on semantic computing (ICSC)*. Newport Beach, USA, January 2019, pp. 204-207, doi: 10.1109/ICOSC.2019.8665583

Rush, A.M., Chopra, S. and Weston, J. (2015) 'A neural attention model for abstractive sentence summarization', *Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Lisbon, Portuga, September 2015, pp. 379-389

See, A., Liu, P.J. and Manning, C.D. (2017) 'Get to the point: Summarization with pointer-generator networks', *55th Annual Meeting of the Association for Computational Linguistics, ACL*. Vancouver, Canada, July 2017, pp. 1073-1083, doi: 10.18653/v1/P17-1099

Shi, Y., Meng, J. and Wang, J. (2019) 'Seq2seq Model with RNN Attention for Abstractive Summarization', In *Proceedings of the 2019 International Conference on Artificial Intelligence and Computer Science (AICS)*. Wuhan, China, July 2019, pp. 348-353, doi:10.1145/3349341.3349429

Wilt, C.M., Thayer, J.T. and Ruml, W. (2010) 'A comparison of greedy search algorithms,' In *third annual symposium on combinatorial search*, Atlanta, Georgia, USA, pp. 34-41.

Yu, S., Su, J., Li, P. and Wang, H. (2016) 'Towards high performance text mining: a TextRank-based method for automatic text summarization', *International Journal of Grid and High-Performance Computing (IJGHPC)*, *8*(2), pp.58-75, doi: 10.4018/IJGHPC.2016040104

Zhou, Q., Yang, N., Wei, F., Huang, S., Zhou, M. and Zhao, T. (2018) 'Neural document summarization by jointly learning to score and select sentences,' *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 18(9), Melbourne, Australia, pp. 59-110, doi: 10.18653/v1/p18-1061