ABSTRACTIVE MULTI-DOCUMENT SUMMARIZATION - PARAPHRASING AND COMPRESSING WITH NEURAL NETWORKS

ELOZINO OFUALAGBA EGONMWAN
Bachelor of Science, University of Benin, 2010
Master of Science, Johannes Kepler University, 2015

A thesis submitted
in partial fulfilment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

THEORETICAL AND COMPUTATIONAL SCIENCE

Department of Mathematics and Computer Science
University of Lethbridge
LETHBRIDGE, ALBERTA, CANADA

ABSTRACTIVE MULTI-DOCUMENT SUMMARIZATION - PARAPHRASING AND COMPRESSING WITH NEURAL NETWORKS

ELOZINO OFUALAGBA EGONMWAN

Date of Defence: December 16, 2020

Dr. Yllias Chali Thesis Supervisor	Professor	Ph.D.
Dr. Wendy Osborn Thesis Examination Committee Member	Associate Professor	Ph.D.
Dr. John Anvik Thesis Examination Committee Member	Associate Professor	Ph.D.
Dr. John Zhang Internal External Examiner Department of Mathematics and Computer Science	Associate Professor	Ph.D.
Dr. Xiaodan Zhu External Examiner Queen's University Kingston, Ontario	Assistant Professor	Ph.D.
Dr. John Sheriff Chair, Thesis Examination Committee	Assistant Professor	Ph.D.

Dedication

To my beloved mother (late), dad,

husband and son (Jayden).

Abstract

This thesis presents studies in neural text summarization for single and multiple documents. The focus is on using sentence paraphrasing and compression for generating fluent summaries, especially in multi-document summarization where there is data paucity. A novel solution is to use transfer-learning from downstream tasks with an abundance of data. For this purpose, we pre-train three models for each of extractive summarization, paraphrase generation and sentence compression. We find that summarization datasets – CNN/DM and NEWSROOM – contain a number of noisy samples. Hence, we present a method for automatically filtering out this noise. We combine the representational power of the GRU-RNN and TRANSFORMER encoders in our paraphrase generation model. In training our sentence compression model, we investigate the impact of using different early-stopping criteria, such as embedding-based cosine similarity and F1. We utilize the pre-trained models (ours, GPT2 and T5) in different settings for single and multi-document summarization.

Preface

This dissertation is an original intellectual product of the author, Elozino Egonmwan. Chapters 2 - 6 are based on work conducted in the University of Lethbridge's Natural Language Processing (NLP) laboratory by Yllias Chali and Elozino Egonmwan. Chapter 7 is based on internship work done at the IBM Watson Research laboratory by Vittorio Castelli, Arafat Sultan and Elozino Egonmwan. A version of Chapter 2, 3 and 5 have been published. (Egonmwan and Chali, 2019a,b). A version of Chapter 4 has been submitted for publication and is currently under review at the *16th Conference of the European Chapter of the Association for Computational Linguistics*. A version of Chapter 6 has been submitted for publication and is currently under review at the *2021 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. A version of Chapter 7 has been published. (Egonmwan et al., 2019).

Contributions of Authors

In Chapters 2 - 5, I was responsible for all major areas of concept/model formation and implementation, as well as manuscript composition. Yllias Chali was the supervisory author and was involved in manuscript edits. In Chapter 6, I was responsible for all major areas of model formation and implementation, as well as manuscript composition. Yllias Chali was the supervisory author and was involved in the concept formation and manuscript edits. In Chapter 7, Elozino Egonmwan, Arafat Sultan and Vittorio Castelli formulated the concept for the manuscript and equally contributed to the manuscript composition. I was largely responsible for the model implementation with substantial contributions from Arafat Sultan.

Acknowledgments

I would like to express my profound gratitude to Prof Yllias Chali for his excellent supervision all through the course of this research. I sincerely appreciate his guidance, deadline reminders, encouragement to continually strive for better results, even though I must admit, it was a tough journey for me. Although each paper rejection made me weaker and believe less in my abilities, his words of encouragement gave me the needed push to keep striving for excellence and to aim for the next submission. I would like to also thank my supervisory committee members - Dr Wendy Osborn and Dr John Anvik for their time and invaluable feedback on my research. They made each committee meeting academically relaxing, such that I could freely express my ideas, progresses and failures. I deeply appreciate their support.

This research was largely funded by the scholarship award provided by Alberta Innovates – Technology Futures (AITF). Thanks to the funding, I was able to attend conferences, finance human evaluations of my experiments, complete my tuition fees and much more. I am immensely grateful to Alberta Innovates – Technology Futures because the funding made it possible for me to focus more on my research, with less worry about my finances. Moreover, I am thankful to Natural Sciences and Engineering Research Council (NSERC) of Canada discovery grant for providing my research lab, a TITAN X GPU machine to run our codes. This made the experiment process not only possible but also faster. Thanks also to the University of Lethbridge School of Graduate Studies Tuition Award, for offsetting part of my tuition fees.

My sincere appreciation also goes to the Computing Research Association (CRA) for awarding me several travel grants to attend a couple of inspiring workshops in the United

States where I got to meet peers and mentors in computing. Thanks to these workshops, I learned useful tips that helped me to better navigate graduate study, and also provided a platform to connect with several industry leaders. In fact, it was through one of these CRA workshops that I got the opportunity to intern at IBM Watson Research Center, New York during the Summer of 2018. An internship that had huge impact on my career.

Words cannot express how grateful I am to Dr Arafat Sultan and Dr Vittorio Castelli – my mentors at IBM Watson Research Center, AI. They essentially made my internship there an amazing experience. The excellent collaboration and team work I experienced there exposed me to several state-of-the-art technologies in the field of AI. I am most grateful to them because in barely four (4) months, I learnt what would have taken me a couple of years outside IBM Research Center.

I am extremely grateful to my husband, Dr Amos Egonmwan, the one I call my pillar of support. This PhD journey would not even have started without his constant motivation. I am thankful for his selflessness and always putting my interests ahead of every and anything else. I am particularly grateful for the comfort and support he always provided especially during low points in my career journey. Most importantly, I am deeply thankful that he stepped in and took care of everything after the birth of our dear son, Jayden. Words are not enough to express my sincere gratitude and appreciation.

On a final note, I would like to thank my parents and siblings; especially Engr (Dr) Godswill Ofualagba and Mrs. Joyce Odjugo for the love and support all through my academic journey. Sadly, my mum saw the beginning of this PhD, but never saw the end. I love and miss you so much mum!

My biggest thanks go to the Almighty God (Jehovah) for granting me with the courage, wisdom and strength to complete this research.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

This thesis is motivated by the need for concise and factual information. Researchers, media analysts, meteorologists and humans in general, often need to access and digest large amounts of data. It will be ideal if machines could automatically process and condense these data into short informative texts, as either abstracts, news digests, weather forecasts or other rich summarized forms. Hence, we seek to develop machine learning models capable of taking news articles as inputs and generating fluent highlights.

We model our approach after the way humans naturally generate summaries. While the art of summary generation by humans could be subjective, it nonetheless, usually encompasses the same key processes such as information extraction and fine-tuning (Cao et al., 2018). Fragments of the text with the most informative content, are taken verbatim to be part of the summary. The selected or extracted texts are then fine-tuned, that is, presented in the words of the summary writer by using grammar techniques like sentence paraphrasing and/or compression to produce abstractive summaries. In line with this, we implement different models for extracting salient parts of a text, generating sentence paraphrases and performing sentence compression.

Due to its real-life utilization appeal coupled with the availability of summarization resources, there is an enormous number of studies on automatic summarization. While this makes it relatively fast to understand the concepts and ideas, on the other hand, innovating novel technologies tends to be increasingly challenging. The focus of this research is

on using existing technologies wrapped with intuitive concepts to produce optimized solutions for automatic summarization. Neural models have shown more promise than manual-feature engineering in Natural Language Processing (NLP) tasks like summarization (Chen and Bansal, 2018; Gehrmann et al., 2018; Zhou et al., 2018). Hence, our methods are all implemented using Neural Networks. We begin with extractive summarization, followed by abstractive summarization using the aforementioned techniques – paraphrasing and compression.

Section 1.2 provides high level definitions of some key terms. We present clearly, the research questions we seek to answer in Section 1.3 and provide highlights of the structure of the entire thesis in Section 1.5.

1.2 Key Terms

– `Neural Network` (NN) is a set of algorithms that endeavour to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. It consists of interconnections of neurons, referred to as nodes which are grouped to form layers. Nodes from one layer pass information to the next layer. A basic NN consists of three (3) layers – input, hidden, and output layers (Kröse et al., 1993).

– `Machine Learning` (ML) is a study of computer algorithms that improve automatically by learning patterns. ML enables self-learning from data (Alpaydin, 2020).

– `Manual Feature Engineering` is the art of manually studying patterns in a set of source data. These patterns are then used to craft features for extracting information from a new set of data with the same patterns as the source. The crafted features could also be used to help facilitate the ML process (Severyn and Moschitti, 2013).

– `Transfer-learning` is the improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned (Torrey and Shavlik, 2010).

– `Text Summarization` is the process of producing a concise or shorter text from a

longer source text, that retains only the most salient parts of the source text (Nenkova and McKeown, 2012).

Source Text: Proghorns defender Daniel Schumann has kept his driving license, telling a court he was speeding 36km over the limit because he was distracted by his sick cat. He drove 96km/h in a 60km/h road works zone on the Southern Eastern expressway in February. He said he didn't see the reduced speed sign because he was so distracted by his cat vomiting violently in the back seat of his car.
Extractive Summary: Proghorns defender Daniel Schumann has kept his driving license, telling a court he was speeding 36km over the limit because he was distracted by his sick cat. **Abstractive Summary:** Proghorns defender Daniel Schumann admits to speeding but says he didn't see road signs because his cat was vomiting in his car.

Figure 1.1: Example of an extractive and abstractive summary.

– `Extractive Text Summarization` reproduces verbatim, parts of the source text which it considers most important (Xiao and Carenini, 2019) as illustrated in Fig 1.1.

– `Abstractive Text Summarization` rewrites the most informative parts of the source text in a concise form and introduces novel words in the summary (Kouris et al., 2019) as illustrated in Fig 1.1.

– `Query-focused Text Summarization` aims at generating a summary with respect to a given query (Egonmwan et al., 2019).

– `Generic Text Summarization` generates a concise version of a source document(s) without any specific query in view (Lee et al., 2009).

– `Single Document Summarization` summarizes a single piece of document (Liu et al., 2019).

– `Multi-document Summarization` summarizes a set of related documents (Goldstein et al., 2000).

– `Indicative versus Informative Summarization`: Indicative summarization identifies the topics of a document while informative summarization represent the concise de-

scription of the original document (Kumar and Salim, 2012). Summarization presented in this thesis are informative.

1.3 Research Questions

At a high level of abstraction, the overarching question we ask in this thesis is:

RQ0 *How can we build a machine learning model that is capable of generating both extractive and abstractive summaries which are grammatically correct and are faithful to the facts contained in the source text?* This has been an open research question for researchers over the years and very interesting solutions have been proposed. However, there is still a need to close the gap between human and machine-generated summaries, especially in terms of readability and factuality (i.e, maintaining the semantics in the source document). As presented in a study by Lebanoff et al. (2019a), 38.3% of the system outputs introduce incorrect facts (see example in Figure 6.1), while 21.6% are ungrammatical (see example in Figure 6.2).

RQ0 gives rise to several sub-questions below:

RQ1 *How can we identify parts of a text that are the most relevant for an extractive summary with improved accuracy?* One challenge with extractive summarization is that the reference summaries in available summarization data-sets are abstractive summaries written by humans. Due to the abundance of external knowledge available to humans, some of these abstractive reference summaries contain information that cannot be directly inferred from the source text. It is infeasible for the ML model to learn patterns from such disparate data pairs. Hence, we must carefully identify all such reference summaries in the dataset, in order to present the ML model with training pairs it can feasibly learn from.

RQ2 *How can we build a paraphrasing model to help with abstractive summarization?* Since abstracting and paraphrasing have the same goal of expressing a text in alternative ways using novel words, the conjecture is that a good paraphrasing model applied on extractive summaries should generate well-formed abstractive summaries.

RQ3 *How can we build a compression model to help improve the conciseness of machine generated summaries?* Extractive summaries are often too lengthy, especially in multi-documents settings. Since whole sentences from the source text are extracted to form the summary, parts of the extracted sentence may contain irrelevant information that are not summary-worthy. It is therefore, necessary to identify these irrelevant words in a sentence. A good sentence compression model appropriately deletes uninformative words in a sentence while retaining the fluency of the resulting compressed sentence.

Source Text: [...] John Abraham has been prohibited from using the title Hamara Bajaj for his home production [...]
Machine Summary: [...] John Abraham has been prohibited from his home [...]
Reference Summary: [...] John Abraham has been prohibited from using the title Hamara Bajaj [...]

Figure 1.2: Example of misleading information in summary.

RQ4 *How can we investigate that our machine generated summaries are factual, that is, that they contain accurate information?* Summarization models are usually evaluated using the ROUGE metric (Lin, 2004), which basically measures the word overlap between the machine and reference outputs. However, as seen in Figure 1.2, a machine generated summary could have high overlap with the reference, but may still contain a high degree of misleading information. We want to be careful to avoid this.

RQ5 *How well does transfer-learning impact multi-document summarization?* Documents for summarization can be presented as single documents or multiple documents discussing the same underlying topic. Since NN models require large training data-set (which is lacking in MDS) for optimized performance, we apply different transfer-learning methods. We compare the gains of using our proposed pretrained models from single document extractive and abstractive summarization versus a language model – GPT2 pretrained on a very large corpus. It is therefore how well these pretrained models perform on MDS.

1.4 Contributions

This thesis makes the following contributions to research in automatic text summarization:

- We presented a simple algorithm for building a sentence-labelled corpus for extractive summarization training that produces more accurate results.
- We proposed a novel framework for the task of extractive single document summarization that improves the current state-of-the-art on two specific datasets.
- We introduced the encode - encode - decode paradigm using two complementary models, TRANSFORMER and SEQ2SEQ for generating paraphrases that improves current top performance on two specific datasets.
- We presented a model that improves performance on extractive sentence compression on GOOGLENEWS compression test dataset.
- We demonstrated the utility of our sentence compression model on a related task – extractive document summarization and in parallel show that our model generalizes relatively well to non-dedicated data.
- We investigated the impact of implementing different stopping criteria on a model trained with the same learning objective, and showed that using the right heuristics is crucial and has a direct bearing on the performance of the model.
- We presented a method for transfer learning by supervised pretraining on paraphrase generation and sentence compression for abstractive MDS.
- We demonstrated the utility of downstream tasks, such as paraphrase generation and sentence compression on the problem of Multi-document summarization.

1.5 Outline

We start with the task of Single Document Extractive Summarization in Chapter 2. After giving background information on this task, we then provide a description of our methodology. First we formulate a method to identify and process disparate data-pairs in the training set that could impede the performance of the extractive summarization ML model. Next, we tune the processed abstractive summarization data-set to be suitable for training an extractive summarizer. To investigate the quality of the processed and tuned data-set, before training of the NN, we perform some evaluations. We evaluate our tuned data-set for accuracy when compared to existing work in literature. In an attempt to address **RQ1**, we describe the architecture and implementation of our NN when trained on the processed data. The outcomes of this chapter are an extractive summarization model that performs well on two (2) data-sets –CNN/DAILYMAIL (Hermann et al., 2015; Nallapati et al., 2016) and Newsroom (Grusky et al., 2018), as well as a good data-filtering technique and an effective method for creating an extractive summarization data-set from an abstractive one. All of the outcomes are used in Chapters 5 and 6.

In Chapters 3 and 4 we build ML models for paraphrasing and compressing, respectively. The paraphrasing model was trained and tested on the QUORA and MSCOCO data-sets (Gupta et al., 2018), while the compression model was implemented using the GOOGLE-NEWS data-set (Filippova et al., 2015; Filippova and Altun, 2013). These two models provide answers to **RQ2**, and **RQ3**. Our compression model also addresses **RQ4**. The paraphrasing and compression models serve as strong foundations for building our abstractive summarization model in Chapters 5 and 6. Specifically, our single and multi-document abstractive summarization model described in Chapters 5 and 6 incorporates the models built in Chapters 2 to 4.

In Chapter 5 we present a neural model for abstractive single document summarization SDS. Our SDS model learns to extract and paraphrase using methods presented in Chapters 2 and 4 in order to generate abstractive summaries.

In Chapter 6 we investigate **RQ5**. Without building a new NN model for multi-document summarization (MDS), we formulate a method for applying our single document summarization (SDS) models iteratively on a MDS data-set – DUC2004 (Paul and James, 2004). While this strengthens the claim that our SDS models perform well both extractively and abstractively, even in MDS settings, it also demonstrates how well our model generalizes to out-of-domain data, which is a desirable model quality.

Chapter 7 explores summarization guided by a query, termed Query Focused Summarization (QFS). Specifically, we explore how we can use knowledge gained from a closely-related NLP task – Question Answering (QA) in QFS. This becomes necessary because QFS suffers from a sparsity of training data. On the other hand, there exists an abundance of data for training QA models.

We conclude this thesis by providing answers to **RQ0** and giving some perspectives for future work in Chapter 8.

Chapter 2

Single Document Extractive Summarization

In this Chapter, we describe concerns and propose solutions to the task of building a machine learning model for single document extractive summarization. Specifically, we aim at providing an answer to **RQ1:** *How can we identify parts of a text that are the most relevant for an extractive summary with improved accuracy?*

2.1 Background

One fast and easy approach to extractive summarization, is simply extracting the leading sentences (first three sentences for example) in a document as the summary, especially in news articles. In fact, this method forms strong baselines for comparison in literature (Grusky et al., 2018; Narayan et al., 2018; See et al., 2017). This is because it is believed that the most important information in a text is usually conveyed at the beginning. However, human styles of writing differ from person to person. While some writers begin a text document by giving the most important details, others might choose to build the main point gradually – that is, start with minor points to aid understanding of the major point which are discussed, towards the end of the text. Hence given this variance in writing, it is difficult to pinpoint the most informative parts of a text only by virtue of the position of a sentence in a text document.

Additionally, informativeness is subjective. What might be important to one reader, might be less relevant to another, based on differences in background knowledge. Hence,

summarization guided by a reader's query – `Query Focused Summarization` (QFS) (Egonmwan et al., 2019) is ideal in such situations. Occasionally however, the user might have no specific query in mind, but rather needs to obtain an overview of the document in order to decide if investing more time into the whole document is relevant. Summarization of this nature is termed `Generic Summarization` (Lee et al., 2009). This is the scope of this thesis, except where otherwise specified (as in Chapter 7). Generic summarization is particularly useful in the context of *news articles*. This is because the highlights of a news article are, in a sense, constant. That is, the highlights of a news article do not vary, based on the reader. In this thesis, all our summarization models were trained, validated and tested on data from news articles. In the context of this thesis, we refer to `generic summarization` as `summarization` for simplicity.

Approaches to solving the task of automatic extractive summarization can be technically classified into two (2) – the use of *Manual Feature Engineering* (MFE) and *Neural Network* (NN). We leave discussion on NN-based methods to Section 2.4 under Related Work, as our approach is also NN-based. In the next section, we seek to understand the idea behind some interesting MFE approaches, how they work, and why we chose a different methodological path.

2.2 MFE-based approaches

MFE-based approaches do not automatically learn features in data like NN approaches do. Rather based on observed patterns in data, features are manually crafted. These features are then applied to an algorithm. Subsections 2.2.1 – 2.2.3 describe some of the algorithms applied on manually chosen features.

2.2.1 Naive-Bayes Classification

The *Naive-Bayes* (NB) classifier (Rish et al., 2001) uses a supervised algorithm that applies the Bayes Theorem in a 'naive' way. The Bayes theorem estimates the likelihood

that an event (A) will happen given that another event (B) has already happened as given in equation 2.1:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}. \tag{2.1}$$

NB relies on the naive assumption that the input variables are statistically independent. Kupiec et al. (1995) describes extractive summarization as a statistical classification problem. A NB classification function is then developed by the authors to estimate the probability that a given sentence is included in the extract. Mathematically, it is formulated as:

Let s be a particular sentence, S the set of sentences that make up the summary, and $F_1, ..., F_k$ the manually chosen features (Das and Martins, 2007), then

$$P(s \in S|F_1, F_2...F_k) = \frac{P(F_1, F_2...F_k|s \in S)P(s \in S)}{P(F_1, F_2...F_k)}. \tag{2.2}$$

Assuming statistical independence of the features, equation 2.2 yields equation 2.3:

$$P(s \in S|F_1, F_2...F_k) = \frac{\prod_{j=1}^{k} P(F_j|s \in S)P(s \in S)}{\prod_{j=1}^{k} P(F_j)}, \tag{2.3}$$

where $P(s \in S)$ is a constant and $P(F_j|s \in S)$ and $P(F_j)$ can be estimated directly from the training set by counting occurrences. New extracts can then be generated by ranking sentences according to the probability in equation 2.3 and selecting a user-specified number of the top scoring ones.

2.2.2 Hidden Markov Models

The *Hidden Markov Model* (HMM) allows for the description of causal factors – both observed events (like words in an input sentence) and hidden events (like *part-of-speech* tags) in its probabilistic model. It makes use of a Markov Chain, which is a model that tells

us something about the probabilities of sequences of random variables. A Markov chain assumes that future predictions in a sequence relies only on the current state (Jurafsky and Martin, 2016).

The problem of extracting sentences from a document is modelled using an HMM by Conroy and O'leary (2001). Given a set of features, the HMM computes an a-posteriori probability that each sentence in a document is a summary sentence. In contrast to NB described in Subsection 2.2.1, which assumes statistical independence, HMM has fewer assumptions of independence. It accounts for local dependencies between the sentences. By using HMM, it is expected that the probability that the next sentence is included in the summary depends only on the inclusion of the current sentence in the summary, as seen in equation 2.4:

$$P(s_i|s_1...s_{i-1}) \approx P(s_i|s_{i-1}). \tag{2.4}$$

Only three (3) manually selected features were used: – position of the sentence in the document, number of terms in the sentence and likeliness of the sentence terms given the document terms. The features are built into the state-structures of the HMM.

2.2.3 Graph-based Models

Here, the document is modelled as a graph – a collection of nodes, connected by edges. Sentences are represented as nodes, and their similarity is modelled by a connecting edge. The weight on an edge indicates how similar the two (2) connected sentences are. This similarity is usually measured by the *cosine similarity* of the sentences based on their vector representations such as *term frequency - inverse document frequency (tf-idf)*, `GloVe`, *word2vec* or *BERT* embedding. Cosine similarity measures the cosine of the angle between two vectors, using equation 2.5 (the smaller the angle, the higher the similarity measure)

$$cos(X,Y) = \frac{X \cdot Y}{\|X\|\|Y\|} = \frac{\sum_{i=1}^{n} X_i Y_i}{\sqrt{\sum_{i=1}^{n} X_i^2}\sqrt{\sum_{i=1}^{n} Y_i^2}}. \tag{2.5}$$

A graph-based algorithm – the *eigenvector centrality*, is then applied to determine the importance of each node (Erkan and Radev, 2004). The *eigenvector centrality* is a measure of the influence of a node in a graph network. It assigns relative scores to all nodes in the graph based on the concept that connections to high scoring nodes contribute more to the node in question than equal connections to low scoring nodes.

2.2.4 Why not MFE

Work presented in the pieces of literature discussed in sub-sections 2.2.1 - 2.2.3 give credence to the science and logic behind MFE-based approaches. To some extent, they are able to identify and extract important sentences in a document. However, they are limited in various ways:

- They perform comparably worse on unseen samples. After learning patterns in training data, and crafting a limited set of features based on these seen data, they are unable to infer unseen relationships/features on new data.

- Not all feature relationships are linear. Some are non-linear and complex. MFE-based approaches are unable to capture such non-linear and complex patterns in data, thereby limiting its performance.

- They are task specific. Engineered features for a specific task cannot exactly be transferred for another closely related task with a different dataset.

- Crafting the features is time consuming, despite how limited in number these features are.

Based on the aforementioned reasons, in addition to improved efficiency and state-of-the art performance reported in models implemented with NN (Gehrmann et al., 2018; Zhou et al., 2018), we present our NN-based single document extractive model in the following section.

2.3 Methodology

First, we state mathematically what the task of single document extractive summarization entails. Next we present in detail our approach to the solution, outlined thus – a preview of the dataset, data filtering/cleaning, pre-processing steps, data tuning, NN architecture, implementation details, results evaluation and analysis of our output. Finally, we discuss how our work differs from existing work in this task.

2.3.1 Task Definition

Given a document $D = (S_1, ..., S_n)$ with n sentences comprising of a set of words $D_W = \{d_1, ..., d_w\}$, the task is to produce an *extractive* summary, S_E , that contains salient information in D, where $S_E \subseteq D_W$.

2.3.2 Datasets

NN models require relatively large datasets to train on, for better performance. Readily available datasets with ample training pairs are the CNN/DAILYMAIL and NEWSROOM summarization corpus.

– **CNN/DAILYMAIL:** It was originally built by Hermann et al. (2015) for the task of Question Answering (QA) but modified by Nallapati et al. (2016) for summarization. The original dataset by Hermann et al. (2015) contains human generated abstractive summary bullets from news-stories in the *CNN* and *DailyMail* websites as questions (with one of the entities hidden), and stories as the corresponding passages from which the QA system is expected to answer the fill-in-the-blank question. Nallapati et al. (2016), restored all summary bullets of each story in the original order to obtain a multi-sentence summary, where each bullet is treated as a sentence. The resulting summarization corpus contains ***286,817 training pairs, 13,368 validation pairs*** and ***11,487 test pairs***. The source documents in the training set have 766 words spanning 30 sentences on average while the summaries consist of 53 words and 4 sentences

(Nallapati et al., 2016). Two versions of the data were released – *anonymized* which has been preprocessed to replace each name entity e.g., *The United States of America*, with a unique id, e.g. `@entity0` and a *non-anonymized* version containing the original text. Work presented in this thesis, make use of the *non-anonymized* version of the released data. Figure 2.1 provides an example from this dataset.

Article: [..] the palestinian authority officially became the 123rd member of the international criminal court on wednesday, a step that gives the court jurisdiction over alleged crimes in palestinian territories. the formal accession was marked with a ceremony at the hague, in the netherlands, where the court is based. the palestinians signed the icc 's founding rome statute in january, when they also accepted its jurisdiction over alleged crimes committed " in the occupied palestinian territory, including east jerusalem, since june 13, 2014. [...] israel and the united states, neither of which is an icc member, opposed the palestinians ' efforts to join the body . [...] **Summary:** membership gives the icc jurisdiction over alleged crimes committed in palestinian territories since last june . israel and the united states opposed the move , which could open the door to war crimes investigations against israelis .'
Article: A star marks the epicentre of a magnitude-7.1 earthquake that struck off the east coast of New Zealand at 4.38am . (US Geological Survey) Tsunami waves of 30cm have hit the east coast of the North Island following a massive undersea earthquake and Civil Defence says the worst has passed. People are still being urged to stay away from the coast and waterways from Northland down to south of Gisborne on Friday morning [...] **Summary:** An earthquake of magnitude 7.2 has struck off the east coast of New Zealand early on Friday morning .

Figure 2.1: Example of a test pair from the **CNN/DM** and **NEWSROOM** datasets.

– **NEWSROOM:** 92% of this dataset was recently released by Connected Experiences Lab[1] (Grusky et al., 2018). The NEWSROOM corpus contains over ***1.3M*** news articles together with various metadata information such as the title, summary, coverage and compression ratio. It was collected over the web and contains summaries from different news sources. The dataset is split into ***995,041 training pairs, 152,479 validation pairs*** and ***152,479 test pairs***. The source documents in the training set have

[1] `https://summari.es`

658 words while the summaries consist of 27 words.

2.3.3 Data Filtering

As seen in Figure 2.1, the summaries in the available datasets are abstractive. Hence it is customary and necessary to create an extractive summarization dataset from the abstractive dataset (Chen and Bansal, 2018; Nallapati et al., 2017). We observe however, that some summaries contain information not found in the corresponding document as can be seen in Figure 2.2.

It is imperative that such pairs are filtered out from the corpus, for two (2) main reasons:

1. Since the extractive labels are usually obtained by performing some *n-gram* overlap matching (details in section 2.3.5), the presence of noisy data samples would result in inaccurate extractive labels.

2. As stated in Section 1.3, leaving such disparate data pairs (for example, see Figure 2.2), where the summary cannot be directly deduced from the document poses challenges for the NN to learn from.

Article: world-renowned chef, author and emmy winning television personality anthony bourdain visits quebec in the next episode of " anthony bourdain : parts unknown, " airing sunday, may 5, at 9 p.m. et. follow the show on twitter and facebook. **Summary:** 11 things to know about quebec. o canada! our home and delicious land.'
Article: each day, cnn producers select a user-submitted photo to be our uniquely you: design of the day entry. click through the gallery above to see creative shots from home decor enthusiasts around the world, and be sure to come back every day for a new image. have inspiring decor ideas from your own home to share? submit them for the gallery at cnn ireport !' **Summary:** see more ireport galleries : travel photos , other worldly landscapes . follow us on twitter @cnnliving . follow us on facebook .'

Figure 2.2: Examples of *noisy* summaries with no corresponding context from the document in a CNN/DM training sample.

We term all disparate pairs as *noisy*, and as such filter out all *noisy* samples as illustrated

in Figure 2.2[2]. In our work, we consider a reference summary R_j as *noisy* if it has zero bigram overlap with the corresponding document D_j, excluding stop words (see equation 2.6).

$$\#bigram(D_j, R_j) == 0. \tag{2.6}$$

The result of this data filtering process is a sub-dataset for summarization with {*document, summary*} pairs, where each *summary* can be directly deduced from the corresponding *document*.

2.3.4 Preprocessing

To avoid errors during *sentence level tokenization* (splitting of text into individual sentences), we ensured words were separated by a space. Special cases where other symbols (e.g '?', '!') marked the end of a sentence, or where a period did not indicate a sentence boundary (as in decimals and abbreviations) were similarly handled. After preprocessing, sentence tokenization was performed automatically by the NLTK library[3].

2.3.5 Data Tuning

The goal here is to create an extractive summarization dataset from the available abstractive summarization dataset. That is, we intend to *tune* the abstractive dataset into an extractive one. Hence, given {*document, summaries*} pairs as in Figure 2.1, can we convert these to {*document, binary labels*} pairs, where *binary labels* refers to a list of 0s and 1s representing a deletion or extraction decision for each sentence in the document?

Summaries in the dataset are referred to as *reference* or *ground-truth* summaries. We hypothesize that each reference summary sentence originates from at least one document sentence. The task therefore, is to identify the most-likely document sentence from which

[2]Filtering is used only for the training set, to ensure that evaluation comparisons on the test set with existing models are fair. We found that about 20% of the training data contained *noisy* samples.

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

the reference summary was abstracted. Different approaches have been used to solve this in literature. Nallapati et al. (2017), greedily selects sentences that maximize the ROUGE (Lin, 2004) score (an evaluation metric with details in section 2.3.8). Chen and Bansal (2018) calculates the individual reference sentence-level score as per its similarity with each sentence in the corresponding document using the ROUGE-L$_{recall}$ score. Our solution is similar to Nallapati et al. (2017) and Chen and Bansal (2018). For each ground-truth summary sentence s_t, we find the most similar document sentence d_t, using:

$$d_t = argmax_i(\#bigram(d_i, s_t)). \tag{2.7}$$

For each sentence in the reference summary, we perform bigram matching with all the sentences in the document. The t^{th} sentence in the document with the highest bigram overlap with the reference summary sentence s_t, is then labelled 1. For example, if the reference summary contains three (3) sentences, we find the three (3) corresponding document sentences respectively and label each as 1. All other sentences in that document are labelled 0. Additionally, for every time both words in the set of bigrams-overlap are stopwords, we decrement the bigram count by 1, for example, *(on, the)* is an invalid bigram-overlap while *(the, President)* is valid. We do this to capture the more important similarities instead of trivial ones.

We refer to the resulting *tuned* dataset with $\{$*document, binary labels*$\}$ suitable for extractive summarization as the *extractive trainer*. The extractive trainer will serve as input to our NN discussed in subsection 2.3.6. An extractive trainer with better performance has a higher chance of resulting in a better performing model. Hence, we evaluate our extractive trainer against Nallapati et al. (2017)'s which is our foundation. Results presented in Table 2.1 in section 2.3.8 (Results and Evaluation) show that our method is able to produce an extractive summarization dataset with improved accuracy.

2.3.6 NN Architecture

Our NN provides an answer to **RQ1:** How can we identify parts of a text that are the most relevant for an extractive summary with improved accuracy?

The built extractive trainer simplifies **RQ1** to **RQ1a:** Given an extractive summarization training set with {*document, binary labels*} pairs, how can we build a NN model capable of learning from this training set and then classifying an unlabelled document with accurate binary labels, where each label represents an extraction or deletion decision per document sentence? We attempt to provide a technical solution with the NN architecture illustrated in Figure 2.3 and explained in detail through the following subsections.

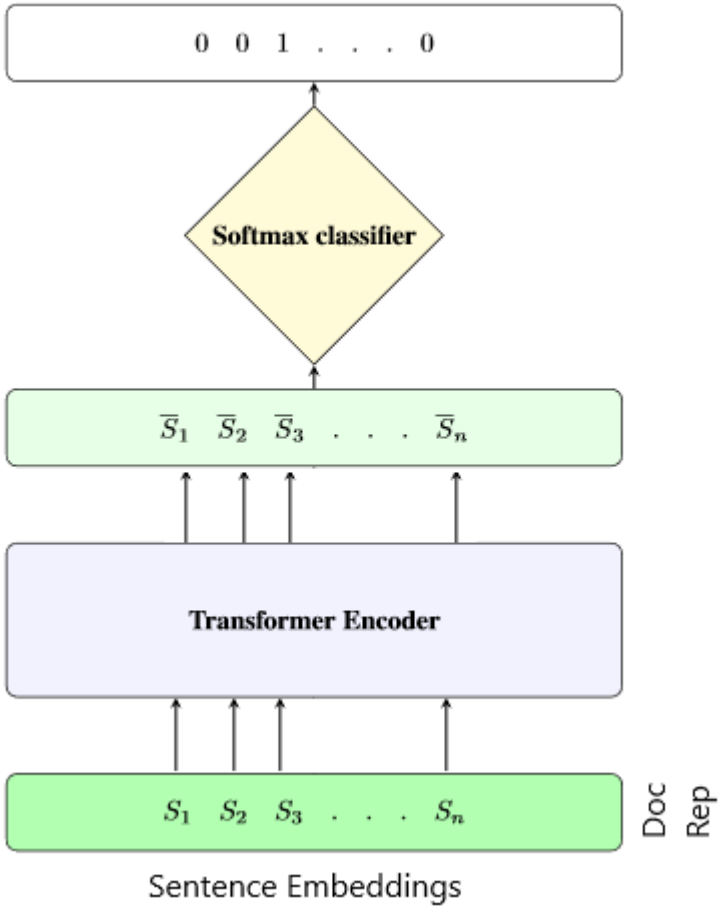

Figure 2.3: NN model architecture for single document extractive summarization.

Embedding

NN models require real value numbers, not text, as input. Hence, we need to transform our input text into a vector of numbers. This is the function of the *embedding* layer. It maps the input to a real-valued space, such that these real-valued numbers are meaningful. When the vector representation of words are visualized in space, semantically related words are spatially close together. The *vector representation* of a word is referred to as its *word embedding*. There exists a variety of word embeddings in literature, such as, `GloVe` [4](Global Vectors for word representation) (Pennington et al., 2014), `word2Vec` (Mikolov et al., 2013), ELMO (Peters et al., 2018) and BERT (Bi-directional Encoder Transformer) (Devlin et al., 2019) learnt through different interesting algorithms. We direct readers to the literature for each of these word embedding types and focus on the one used in this thesis – `GloVe`. The architectural decision to use `GloVe` was based on a few experiments on this specific task.

Training of `GloVe` embedding is performed on global word-word co-occurrence statistics from a corpus. The main intuition underlying the model is the observation that ratios of word-word co-occurrence probabilities have the potential for encoding some form of meaning. The training objective of GloVe is to learn word vectors such that their dot product equals the logarithm of the words' probability of co-occurrence. Owing to the fact that the logarithm of a ratio equals the difference of logarithms, this objective associates (the logarithm of) ratios of co-occurrence probabilities with vector differences in the word vector space. Because these ratios can encode some form of meaning, this information gets encoded as vector differences as well (Pennington et al., 2014).

`GloVe` word embeddings come in different dimensions. The vector representation of a word can be of length 50, 200, 300 or 500. The length of the vector representation is referred to as the *embedding dimensionality*. Theoretically, larger vectors can store more information since they have more possible states. However, experiments show that not

[4]`https://nlp.stanford.edu/projects/glove/`

much is gained beyond a size of 300-500. Our model was implemented using pretrained 300-dimensional `gloVe` word embeddings. These embeddings were trained on 6 billion words with a vocabulary size of 400k from Wikipedia[5] and Gigaword[6].

Let D_j be a document with n sentences and S_i be a sentence with m maximum number of words. First we obtain the sentence embedding by averaging the word embeddings and then concatenate the sentence embeddings to obtain the vector representation of the document as in equation 2.8:

$$\begin{aligned} S_i &= 1/m \sum_{i=1}^{m} w_i, \\ D_j &= S_1 \| S_2 \| \ldots \| S_n. \end{aligned} \tag{2.8}$$

Encoding

The function of the *encoding layer* performed by an *encoder* is to learn *latent* or *hidden* representations of the input. In this case, the input is the set of document embeddings in Subsection 2.3.6. These representations are not directly understandable, but are proven to hold important features of the input (Li et al., 2016). There exists a variety of encoders – AUTOENCODERS (Pu et al., 2016), CNN (Convolutional Neural Network), RNN (Recurrent Neural Network) (Cho et al., 2014), LSTM (Long Short Term Memory), GRU (Gated Recurrent Units) (Chung et al., 2014) and most recently TRANSFORMERS (Vaswani et al., 2017b, 2018). These different encoders all work relatively well, based on specific tasks. They also have certain drawbacks, which later encoder models attempt to resolve. We direct readers to the cited literature for detailed explanations of these different encoder types. We, however, provide more explanations on the TRANSFORMER encoder, as this is our encoder choice for this task.

The **TRANSFORMER Encoder** differs from other encoder types mainly because of its

[5] `https://dumps.wikimedia.org/enwiki/20140102/`

[6] `https://catalog.ldc.upenn.edu/LDC2011T07`

Self Attention mechanism. First we explain *Attention*, how it is generally applied, then we highlight the process of *Self Attention* and how it is implemented in the TRANSFORMER. Finally, we give the architecture of the TRANSFORMER network.

Attention in NN is analogous to attention in human reading. Humans often pay attention to certain parts of a text while reading because these parts might hold information necessary to understand the whole text. Now, in NN applications, it is often required to generate one sequence from another (for example, translating a text from English to German as in Neural Machine Translation (NMT) (Neubig, 2017) or generating a summary from a document as in summarization). Tasks of this nature are referred to as *Sequence to Sequence* (SEQ2SEQ) problems (Sutskever et al., 2014) and are usually solved using an *encoder – decoder* framework implemented using a RNN. The concept is to encode the input into a hidden state that holds information about the entire sequence and then decode this hidden state into the desired output. An attention mechanism is introduced between the encoder and decoder resulting in; *encode – attend – decode* (Bahdanau et al., 2015; Luong et al., 2015). The idea is to guide the decoder to important parts of the hidden state produced by the encoder. In other words, the decoder should pay attention to certain parts. This gives rise to the question – **How does the decoder network know what parts should be *attended* to?** The input sequence is processed per timestep, t, where the number of timesteps, N, is the length of the sequence (for example, number of words in a sentence).

Given the encoder hidden states, $h_1, h_2, ..., h_N \in \mathbb{R}^h$, and the decoder hidden state, $s_t \in \mathbb{R}^h$, we compute the attention score e^t, for timestep t, with:

$$e^t = [s_t^T h_1, ..., s_t^T h_N] \in \mathbb{R}^N. \tag{2.9}$$

Equation 2.9 scores how well each encoded input, h_i, matches the current output of the decoder, s_t. These attention scores e^t, are normalized in Equation 2.10 using a softmax function, to obtain the attention distribution α^t:

$$\alpha^t = softmax(e^t) \in \mathbb{R}^N. \tag{2.10}$$

The final representation, c_t, known as the *context vector*, passed by the encoder is the weighted sum of the encoder hidden states, where the weights are the attention distribution α^t, computed with:

$$c_t = \sum_{i=1}^{N} \alpha_i^t h_i \in \mathbb{R}^h \tag{2.11}$$

Lastly, the *context vector* c_t, is concatenated with the decoder hidden state s_t:

$$[c_t; s_t] \in \mathbb{R}^{2h}. \tag{2.12}$$

Having explained the fundamentals of attention mechanism, it becomes necessary to ask the question – **How does Attention differ from Self-Attention?** Self-attention, sometimes called intra-attention is an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence (Vaswani et al., 2017b). On a high level of abstraction, we see that attention in SEQ2SEQ models are basically implemented between two layers – the encoder *and* the decoder, as a unit. However, for ***Self-Attention*** the implementation is within a layer – the encoder *and/or* decoder respectively. When self- attention is applied - the self-attention mechanism looks at the inputs of the same layer where it is being applied.

How is Self-Attention implemented in a TRANSFORMER? Self-attention could be applied in different levels of granularity based on the input. That is, it could be applied to each word in an input sentence, or each sentence in an input document. In this specific use case, we want to encode a document of n sentences. Here, the goal of self-attention is to learn dependencies between sentences in the document and use that information to capture the internal structure of the document (Cloud, 2018).

Figure 2.4 illustrates the architecture of the TRANSFORMER encoder. For each sentence

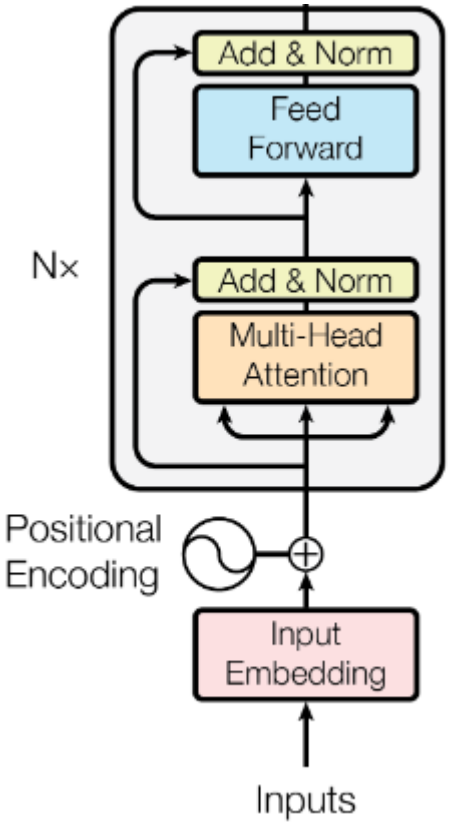

Figure 2.4: TRANSFORMER encoder model architecture (Vaswani et al., 2017b).

embedding S_i (equation 2.8), a *positional encoding*, $e_p(S_i)$ is added:

$$\widetilde{S}_i = S_i + e_p(S_i), \tag{2.13}$$

where $e_p(S_i)$ is learned during training[7]. The idea is that the contribution of a sentence to a document is a summation of the sentence salience (S_i) and position ($e_p(S_i)$) in the document (Lebanoff et al., 2019b). Next, from $\widetilde{S}_i$, three vectors – *Query Vector (Q), Key Vector (K)* and *Value Vector (V)* from each of the document vectors (in this case, the embedding of each sentence) are created. The *Query* and *Key Vectors* both have a dimension d_k while the *Value Vector* has a dimension, d_v[8]. These vectors are created by multiplying the document

[7]The authors (Vaswani et al., 2017b) experimented with fixed and learned positional embeddings, with identical results, but preferred the former. We chose to use the learned positional embeddings instead.

[8]$d_k = d_v = d_{model}/h = 64$, where $d_{model} = 512$ and h the number of attention layers is 8 (Vaswani et al., 2017b)

embedding by three matrices. The matrices are simply parameters learned during training of the network (Giacaglia, 2019). The score for each sentence is computed by applying equation 2.14 (Vaswani et al., 2017b):

$$Attention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_k}})V, \tag{2.14}$$

where QK^T is the dot product between the *Key* and *Query* vectors. QK^T scores each sentence of the input document against a particular sentence being processed. The score determines how much focus to place on other parts of the input document as we encode a sentence at a certain position. Dividing this score by $\sqrt{d_k}$ leads to having more stable gradients (Giacaglia, 2019). Finally passing the result through a $softmax$ operation normalizes the scores allowing them to be treated as probabilities. Because eight (8) attention layers are applied (Vaswani et al., 2017b), this results in a set of eight (8) QKV vectors. In order to arrive at one vector which is passed to the *feed-forward network layer* (FFN), these eight (8) attention vectors are concatenated and multiplied with another learned weight matrix, W^O as in equation 2.15 (Vaswani et al., 2017b):

$$\begin{aligned} MultiHead(Q,K,V) &= Concat(head_1,...,head_h)W^O, \\ head_i &= Attention(QW_i^Q, KW_i^K, VW_i^V). \end{aligned} \tag{2.15}$$

This is referred to as the **Multi-head attention** mechanism of the TRANSFORMER. In a nutshell, the multi-head attention expands the model's ability to focus on different positions of the input embedding vector (Jay, 2018). Lastly, the output of the multi-head attention is passed to the FFN to produce the final encoding vector by the TRANSFORMER, given in equation 2.16:

$$FFN(x) = max(0, xW_1 + b_1)W_2 + b_2, \tag{2.16}$$

where W_1, W_2 are learned weights and b_1, b_2 are biases. The biases in FFN help to add flexibility to the network. As seen in the TRANSFORMER model architecture in Figure 2.4, the output of each sub-layer (Multi-head and FFN) are layer-normalized (Ba et al., 2016) before being passed to sub-layers above. That is, the output of each sub-layer is $LayerNorm(x + Sublayer(x))$, where $Sublayer(x)$ is the function implemented by the sub-layer itself (Vaswani et al., 2017b).

The TRANSFORMER encoder is made up of six (6) stacked layers. Each layer contains a *Mutihead Self-Attention* and FFN sub-layers as illustrated in Figure 2.4.

Classifying

Given the learned representations from document embedding to encoding, we aim to use these representations in reaching a binary decision of extracting or not extracting a document sentence. We estimate the predicted probability, y_i^p, of extracting the i^{th} sentence in a document using a *Softmax Classifier* with equation 2.17:

$$y_i^p = softmax(WS_i' + b), \tag{2.17}$$

where W and b are trainable parameters representing the weights and bias of the model and S_i' is the TRANSFORMER encoded representation of the i^{th} document sentence. The input to the *softmax classifier* is the output of our TRANSFORMER encoder (S_i') described in section 2.3.6. The *softmax* function outputs probabilities between 0 and 1 for each of the document's sentences. Although highly subjective and dataset specific, since the reference summaries are typically 3 or 4 sentences long, we extract the top three (3) sentences by default, but proceed to additionally extract a 4^{th} sentence if the probability score from the *softmax* function is greater than 0.55[9].

[9]We arrived at this score after experiments on the datasets.

Training

The model is trained by minimizing the cross-entropy loss (Zhang and Sabuncu, 2018), L:

$$L = -(y_t log(y_p) + (1 - y_t) log(1 - y_p)), \tag{2.18}$$

between the predicted probabilities, y^p from equation 2.17 and the true sentence binary labels, y_t.

2.3.7 Implementation Details

We present the technologies used for implementing the aforementioned method and architecture.

Programming Language and Libraries

The entire thesis, including this chapter, was implemented using the Python programming language (v 3.6)[10]. We employed a handful of open source libraries, but primarily, Tensorflow (v1.10) [11].

Mini-batch size

We split our samples into mini-batches, each with a size of 100, after experimenting with various mini-batch sizes. A mini-batch size represents the number of samples that are processed by the NN model at a time. The size of a mini-batch impacts the number of iterations needed to go through the samples, which impacts the training time. It also has an effect on the performance of the model. The effects of trade-offs between smaller mini-batch sizes versus longer training time are nicely discussed in Hoffer et al. (2017) and Keskar et al. (2016).

[10] https://www.python.org/
[11] https://www.tensorflow.org/

Optimization Algorithm

Recall that our methodology involved the use of different *weight* matrices and vectors. These *weights* are randomly initialized. The goal of the NN training process is to update the weights such that when they are passed appropriately together with features (embeddings, encoded vectors) through the respective functions (TRANSFORMER encoder, *softmax* classifier), the outputs are closest to the reference. The process of updating the *weights* is referred to as *optimization*, for which several algorithms exist – *AdamOptimizer*, *Momentum* (Sutskever et al., 2013), *Mini-batch gradient descent* (Tieleman and Hinton, 2012), *Stochastic gradient descent* (Ruder, 2016). We used `AdamOptimizer` (Kingma and Ba, 2014) as the optimization algorithm with a fixed *learning rate* of 0.0001 [12]. The learning rate defines the magnitude at which the updates to the weights are made.

Hyperparameters

Hyperparameters are parameters whose values are set before the training process begins (for example, the mini-batch size, learning rate, number of hidden units, etc.). Some of our hyperparameter values have been stated in other subsections (see Subsection 2.3.7). The TRANSFORMER encoder was setup with the $transformer\_base$ hyperparameter setting from the `tensor2tensor library`[13] (Vaswani et al., 2018), but the hidden size and *dropout* were reset to 300 and 0.0 respectively. *Dropout* is a regularization technique patented by Hinton et al. (2016) for reducing overfitting in neural networks by preventing complex co-adaptations on training data (Baldi and Sadowski, 2013). Dropout is performed by dropping out units from the network along with their connections during training. Choosing which units to drop is usually random. We apply gradient clipping at 5.0 (Pascanu et al., 2013). It is important the gradients are clipped so they do not become extremely high (explode) during training. Clipping helps to ensure that a threshold is maintained, positively impacting the performance and training time (Pascanu et al., 2013).

[12] We experimented with decaying the learning rate, with no significant performance improvement, only a longer training time.

[13] `https://github.com/tensorflow/tensor2tensor`

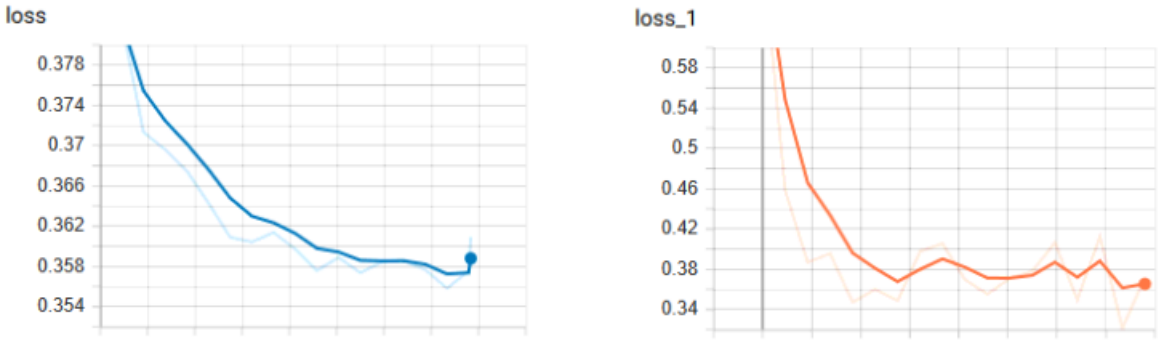

Figure 2.5: TENSORBOARD visualization of our loss scores during training (ORANGE line) and validation (BLUE line).

Training and Inference

We perform training and validation concurrently. This is done so we can tune the hyper-parameters on the validation set and more importantly implement *early stopping* when the validation loss does not decrease after a certain number of epochs (5 for this task). Early stopping is a technique to curtail *overfitting*. Overfitting occurs when the model performs well on training samples, but poorly on unseen samples such as test and validation samples. We also say that the model fails to *generalize* when this happens. We trained our models on a Nvidia TITAN X GPU card with 12G RAM. It took an average of 60 minutes to train our model on each dataset. Figure 2.5 visualizes the learning curve of our model during training and validation as per the loss score. In the validation loss curve, after a steady decline in the loss score for some time, it then starts increasing. It is at this point, training is stopped.

2.3.8 Results and Evaluation

We present some samples of our model output in Appendices A and B. Following previous works (Chen and Bansal, 2018; Nallapati et al., 2017; See et al., 2017), we evaluate our single models on both datasets – CNN/DM and NEWSROOM discussed in section 2.3.2 using standard ROUGE-1, ROUGE-2 and ROUGE-L evaluation metrics (Lin, 2004).

ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation. It calculates the appropriate n-gram word-overlap between the reference and system summaries. ROUGE-1

Table 2.1: ROUGE -F1 (%) scores (with 95% confidence interval) of extractive trainers using CNN/DM dataset.

Extractive Trainer	R-1	R-2	R-L
Ours	49.5 [49.2 - 49.7]	27.8 [27.6 - 28.1]	45.8 [45.6 - 46.1]
Ours + filter	**51.4** [50.9 - 51.8]	**31.7** [31.4 - 31.8]	**50.3** [49.4 - 50.6]
(Nallapati et al., 2017)	48.4 [47.4 - 49.4]	27.5 [26.3 - 28.7]	44.4 [43.4 - 45.6]

Table 2.2: Average ROUGE-F1 (%) scores of various extractive models on the **CNN/DM** test set. The first and second sections show LEAD-3 and baseline model scores respectively.

Extractive Model	R-1	R-2	R-L
LEAD (See et al., 2017)	40.3	17.7	36.5
LEAD (Narayan et al., 2018)	39.6	17.7	36.2
LEAD (ours)	40.1	17.6	36.0
SUMMARUNNER(Nallapati et al., 2017)	39.6	16.2	35.3
REFRESH (Narayan et al., 2018)	40.0	18.2	36.6
NEUSUM (Zhou et al., 2018)	41.6	19.0	37.0
CONTENT SELECTOR (Gehrmann et al., 2018)	42.0	15.9	37.3
HIBERT$_M$ (Zhang et al., 2019b)	42.3	19.9	**38.8**
TRANS-ext	41.0	18.4	36.9
TRANS-ext + filter	**42.8**	**21.1**	38.4

Table 2.3: Average ROUGE-F1 (%) scores of various extractive models on the **NEWSROOM** released test set*.

Extractive Model	R-1	R-2	R-L
LEAD* (Grusky et al., 2018)	30.49	21.27	28.42
TextRank* (Barrios et al., 2016)	22.77	9.79	18.98
TRANS-ext	37.21	25.17	32.41
TRANS-ext + filter	**41.52**	**30.62**	**36.96**

* marks results taken from Grusky et al. (2018).

refers to the overlap of *unigrams* between the system summary and reference summary. ROUGE-2 refers to the overlap of *bigrams* between the system and reference summaries. ROUGE-L measures the longest matching sequence of words. It does not require consecu-

Table 2.4: 95% confidence intervals of our SDS extractive systems.

CNN/DM	R-1	R-2	R-L
TRANS-ext	41.0 ± 0.187	18.4 ± 0.149	36.9 ± 0.206
TRANS-ext + filter	42.8 ± 0.268	21.1 ± 0.262	38.4 ± 0.262
NEWSROOM			
TRANS-ext	37.21 ± 0.062	25.17 ± 0.033	32.41 ± 0.108
TRANS-ext + filter	41.52 ± 0.073	30.62 ± 0.045	36.96 ± 0.113

tive matches but only in-sequence matches that reflect sentence level word order. Since it automatically includes longest in-sequence common n-grams, a predefined n-gram length is not needed (Lin, 2004). We considered the widely used ROUGE evaluation F-measure for our evaluation task. F-measure is the harmonic mean of precision and recall:

$$f - measure = 2 \,.\, \frac{precision \,.\, recall}{precision \,+\, recall} \tag{2.19}$$

Recall is defined as the ratio of the number of units (sentences/words) of the system-generated summaries in common with the reference summaries to the total number of units in the reference summary. For a given sentence, it is evaluated as:

$$recall = \frac{|\{system\text{-}generated\ words\}| \cap |\{reference\ words\}|}{|\{reference\ words\}|}. \tag{2.20}$$

Precision is the ratio of the number of units of system-generated summaries in common with the reference summaries to the total number of units in the system-generated summaries. For a given sentence, it is evaluated as:

$$precision = \frac{|\{system\text{-}generated\ words\}| \cap |\{reference\ words\}|}{|\{system\text{-}generated\ words\}|} \tag{2.21}$$

ROUGE values were computed using the official `pyrouge` script[14] with options *-n 2 -w 1.2 -m -a -c 95*. We present the ROUGE evaluation scores on both datasets in Tables 2.2 and 2.3.

2.3.9 Analysis

In order to identify what differences in the competing scores for the extractive trainers are significant, we compare the **Confidence Intervals** (CI) for each mean. In Table 2.1, we show the 95% confidence intervals[15] to report statistical significance for doing meaningful comparison. Two systems can be judged as statistically significantly different if one of the two criteria holds: 1) their confidence intervals for the estimates of the means do not overlap at all, or 2) the two intervals overlap but neither contains the best estimate for the mean of the other (Schenker and Gentleman, 2001). Analyzing the confidence intervals of the different methods in Table 2.1, we see that our filtered extractive trainer has the best performance by considering both criteria. We also observe that our non-filtered extractive trainer does not differ significantly from Nallapati et al. (2017)'s by the first criterion as their confidence intervals overlap. However, according to the second criterion, our non-filtered extractive trainer outperforms Nallapati et al. (2017)'s with exception on the ROUGE-2 score.

Tables 2.2 and 2.3 present extractive summarization results on the CNN/DM and NEWSROOM datasets respectively. For clarity, we tabulate separately for each dataset. Our baseline non-filtered extractive (TRANS-ext) model is highly competitive with top models. Our TRANS-ext + filter produces an average of about +1 and +9 points across reported ROUGE variants on the CNN/DM and NEWSROOM datasets respectively, showing that our model does a better job at identifying the most salient parts of the document than existing state-of-the-art extractive models, thus adequately providing an answer to **RQ1:** *How can we identify parts of a text that are the most relevant for an extractive summary with improved*

[14] `https://github.com/andersjo/pyrouge/tree/master/tools/ROUGE-1.5.5`

[15] This is computed by specifying the option -c 95 when running the ROUGE script.

accuracy? raised at the onset. We observe the large margin in the NEWSROOM dataset results, as existing baselines are just the LEAD-3 and TEXTRANK of Barrios et al. (2016). The NEWSROOM dataset was recently released and is yet to be thoroughly explored, however it is a larger dataset and contains more diverse summaries as analyzed by Grusky et al. (2018).

Switching the TRANSFORMER encoder with a SEQ2SEQ encoder resulted in a drop of about 2 ROUGE points, showing that the TRANSFORMER encoder does learn features that add meaning to the vector representation of the input sequence. In Table 2.4, we report the 95% confidence intervals for ROUGE-1, ROUGE-2 and ROUGE-L to show the statistical significance of our results.

2.4 Related Work

There exists ample literature in extractive summarization (Bui et al., 2016; Cheng and Lapata, 2016; Dorr et al., 2003; Durrett et al., 2016; Jing and McKeown, 2000; Knight and Marcu, 2000). Section 2.2 discusses some works that are manual-feature engineering based. In this section we focus on single document extractive summarization models implemented with NN on the same dataset as ours – CNN/DM and NEWSROOM. Specifically, we review the baseline models our system compares to in Tables 2.2 and 2.3. Basically, for the different baseline models, we highlight differences/similarities in terms of the general approach, embeddings used, encoder architecture and dataset tuning.

Similar to our approach, SUMMARUNNER (Nallapati et al., 2017) treats extractive summarization as a sequence classification problem where each sentence is visited sequentially in the original document order and a binary decision is made in terms of whether or not it should be included in the summary. They use a 100-*d word2vec* word embedding. Encoding is done in a two-layer of GRU-based RECURRENT NEURAL NETWORK (RNN), one that runs on word level and the other on sentence level. We explain how the GRU-RNN works in Chapter 3. The final document encoding is as a non-linear transformation of the average

pooling of the concatenated hidden states of the sentence-level encoding. Different from our approach, data is not filtered in SUMMARUNNER. Although motivated by their data tuning approach (see Section 2.3.5), their criteria for selecting sentences for the extractive trainer differs from the criteria we employed. Their approach is based on the idea that the selected sentences from the document should be the ones that maximize the ROUGE score with respect to reference summaries while ours is based on the idea that each reference summary sentence originates from at least one document sentence, and that document sentence is the one with the maximum *bigram* overlap with the reference summary excluding non-informative words (i.e stop words).

REFRESH (Narayan et al., 2018) handles the task as a sequence ranking problem. They trained the word-embeddings with the SKIP-GRAM model from Mikolov et al. (2013), however known words were initialized with 200-*d* pretrained *word2vec* embeddings and unknown words with zero. Encoding was done in 2 layers similar to Nallapati et al. (2017). The first layer catered to sentence level encoding using CONVOLUTIONAL NEURAL NETWORKS (CNN) while document encoding was implemented with LSTMS. Ranking was done by a *softmax* classifier on top of an LSTM-RNN layer and trained using REINFORCEMENT LEARNING (external to the scope of this thesis). The top three (3) and four (4) sentences were selected for CNN and DM datasets respectively. As per generating an extractive dataset from the available data, during training they used the reference summaries to generate high scoring extracts and to estimate rewards for them, but during testing, they are used as reference summaries to evaluate the model.

NEUSUM (Zhou et al., 2018) learns to score and select sentences jointly for extractive summarization. They used pretrained 50-*d gloVe* word embeddings. They employ a hierarchical document encoder to represent the sentences in the input document. The document encoding was also done in two levels similar to Narayan et al. (2018), i.e., sentence level encoding and document level encoding each with a BIDIRECTIONAL GRU. To score the document sentences considering both their importance and partial output summary, the model

was imbued with two abilities: 1) remembering the information of previous selected sentences; 2) scoring the remaining document sentences based on both the previously selected sentences and the importance of remaining sentences. Therefore, they employed another GRU as the recurrent unit to remember the partial output summary, and use a Multi-Layer Perceptron (MLP) to score the document sentences, with a training objective to minimize the *Kullback-Lieber* (KL) divergence (Inan et al., 2016). In creating an extractive dataset for training they followed the same approach as Nallapati et al. (2017).

CONTENT SELECTOR (Gehrmann et al., 2018) formulated the extractive module of their summarization model as a sequence tagging problem, but this time on word-level not sentence-level as in our approach and Nallapati et al. (2017). Two (2) word-embeddings were concatenated – gloVe (100-*d*) and ELMO (1024-*d*) to form the input to a BIDIRECTIONAL LSTM encoder. The probability of extracting a word is then computed using a *sigmoid* function. They generate extractive training data by aligning the reference summaries to the document and define a word as copied if it is part of the longest possible sub-sequence of tokens.

In line with a popular approach, HIBERT$_M$ (Zhang et al., 2019b) also modelled extractive summarization task as a sequence labelling problem. The word-embeddings are randomly initialized, then trained using BERT (Devlin et al., 2019). On top of this are two layers of TRANSFORMER encoders for sentence and document encoding respectively. Classification is then performed by a *softmax* function. To create sentence level labels for extractive summarization, they used a strategy similar to Nallapati et al. (2017) – label the subset of sentences in a document that maximizes ROUGE.

We see from existing literature, that the approaches to the extractive summarization task are fundamentally similar with slight variations in architectural choices and of course hyperparameter settings. However, a significant difference in our model is the data-filtering stage. Though simple and might be considered trivial, it is crucial to the performance of the model, as any model is unable to learn efficiently from disparate data pairs.

2.5 Summary

In this chapter we presented our NN model in an attempt to answer **RQ1:** *How can we identify parts of a text that are the most relevant for an extractive summary with improved accuracy?* We stated the reasons for choosing the NN path in contrast to the traditional MFE based approaches after a review of existing MFE based literature on single document extractive summarization. We explained the rationale and yielded benefit of filtering our dataset before application on our model – if data is flawed, then model prediction will be flawed. We gave evidence of some flawed sample pairs in the dataset. We implemented an improved method for tuning the available abstractive summarization dataset for extractive summarization. Finally, we applied our filtered and tuned dataset on the NN model – a TRANSFORMER-based classifier. The TRANSFORMER comes with the merits of being able to learn syntactic and semantic relationship within its input, thereby giving a richer encoded representation to enable better classification by the *softmax* layer. Evaluation of our presented NN model on ROUGE metric showed improved performance over existing models. Part of this work was published at the *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Workshop on Neural Generation and Translation, pages 70 - 79. Hong Kong, China* (Egonmwan and Chali, 2019a).

Chapter 3

Paraphrase Generation

This chapter describes the first of two (2) sub-models for our abstractive summarization model – *Paraphrasing*. The goal is to provide an answer to **RQ2:** *How can we build a paraphrasing model to help with abstractive summarization?* But why do we need a paraphrasing model for abstractive summarization? Abstractive summaries are simply paraphrases of salient parts of the source text. Hence, if we can successfully identify these salient parts with the aid of the extractive summarization NN model described in the previous chapter (Chapter 2), then we can proceed to paraphrasing those extracts to produce abstractive summaries.

3.1 Introduction

Paraphrase generation aims to improve the clarity of a sentence by using different wording, while retaining the meaning of the original sentence. For example, the sentence – *"Stuffiness and elevated temperature are signs of the flu"* can be paraphrased as *"Symptoms of the flu include fever and nasal congestion"*. Paraphrasing is a key abstraction technique used in Natural Language Processing (NLP). While capable of generating novel words, it also learns to compress or remove unnecessary words along the way. Thus, it gainfully lends itself to abstractive summarization (Chen and Bansal, 2018; Gehrmann et al., 2018). Paraphrasing requires a deep understanding of the underlying semantics in a sentence. Hence knowledge gained from a paraphrasing model can be used in other tasks like, Question Generation (QG) (Song et al., 2018) and Machine Reading Comprehension (MRC)

(Dong et al., 2017). Paraphrases can also be used as simpler alternatives to input sentences for Machine Translation (MT) (Callison-Burch et al., 2006) as well as the evaluation of Natural Language Generation (NLG) texts (Apidianaki et al., 2018).

Despite the importance of paraphrase generation, there has been relatively little prior work in the literature. A larger amount of work exists on the paraphrase detection problem (Gupta et al., 2018). Existing methods for generating paraphrases fall into one of these broad categories – rule-based (McKeown, 1983), SEQ2SEQ (Prakash et al., 2016), reinforcement learning (Li et al., 2018b), deep generative models (Iyyer et al., 2018) and a varied combination (Gupta et al., 2018; Mallinson et al., 2017) of the later three (3). In our work, we propose a novel framework for paraphrase generation that utilizes the TRANSFORMER model (Vaswani et al., 2017b) and the SEQ2SEQ model (Sutskever et al., 2014), specifically Gated Recurrent Units (GRU) (Cho et al., 2014).

3.2 Methodology

In this section, we begin with a formal definition of *paraphrase generation*. Next, we describe the datasets used for training and testing our model and the necessary preprocessing steps performed on the data. We then present our architecture of the NN model for the task of paraphrase generation, explaining the technologies employed and implementation details. Sections 3.2.7 and 3.2.8 evaluates and analyzes the output of our model. Finally we discuss published work related to ours, on the task of paraphrase generation.

3.2.1 Task Definition

Given an input sentence $S = (s_1, ..., s_n)$ with n words, the task is to generate an alternative output sentence $Y = (y_1, ..., y_m) \mid \exists y_i \notin S$ with m words that convey similar semantics as S.

3.2.2 Datasets

We used two (2) datasets for training, validating and testing our framework:

– **MSCOCO** (Lin et al., 2014)[16]: We use the 2014 version which contains human annotated captions of over 120,000 images. Each image contains five captions from five different annotators. This dataset is a standard benchmark dataset for the image caption generation task. In the majority of the cases, annotators describe the most prominent object/action in an image, which makes this dataset suitable for the paraphrase generation task. Following Gupta et al. (2018), from the five captions accompanying each image, we randomly omit one caption, and use the other four as training instances (by creating two source-reference pairs). This results in over 400, 000 sample pairs. For consistency with Gupta et al. (2018) and Prakash et al. (2016), long captions are reduced to the first 15 words. We randomly choose ***200,000 pairs for training, 10,000 for validating*** and ***10,000 for testing***.

– **QUORA**: This is a more recent dataset released in January 2017[17]. The dataset consists of over 400,000 lines of potential duplicate question pairs. Each line contains IDs for each question in the pair, the full text for each question, and a binary value that indicates whether the questions in the pair are truly duplicates of each other. Wherever the binary value is 1, the question in the pair are not duplicates; they are rather paraphrases of each other (Gupta et al., 2018). We choose all such question pairs with binary value 1. There are a total of 155,000 such questions. In line with (Gupta et al., 2018), we also evaluate our model on ***50,000, 100,000 and 150,000 training*** dataset sizes and ***4,000 test pairs*** of paraphrases.

[16]http://cocodataset.org/

[17]https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs

3.2.3 Preprocessing

We removed stop words (e.g *and, or, it*) as these words add very little to the meaning of a sentence. After removal of stop words, we perform *word tokenization* (splitting a sentence into words). Word level tokenization was automatically done by the NLTK tool kit[3].

3.2.4 NN Architecture

We describe our NN architecture to solve the task of paraphrase generation. First, we embed, then encode, and lastly decode into the output sentence. Each process is explained below.

Embedding

For each word in a sentence, we use its 300-*d* `GloVe` embedding (see section 2.3.6). The sentence embedding is then the concatenation of the individual word embeddings. Let S_i be a sentence with n words with each word having an embedding representation w_i', then the emebedding, S_i' is given by:

$$S_i' = w_1' \| w_2' \| \ldots \| w_n'. \tag{3.1}$$

Encoding

Our model intrinsically differs from existing paraphrase generation models in the encoding layer. In contrast to existing models, we stack two (2) encoding layers with differing architectures – TRANSFORMER and GRU-RNN. Stacking the TRANSFORMER and GRU-RNN aims at combining the representational power of RNNs and self-attention. The idea is to enrich a set of stateful representations by cascading a feature extractor with a focus on vertical mapping (Chen et al., 2018). Additionally, we hypothesize that the probability of the decoder generating better quality paraphrases rests extensively on how well the information was captured by the encoder, hence the need for rich representations. Similar to Chen et al.

(2018), we observed that the TRANSFORMER and SEQ2SEQ model complement themselves adequately, making for a richer encoded vector representation.

The first encoding layer is the **TRANSFORMER encoder**. The input to the TRANSFORMER encoder are the respective sentence embeddings from Equation 3.1. We refer readers to Section 2.3.6 for details on the TRANSFORMER encoder.

The second layer of the encoder is a **GRU-RNN encoder** whose input is the output of the TRANSFORMER. A GRU is a special kind of RNN developed to solve a peculiar problem in basic RNNs. So first, we explain how an RNN works. Then we highlight the major challenge with vanilla RNNs and how the GRU-RNN attempts to tackle the challenge.

Traditional NNs are often referred to as *Feed Forward Networks* (FFNs). FFNs have a minimum of three (3) layers – input, hidden and output, with connections going from one layer to the next, immediately above or below it. Connections do not exist within the same layer. **RNN**s generalise FFNs to be able to model sequential data. FFNs take an input (e.g. an image) and immediately produce an output (e.g. probabilities of different classes). RNNs, on the other hand, consider the data sequentially, and can remember what they have seen earlier in the sequence to help interpret elements from later in the sequence. The presence of loops in RNNs, as illustrated in Figure 3.1 enable this functionality.

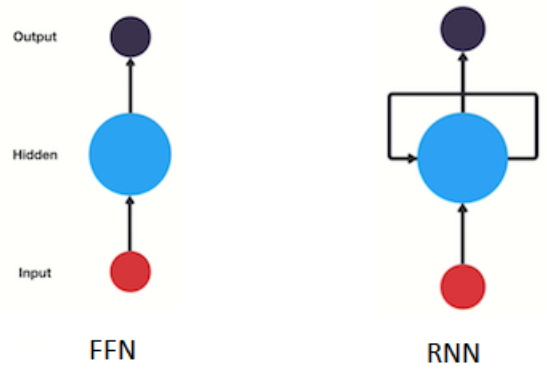

Figure 3.1: FNN vs RNN (Phi, 2018).

An unrolled RNN is illustrated in Figure 3.2. In Figure 3.2, h_t, which is the output of A, is simply a non-linear function of x_t and h_{t-1} as in Equation 3.2.

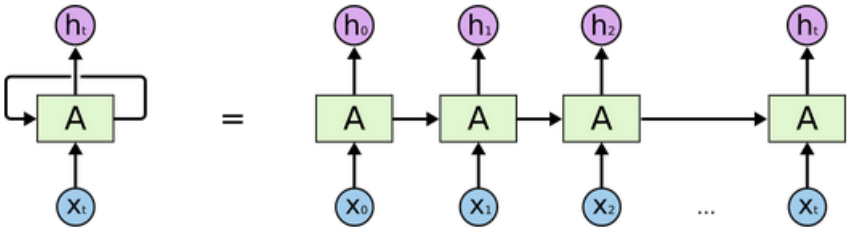

Figure 3.2: An unrolled RNN (Olah, 2015).

$$h_t = f(x_t, h_{t-1}). \tag{3.2}$$

At each time step t, the RNN uses the hidden state from the previous timestep $t-1$, and the current input x_t, to compute the hidden state, h_t, for that timestep. It uses information from the past (h_{t-1}) to capture the present. Hence, its great utility in sequential problems. However, this major advantage of RNNs being able to capture past information leads also to its major challenge – *short term memory*, that is, when the sequence is too long, it tends to forget information from way back in the past. Let us illustrate this with an example taken from Brundyn (2018). Imagine we are trying to predict the next word in the following piece of text:

```
I was born in France, but I have been working in South Africa working
for ... (another 200 words) ... Therefore my mother tongue is:
```

Figure 3.3: Long text sequence to illustrate the short term memory problem of RNNs (Brundyn, 2018).

In the illustration in Figure 3.3, the RNN may be able to detect that it should predict the name of a language but it may also leave out important information from the beginning of the text like the fact that the speaker grew up in France (Brundyn, 2018). The *short term memory* problem in RNNs is a consequence of the *vanishing gradient problem*. This occurs when the gradient for updating the network's weights become so small (i.e vanishing), it

has no impact on the weights and hence the learning process of the NN. In an attempt to lessen this effect, variants of RNN were introduced – LSTM and GRU. I refer to GRU as a more computationally effective variant of LSTM. The GRU (Cho et al., 2014; Chung et al., 2014) adds two (2) gates to the basic RNN – an *update gate* computed as:

$$z = \sigma(x_t U^z + h_{t-1} W^z), \tag{3.3}$$

which decides what information to discard and what new information to add, and a *reset gate* computed as:

$$r = \sigma(x_t U^r + h_{t-1} W^r), \tag{3.4}$$

which is used to decide how much past information to forget. The current hidden state, h_t, is then computed with equations:

$$h = tanh(x_t U^h + (h_{t-1} \odot r) W^h), \tag{3.5}$$

$$h_t = (1-z) \odot h + z \odot h_{t-1}, \tag{3.6}$$

where W and U from equations 3.3 - 3.6 are all of the network parameters learned during training, h_{t-1} is the hidden state at the previous timestep, x_t is the input vector (in this case, the output of the TRANSFORMER encoder) and $\odot$ represents the Hadamard (or element-wise) (Million, 2007) product. The GRU-RNN is nicely illustrated in figure 3.4.

Decoding

The initial state of the decoder is the output of the GRU-RNN encoder, which is a fixed length latent vector. The goal of the decoder is to map this encoded vector into a readable sequence of varied length. We also use a GRU-RNN architecture for our decoder. It works fundamentally the same as described in section 3.2.4 with a few essential differences. At each timestep, the decoder produces two outputs: i) a hidden state s_t, computed in the same

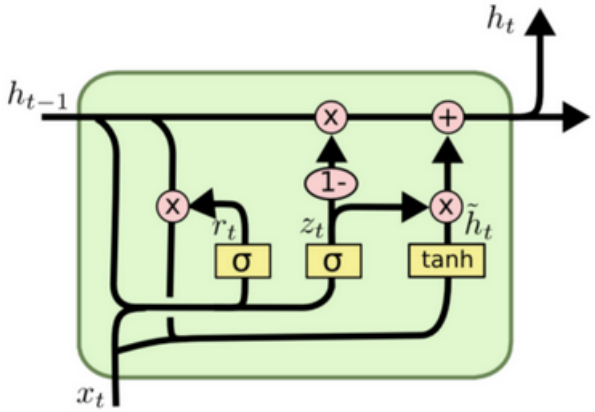

Figure 3.4: GRU architecture (Rathor, 2018).

fashion as h_t in equation 3.6 and ii) a readable output w_t^p, for example, a word, if the output sequence is a sentence. w_t^p is the word with the highest probability y_t^p, where y_t^p is derived by taking a softmax of s_t over all the words in the target sequence vocabulary given by equation 3.7:

$$y_t^p = softmax(W^s s_t), \tag{3.7}$$

where, W^s is a learned weight. Equation 3.7 calculates the probability that each word in the target vocabulary set is the output word for that timestep. In contrast to the encoder, where x_t (equations 3.3 - 3.5) at each time step is always the input sequence, for the decoder, x_t could be the target output word y_t^w, during training or the previous output word y_{t-1}^p, during inference.

Training

The model is trained by minimizing the error between the probability of the predicted output word y_t^p, and the *id* y_t^d, of the target output word, using the SEQ2SEQ loss from the `tensorflow library`[18].

[18] `https://www.tensorflow.org/api_docs/python/tf/contrib/seq2seq/sequence_loss`

3.2.5 Implementation Details

For the TRANSFORMER encoder, we used the *transformer_base* hyperparameter setting from the `tensor2tensor library`[19] (Vaswani et al., 2018), but set the hidden size to 300. Both the GRU-RNN encoder and decoder contain 300 hidden units. We set dropout to 0.0 and 0.7 for the MSCOCO and QUORA datasets respectively. We used a large dropout for QUORA because the model tends to over-fit to the training set (see figure 3.6).

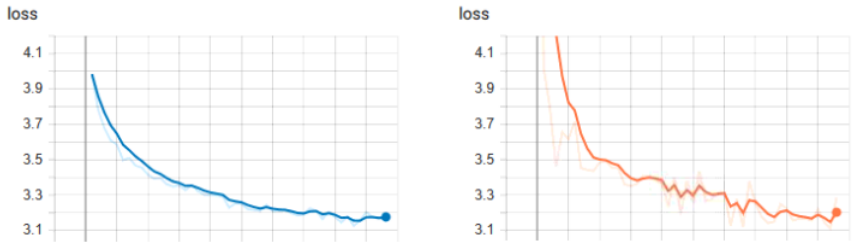

Figure 3.5: TENSORBOARD visualization of our paraphrase generation loss scores during training (ORANGE line) and validation (BLUE line) on the MSCOCO dataset.

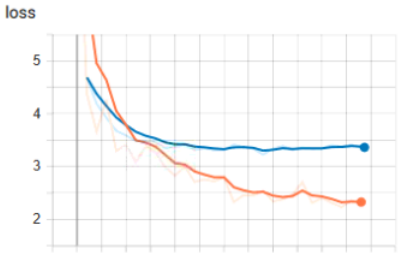

Figure 3.6: TENSORBOARD visualization of our paraphrase generation loss scores during training (ORANGE line) and validation (BLUE line) on the QUORA dataset superimposed*.
*The large gap between training and validation shows how the model tends to overfit on the training data, hence the need for using a dropout of 0.7.

We pre-process our datasets, and do not use the pre-processed/tokenized versions of the datasets from the `tensor2tensor library`. Our target vocabulary is a set of approximately 15,000 words. It contains words in our target training and test sets that occur at least twice. Using this subset of vocabulary words, as opposed to over 320,000 vocabulary words contained in `GloVe`, improves both training time and performance of the model.

[19] `https://github.com/tensorflow/tensor2tensor`

We train and evaluate our model after each epoch with a fixed learning rate of 0.0005, and stop training when the validation loss does not decrease after 5 epochs. We use *greedy-search decoding* during training and validation and set the maximum number of iterations to 5 times the target sentence length. For testing/inference we use *beam-search decoding*. *Greedy-search decoding* selects the most likely word at each step in the output sequence. *Beam-search decoding* returns a list of most likely output sequences. Instead of greedily choosing the most likely next step as the sequence is constructed, the beam search expands all possible next steps and keeps the k most likely, where k is a user-specified parameter (we used 6 for QUORA dataset and 10 for MSCOCO) and controls the number of parallel searches through the sequence of probabilities. By increasing the beam size, the performance might improve at the expense of significantly reducing the decoder speed (Brownlee, 2018). We found that increasing the beam size greater than 6 for the QUORA dataset did not lead to significant performance improvements. However, a beam size of 10 yielded the best result on MSCOCO dataset[20]. We subjectively attribute this to the nature of the datasets. The MSCOCO dataset contains paraphrase pairs with more novel words, as well as syntactic manipulations, than the QUORA pairs making it a more challenging corpora which is likely the reason we needed a more extensive search in the MSCOCO dataset.

Figures 3.5 and 3.6 visualizes the validation and training on the MSCOCO and QUORA datasets respectively. These figures show that we stop training after the validation loss starts decreasing consistently.

3.2.6 Baselines

We compare our model with recent models (Gupta et al., 2018; Li et al., 2018b; Prakash et al., 2016) including the current state-of-the-art (SOTA) in the field. To further highlight the gain of stacking 2 encoders, we use each component – TRANSFORMER (TRANS) and SEQ2SEQ (SEQ) as baselines.

[20]We could not experiment with more than 10 beams due to limitations on our machine's computing power.

- VAE-SVG-EQ (Gupta et al., 2018): This is the current state-of-the-art in the field, with a variational autoencoder as its main component.
- RbM-SL (Li et al., 2018b): Different from the encoder-decoder framework, this is a generator-evaluator framework, with the evaluator trained by reinforcement learning.
- Residual LSTM (Prakash et al., 2016): This implements a stacked residual long short term memory networks (LSTM).
- TRANS: Encoder-decoder framework as described in Section 3.2.4 but with a single TRANSFORMER encoder layer.
- SEQ: Encoder-decoder framework as described in Section 3.2.4 but with a single GRU-RNN encoder layer.

3.2.7 Results and Evaluation

For quantitative analysis of our model, we use popular automatic metrics such as BiLingual Evaluation Understudy (BLEU) (Papineni et al., 2002), ROUGE (described in section 2.3.8) and Metric for Evaluation of Translation with Explicit ORdering (METEOR) (Banerjee and Lavie, 2005; Denkowski and Lavie, 2014).

Table 3.1: Performance of our model against various models on the **MSCOCO** dataset. R-L refers to the ROUGE-L F1 score.

MODEL	BLEU	METEOR	R-L	EACS	GMS
Residual LSTM (Prakash et al., 2016)	37.0	27.0	-	-	-
VAE-SVG-EQ (Gupta et al., 2018)	41.7	31.0	-	-	-
TRANS (ours)	41.8	38.5	33.4	79.6	70.3
SEQ (ours)	40.7	36.9	35.8	78.9	70.0
TRANSEQ (ours)	43.4	38.3	37.4	80.5	71.1
TRANSEQ + beam (size=10) (ours)	**44.5**	**40.0**	**38.4**	**81.9**	**71.3**

BLEU is similar to ROUGE in that, they both measure similarity between the reference and system generated sequence based on an $n-gram$ match. However, the counting of

matching $n-grams$ in BLEU is modified to ensure that it takes the occurrence of the words in the reference text into account, not rewarding a candidate translation that generates an abundance of reasonable words. This is referred to by Papineni et al. (2002) as *modified n-gram precision*. While ROUGE is more recall-oriented, BLEU is precision-oriented. This means, BLEU measures how frequently the words (and/or n-grams) in the machine generated sequence appeared in the human reference sequence. ROUGE measures how often the words (and/or n-grams) in the human reference sequence appeared in the machine generated sequence. BLEU also adds the brevity penalty when a sequence generation is too short. We used the implementation provided in the NLTK[3] library, considering up to $4-gram$ matching. Since BLEU and ROUGE both measure $n-gram$ word-overlap with slight differences, we only report the BLEU and the ROUGE-L value.

METEOR is based on the harmonic mean of unigram precision and recall, with recall weighted higher than precision. It also has several features that are not found in other metrics, such as stemming and synonymy matching, along with the standard exact word matching. The metric was designed to produce good correlation with human judgement at the sentence level. We used the open source implementation by Denkowski and Lavie (2014)[21].

Besides using evaluation metrics that are word-overlap based, we evaluated our models on two (2) additional recent qualitative metrics – Greedy Matching Score (**GMS**) and Embedding Average Cosine Similarity (**EACS**) by Sharma et al. (2017)[22] that measure the similarity between the reference and generated paraphrases based on the cosine similarity of their embeddings on word and sentence levels respectively. Results are presented in Tables 3.1 and 3.2. In table 3.3, we report the 95% confidence intervals for ROUGE-L, to show the statistical significance of our results.

[21] https://www.cs.cmu.edu/~alavie/METEOR/
[22] https://github.com/Maluuba/nlg-eval

Table 3.2: Performance of our model against various models on the **QUORA** dataset with 50k,100k,150k training examples. R-L refers to the ROUGE-L F1 score.

		50K			
MODEL	**BLEU**	**METEOR**	**R-L**	**EACS**	**GMS**
VAE-SVG-EQ (Gupta et al., 2018)	17.4	22.2	-	-	-
RbM-SL (Li et al., 2018b)	35.81	28.12	-	-	-
TRANS (ours)	35.56	**33.89**	27.53	79.72	62.91
SEQ (ours)	34.88	32.10	29.91	78.66	61.45
TRANSEQ (ours)	37.06	33.73	**30.89**	80.81	**63.63**
TRANSEQ + beam (size=6) (ours)	**37.12**	33.68	30.72	**81.03**	63.50
		100K			
MODEL	**BLEU**	**METEOR**	**R-L**	**EACS**	**GMS**
VAE-SVG-EQ (Gupta et al., 2018)	22.90	25.50	-	-	-
RbM-SL (Li et al., 2018b)	**43.54**	32.84	-	-	-
TRANS (ours)	37.46	**36.04**	29.73	80.61	64.81
SEQ (ours)	36.98	34.71	32.06	79.65	63.49
TRANSEQ (ours)	38.75	35.84	**33.23**	81.50	**65.52**
TRANSEQ + beam (size=6) (ours)	38.77	35.86	33.07	**81.64**	65.42
		150K			
MODEL	**BLEU**	**METEOR**	**R-L**	**EACS**	**GMS**
VAE-SVG-EQ (Gupta et al., 2018)	38.30	33.60	-	-	-
TRANS (ours)	39.00	**38.68**	32.05	81.90	65.27
SEQ (ours)	38.50	36.89	34.35	80.95	64.13
TRANSEQ (ours)	**40.36**	38.49	**35.84**	**82.84**	**65.99**
TRANSEQ + beam (size=6) (ours)	39.82	38.48	35.40	82.48	65.54

Table 3.3: 95% confidence intervals of our paraphrase generation systems.

Models	**MSCOCO**	**QUORA (150k)**
TRANS	33.4 ± 0.294	32.05 ± 0.279
SEQ	35.8 ± 0.216	34.35 ± 0.217
TRANSEQ	37.4 ± 0.196	35.84 ± 0.155
TRANSEQ + beam	38.4 ± 0.176	35.40 ± 0.186

3.2.8 Analysis

Tables 3.1 and 3.2 report scores of our model on both datasets. Our model pushes the benchmark on all evaluation metrics when compared against current published top models

evaluated on the same datasets. Since several words could connote similar meaning, it is more logical to evaluate with metrics that match with embedding vectors capable of measuring this similarity. Hence we also report GMS and EACS scores as a basis of comparison for future work in this direction.

S: What are the dumbest questions ever asked on Quora? **G:** what is the stupidest question on quora? **R:** What is the most stupid question asked on Quora?
S: How can I lose fat without doing any aerobic physical activity **G:** how can i lose weight without exercise? **R:** How can I lose weight in a month without doing exercise?
S: How did Donald Trump won the 2016 USA presidential election? **G:** how did donald trump win the 2016 presidential **R:** How did Donald Trump become president?

Figure 3.7: Examples of our generated paraphrases on the **QUORA** sampled test set, where **S**, **G**, **R** represents Source, Generated and Reference sentences respectively.

S: Three dimensional rendering of a kitchen area with various appliances. **G:** a series of photographs of a kitchen **R:** A series of photographs of a tiny model kitchen
S: a young boy in a soccer uniform kicking a ball **G:** a young boy kicking a soccer ball **R:** A young boy kicking a soccer ball on a green field.
S: The dog is wearing a Santa Claus hat. **G:** a dog poses with santa hat **R:** A dog poses while wearing a santa hat.
S: the people are sampling wine at a wine tasting. **G:** a group of people wine tasting. **R:** Group of people tasting wine next to some barrels.

Figure 3.8: Examples of our generated paraphrases on the **MSCOCO** sampled test set, where **S**, **G**, **R** represents Source, Generated and Reference sentences respectively.

Besides quantitative values, Figures 3.7 and 3.8 show that our paraphrases are well formed, abstractive (*e.g dumbest – stupidest, dog is wearing – dog poses*) and capable

of performing syntactic manipulations (*e.g in a soccer uniform kicking a ball – kicking a soccer ball*) and compression.

3.3 Related Work

Our baseline models – VAE-SVG-EQ (Gupta et al., 2018) and RbM-SL (Li et al., 2018b) are both deep learning models. While the former uses a variational-autoencoder and is capable of generating multiple paraphrases of a given sentence, the later uses deep reinforcement learning. In tune with part of our approach, ie, SEQ2SEQ, there exists ample models with interesting variants – residual LSTM (Prakash et al., 2016), bi-directional GRU with attention and special decoding tweaks (Cao et al., 2017) and attention from the perspective of semantic parsing (Su and Yan, 2017). MT has been greatly used to generate paraphrases (Quirk et al., 2004; Zhao et al., 2008) due to the availability of large corpora. Earlier works have explored the use of manually drafted rules (Hassan et al., 2007; Kozlowski et al., 2003).

Similar to our model architecture, Chen et al. (2018) combined TRANSFORMERS and RNN-based encoders for MT. Recently, the TRANSFORMER model was used for paraphrasing on different datasets – TURK and NEWSELA (Zhao et al., 2018a). We experimented using solely a TRANSFORMER but got better results with TRANSEQ. To the best of our knowledge, our work is the first to cross-breed the TRANSFORMER and SEQ2SEQ for paraphrase generation.

3.4 Summary

In this chapter we presented our NN model in an attempt to answer **RQ2:** *How can we build a paraphrasing model to help with abstractive summarization?* We proposed a novel framework, TRANSEQ that combines the efficiency of a TRANSFORMER and SEQ2SEQ model and improves the current state-of-the-art on the QUORA and MSCOCO paraphrasing datasets. Besides quantitative results, we presented samples that highlight the syntactic and

semantic quality of our generated paraphrases (more examples are presented in Appendices C and D). This framework was applied in chapter 5 of this thesis for the task of abstractive text summarization. Part of this work was published at the *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Workshop on Neural Generation and Translation, pages 249 - 255 (2019). Hong Kong, China* (Egonmwan and Chali, 2019b).

Chapter 4

Sentence Compression

This chapter describes the second sub-model for our abstractive summarization model – *Sentence Compression*. The goal is to provide an answer to **RQ3:** *How can we build a compression model to help improve the conciseness of machine generated summaries?* Summaries are shorter than the source text but might still be longer than desired, especially when compared to the length of the gold or reference summaries. Hence sentence compression becomes a necessary tool to further improve the conciseness of the machine generated summaries. However, in the process of sentence compression, it is likely that words which affect the semantics and factual-correctness of the sentence are sometimes deleted. As a result, we also address **RQ4:** *How can we investigate that our machine generated outputs are factual, that is, that they contain accurate information?*

4.1 Introduction

Sentence compression is a Natural Language Processing (NLP) task that aims at producing a shorter version of a sentence. The shortened sentence could be an abstract (Fevry and Phang, 2018; Yu et al., 2018) or extract (Filippova et al., 2015; Knight and Marcu, 2000) of the source sentence. Abstractive sentence compression is similar to paraphrasing which has been addressed in the previous chapter (see chapter 3). For example, while the sentence – *"Symptom of the flu include fever and nasal congestion"* can be paraphrased as *"Elevated temperature is a sign of the flu"*, while an extractive compressed version is *"Symptom of the flu include fever"*. The focus of this chapter is on extractive sentence compression,

which as seen in the example, solely involves deleting appropriate words from the sentence without substituting with novel words. One major challenge with sentence compression is maintaining the factuality of the source text after deletion of word(s). As seen in Figure 4.1 a single erroneous deletion of the word *was* results in a sentence which implies that the man did the shooting, when in fact he was shot.

Input Sentence: A man who was shot in the head in front of his teammates as he walked off a soccer pitch knew his life was in danger, an inquest heard.
Erroneous Compression: A man who shot in the head knew his life was in danger.
Correct Compression: A man who **was** shot in the head knew his life was in danger.

Figure 4.1: Example of the consequence of erroneous compression on factuality

Existing models for deletion-based sentence compression give little or no consideration to the factuality of the compression. Sentences are pruned manually based on the constituency (Berg-Kirkpatrick et al., 2011), dependency (Filippova, Katja and Strube, Michael, 2008) and by a parse tree or a noisy-channel model (Knight and Marcu, 2000). Features can be automatically learned by training Long Short Term Memory (LSTM) models on a large sentence compression dataset (Filippova et al., 2015; Wang et al., 2017; Zhao et al., 2018b).

We formulate the sentence compression task as a word-level binary classification problem. For each word in the source sentence, we decide if to delete (0) or retain (1) it during compression by training a bi-directional transformer neural network (Vaswani et al., 2017b) on a large sentence compression dataset – GOOGLENEWS (Filippova et al., 2015). We attempt to mitigate factual inconsistency by triggering a stop in the training of our sentence compression model once a divergence in meaning between the original and machine-compressed sentence starts to occur. The semantic similarity between the sentences is measured by their embedding-based cosine-similarity (Kenter and De Rijke, 2015). We also observe that binary-labels of the words in a sentence are biased (more retains (1) than deletes (0)). Therefore, the F1- score better reflects the performance of the model (Lipton

et al., 2014). Hence, we experiment with the impact on our model when we stop training based on the F1-score instead of the typical loss score.

4.2 Methodology

First, we formally define the task of *sentence compression*. Next, we describe the datasets used for training and testing our sentence compression model including preprocessing steps. We then present our NN architecture for solving the task, providing details of all settings used during implementation. Since compressed sentences are prone to factual inconsistencies, we give details on this, and provide a method to reduce the factual incorrectness of system compressed sentences. Sections 4.2.7 and 4.2.8 evaluates and analyzes the output of our model. Finally we discuss published work related to ours, on the task of sentence compression.

4.2.1 Task Definition

Given an input sentence $X = (x_1, ..., x_n)$ with n words, the task is to generate a compressed output sentence $\overline{X} = (\overline{x}_1, ..., \overline{x}_m)$ with m words that conveys similar semantics as X and $m \leq n$.

4.2.2 Dataset

Our model was trained and tested using the GOOGLENEWS dataset only. The GOOGLENEWS sentence compression corpus contains about two million sentence compression pairs (i.e., full naturally formed sentences with its compressed version) collected by Filippova et al. (2015) from Google News articles between 2013 and 2014 on the web, by using a method proposed in Filippova and Altun (2013). Due to performance limitations of our machine[23] we trained on only $200,000$ randomly selected samples. For testing, we use the publicly released set containing $10,000$ sentence compression pairs[24]. Similar to Filippova

[23] Our experiments were implemented on a Nvidia TITAN X GPU card with 12G RAM.

[24] https://github.com/google-research-datasets/sentence-compression

et al. (2015), we perform automatic evaluation on the first 1,000 sentences.

4.2.3 Preprocessing

Articles and stop words are relevant in this task, hence they were not removed. Word-level tokenization was performed by the NLTK[3] library. The GOOGLENEWS dataset comes in pairs of sentences and their compression. We convert the compressed reference sentences into a sequence of 0's and 1's as illustrated in Figure 4.2.

Input Sentence (X): Actor Philip Seymour Hoffman has been found dead in his New York apartment from an apparent drug overdose
Labels (Y): 0 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0
Compressed Sentence ($\overline{X}$): ~~Actor~~ Philip Seymour Hoffman has been found dead ~~in his New York apartment from an apparent drug overdose~~

Figure 4.2: Example of a training pair from the GoogleNews dataset.

In Figure 4.2, X is the input sentence and Y corresponds to the binary labels $(0,1)$ matching the compressed reference sentence, $\overline{X}$. 0 represents a word that was deleted from the input sentence and 1 represents a word that was retained. Labels Y, are generated by simply performing a 1-gram word overlap between the source sentence X, and the compressed reference sentence, $\overline{X}$.

4.2.4 NN Architecture

The NN architecture is modelled to correctly classify each word in a sentence as worthy of being present in the compressed sentence (meaning it should be retained) or otherwise (meaning it should be deleted). Hence, the topmost layer is a softmax classifier. However, the classifier needs features to enable its decision making. These features are automatically learned by a BI-DIRECTIONAL TRANSFORMER encoder. Prior to encoding, each word is embedded into a vector representation using BERT embedding. We explain each of these components of our NN architecture below.

Embedding

We explored instantiating the sentence encoder with non-contextualized `GloVe` (Pennington et al., 2014) word-embeddings. However, for this task, `GloVe` proved sub-optimal compared to using deep contextualized word-embeddings like BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2018).

BERT uses a "masked language model" (MLM) which randomly masks some tokens from the input and tries to predict the vocabulary-id of the masked token based only on its context (Devlin et al., 2018).

To obtain the input sentence representation, we concatenate the word embeddings as in equation 4.1:

$$\widetilde{X_j} = \tilde{x_1} \| \tilde{x_2} \| \ldots \| \tilde{x_n}, \tag{4.1}$$

where $\tilde{x_i}$ refers to the word-embedding of the i^{th} word in sentence X_j.

Encoding

The BERT embeddings are fed to a TRANSFORMER encoder to learn hidden representations. Please refer to Section 2.3.6 for details on the TRANSFORMER encoder.

Classifying and Training

The encoded vector representation serves as input to the softmax classifier. Recall in Chapter 2, we stated the softmax equation (2.17) and the loss function (2.18) on which the softmax classifier is trained. We use the same softmax classifier equation and cross entropy loss function for our sentence compression model.

Maintaining Factuality

Here, we attempt to address **RQ4:** *How can we investigate that our machine generated outputs are factual, that is, they contain accurate information?* Intuitively, it is easy

to see how a shorter sentence can easily omit or misconstrue information contained in a longer source sentence. Hence, there is a need to strike a balance between the correctness of information retained in the compression and its length. We apply a preemptive approach to tackle this. The idea is that the model learns relatively well what tokens to delete at each training step, but needs a better trigger to communicate when to stop. Otherwise, it over-compresses, resulting in "shorter-than-is-ideal" sentences, which distort the facts contained in the original sentence and/or introduces grammatical errors, negatively affecting its readability. We observe evidence of this in Figures 4.1 and 4.3. Hence, we implement early-stopping to mitigate factual inconsistencies. Generally, it is used to curtail the effects of over-fitting (Hawkins, 2004; Tetko et al., 1995) such as failure of the model to generalize to unseen samples.

One common early-stopping criteria is the loss score on the validation set. The model is setup to stop training if the loss on a held-out validation set does not decrease after a fixed number of epochs (Li et al., 2019; Prechelt, 1998a,b). However, due to the imbalanced nature of our labelled data and the desire to maintain the factuality of our compressed sentence, we sought to investigate the impact of implementing two (2) other early-stopping criteria, explained in the following sub-sections.

Cosine Similarity

While making a few delete or retain decision errors could minimally affect the accuracy and F1-score, there could be a greater impact on the meaning of the resulting compressed sentence. Due to the natural ambiguous characteristic of English sentences, a few misplaced word(s) could entirely distort the semantics of the sentence. As seen in the example in Figure 4.1, a single erroneous deletion of the word *was*, results in a sentence which implies that the man did the shooting, when in fact he was shot.

To reduce the semantic disparity between the system and reference compressed sentence, we measure their embedding-based cosine similarity (Kenter and De Rijke, 2015),

and implement a stop criteria when the similarity score begins to diverge.

F1-score

Here, we try to solve the problem that comes with the biased nature of the word-labels – usually more retain than delete decisions. For instance, retaining all words in the source sentence might result in high recall and moderate accuracy scores, even when no compression was actually done. For binary classification problems of this nature, the F1-score is a better evaluation metric as it balances both the precision and recall of predictions (Lipton et al., 2014). Hence, we choose to validate our model based on the F1-score. The words with probabilities greater than 0.5[25] are labelled 1 (retain), else 0 (delete). Next, we calculate the F1-score of system labels and stop training when the F1-score on the validation set fails to improve after a defined number of training epochs.

4.2.5 Implementation Details

We use the `API` from `pypi`[26] to obtain pre-trained BERT word-embeddings. For the TRANSFORMER encoder, we used the *transformer_base* hyperparameter setting from the tensor2tensor library (Vaswani et al., 2018)[27]. We set the dropout to 0.0. We train and evaluate our model after each epoch using a fixed learning rate of 0.0001, with a batch size of 50 and stop training when either of the heuristics described in Sections 4.2.4 do not improve after 10 epochs.

4.2.6 Baselines

We compare our compression model against some recent models and strong existing baselines [28]. To contrast the effect of the different stopping criteria as described in Section 4.2.4 on our models' performance, we evaluate the output of the model when stopped, based on each of the different criteria.

[25] We empirically arrived at this score after experiments on the datasets.

[26] `https://pypi.org/project/bert-embedding/`

[27] `https://github.com/tensorflow/tensor2tensor`

[28] None of these existing models address factuality issues in sentence compression.

- Evaluator-SLM (Zhao et al., 2018b): To the best of our knowledge, this is the current state-of-the-art for deletion-based sentence compression. The authors trained a syntactic neural language model integrated with Part-of-Speech (POS) tags and dependency relations. Subsequently, they formulated the deletion-based sentence compression as a series of trial-and-error deletion operations through reinforcement learning using about 200,000 compression pairs from GoogleNews dataset. The policy network receives a reward from the language model for updating the network.

- LSTM+PAR+PRES (Filippova et al., 2015): This is a strong foundational baseline for deletion-based sentence compression. They trained a softmax classifier on top of an LSTM model devoid of any syntactic features using about two million sentence compression pairs from the GoogleNews dataset.

- BI-LSTM+SynFeat (Wang et al., 2017): Similar to Filippova et al. (2015), the authors trained an LSTM based model, but with a bi-directional architecture and fewer training samples (8,000 sentence pairs). Additionally, syntactic features (POS and dependency embeddings) were introduced to the model both explicitly and by constraints through Integer Linear Programming (ILP).

Our models

- Bi-TRANS: Our implementation of bi-directional transformer + softmax classifier, without any syntactic features. In this setting, training is stopped based on the loss score as explained in section 4.2.4.

- Bi-TRANS+COSSIM: Same implementation of Bi-TRANS but with a stopping criteria based on the embedding-based cosine similarity of predicted and reference sentences as explained in section 4.2.4.

- Bi-TRANS+F1: Same implementation of Bi-TRANS but with a stopping criteria based on the F1-score of predicted and reference token labels per sentence as explained in

section 4.2.4.

Input Sentence: John Abraham has been prohibited from using the title Hamara Bajaj for his home production . **Bi-TRANS:** John Abraham has been prohibited from his home **Bi-TRANS+F1:** John Abraham has been prohibited from using his home production **Bi-TRANS+COSSIM:** John Abraham has been prohibited from using the title Hamara **Reference Compression:** John Abraham has been prohibited from using the title Hamara Bajaj
Input Sentence: Google is developing smart contact lenses that measure the glucose levels in diabetics' tears. **Bi-TRANS:** Google is developing glucose **Bi-TRANS+F1:** Google is developing smart contact **Bi-TRANS+COSSIM:** Google is developing smart lenses **Reference Compression:** Google is developing smart contact lenses

Figure 4.3: Example outputs of our implemented baselines from GOOGLENEWS dataset.

4.2.7 Results and Evaluation

We carry out both automatic and human evaluation on our model.

Automatic Evaluation

Following previous work on sentence compression, we evaluate our model on three quantitative metrics – F1-score, Accuracy and Compression Rate (CR) (Napoles et al., 2011). F1-score is the harmonic mean of precision and recall. While precision measures the percentage of accurately retained words in the machine-compressed sentence, recall measures the percentage of retained words in the reference that were accurately produced by the machine. Accuracy on the other hand, measures the percentage of machine-predictions (both retains and deletes) that are correct. CR is the Compression Ratio, which translates to the number of tokens in the compressed sentence divided by the number of tokens in

Table 4.1: Quantitative Evaluation of our Compression model against various models on the **GOOGLENEWS** dataset†.

SENT. COMPRESSION	**F1**	**ACC**	**CR**
LSTM+PAR+PRES (Filippova et al., 2015)	0.82	0.34	0.38
Bi-LSTM-CRF (Lai et al., 2017)	0.79	-	0.38
Bi-LSTM+SynFeat (Wang et al., 2017)	0.80	0.82	0.43
LK-GNN (Zhao et al., 2017)	0.83	-	-
CR-BILSTM-TA (Lu et al., 2017)	0.80	0.33	-
Evaluator-SLM (Zhao et al., 2018b)	0.85	-	0.39
Bi-TRANS (ours)	0.83	0.80	0.40
Bi-TRANS+F1 (ours)	0.85	0.84	0.42
Bi-TRANS+COSSIM (ours)	**0.88**	**0.85**	0.45

†F1 score for all models are word-level based. Accuracy (ACC) in Filippova et al. (2015) and Lu et al. (2017) are sentence-level based, while ours and Wang et al. (2017)'s are word-level based.

the source sentence. So a model with 0.38 CR value, means 62% of the words in the sentence were deleted. We aim for moderate CR values, as too low CR scores indicates over-compression, and too high CR values indicates under-compression. The reference sentences compression had an average of 0.47 CR value.

Table 4.2: Qualitative Evaluation of our Sentence Compression Model.

MODEL	**EACS**	**GMS**
Bi-TRANS	86.8	88.1
Bi-TRANS+F1	88.9	89.4
Bi-TRANS+COSSIM	**89.1**	**89.9**

Additionally, we evaluate on two qualitative metrics – Greedy Matching Score (GMS) and Embedding Average Cosine Similarity (EACS) (Sharma et al., 2017). GMS and EACS measure the similarity between the reference and machine produced sentence compressions, based on the cosine similarity of their embeddings on word and sentence levels respectively, thus having a stronger correlation with human evaluation.

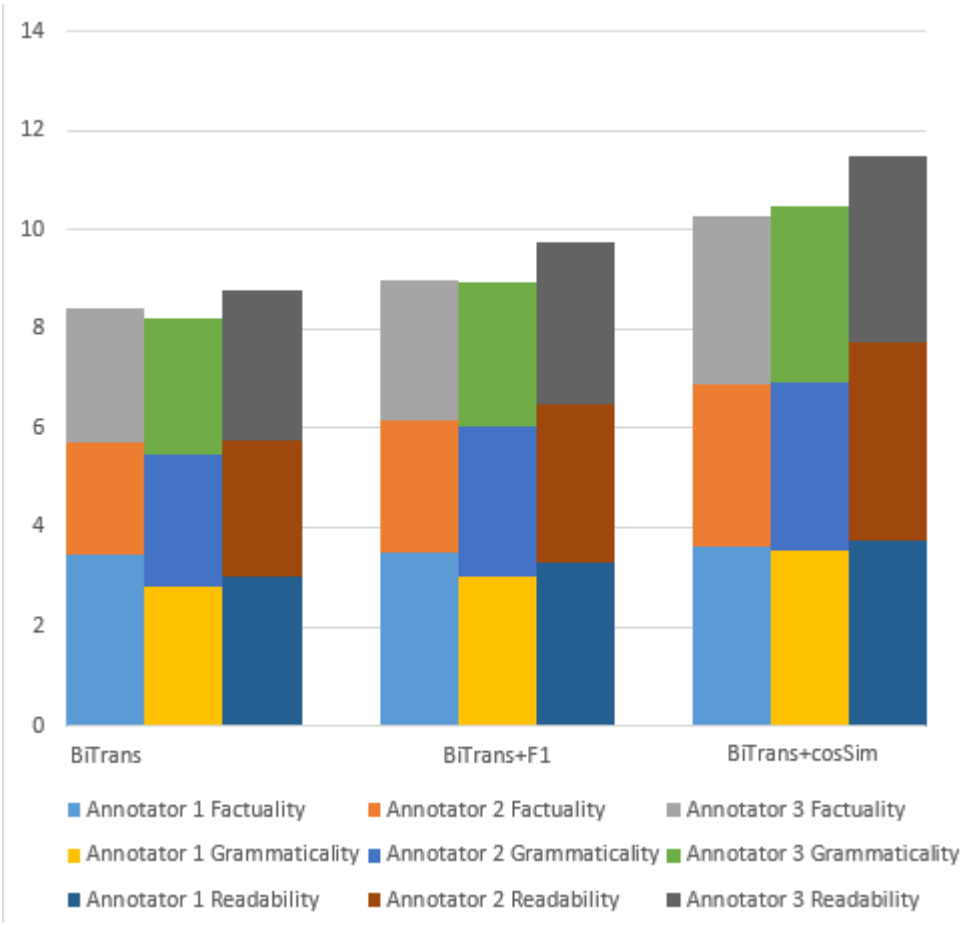

Figure 4.4: Stacked bar charts showing human evaluations of some samples from each model's output. The three stacks in each bar corresponds to each annotator's score per criterion – factuality, grammaticality and readability.

Human Evaluation

To further validate our experiments, we conduct human evaluation. We measure *factuality, grammaticality* and *readability*. Factuality measures how well the compressed sentence maintains facts stated in the original sentence. Grammaticality measures how well the compressed sentence adheres to grammar rules, while readability estimates how easy to read the compressed sentence is. To evaluate each of these criteria, we design the following Amazon MTurk experiment: we randomly select 200 samples from the GOOGLENEWS test set and ask the human testers (3 per sample) to rank between outputs (for the three (3) models – Bi-TRANS, Bi-TRANS+F1 and Bi-TRANS+COSSIM). For the output X, of each corresponding

model we posed three statements and required the testers[29] to agree or disagree on a scale of 1 – 5 (1 = Strongly disagree, 5 = Strongly agree). The three statements were:

i) *X adequately compresses and maintains the factuality contained in Source Text.*

ii) *X is grammatically correct.*

iii) *X is readable.*

The human produced reference sentence compressions were not presented to the testers. From the results in Figure 4.4 and Table 4.3, we see that training our sentence compression model with the discussed stop criteria – F1 and cosine similarity yields improvements.

Table 4.3: Human Evaluation of our Sentence Compression Model, where FT., GM. and RD. stand for Factuality, Grammaticality and Readability respectively†.

MODEL	FT.	GM.	RD.
Bi-TRANS	2.80	2.73	2.93
Bi-TRANS+F1	2.99	2.99	3.29
Bi-TRANS+COSSIM	**3.43**	**3.49**	**3.83**

†The scores from all three (3) annotators were averaged to yield the values presented in the Table.

4.2.8 Analysis

The theory behind our experiment is that over-compression possibly leads to inconsistencies in facts between the system compression and original sentence. Qualitative evaluation of our models presented in Table 4.2 gives evidence that stopping sentence compression appropriately sooner, better maintains the semantics between the original and compressed sentence. Also, we observe higher F-1 and accuracy scores for models with a higher CR as seen in Table 4.1. Although the F-1 and accuracy metrics do not necessarily represent the coherence level of the compressed sentence, they indicate that the machine-produced compressed sentence is very much close to the human-compression, based on token overlap. Human evaluations substantiate the presence of factual errors in Bi-TRANS and a

[29]We selected testers who were located in US or Canada, have Amazon Mechanical Turk Masters qualification and had HIT approval rate greater than or equal to 95%.

reduction in these errors with Bi-TRANS+F1 and Bi-TRANS+COSSIM. We also see a corresponding increase in the readability and grammaticality in the outputs of Bi-TRANS+F1 and Bi-TRANS+COSSIM as seen in Table 4.3 and Figure 4.4.

4.3 Related Work

The crux of the extractive sentence compression task is deciding what words/tokens are least important in a sentence, such that when deleted, the meaning of the original sentence remains while still being grammatically correct. Knight and Marcu (2000) assumed that the input sentence has been infiltrated with noise or blown up. Hence, they use a noisy channel model to filter out the noise. The compressed output is learned from training a parallel corpus that consists of {texts, abstracts} tuples. This model is highly statistical and dependent on the language model used. Motivated by the belief that the grammaticality of a sentence can be better ensured by compressing trees – Filippova, Katja and Strube, Michael (2008) implemented tree pruning based on the dependency parse of the input sentence. Their method was aimed at preserving dependencies which are required to retain the grammatical correctness of the output or have important words as the dependent. In a similar light Berg-Kirkpatrick et al. (2011) parses the sentence to be compressed into its constituency tree. A small set of manually formulated features with learned weights from a small dataset, are then used to decide if a word should be deleted or not. The features were chosen by looking at human annotated data and observing the most common types of edit. However, the performance of these methods rest largely on the efficiency of the dependency or constituency parser, which unfortunately, are usually error-prone.

Recent methods are less dependent on parse-trees or manually crafted features, but rather employ deep learning techniques using variants of recurrent neural networks (RNN) (Filippova et al., 2015; Wang et al., 2017; Zhao et al., 2018b). Different from existing models, we attempt to address factuality in sentence compression and utilize contextualized word-embeddings to better capture semantics.

4.4 Summary

In this chapter we proposed solutions to two (2) of our key research questions – **RQ3:** *How can we build a compression model to help improve the conciseness of machine generated summaries?* and **RQ4:** *How can we investigate that our machine generated outputs are factual, that is, they contain accurate information?* Specifically, we showed that factual errors present in compressed sentences could be reduced by stopping the training process at an 'ideal' time to preempt over-compression. The 'ideal' time was experimented as a measure of the F1 and embedding-based cosine similarity scores that help to capture the imbalance in token labels and the semantic relatedness between the compressed and original sentence. We evaluated our model using both automatic and human evaluation methods. All results show improved performance of our system over existing baselines and models stopped earlier with our discussed heuristics proved superior. Examples of the outputs of our models are presented in Appendix E. This framework was applied in Chapter 6 of this thesis for the task of multi-document abstractive text summarization. Part of this work has been submitted for publication at the *16th Conference of the European Chapter of the Association for Computational Linguistics*. 2021.

Chapter 5

Single Document Abstractive Summarization

In this Chapter, we address a part of the overarching research question, **RQ0:** *How can we build machine learning model capable of generating* ***abstractive*** *summaries?* In simple and straightforward terms, the solution is that, we extract and paraphrase using the appropriate models from Chapter 2 and 3 respectively. We provide a more detailed methodology in subsequent sections.

5.1 Introduction

The task of abstractive summarization can be expressed as writing summaries in *your* own words. Where, *your* refers to the summary writer, which may or may not be the author of the document. It is important that *own words* introduced convey a similar meaning as those in the original document. However, the summary writer chooses to use those words for succinctness and/or clarity. So in contrast to extractive summarization, which is 100% *copying and pasting* of salient points in the document, abstractive summarization involves some degree of *copying and pasting* but a larger degree of *rewriting* and *rephrasing*. Exactly how much rewriting and rephrasing defines a summary as abstractive is an unaddressed research question. In general, the more novel words that are introduced, the higher the abstractive quality of the summary.

We consider the art of abstracting essentially the same as paraphrasing. Hence, we model the abstractive summarization task as one of generating paraphrases after the most

salient sentences have been identified by an extractive summarizer. This approach has been shown to be valid in literature (Chen and Bansal, 2018; Hsu et al., 2018; Liu et al., 2018). Having demonstrated that our chosen model architecture is viable for generating meaningful paraphrases and testing on a dedicated paraphrase dataset (see Chapter 3) we apply the same model architecture for this task as for generating abstractive summaries. Details of the application are given subsequently.

5.2 Methodology

The datasets used for the task of single document abstractive summarization are the same as those used for the task of single document extractive summarization – CNN/DM and NEWSROOM datasets. Preprocessing is performed during each sub-task – extraction and paraphrasing as described in Chapters 2 and 3 respectively. The input to our abstraction module is a subset of the document's sentences which is comprised of the output of the extraction phase from Chapter 2. For each document D_j, initially comprising of n sentences, we abstract its extracted sentences,

$$S_j^E = \{S_j^1, S_j^2, ..., S_j^m\} \tag{5.1}$$

where $m < n$ and $S_j^E \subseteq D_j$, by learning to paraphrase using the paraphrase architecture explained in Chapter 3. The paraphrasing module requires single sentences as input. However, not all summary sentences are paraphrases of a document sentence. While some are paraphrases of a single sentence, others are a fusion of multiple sentences (Lebanoff et al., 2019b). Hence, we first identify document-summary sentence pairs which are possible paraphrases by using the similarity measure given in equation 2.7, which is a measure of the bi-gram overlap between the two (2) sentences. We found about 60% of the summary sentences were paraphrases of the source sentence. Sentences which were not paraphrased, were written in the output summary as-is (extracted). It is important to note that, what

we did was to apply the same architecture used for paraphrasing to the task of generating abstractive summaries. The model was trained from scratch on the appropriate dataset for the abstractive summarization task. This is significantly different from the more popular transfer learning setting, where the model is trained on a specific dataset for a certain task, and then used with the same learned weights on a different task. This is particularly useful in use-cases where data in one task is limited or scarce. Hence knowledge is gained from a largely available dataset on a related task or problem. We employ this transfer learning (Mahajan et al., 2018; Van Den Oord et al., 2014) method in multi-document settings, because of the paucity of multi-document summarization corpus.

We experimented with adding a sentence compression layer (with the same architecture explained in Chapter 4) before or after the layer responsible for generating paraphrases. However, this produced no significant improvements in results as seen in Tables 5.2 and 5.3. As we will see later in Chapter 6, sentence compression proved very useful in multi-document settings. In Table 5.1, we compare the statistics of CNN/DM and MULTINEWS datasets for single and multi-document summarization respectively, to gain intuitive understanding behind the observed behavior – sentence compression yielding significant gains when used in conjunction with sentence paraphrasing for multi-document summarization but otherwise for single document abstractive summarization model.

Table 5.1: Statistics of the single document summarization dataset (CNN/DM) test samples versus multi-document summarization dataset (MULTINEWS) test samples.

	CNN/DM (SDS)	**MULTINEWS (MDS)**
Avg. #words/doc.	766	2100
Avg. #words/summ.	30	264
Avg. #sents/doc.	53	83
Avg. #sents/summ.	3.75	10

As seen in Table 5.1, the sentences in SDS are significantly shorter than in MDS, both in the source documents and summaries. Hence, in SDS settings, sentence compression might

O: the two clubs, who occupy the top two spots in spain's top fight, are set to face each other at the nou camp on sunday. **G:** real madrid face barcelona in the nou camp **R:** real madrid will travel to the nou camp to face barcelona on sunday.
O: dangelo conner, from new york, filmed himself messing around with the powerful weapon in a friend's apartment, first waving it around, then sending volts coursing through a coke can . **G:** dangelo conner from new york was fooling around with his gun **R:** dangelo conner, from new york ,was fooling around with stun gun.
O: jamie peacock broke his try drought with a double for leeds in their win over salford on sunday. **G:** jamie adam scored to win over salford for leeds **R:** jamie peacock scored two tries for leeds in their win over salford.
O: britain's lewis hamilton made the perfect start to his world title defense by winning the opening race of the f1 season in australia sunday to lead a mercedes one-two in melbourne . **G:** lewis hamilton wins first race of season in australia **R:** lewis hamilton wins opening race of 2015 f1 season in australia .

Figure 5.1: Examples of some of our abstractive sentences from the CNN/DM dataset, where **O**, **G**, **R** represents Originating document sentence, our model's Generated abstract and Reference sentences from the ground-truth summary respectively.

not be of high utility especially if/when the model is capable of generating paraphrases[30]. In the CNN/DM dataset, with an average of 30 words/summary spread over about 3 to 4 sentences, each summary sentence contains an estimate of about 10 words. Using our extract and paraphrase model, our generated summaries are an average of about 36 words. Therefore, adding compression capabilities on top of this only results in shorter-than-desired sentences with negative effects on the ROUGE-RECALL score and thus the overall performance (ROUGE-F1 score).

5.2.1 Results and Analysis

Following the usual practice in literature and this thesis, we evaluate our generated summaries against the reference using ROUGE. Results are presented in Tables 5.2 and

[30]Recall that our sentence paraphrasing model is able to jointly perform sentence compression to some extent as observed in some generated samples (see Figures 3.7, 3.8, Appendices C and D).

5.3. Recall from Chapter 2 that TRANS-ext and TRANS-ext + filter refer to our extractive systems with non-filtered and filtered training sets respectively, where filtering was done mainly to reduce noise in the data and remove disparate document-summary pairs. +abs refers to abstraction by paraphrasing while +absComp refers to abstraction by paraphrasing and compressing.

Table 5.2: Average ROUGE-F1 (%) scores of various abstractive models on the **CNN/DM** test set. The first section shows results from literature while the second section presents results from our models.

Abstractive Model	**R-1**	**R-2**	**R-L**
RL+Intra-Att (Paulus et al., 2018)	41.16	15.75	**39.08**
KIGN+Pred (Li et al., 2018a)	38.95	17.12	35.68
FAST (Chen and Bansal, 2018)	40.88	17.80	38.54
Bottom-Up (Gehrmann et al., 2018)	41.22	18.68	38.34
TRANS-ext + absComp	40.75	17.55	36.45
TRANS-ext + filter + absComp	40.85	17.72	36.67
TRANS-ext + abs	41.05	17.87	36.73
TRANS-ext + filter + abs	**41.89**	**18.90**	38.92

Table 5.3: Average ROUGE-F1 (%) scores of various abstractive models on the **NEWSROOM** released test set. * marks results taken from Grusky et al. (2018).

Abstractive Model	**R-1**	**R-2**	**R-L**
Abs-N* (Rush et al., 2015)	5.88	0.39	5.32
Pointer* (See et al., 2017)	26.02	13.25	22.43
TRANS-ext + absComp	32.37	14.72	25.69
TRANS-ext + filter + absComp	32.89	15.03	26.65
TRANS-ext + abs	33.81	15.37	28.92
TRANS-ext + filter + abs	**35.74**	**16.52**	**30.17**

We highlight examples of some of the abstractive sentences in Figure 5.1. Figure 5.1 show that our outputs are well formed, abstractive (*e.g powerful weapon – gun, messing around – fooling around*) and capable of performing syntactic manipulations (*e.g for leeds in their win over sadford – win over salford for leeds*). In table 5.4, we report the 95% confidence intervals for ROUGE-1, ROUGE-2 and ROUGE-L to show the statistical significance of our results.

Table 5.4: 95% confidence intervals of our SDS abstractive systems.

CNN/DM	R-1	R-2	R-L
TRANS-ext + absComp	40.75 ± 0.037	17.55 ± 0.039	36.45 ± 0.031
TRANS-ext + filter + absComp	40.85 ± 0.056	17.72 ± 0.054	36.67 ± 0.051
TRANS-ext + abs	41.05 ± 0.018	17.87 ± 0.022	36.73 ± 0.021
TRANS-ext + filter + abs	41.89 ± 0.014	18.90 ± 0.016	38.92 ± 0.015
NEWSROOM			
TRANS-ext + absComp	32.37 ± 0.045	14.72 ± 0.043	25.69 ± 0.039
TRANS-ext + filter + absComp	32.89 ± 0.035	15.03 ± 0.042	26.65 ± 0.037
TRANS-ext + abs	33.81 ± 0.025	15.37 ± 0.026	28.92 ± 0.027
TRANS-ext + filter + abs	35.74 ± 0.018	16.52 ± 0.019	30.17 ± 0.021

5.3 Related Work

Before the ubiquitous use of neural networks, manually crafted rules and graph techniques were utilized with considerable success. Barzilay and McKeown (2005) and Cheung and Penn (2014) fused two sentences into one using their dependency parsed trees. Recently, sequence-to-sequence models (Sutskever et al., 2014) with attention (Bahdanau et al., 2014; Chopra et al., 2016), copy mechanism (Gu et al., 2016; Vinyals et al., 2015), pointer-generator (See et al., 2017) and graph-based attention (Tan et al., 2017) have been explored. Since the system generated summaries are usually evaluated on ROUGE, it has been beneficial to directly optimize this metric during training via a suitable policy using reinforcement learning (Celikyilmaz et al., 2018; Paulus et al., 2018). In order to deal with the problem of repetition that often comes with attentional RNN-based encoder-decoder models, especially when processing long sequences, Paulus et al. (2018) introduced a neural network model with intra-attention that attends over the input and continuously generated output separately. Li et al. (2018a) on the other hand, focused on the difficulty of these attentional encoder-decoder models in keeping key information during generation due to lack of guidance. Hence, they proposed a guiding generation model that combines extraction and abstraction. They first obtain keywords from the text during extraction. Afterwards, a

Key Information Guide Network (KIGN), which encodes the keywords to the key information representation to guide the process of generation is introduced.

Our work is conceptually similar to Chen and Bansal (2018) and Rush et al. (2015), where abstractive summaries are generated by simplifying extracted sentences. We simplify by paraphrasing, but different from existing models purely based on RNN, we implement a blend of two proven efficient models – TRANSFORMER and GRU-RNN encoder. Additionally, while the model of Rush et al. (2015), generates single sentence summaries, our TRANSFORMER-based model generates multi-sentence summaries.

5.4 Summary

In this Chapter we applied our extractive and paraphrasing NN architectures for abstractive summarization as our solution to **RQ0:** How can we build a machine learning model capable of generating ***abstractive*** summaries? Besides quantitative results, we presented samples that highlight the syntactic and semantic quality of some of our abstractive sentences (more examples are presented in Appendices F). Part of this work was published at the *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Workshop on Neural Generation and Translation, pages 249 - 255 (2019). Hong Kong, China.* (Egonmwan and Chali, 2019a)

Chapter 6

Multi-document Summarization

In this Chapter we address summarization of documents from multiple sources – multi-document summarization (MDS). Owing to the paucity of MDS data, we apply transfer-learning methods, using our pre-trained models (paraphrase generation and sentence compression) versus an open-source pre-trained transformer-based models for language modelling (LM) and text-to-text generation – GPT2 and T5 models respectively[31]. Specifically, we investigate **RQ5:** *How well does transfer-learning impact multi-document summarization?*

6.1 Introduction

In MDS the multiple sources of text convey a central idea or topic. Examples include news article from different sources on a defined topic (Hong et al., 2014), questionnaires completed by various individuals (Luo and Litman, 2015; Luo et al., 2016) or varied reviews from different users on a certain product (Gerani et al., 2014). As stated in Chapter 1, this thesis addresses summarization of news articles. In MDS settings, these are news articles on the same topic. These input articles are usually lengthy. For example, the two (2) major MDS datasets – DUC 2004 and MULTINEWS contain over $4,600$ and $2,100$ words per input document respectively, in contrast to about 800 words in the CNN/DM SDS dataset. This lengthy size of MDS source texts, makes it very challenging (due to memory constraints of machines (Liu and Lapata, 2019)) to directly encode and decode in an end-to-end fashion.

[31] Available from the open-source library – hugging face transformers at `https://huggingface.co/transformers/model_doc/gpt2.html`

To address this, existing work either truncates the input documents to a maximum length (Fabbri et al., 2019; Li et al., 2020) or first extracts a subset of salient passages from the source text before abstracting (Liu et al., 2018; Liu and Lapata, 2019). We employ the latter approach – extract and abstract. Besides the fact that truncating the input documents might lead to loss of valuable information, we believe that the "*extract and abstract*" approach is more similar to the way humans summarize both single and multiple documents.

Transfer learning has proven useful for low-resource tasks (Radford et al., 2019; Zoph et al., 2016; Dai et al., 2007). Transfer learning uses knowledge from a learned task to improve the performance on a related task, typically reducing the amount of required training data (Torrey and Shavlik, 2010). The choice of task to transfer knowledge from is crucial. Inspired by the need for coherency, faithfulness (to the source) and conciseness of our summaries, we choose tasks with learning objectives centered around these qualities including – Language Modeling (with OpenAI's GPT2 model), Text-to-Text Generation (with Google's T5 model) and Paraphrase generation/Sentence Compression (with our pre-trained models from chapters 3 and 4) respectively.

Radford et al. (2019) demonstrated that language models (LM) are capable of learning most NLP tasks – without any explicit supervision when trained on a new dataset of millions of webpages called WebText. Raffel et al. (2020) introduced a unified framework – T5 model pre-trained on a multi-task mixture of tasks and for which each task is converted into a text-to-text format, i.e. taking text as input and producing new text as output. The output text is expected to be faithful to the information contained in the input text. Egonmwan and Chali (2019a) and Xu and Durrett (2019) have shown the utility of paraphrase generation/sentence compression in producing concise outputs for text summarization. While all of these pre-trained models have been tested in single document summarization settings with ample training data for fine-tuning, similar experiments are yet to be carried out in MDS settings where there is a real need due to data paucity. Hence we investigate the performance of these pre-trained models for MDS.

6.2 Transfer Learning for MDS

6.2.1 Datasets

DUC 2004[32] (Paul and James, 2004): This is a test corpus provided by the National Institute for Standards and Technology (NIST) for Task 2 – Multi-document summarization. It contains 50 document clusters, with 10 documents per cluster. The documents came from the Associated Press and New York Times newswire. The documents contain about 4,600 words spanning 173 sentences on an average, while the summaries consist of about 110 words and 3 sentences.

MULTINEWS[33] (Fabbri et al., 2019): This is a multi-document news summarization dataset with about 2 – 10 documents per document cluster. The dataset consists of news articles and human-written summaries of these articles from the site `newser.com`. It contains *44,972 training, 5,622 validation* and *5,622 test* document cluster pairs. The documents contain about 2,100 words spanning 83 sentences on average, while the summaries consist of about 264 words and 10 sentences.

6.2.2 Using OpenAI's Pre-trained Language Model – GPT2

GPT2 is a TRANSFORMER-based language model proposed by Radford et al. (2019), which was pre-trained on a very large corpus of about 40GB of text from 8 million web pages [34]. Same as the objective of LMs, the learning objective of GPT2 is to predict the next word given all the previous words within some text. Formally, a language model (LM) is a probabilistic model that predicts the next token(s) in a sequence given the preceding tokens. LM is usually framed as unsupervised distribution estimation from a set of examples $(x_1, x_2, ..., x_n)$ each composed of variable length sequences of symbols $(s_1, s_2, ..., s_n)$ (Radford et al., 2019). Since language has a natural sequential ordering, it is common to factorize the joint probabilities over symbols as the product of conditional probabilities

[32] `https://duc.nist.gov/duc2004/tasks.html`

[33] `https://github.com/Alex-Fabbri/Multi-News`

[34] GPT2 is a direct scale up of GPT (*generative pre-training*) with 10x more parameters and 10x more data (Radford et al., 2019).

(Bengio et al., 2003):

$$p(x) = \prod_{i=1}^{n} p(s_i|s_1, ..., s_{n-1}). \tag{6.1}$$

Why is a pre-trained LM suitable for text summarization? Firstly, similar to LM, the task of text summarization can be expressed in a probabilistic framework as – $p(summary|document)$, that is, learning the conditional distribution of a summary given some document(s). Secondly, since GPT2 is great at predicting the next token in a sequence, leveraging this feature allows GPT2-based generated summaries to be syntactically coherent text (Radford et al., 2019).

Fine-tuning GPT2 for MDS. We fine-tune GPT2 on our MDS datasets by training it with the same architecture and hyperparamter settings as GPT2 and initializing the weights with those learned during pre-training of the GPT2 LM on the large 40GB of WebText data. GPT2 is a decoder-only transformer model that comes in four (4) different sizes[35]. We use the model size with $345M$ parameters with 24 layers and 1024 dimensional states. The checkpoint containing the weights of this pre-trained GPT2 model is publicly available from the `hugging face transformer`[31] library. We transform the $\{document, summary\}$ pairs into a contiguous sequence of texts suitable for the GPT2 LM model by appending each summary to its source document article along with a delimiter (Khandelwal et al., 2019; Radford et al., 2018). We generate 200 and 150 tokens for the MULTINEWS and DUC04 datasets respectively. Similar to Radford et al. (2019), we use Top-k random sampling[36] (Fan et al., 2018) with k=2, which the authors state helps to reduce repetition and encourages more abstractive summaries than greedy decoding.

Tokenization is performed by the GPT2 tokenizer. GPT2 tokenizer is based on byte-level Byte-Pair-Encoding (BPE) (Sennrich et al., 2016). This tokenizer has been trained to treat spaces like parts of the tokens so a word will be encoded differently whether it is at

[35]More details as to the architecture can be obtained from Radford et al. (2019) and Radford et al. (2018).
[36]Sampling from the top-k most likely words.

the beginning of the sentence (without space) or not (Radford et al., 2019). GPT counts the frequency of each word in the training corpus. It then begins from the list of all characters, and will learn merge rules to form a new token from two symbols in the vocabulary until it has learned a vocabulary of the desired size (this is a hyperparameter setting). We look at the following example from the `hugging face transformer`[37] library.

Let's say that after the pre-tokenization we have the following words with their respective frequencies:

('fun', 10), ('run', 5), ('pun', 12), ('tun', 4), ('tuns', 5)

The base vocabulary will consist of ['f', 'n', 'p', 'r', 's', 't', 'u']. We then split the words into characters to obtain ('f' 'u' 'n', 10), ('r' 'u' 'n', 5), ('p' 'u' 'n', 12), ('t' 'u' 'n', 4), ('t' 'u' 'n' 's', 5). Next, we begin merging based on the frequency of the symbols. Possible merges with their frequencies include: ('fu', 10), ('un', 36), ('ru', 5), ('tu', 9), ('pu', 12). 'u' and 'n' – 'un', has the most frequency, hence the first merge will yield:

('f' 'un', 10), ('r' 'un', 5), ('p' 'un', 12), ('t' 'un', 4), ('t' 'un' 's', 5)

'un' is then added to the vocabulary to obtain ['f', 'n', 'p', 'r', 's', 't', 'u', 'un']. The next most frequent pair of symbol is 'pu' and 'n' – 'pun' appearing 12 times. Consequently, in the next merge our corpus becomes:

('f' 'un', 10), ('r' 'un', 5), ('pun', 12), ('t' 'un', 4), ('t' 'un' 's', 5)

and 'pun' is added to the vocabulary to yield ['f', 'n', 'p', 'r', 's', 't', 'u', 'un', 'pun']. This process continues until there are no more possible merges or we have obtained the fixed total number of merges (for example, 40,378 in GPT). If there are new words left in the corpus, but merging has been stopped because we have reached the maximum limit, then the learned rules are applied to the new words. Words with characters not found in the base vocabulary are tokenized as 'unk', which stands for unknown. For example, if we stopped at the second merge and we encounter the word 'gun', it will be tokenized as ['<unk>', 'un'] because 'g' is not in our vocabulary.

[37] `https://huggingface.co/transformers/master/tokenizer_summary.html`

GPT2 uses bytes as the base vocabulary instead of characters to deal with the fact that the base vocabulary needs to get all base characters, which can be quite large if one allows for all unicode characters. With some additional rules to deal with punctuation, the GPT2 manages to tokenize every text without needing an unknown token. The GPT2 model has a vocabulary size of 50,257, which corresponds to the 256 bytes base tokens, a special end-of-text token and the symbols learned with 50,000 merges [37] (Radford et al., 2019).

Training details. Due to memory constraints we use a batch size of 1[38]. We train for 5 epochs with 32 `gradient_accumulation_steps` and a learning rate of 5e-5. We observe that fine-tuning the GPT2 model tends to exhibit a tendency referred to as *catastrophic forgetting* (Kirkpatrick et al., 2017), which is the tendency of the model to lose previous learnt knowledge abruptly while it may incorporate information relevant to target tasks, leading to overfitting (Chen et al., 2019). Hence, following Khandelwal et al. (2019) we train only 3000 samples over 5 epochs. Owing to the limit of 1024 maximum tokens for GPT2, we randomly choose 3000 samples from the MDS dataset that have a maxiumum token length of 1024 after tokenization by the GPT2 tokenizer.

6.2.3 Using Google's Pre-trained Text-to-Text Transfer Transformer Model – T5

T5 is a TRANSFORMER-based encoder-decoder model proposed by Raffel et al. (2020), pre-trained on a multi-task mixture of unsupervised and supervised tasks. The T5 model treats every text processing problem as a "text-to-text" problem, i.e. taking text as input and producing new text as output. This allows direct application of the same model, objective, training procedure, and decoding process to every other downstream task (Raffel et al., 2020). This framework makes T5 essentially useful in MDS, as the task of text summarization can be cast as that of generating an output text (summary) from an input text (document).

In order to gain generalizable knowledge that will be useful in downstream tasks, the authors performed unsupervised training on the "Colossal Clean Crawled Corpus" (C4),

[38]Experiments were run on Google Colab Pro at `https://colab.research.google.com/drive/`

an unlabelled data set consisting of hundreds of gigabytes of clean English text scraped from the web (Raffel et al., 2020). In the unsupervised pre-training setting, the objective is similar to Devlin et al. (2018)'s "denoising" objective, where the model is trained to predict missing or otherwise corrupted tokens in the input. However, in Raffel et al. (2020)'s objective formulation, consecutively masked tokens are replaced by a single sentinel token to maximize computational cost of pre-training (Raffel et al., 2020). For example, the sentence *"Text summarization is a challenging problem in* NLP*"* with the masks put on "text summarization" and "challenging" will be processed as:

<extra_id_0> is a <extra_id_1> problem in NLP

The target sequence is formed as a concatenation of the same sentinel tokens and the actual masked tokens. Thus the target sequence for the above example will be:

<extra_id_0> text summarization <extra_id_1> challenging <extra_id_2>

The model is then trained with maximum likelihood to predict the target sequence.

After unsupervised pre-training of the model, it is then fine-tuned on a number of downstream tasks including abstractive text summarization using the CNN/DM SDS dataset. Hence, we directly[39] applied the fine-tuned model to the MDS dataset by prepending the prefix *"summarize:"*[40] to the input documents[41].

6.2.4 Using our Pre-trained Models – Paraphrase Generation and Sentence Compression

In this experiment, we use our pre-trained paraphrasing and compression models presented in chapters 3 and 4 respectively, in zero-shot settings[42] on the MDS data. But first, following Liu et al. (2018), we extract salient passages from the multiple documents by applying our single-document extractive summarization model using algorithm 1. It is

[39] fine-tuning directly on the MDS data through training, did not yield any gain.

[40] Raffel et al. (2020) provides specific prefixes to be prepended to the input sequence for each downstream task.

[41] similar to Fabbri et al. (2019); Liu et al. (2018), we concatenated the multiple-documents into one mega-document as a flat sequence.

[42] no training, just inference using the pre-trained model

important to note that while this methodology bears major similarity with our single document abstractive summarization system where we extract and paraphrase, the application in multi-document settings is significantly different. Here, we do not apply the architectures of our existing models directly on MDS data. This proved to be sub-optimal due to the small number of training samples for MDS summarization. Rather, each sub-model was trained on its dedicated datasets (for example, extraction – CNN/DM, paraphrasing – QUORA data, compression – GOOGLENEWS) and then the MDS data was applied on each phase as is done during inference. However, because the extractive summarization system was modelled for single document summarization specifically, we used Algorithm 1 to iteratively apply the SDS extractive system to MDS data.

Algorithm 1 : Iteratively applying SDS extractive model on MDS data for Extractive Multi-document Summarization

procedure SDS4MDS($\text{SDS}_M, \text{MDS}_d$) ▷ SDS extractive model, SDS_M; MDS data, MDS_d
  interimSummaries $\leftarrow \{\}$
  for *document* `in` MDS_d **do**
    $s \leftarrow$ `apply` SDS_M `on` *document*
    `put` s `in` *interimSummaries*
  end for
  finalSummaries $\leftarrow$ `apply` SDS_M `on` *interimSummaries*
  return *finalSummaries*
end procedure

We preprocess the MDS dataset before application on the pre-trained models by ensuring all sentences are delimited by a space, prior to sentence tokenization by the NLTK[43] library. After salient sentences per document are extracted, we prepare them for paraphrase generation and compression. Because the paraphrase generation and sentence compression models are single sentence-level models, we split the extracted MDS data into single sentences with document markers per sentence [44]. After paraphrase generation and compression, we reassign sentences to documents with the aid of the document markers[45].

[43] `https://www.nltk.org/`

[44] this way, we know what sentences initially belonged to what documents.

[45] This is common practice for abstractive MDS (Fabbri et al., 2019; Lebanoff et al., 2018; Liu et al., 2018).

Table 6.1: ROUGE-1 results of the ablation test on test samples from DUC 04 and MULTI-NEWS.

MODELS	DUC 04	MULTINEWS
EX	31.58	45.99
EX-COM	31.71	47.20
EX-PAR	31.81	47.49
EX-COM-PAR	31.89	47.94
EX-PAR-COM	**38.01**	**49.66**

Ablation Studies

In order to find the optimal order of applying the pre-trained models, ie, paraphrase generation preceding sentence compression or otherwise, we conduct an ablation test by removing modules from EX-PAR-COM step by step and alternating the order of paraphrasing and compressing. Consequently, we obtain the following models – basic extraction (EX), extraction and sentence compression only (EX-COM), extraction and paraphrase generation only (EX-PAR), extraction then compression+paraphrase (EX-COM-PAR) and extraction then paraphrase+compression (EX-PAR-COM). The results are tabulated in Table 6.1.

6.2.5 Baselines

We compare our system with existing models in literature. For our abstractive MDS models, we experiment with swapping the order of application on the supervised pre-trained models for paraphrase generation and sentence compression.

- PEGASUS (Zhang et al., 2019a): This is a large Transformer-based encoder-decoder model pre-trained on a massive text corpora with a new self-supervised objective. Quite similar to MLM for BERT, where words are masked, in PEGASUS, important sentences are masked from an input document and are generated together as one output sequence from the remaining sentences.

- MGSum-abs (Jin et al., 2020): This is a multi-granularity interaction network that encodes semantic representations for documents, sentences, and words. It unifies the extractive and abstractive summarization by utilizing the word representations to generate the abstractive summary and the sentence representations to extract sentences.

- HI-MAP by Fabbri et al. (2019) stands for Hierarchical (Marginal Maximal Relevance) MMR-Attention Pointer-generator model. It expands the existing pointer-generator network model into a hierarchical network, which allows for calculation of sentence-level MMR scores. It consists of a pointer-generator network and an integrated MMR module.

- PG-MMR: Pointer-generator MMR network model by Lebanoff et al. (2018) incorporates the MMR algorithm (Carbonell and Goldstein, 1998), into pointer-generator networks (See et al., 2017) by adjusting the network's attention values.

- EX-COM-PAR: Extract – Compress – Paraphrase, our neural abstractive MDS model, that uses supervised pre-trained models to perform extraction, sentence compression and paraphrase generation in that order.

- EX-PAR-COM: Extract – Paraphrase – Compress, our neural abstractive MDS model, that uses supervised pre-trained models to perform extraction, paraphrase generation and sentence compression in that order.

6.2.6 Results and Evaluation

Automatic Evaluation

We measure the performance of our models using the ROUGE automatic evaluation metric (Lin, 2004) with the official `pyrouge`[46] script using option[47]. It measures the overlap of unigrams (R-1), bigrams (R-2) and skip bigrams with a max distance of four words (R-SU) between the system summary and reference summaries. The results are presented in

[46] `https://github.com/andersjo/pyrouge/tree/master/tools/ROUGE-1.5.5`
[47] -c 95 -n 2 -u -U -m -a -x -2 4

Tables 6.2 and 6.4. In table 6.3, we report the 95% confidence intervals for ROUGE-1, ROUGE-2 to show the statistical significance of our results.

Table 6.2: Average ROUGE-F1 (%) scores of various MDS abstractive models on the **DUC04** test set†.

DUC04	**R-1**	**R-2**	**R-SU4**
HI-MAP (Fabbri et al., 2019)	35.78	8.90	11.43
PG-MMR (Lebanoff et al., 2018)	36.42	**9.36**	13.23
PG-BRNN* (Gehrmann et al., 2018)	29.47	6.77	7.56
(Zhang et al., 2018)	36.70	7.83	12.40
CopyTransformer* (Gehrmann et al., 2018)	28.54	6.38	7.22
EX	31.58	6.27	10.16
EX-COM	31.71	6.42	10.27
EX-PAR	31.81	6.75	10.60
EX-COM-PAR	31.89	6.79	10.94
EX-PAR-COM	**38.01**	7.59	13.58
GPT2	28.02	4.16	**15.12**
T5	34.24	6.10	14.77

†* marks results taken from Fabbri et al. (2019). The first, second, third and fourth set of values presents abstractive baselines in literature, our extractive baselines, our abstractive systems using our pre-trained models and the results from using open-source pre-trained models respectively.

Table 6.3: 95% confidence intervals of our MDS systems.

	DUC04		**MULITINEWS**	
	R-1	**R-2**	**R-1**	**R-2**
EX	31.58 ± 0.043	6.27 ± 0.026	45.99 ± 0.089	14.14 ± 0.046
EX-COM	31.71 ± 0.035	6.42 ± 0.024	47.20 ± 0.010	14.37 ± 0.035
EX-PAR	31.81 ± 0.034	6.75 ± 0.007	47.49 ± 0.028	14.60 ± 0.005
EX-COM-PAR	31.89 ± 0.026	6.79 ± 0.006	47.94 ± 0.032	14.63 ± 0.005
EX-PAR-COM	38.01 ± 0.035	7.59 ± 0.005	49.66 ± 0.035	14.68 ± 0.004
GPT2	28.02 ± 0.041	4.16 ± 0.016	27.66 ± 0.053	6.49 ± 0.046
T5	34.24± 0.034	6.10 ± 0.025	33.73 ± 0.017	8.61 ± 0.018

Table 6.4: Average ROUGE-F1 (%) scores of various MDS abstractive models on the **MULTINEWS** test set†.

MULTINEWS	R-1	R-2	R-SU4
PEGASUS (Zhang et al., 2019a)	47.52	**18.72**	**24.91**
MGSum-abs (Jin et al., 2020)	46.00	16.81	20.09
HI-MAP (Fabbri et al., 2019)	43.47	14.89	17.41
CopyTransformer* (Gehrmann et al., 2018)	43.57	14.03	17.37
PG-ORIGINAL* (Lebanoff et al., 2018)	41.85	12.91	16.46
EX	45.99	14.14	18.84
EX-COM	47.20	14.37	20.18
EX-PAR	47.49	14.60	20.81
EX-COM-PAR	47.94	14.63	21.41
EX-PAR-COM	**49.66**	14.68	21.86
GPT2	27.66	6.49	17.22
T5	33.73	8.61	17.38

†* marks results taken from Fabbri et al. (2019). The table is subdivided in same fashion as Table 6.2.

Human Evaluation

We carried out qualitative evaluation by means of human assessment. We design the following Amazon MTurk experiment: we randomly select 50 samples (Luo et al., 2019) from the DUC04 and MULTINEWS test sets and ask the human testers (3 per sample) to rank between outputs for the three (3) models – EX-PAR and EX-COM-PAR and EX-PAR-COM. We presented the testers[48] with the reference summary and our system's summary, X, of each model. The testers were required to scale (1 – 5, with 5 being of superior quality to 1) the system's output on *informativeness* (how well does it cover the information in the reference), *fluency* (how well do the information in the systems summary flow) and *non-redundancy* (how well are information not being repeated). The results are presented in Table 6.5.

[48]We selected testers who were located in US or Canada, have Mechanical Turk Masters qualification and had HIT approval rate greater than or equal to 95%.

Table 6.5: Human Evaluation scores of our top abstractive MDS models based on Informativeness, Fluency and Non-Redundancy.

Models	Informativeness	Fluency	Non-Redundancy
EX-PAR	3.01	3.00	2.98
EX-COM-PAR	3.19	3.20	3.11
EX-PAR-COM	3.30	3.33	3.25

Source document (truncated): the second ex-wife of peter cook has issued a written apology to his first wife — supermodel christie brinkley — for ignoring her warning about the architect' s philandering . "christie and i have talked recently and i have privately apologized to her, but, given the public nature of their divorce and custody battle, i feel a public apology is also appropriate and deserved , " suzanne shaw wrote in a letter obtained by the post thursday [...] christie ' s six-year marriage to cook imploded in 2007 after she learned of his affair with 18-year-old office clerk diana bianchi [...]

GPT2 summary: the second ex-wife of peter and jennifer davis, who was also a lawyer. The couple divorced in 2005 after the divorce court ruled that she had been cheating on him with another man for years before they separated last year.

Reference summary (truncated): – christie brinkley' s marriage to peter cook ended in 2008 after he had an affair with an 18-year-old ; he went on to wed suzanne shaw [..] shaw and cook filed for divorce this year ... after cook allegedly cheated on shaw . now shaw has apologized to brinkley for bad-mouthing her years ago [...]

Figure 6.1: An example of a summary generated by fine-tuning **GPT2**. Red colored texts shows hallucinated content in the summary, with no corresponding context in the source document.

6.2.7 Analysis

Although summaries generated by fine-tuning GPT2 were fluent, due to the language modelling capabilities of GPT2, we observe that they mostly contained hallucinations as can be observed in Figure 6.1. Similar to Khandelwal et al. (2019), we infer that the GPT2 model is not fully capable of using information from the source but rather behaves more like a general domain language model.

The T5 model on the other hand does a better job at generating factual summaries, but however starts to suffer from repetition and lack of fluency at some point (see figure 6.2). In

Source document (truncated): a man has doused himself with gasoline and set himself on fire in st. peter 's square in vatican city . the vatican said thursday the 51-year-old man was treated at a nearby hospital and transferred to a larger one , with serious burns to the upper part of his body . a vatican statement says a jesuit priest initially tried to smother the flames with a jacket . he was later joined by two police officers who used a fire extinguisher to put out the blaze [..] the man had a piece of paper with the phone number of his daughter . [...]
T5 summary: the 51-year-old man was treated at a nearby hospital and transferred to a larger one, with serious burns to the upper part of his body . a vatican statement says a jesuit priest initially tried to smother the flames with a jacket . the man was later joined by two police officers who used a fire extinguisher to put out the blaze . police found a piece of paper with his daughter's phone number written on it at the scene '. .. ... the a. the . at around . local . and . with . severe burns . to the lower part of . his . they a large , the , to . of his a the s e a priest's 'the man . in . it a police . after
Reference summary (truncated): – a man is in " grave " condition after he doused himself in gasoline and set himself on fire in st. peter ' s square today around 8 : 30am local time , ansa reports . the 51-year-old man was treated at a nearby hospital , then transferred to a larger one , with serious burns to his upper body , the ap notes . a jesuit priest was first on the scene and threw his jacket on the man before two police officers arrived to extinguish the flames ; both were treated for smoke inhalation and injuries to their hands . the motive for the man ' s act isn ' t clear , though a piece of paper with his daughter ' s phone number on it was found nearby .

Figure 6.2: An example of a summary generated by the **T5** model. Red colored texts shows content with repetition and grammatical errors.

other cases, we observe that although the T5 generated summaries are accurate by human observation, they fall-short by the ROUGE metric because of differences in the vocabulary used, as the ROUGE metric is unable to capture synonymy.

While the EXPARCOM model performed best amongst our models by ROUGE standards, we observed that most of its summaries were more extractive in comparison to other models – GPT2 and T5. It however managed to introduce a few novel words as can be seen in Figure 6.3 and was effective at deleting irrelevant words from a sentence to improve conciseness.

On the effect of paraphrasing and compression on MDS, results from Tables 6.1, 6.2 and 6.4 all show that paraphrase generation and sentence compression improve the quality

Source document (truncated): speaking at a conference in sweden's third-largest city of malmö , home to a large immigrant population , the dalai lama – who won the nobel peace prize in 1989 – said europe was " morally responsible " for helping " a refugee really facing danger against their life " . " receive them , help them , educate them . . . but ultimately they should develop their own country , " said the 83-year-old tibetan who fled the capital lhasa in fear of his life after china poured troops into the region to crush an uprising . " i think europe belongs to the europeans , " he said , adding they should make clear to refugees that " they ultimately should rebuild their own country " . the dalai lama at the conference in malmö [...]

EX-PAR-COM summary: speaking at a conference in the city of malmö, home to a lot of immigrants, the dalai lama – who won the nobel peace prize – said europe was "morally responsible" for assisting "a refugee really facing danger against their life" . " receive them , help them , educate them . . . but finally they should develop their countries, " said the 83-year-old tibetan . " i think europe belongs to the europeans , " he said , adding refugees should know that " they ultimately should rebuild their own country ".

Reference summary: addressing a conference in malmö , sweden , home of many immigrants , the dalai lama said europe was " morally responsible " for helping refugees who are in danger — but that ultimately those refugees should return to their homelands . " receive them , help them , educate them ... but ultimately they should develop their own country , " he said , per the local . " i think europe belongs to the europeans , " and that refugees " ultimately should rebuild their own country , " he added . the 83-year-old buddhist spiritual leader and nobel peace prize winner fled tibet as chinese communist troops took over the area in 1959 , and settled in india , where he was granted asylum , the daily caller notes .

Figure 6.3: An example of a summary generated by EXPARCOM model. Red colored texts shows few novel words generated.

of summaries, giving credence to the utility of transfer learning from these specific tasks for MDS. From Table 6.1, we observe an average increase of about 0.2 ROUGE-1 points on top each previous output when a module is added. While using only one of paraphrase generation or sentence compression still improves over basic extraction, a combined use of both, proved to be better. Intuitively, the manner humans summarize explain the results. Mostly, we first highlight the main points, rephrase in our own words, and sketch through to see if we could further compress the summary by deleting unimportant words or sentences.

6.3 Related Work

Existing MDS methods are mostly extractive. These extractive methods are primarily modelled as graph operations with peculiarities on edge weight assignment. Christensen et al. (2013) assigned weights based on discourse relations while Erkan and Radev (2004); Lin and Bilmes (2010) measured their cosine similarity. Different algorithms, such as eigen-vector centrality (Erkan and Radev, 2004), ILP (Lin and Bilmes, 2010, 2011), are then applied on the resulting graph, to rank and retrieve highest scoring nodes. A Graph Convolutional Neural (GCN) network with sentence embeddings obtained from RNNs as input node features was recently proposed by Yasunaga et al. (2017).

Abstractive MDS on the other hand, has met with limited research due to data limitations. Liu and Lapata (2019), proposed a neural model which is capable of encoding multiple input documents hierarchically. Liu et al. (2018) handled MDS in two stages – extract and abstract. Abstraction was performed by a decoder-only sequence transduction model. Our approach is much similar to Lebanoff et al. (2018) and Zhang et al. (2018) that adapt the neural model trained on SDS for MDS by fine-tuning on the MDS dataset. We use the SDS model as-is in the extractive stage, making no changes to the encoder or decoder. Additionally, different from their methods, we incorporate other downstream tasks such as paraphrasing and sentence compression.

6.4 Summary

In this Chapter we applied transfer-learning using carefully chosen pre-trained models. We found that, while transfer-learning is generally useful for low-resource tasks like MDS, the choice of task to transfer knowledge from is crucial and has great impacts on the performance of the model. We conducted several experiments using both open-source pre-trained models versus ours from previous chapters and presented our findings with respect to: **RQ5:** *How well does transfer-learning impact multi-document summarization?* We carried out ablation studies to understand the impact of each of the components of our pre-

trained model (paraphrase generation and sentence compression) on the MDS model output. Additionally, by means of the ablation studies we sought to identify the proper sequential application order of these pre-trained models, in order to yield the best results. Examples of some of our MDS model output are presented in Appendices G and H. Part of this work is being prepared to be submitted for publication to the *2021 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. NAACL 2021.

Chapter 7

Cross-Task Knowledge Transfer for Query Focused Summarization

In this chapter, we present work done as part of a Ph.D. internship research program at the IBM Thomas J. Watson Research Center, New York during the summer of 2018. It covers a specific part of summarization – *Query Focused Summarization* (QFS), which basically refers to summarization guided by user questions or query. The summaries seek to provide answers to the given question, hence are not generic. The methodology applied demonstrates the viability of knowledge transfer between two related tasks: machine reading comprehension (MRC) and query-focused text summarization. Similar to the methods used in our generic abstractive document summarization, we also extract, compress and paraphrase for query focused summarization. However, since the goal is to investigate how well we can use knowledge from off-the-shelf models, none of the sub-components (extraction, compression, paraphrase generation) was implemented from scratch. Rather, we used already existing models[49].

7.1 Introduction

Query-based single-document text summarization is the process of selecting the most relevant points in a document for a given query and arranging them into a concise and coherent snippet of text. The query can range from an individual word to a fully formed

[49]It is important to note that the paraphrase generation and sentence compression models presented in Chapters 3 and 4 were completed after this internship program, hence could not be used as off-the-shelf components for this task.

natural language question. *Extractive* summarizers select verbatim the most relevant span of text in the source, while *abstractive* summarizers further paraphrase the selected content for better clarity and brevity.

By and large, existing approaches train models using summarization data corpora (Hasselqvist et al., 2017; Nema et al., 2017), which are of moderate size. At the same time, large corpora are available for related tasks, specifically machine reading comprehension (MRC) and machine translation (MT). To find out if such corpora have utility for summarizers, we propose methods to directly produce extractive and abstractive query-based summaries from pretrained MRC and MT modules, requiring no further adaptation or transfer learning steps.

In our experiments, this approach outperforms existing methods, suggesting a novel route to query-based summarization: pre-training systems on such related tasks, where an abundance of training data is enabling extremely rapid progress (Sun et al., 2018; Vaswani et al., 2017a; Wang et al., 2018), and using summarization-specific corpora for transfer learning.

7.2 Methodology

Our proposed system comprises of three modules for extractive summarization: retrieval of candidate answer phrases using a reading comprehension system, sentence extraction, and sentence compression. Additionally we utilize two (2) Machine Translation (MT) modules (English to Spanish and back) to paraphrase for abstractive summarization.

7.2.1 Task Definition

Given a document $D = (S_1, ..., S_n)$ with n sentences comprising of a set of words $D_W = \{d_1, ..., d_w\}$, and a query $Q = (q_1, ..., q_m)$ with m words, one desires to produce an *extractive* (S_E) or *abstractive* (S_A) summary that provides information about the answer to Q, where $S_E \subseteq D_W$ and $S_A = \{w_1, ..., w_s\} \mid \exists w_i \notin D_W$. Tables 7.1 and 7.2 show examples of our

abstractive and extractive query-based summaries respectively.

Table 7.1: Example/comparison of our *abstractive* summary on a Debatepedia sample with the output of the diversity driven attention model of Nema et al. (2017). Our generated summary is relevant to the query.

Passage: people whether overweight or not are still people. you can not compare a person with a suitcase. suitcases don't live and breathe. this rule is the same with weight. excess weight in a suitcase is not comparable with a fat person . **Query:** is it necessary to charge fat passengers extra when flying? **Reference Summary:** there is no comparison between a person and a suitcase.
Our method (abstractive) : The overweight in the bag can't be compared with the fat guy. **Diversity driven attention model:** beings are definitely by the <unk> to illegal illegal.

7.2.2 Datasets

Debatepedia[50](Nema et al., 2017): This dataset was created from Debatepedia, which is an encyclopedia of pro and con arguments and quotes on about 663 critical debate topics. Each topic has about 5 queries and each query is associated with an average of 4 documents. The authors crawled **12,695** {*document, query, summary*} triples. An example is given in Table 7.1.

CNN/DM[51](Hasselqvist et al., 2017): This dataset is the same as the one described in Section 2.3.2 for generic text summarization, but with a few adaptations for QFS. For each of the human-written abstractive highlight, Hasselqvist et al. (2017) consider one named entity to be the query. For every occurrence of an entity in a highlight, they construct a document-query-summary triple for QFS. An example of a document-query-summary triple from the CNN/DM dataset is given in Table 7.2. This adaptation yielded **1,294,730; 19,827; 20,046** training, validation and test document-query-summary triples respectively from the original CNN/DM corpus.

[50] https://github.com/PrekshaNema25/DiverstiyBasedAttentionMechanisml

[51] https://github.com/helmertz/querysum-data

Table 7.2: Example of our *extractive* summary on an example from the query-based version of CNN/DAILYMAIL (Hermann et al., 2015).

Passage (truncated): [...] offensive italian football expert and author john foot explained how paulo berlusconi 's words were offensive on several levels . " it is an insult , " foot told cnn [...] **Query:** john foot **Reference Summary:** italian football expert and author john foot says paulo berlusconi 's words are offensive on several levels .
Our method (extractive) : offensive italian football expert and author john foot explained how paulo berlusconi 's words were offensive on several levels .

7.2.3 Machine Reading Comprehension (MRC)

MRC requires the identification of a contiguous span of words in a passage that answers a given query (Hu et al., 2017; Rajpurkar et al., 2016; Wang et al., 2018). We use the MRC model by Wang et al. (2016b) trained on the SQuAD1.1 dataset (Rajpurkar et al., 2016) to identify the top n (empirically: n=5) possibly overlapping candidate answer phrases, or *chunks*, for the given query. The chunks are typically short, 3.2 words on average in the training set. Obviously, chunks from MRC are not meant to be summaries, but in our system they help the summarizer focus on the regions of the input document that appear related to the query.

7.2.4 Sentence Extraction

Sentence extraction consists of selecting the sentences containing the top n chunks produced by MRC. This is in contrast to methods based on sentence ranking algorithms such as those used in Boudin et al. (2015); Cheng and Lapata (2016); Nallapati et al. (2017); Parveen and Strube (2015). For our experiments, we impose the constraint that the candidate answer chunks for each query be contained in a single sentence. Hence, starting from $n = 5$, we iteratively reduce n until the top n candidate chunks are all contained in one sentence.

Table 7.3: Statistics of the dataset test samples after processing by the Wang et al. (2016b) MRC system's pre-processing module†.

	CNN.	**Deb.**
Test	14,725	979
Avg. #words/psg.	776	70
Avg. #words/query	2	11
Avg. #words/summ.	14	10

†Note that the preprocessor fails to parse 2-3% of the test samples in each dataset.

7.2.5 Sentence Compression

Sentence extraction often produces results that are much longer than those in the reference summaries—the training data (Table 7.3) suggests that 20 words is a good upper limit for the length of the summaries. We address this problem by introducing a novel sentence compression framework based on pruning the dependency parses of the sentences. Our approach is partially inspired by the work of Wang et al. (2016a), which performs sentence compression based on constituency parses.

Given a summary with length ≥ 20, we obtain the dependency parses of its sentences using the IBM Watson NLU toolkit [52]. Next, we remove words in the sentences (starting from the rear) that are not in a dependency relationship with any of the candidate phrases, until the summary length limit is reached.

7.2.6 Back Translation

Recent research has shown gains in leveraging on the enormous corpora in machine translation (MT) for paraphrasing (Mallinson et al., 2017; Wieting and Gimpel, 2017). Inspired by such research and our fundamental goal of investigating the viability of cross-task knowledge transfer for query-based summarization, we paraphrase our extracts using an off-the-shelf MT system[53]. The final English paraphrase of the input sentence is obtained by translating it into Spanish and back-translating the translation into English. We exper-

[52]this is a proprietary tool belonging to IBM with no granted access to the public.

[53]The MT engine is used in the IBM Watson Language Translator service.

Table 7.4: Examples of some of our paraphrased sentences using an MT system. Bolded words are novel.

Input Sentence: it is ridiculous to suggest governments should restrict their own ability to help their economies. **Paraphrase (with MT):** It is **absurd** to suggest that governments **impose limits** on their ability to help their economies.
Input Sentence: this favoritism would only increase that of which the laws are trying to suppress . **Paraphrase (with MT):** These **nepotism** will only increase the laws that you try to suppress.

imented with English-French-English and English-Italian-English as well as with multi-hops approaches before settling on the English-Spanish pair, based on subjective analysis of the results. Table 7.4 shows examples of paraphrased sentences using back-translation.

7.3 Experiments

We test our approach on two publicly available datasets—Debatepedia (Nema et al., 2017) for abstractive summarization, and the version of CNN/DM that was adapted in Hasselqvist et al. (2017); Hermann et al. (2015) for both extractive and abstractive summarization. No training was involved; the test sets were simply passed through the modules discussed in Section 7.2.

7.3.1 Evaluation

As customary in summarization tasks, we evaluate our system using ROUGE (Lin, 2004)—a family of metrics that compute the textual overlap between the output and the reference summary.

7.3.2 Results

Tables 7.5 and 7.6 summarize the performances of our model and other published models on Debatepedia and CNN/DM, respectively. Our models, both extractive and abstractive,

Table 7.5: Average ROUGE-F1 (%) performances of our model and competing models on the Debatepedia dataset.

Abstractive	**R-1**	**R-2**	**R-L**
Diversity (Nema et al., 2017)	41.26	18.75	40.43
RSA (Baumel et al., 2018)	53.09	16.10	46.18
Ours	**64.43**	**18.93**	**46.80**

Table 7.6: Average ROUGE-F1 (%) scores of our models and the competing model on the CNN/DAILYMAIL dataset.

Extractive	**R-1**	**R-2**	**R-L**	**R-SU4**
QSum (Hasselqvist et al., 2017)	33.81	18.19	29.22	17.49
Ours	**65.45**	**30.07**	**60.40**	**36.62**
Abstractive				
QSum (Hasselqvist et al., 2017)	18.25	5.04	16.17	6.13
Ours	**58.46**	**25.12**	**54.32**	**32.06**

outperform the published results on both test sets.

The extractive performance on CNN/DM indicates that the combination of a reading comprehension system and a syntax-driven compression module can be highly effective in identifying regions in a document that contain key information with respect to a given query. Moreover, the abstractive performances on both test sets show the effectiveness of machine translation as a paraphrasing component for abstractive summarization. In particular, in the CNN/DM test set the improvement over the baseline is greater in the abstractive than in the extractive case, again suggesting that both text selection and MT-based paraphrasing contribute to the gain.

7.4 Related Work

Text summarization has long been an active area of research and query-based summarization has gained momentum more recently. Classical summarization models usually identify salient parts of a text by encapsulating manually crafted rules into linear functions (Lin and Bilmes, 2011) which are solved using integer linear programming (ILP) (Boudin

et al., 2015; Nayeem and Chali, 2017), conditional random fields (CRF) (Shen et al., 2007), or graph algorithms (Erkan and Radev, 2004; Parveen and Strube, 2015). More recently, neural networks, mostly with an encoder-decoder framework (Bahdanau et al., 2014), have been used to learn the underlying features (Jadhav and Rajan, 2018; Nallapati et al., 2016) trained by minimizing the cross-entropy loss (Nallapati et al., 2017) or reinforcement learning (Narayan et al., 2018; Paulus et al., 2018).

Our baseline models for query-based summarization (Hasselqvist et al., 2017; Nema et al., 2017) are both implemented on the encoder-decoder framework with the former incorporating a diversity function in their model aimed at minimizing the problem of repetitive word generation inherent in encoder-decoder models. However our approach is similar to neither, as our goal is not to train a query-based summarizer from scratch but rather to investigate the competitiveness of using pre-trained models for closely related tasks—i.e., MRC and MT—on query-based summarization.

7.5 Summary

We described an approach to extractive and abstractive summarization that relies on components designed for different tasks: MRC, sentence compression, and MT. We have shown that retrieving the top *n* answer chunks from a passage with an MRC system and trimming the corresponding sentences using their dependency trees yields an extractive summarizer that outperforms published results on a publicly available dataset. We also showed that using MT to produce a paraphrase of the answers yields a high-performance abstractive summarization method. This work lays the foundations for transfer learning based approaches that use summarization data to adapt MRC models for summarization. This work was published at the *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Workshop on Machine Reading and Question Answering, pages 72 - 77 (2019). Hong Kong, China* (Egonmwan et al., 2019).

Chapter 8

Conclusion

In this thesis, we provided a solution to the overarching research question **RQ0:** *How can we build a machine learning model that is capable of generating both extractive and abstractive summaries which are grammatically correct and are faithful to the facts contained in the source text?*

While there are ample interesting proposed models for the task of text summarization, there is still room for performance improvement on the task. We found that implementing seemingly simple approaches yielded some improvements over existing models. For example, paying more attention to the dataset by filtering out flawed samples before application on the machine learning model.

In the introduction of this thesis, we hypothesized that some key abstraction techniques such as paraphrase generation and sentence compression could possibly improve machine generated summaries as humans often generate summaries by paraphrasing and sometimes compressing sentences to the desired summary length. Hence, we implemented each of these techniques independently and incorporated them for the task of single and multi-document summarization. Evaluation and ablation studies supported our hypothesis and showed the useful impact of these techniques on performance and a good combination order to yield better results.

The concept of extract, paraphrase and compress proved to be even more useful in multi-document settings where large training data is non-existent. It was therefore very beneficial to implement transfer learning for multi-document summarization, by applying

these pre-trained models on the few available MDS data.

One challenge we experienced in this work, was the choice of a NN architectural model. There exists a variety of NN models – Deep Belief Nets (DBN), Convoluted Neural Network CNN, RNN (LSTM, GRU) and more recently, the TRANSFORMER. Literature proves that all of these models work quite well, however making an informed decision as to which would give the best desired result for a particular task usually requires series of non-exhaustive experiments. We found that in some cases, an hybrid of these models work best. For example, we used a stack of the TRANSFORMER and GRU encoders for paraphrase generation, as experiments showed that this hybrid performed better than using only either one of those model choices.

While it is common practice to implement early-stopping during training in order to prevent the model from over-fitting on the training set, we found that deciding on what basis to stop training on the model is often less thoroughly considered. We carried out experiments on other early-stopping criteria besides the typical loss score, such as the F1-measure and the embedding-based cosine similarity. We found that using these other mentioned heuristics, not only improved overall performance but helped to maintain the grammaticality and accuracy of facts of the machine generated outputs.

In totality, we provided improved solutions to the task of summarization in both single and multi-document settings. The next section provides a concise summary of our work in line with the theme of this thesis. Finally, I am interested in applying the knowledge gained from working on text summarization in other NLP problems like Question Answering (QA), Dialogue and Discourse Processing.

8.1 Summary

To guide us through the major contributions of this thesis, we raised five (5) research questions at the onset in Chapter 1. These questions were answered in details through Chapters 2 to 6. We re-iterate the research questions and highlight the solutions provided.

RQ1: *How can we identify parts of a text that are the most relevant for an extractive summary with improved accuracy?* We identified the challenge of the presence of noisy samples in available corpus, where the summaries cannot be directly inferred from the documents, either due to error during data collection or the presence of external information by the humans who wrote the abstractive reference summaries. Hence, we filtered the dataset to remove flawed samples (Section 2.3.3). Next, we converted the input document containing a sequence of sentences into a sequence of labels for classification of the sentences as extraction-worthy or otherwise (Section 2.3.5). Finally, we trained a TRANSFORMER-based binary classifier to automatically label sentences for extraction into summaries (Section 2.3.6). See results of the evaluated model on the CNN/DM and NEWSROOM datasets in Tables 2.2 and 2.3 respectively.

We hypothesized that two (2) key abstraction techniques – sentence paraphrasing and compression could improve the quality of machine-generated abstractive summaries. Hence, we carried out investigations with the following research questions:

RQ2: *How can we build a paraphrasing model to help with abstractive summarization?* We proposed a model capable of generating paraphrases of sentences trained on dedicated paraphrasing datasets (Section 3.2.2). The model combined the efficiency of two (2) interesting neural networks – TRANSFORMER and GRU-RNN (Section 3.2.4). See results of the evaluated model on the MSCOCO and QUORA paraphrasing datasets in Tables 3.1 and 3.2 respectively. Having ascertained the efficiency of the architectural choice for paraphrasing task, we applied it on summarization, by paraphrasing the extracted sentences from Section 2.3.6 (See Section 3.2.4). Evaluation results of our abstractive single document summarization model on the CNN/DM and NEWSROOM datasets were presented in Tables 5.2 and 5.3 respectively.

RQ3: *How can we build a compression model to help improve the conciseness of machine generated summaries?* We modelled the problem as a word-level binary classification task, trained on a TRANSFORMER-based encoder and softmax classifier (Section

4.2.4). One major challenge with the task, is ensuring that the compressed sentence is faithful to the facts in the original sentence. Hence, it was necessary to address the next question.

RQ4: *How can we investigate that our machine generated summaries are factual, that is, that they contain accurate information?* To tackle this, early-stopping based on three (3) different heuristics – loss, F1-score and embedding-based cosine similarity were employed (Section 4.2.4). We found that stopping the training process based on the embedding-based cosine similarity of the machine-generated sentence compression and the reference sentence compression, preserved the facts and grammaticality of the original sentence the most. Besides automatic evaluations, human evaluations designed on Amazon Mechanical Turk were performed (Section 4.2.8).

In this Chapter 6 we applied transfer learning by using pretrained models from other related tasks such as – extractive summarization, sentence paraphrasing and compression on MDS as our solution to:

RQ5: *How well does transfer-learning impact multi-document summarization?* We carried out experiments investigating the impact of different pre-trained models on MDS, such as – GPT2, T5 and Paraphrase generation/sentence compression. Using our paraphrase generation and sentence compression models were comparatively better. We then performed ablation studies (Section 6.2.4) to understand the impact of each of these components (paraphrase generation and sentence compression) on the output, and find out the optimal application order, that is, extract-paraphrase and compress or extract-compress and paraphrase. The former produced better results in our experiments when tested on the DUC 2004 and MULTINEWS MDS datasets (Section 6.2.6).

8.2 Future Work

In this work, we found that, while interesting concepts and ideas could make for improved performance in NLP tasks, the efficiency of the specific neural network being em-

ployed is very crucial. For the most part, we simply used existing neural networks without making any changes to the internal functionality. In future research, it would be intriguing to dive deeper into these "black boxes" of neural networks and study what can be tweaked to yield better performance or better still, come up with a new NN structure from scratch. One that is better able to deal with the ambiguity in English language and capture semantics in sentences more efficiently.

Secondly, creation of corpora for certain downstream problems like *Sentence fusion* would be beneficial to summarization task. In sentence fusion, the goal is to coherently combine two (2) sentences into one. A model capable of accurately fusing two sentences into one without redundancy will have great utility for summarization. Currently, no such large sentence fusion dataset exists.

Finally, it will be interesting to apply the knowledge gained from working on text summarization in other NLP problems like Question Answering (QA), Dialogue and Discourse Processing.

Bibliography

Ethem Alpaydin. 2020. *Introduction to machine learning*. MIT press.

Marianna Apidianaki, Guillaume Wisniewski, Anne Cocos, and Chris Callison-Burch. 2018. Automated paraphrase lattice creation for HyTER machine translation evaluation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 480–485.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*.

Pierre Baldi and Peter J Sadowski. 2013. Understanding dropout. *Advances in neural information processing systems*, 26:2814–2822.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Federico Barrios, Federico López, Luis Argerich, and Rosa Wachenchauzer. 2016. Variations of the similarity function of textrank for automated summarization. *CoRR*.

Regina Barzilay and Kathleen R McKeown. 2005. Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–328.

Tal Baumel, Matan Eyal, and Michael Elhadad. 2018. Query Focused Abstractive Summarization: Incorporating Query Relevance, Multi-Document Coverage, and Summary Length Constraints into seq2seq Models. *arXiv preprint arXiv:1801.07704*.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. 2011. Jointly learning to extract and compress. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 481–490. Association for Computational Linguistics.

Florian Boudin, Hugo Mougard, and Benoit Favre. 2015. Concept-based summarization using integer linear programming: From concept pruning to multiple optimal solutions. In *Conference on Empirical Methods in Natural Language Processing (EMNLP) 2015*.

Jason Brownlee. 2018. How to Implement a Beam Search Decoder for Natural Language Processing. `https://machinelearningmastery.com/beam-search-decoder-natural-language-processing/`. Accessed: 2020-05-15.

Annika Brundyn. 2018. The Beginner's Guide to Recurrent Neural Networks and Text Generation. https://medium.com/@annikabrundyn1/the-beginners-guide-to-recurrent-neural-networks-and-text-generation-44a70c34067f. Accessed: 2020-05-12.

Duy Duc An Bui, Guilherme Del Fiol, John F Hurdle, and Siddhartha Jonnalagadda. 2016. Extractive text summarization system to aid data extraction from full text in systematic review development. *Journal of biomedical informatics*, 64:265–272.

Chris Callison-Burch, Philipp Koehn, and Miles Osborne. 2006. Improved statistical machine translation using paraphrases. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 17–24. Association for Computational Linguistics.

Ziqiang Cao, Wenjie Li, Sujian Li, and Furu Wei. 2018. Retrieve, rerank and rewrite: Soft template based neural summarization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 152–161.

Ziqiang Cao, Chuwei Luo, Wenjie Li, and Sujian Li. 2017. Joint copying and restricted generation for paraphrase. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336.

Asli Celikyilmaz, Antoine Bosselut, Xiaodong He, and Yejin Choi. 2018. Deep Communicating Agents for Abstractive Summarization. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1662–1675.

Mia Xu Chen, Orhan Firat, Ankur Bapna, Melvin Johnson, Wolfgang Macherey, George Foster, Llion Jones, Mike Schuster, Noam Shazeer, Niki Parmar, et al. 2018. The Best of Both Worlds: Combining Recent Advances in Neural Machine Translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–86.

Xinyang Chen, Sinan Wang, Bo Fu, Mingsheng Long, and Jianmin Wang. 2019. Catastrophic forgetting meets negative transfer: Batch spectral shrinkage for safe transfer learning. In *Advances in Neural Information Processing Systems*, pages 1908–1918.

Yen-Chun Chen and Mohit Bansal. 2018. Fast Abstractive Summarization with Reinforce-Selected Sentence Rewriting. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–686.

Jianpeng Cheng and Mirella Lapata. 2016. Neural Summarization by Extracting Sentences and Words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 484–494.

Jackie Chi Kit Cheung and Gerald Penn. 2014. Unsupervised sentence enhancement for automatic summarization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 775–786.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Sumit Chopra, Michael Auli, and Alexander M Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 93–98.

Janara Christensen, Stephen Soderland, Oren Etzioni, et al. 2013. Towards coherent multi-document summarization. In *Proceedings of the 2013 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pages 1163–1173.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Alibaba Cloud. 2018. Self-Attention Mechanisms in Natural Language Processing Alibaba Cloud. `https://medium.com/@Alibaba_Cloud/self-attention-mechanisms-in-natural-language-processing-9f28315ff905`. Accessed: 2020-04-25.

John M Conroy and Dianne P O'leary. 2001. Text summarization via hidden markov models. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 406–407.

Wenyuan Dai, Qiang Yang, Gui-Rong Xue, and Yong Yu. 2007. Boosting for transfer learning. In *Proceedings of the 24th international conference on Machine learning*, pages 193–200.

Dipanjan Das and Andre FT Martins. 2007. A Survey on Automatic Text Summarization. Language Technologies Institute.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the ninth workshop on statistical machine translation*, pages 376–380.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Li Dong, Jonathan Mallinson, Siva Reddy, and Mirella Lapata. 2017. Learning to Paraphrase for Question Answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 875–886.

Bonnie Dorr, David Zajic, and Richard Schwartz. 2003. Hedge trimmer: A parse-and-trim approach to headline generation. In *Proceedings of the HLT-NAACL 03 on Text summarization workshop-Volume 5*, pages 1–8. Association for Computational Linguistics.

Greg Durrett, Taylor Berg-Kirkpatrick, and Dan Klein. 2016. Learning-Based Single-Document Summarization with Compression and Anaphoricity Constraints. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1998–2008.

Elozino Egonmwan, Vittorio Castelli, and Md Arafat Sultan. 2019. Cross-Task Knowledge Transfer for Query-Based Text Summarization. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 72–77.

Elozino Egonmwan and Yllias Chali. 2019a. Transformer-based Model for Single Documents Neural Summarization. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 70–79.

Elozino Egonmwan and Yllias Chali. 2019b. Transformer and seq2seq model for Paraphrase Generation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 249–255.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479.

Alexander Richard Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. 2019. Multi-News: A Large-Scale Multi-Document Summarization Dataset and Abstractive Hierarchical Model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1074–1084.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical Neural Story Generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898.

Thibault Fevry and Jason Phang. 2018. Unsupervised Sentence Compression using Denoising Auto-Encoders. *CoNLL 2018*, page 413.

Katja Filippova, Enrique Alfonseca, Carlos A Colmenares, Lukasz Kaiser, and Oriol Vinyals. 2015. Sentence compression by deletion with lstms. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 360–368.

Katja Filippova and Yasemin Altun. 2013. Overcoming the Lack of Parallel Data in Sentence Compression. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1481–1491.

Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. 2018. Bottom-Up Abstractive Summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4098–4109.

Shima Gerani, Yashar Mehdad, Giuseppe Carenini, Raymond Ng, and Bita Nejat. 2014. Abstractive summarization of product reviews using discourse structure. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1602–1613.

Giuliano Giacaglia. 2019. How Transformers Work. `https://towardsdatascience.com/transformers-141e32e69591`. Accessed: 2020-04-26.

Jade Goldstein, Vibhu O Mittal, Jaime G Carbonell, and Mark Kantrowitz. 2000. Multi-document summarization by sentence extraction. In *NAACL-ANLP 2000 Workshop: Automatic Summarization*.

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A Dataset of 1.3 Million Summaries with Diverse Extractive Strategies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 708–719.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating Copying Mechanism in Sequence-to-Sequence Learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1631–1640.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2018. A deep generative framework for paraphrase generation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Samer Hassan, Andras Csomai, Carmen Banea, Ravi Sinha, and Rada Mihalcea. 2007. Unt: Subfinder: Combining knowledge sources for automatic lexical substitution. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 410–413.

Johan Hasselqvist, Niklas Helmertz, and Mikael Kågebäck. 2017. Query-Based Abstractive Summarization Using Neural Networks. *arXiv preprint arXiv:1712.06100*.

Douglas M Hawkins. 2004. The problem of overfitting. *Journal of chemical information and computer sciences*, 44(1):1–12.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in neural information processing systems*, pages 1693–1701.

Geoffrey E Hinton, Alexander Krizhevsky, Ilya Sutskever, and Nitish Srivastva. 2016. System and method for addressing overfitting in a neural network. US Patent 9,406,017.

Elad Hoffer, Itay Hubara, and Daniel Soudry. 2017. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. In *Advances in Neural Information Processing Systems*, pages 1731–1741.

Kai Hong, John Conroy, Benoit Favre, Alex Kulesza, Hui Lin, and Ani Nenkova. 2014. A Repository of State of the Art and Competitive Baseline Summaries for Generic News Summarization. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1608–1616.

Filippova, Katja and Strube, Michael. 2008. Dependency tree based sentence compression. In *Proceedings of the Fifth International Natural Language Generation Conference*, pages 25–32. Association for Computational Linguistics.

Wan-Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. 2018. A Unified Model for Extractive and Abstractive Summarization using Inconsistency Loss. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 132–141.

Minghao Hu, Yuxing Peng, Zhen Huang, Xipeng Qiu, Furu Wei, and Ming Zhou. 2017. Reinforced Mnemonic Reader for Machine Reading Comprehension. *arXiv preprint arXiv:1705.02798*.

Hakan Inan, Khashayar Khosravi, and Richard Socher. 2016. Tying word vectors and word classifiers: A loss framework for language modeling. *arXiv preprint arXiv:1611.01462*.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial Example Generation with Syntactically Controlled Paraphrase Networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885.

Aishwarya Jadhav and Vaibhav Rajan. 2018. Extractive Summarization with SWAP-NET: Sentences and Words from Alternating Pointer Networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 142–151.

Alammar Jay. 2018. The Illustrated Transformer. http://jalammar.github.io/illustrated-transformer/. Accessed: 2020-04-27.

Hanqi Jin, Tianming Wang, and Xiaojun Wan. 2020. Multi-granularity interaction network for extractive and abstractive multi-document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6244–6254, Online. Association for Computational Linguistics.

Hongyan Jing and Kathleen R McKeown. 2000. Cut and paste based text summarization. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 178–185. Association for Computational Linguistics.

Daniel Jurafsky and James H Martin. 2016. Hidden markov models. *Speech and Language Processing*, page 1024.

Tom Kenter and Maarten De Rijke. 2015. Short text similarity with word embeddings. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, pages 1411–1420. ACM.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. 2016. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*.

Urvashi Khandelwal, Kevin Clark, Dan Jurafsky, and Lukasz Kaiser. 2019. Sample efficient text summarization using a single pre-trained transformer. *arXiv preprint arXiv:1905.08836*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

Kevin Knight and Daniel Marcu. 2000. Statistics-based summarization-step one: Sentence compression. *AAAI/IAAI*, 2000:703–710.

Panagiotis Kouris, Georgios Alexandridis, and Andreas Stafylopatis. 2019. Abstractive text summarization based on deep learning and semantic content generalization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5082–5092.

Raymond Kozlowski, Kathleen F McCoy, and K Vijay-Shanker. 2003. Generation of single-sentence paraphrases from predicate/argument structure using lexico-grammatical resources. In *Proceedings of the second international workshop on Paraphrasing-Volume 16*, pages 1–8. Association for Computational Linguistics.

Ben Kröse, Ben Krose, Patrick van der Smagt, and Patrick Smagt. 1993. An introduction to neural networks.

Yogan Jaya Kumar and Naomie Salim. 2012. Automatic multi document summarization approaches. In *KS Gayathri, Received BE degree in CSE from Madras University in 2001 and ME degree from Anna University, Chennai. She is doing Ph. D. in the area of Reasoning in Smart*. Citeseer.

Julian Kupiec, Jan Pedersen, and Francine Chen. 1995. A trainable document summarizer. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 68–73.

Dac-Viet Lai, Nguyen Truong Son, and Nguyen Le Minh. 2017. Deletion-based sentence compression using Bi-enc-dec LSTM. In *International Conference of the Pacific Association for Computational Linguistics*, pages 249–260. Springer.

Logan Lebanoff, John Muchovej, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. 2019a. Analyzing Sentence Fusion in Abstractive Summarization. *EMNLP-IJCNLP 2019*, page 104.

Logan Lebanoff, Kaiqiang Song, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. 2019b. Scoring Sentence Singletons and Pairs for Abstractive Summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2175–2189.

Logan Lebanoff, Kaiqiang Song, and Fei Liu. 2018. Adapting the Neural Encoder-Decoder Framework from Single to Multi-Document Summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4131–4141.

Ju-Hong Lee, Sun Park, Chan-Min Ahn, and Daeho Kim. 2009. Automatic generic document summarization based on non-negative matrix factorization. *Information Processing & Management*, 45(1):20–34.

Chenliang Li, Weiran Xu, Si Li, and Sheng Gao. 2018a. Guiding generation for abstractive text summarization based on key information guide network. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 55–60.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2016. Visualizing and Understanding Neural Models in NLP. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 681–691.

Mingchen Li, Mahdi Soltanolkotabi, and Samet Oymak. 2019. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. *arXiv preprint arXiv:1903.11680*.

Wei Li, Xinyan Xiao, Jiachen Liu, Hua Wu, Haifeng Wang, and Junping Du. 2020. Leveraging Graph to Improve Abstractive Multi-Document Summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6232–6243, Online. Association for Computational Linguistics.

Zichao Li, Xin Jiang, Lifeng Shang, and Hang Li. 2018b. Paraphrase Generation with Deep Reinforcement Learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3865–3878.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*.

Hui Lin and Jeff Bilmes. 2010. Multi-document summarization via budgeted maximization of submodular functions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 912–920.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 510–520. Association for Computational Linguistics.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Zachary C Lipton, Charles Elkan, and Balakrishnan Naryanaswamy. 2014. Optimal thresholding of classifiers to maximize F1 measure. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 225–239. Springer.

Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. *arXiv preprint arXiv:1801.10198*.

Yang Liu and Mirella Lapata. 2019. Hierarchical Transformers for Multi-Document Summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Yang Liu, Ivan Titov, and Mirella Lapata. 2019. Single document summarization as tree induction. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1745–1755, Minneapolis, Minnesota. Association for Computational Linguistics.

Zhonglei Lu, Wenfen Liu, Yanfang Zhou, Xuexian Hu, and Binyu Wang. 2017. An Effective Approach of Sentence Compression Based on "Re-read" Mechanism and Bayesian Combination Model. In *Chinese National Conference on Social Media Processing*, pages 129–140. Springer.

Ling Luo, Xiang Ao, Yan Song, Feiyang Pan, Min Yang, and Qing He. 2019. Reading Like HER: Human Reading Inspired Extractive Summarization. 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)*, pages 3024–3034.

Wencan Luo and Diane Litman. 2015. Summarizing student responses to reflection prompts. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1955–1960.

Wencan Luo, Fei Liu, Zitao Liu, and Diane Litman. 2016. Automatic Summarization of Student Course Feedback. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 80–85.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.

Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. 2018. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 181–196.

Jonathan Mallinson, Rico Sennrich, and Mirella Lapata. 2017. Paraphrasing revisited with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 881–893.

Kathleen R McKeown. 1983. Paraphrasing questions using given and new information. *Computational Linguistics*, 9(1):1–10.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Elizabeth Million. 2007. The hadamard product. *Course Notes*, 3(6).

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. SummaRuNNer: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents. In *AAAI*, pages 3075–3081.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Ça glar Gulçehre, and Bing Xiang. 2016. Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond. *CoNLL 2016*, page 280.

Courtney Napoles, Benjamin Van Durme, and Chris Callison-Burch. 2011. Evaluating sentence compression: Pitfalls and suggested remedies. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 91–97.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Ranking Sentences for Extractive Summarization with Reinforcement Learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 1747–1759.

Mir Tafseer Nayeem and Yllias Chali. 2017. Extract with Order for Coherent Multi-Document Summarization. In *Proceedings of TextGraphs-11: the Workshop on Graph-based Methods for Natural Language Processing*, pages 51–56.

Preksha Nema, Mitesh M Khapra, Anirban Laha, and Balaraman Ravindran. 2017. Diversity driven attention model for query-based abstractive summarization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1063–1072.

Ani Nenkova and Kathleen McKeown. 2012. A survey of text summarization techniques. In *Mining text data*, pages 43–76. Springer.

Graham Neubig. 2017. Neural machine translation and sequence-to-sequence models: A tutorial. *arXiv preprint arXiv:1703.01619*.

Christopher Olah. 2015. Understanding LSTM Networks. `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`. Accessed: 2020-05-13.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Daraksha Parveen and Michael Strube. 2015. Integrating Importance, Non-Redundancy and Coherence in Graph-Based Extractive Summarization. In *IJCAI*, pages 1298–1304.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318.

Over Paul and Yen James. 2004. An introduction to duc-2004. In *Proceedings of the 4th Document Understanding Conference (DUC 2004)*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A Deep Reinforced Model for Abstractive Summarization. In *International Conference on Learning Representations*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

Michael Phi. 2018. Illustrated Guide to Recurrent Neural Networks. https://towardsdatascience.com/illustrated-guide-to-recurrent-neural-networks-79e5eb8049c9. Accessed: 2020-05-12.

Aaditya Prakash, Sadid A Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2016. Neural paraphrase generation with stacked residual LSTM networks. *arXiv preprint arXiv:1610.03098*.

Lutz Prechelt. 1998a. Early stopping-but when? In *Neural Networks: Tricks of the trade*, pages 55–69. Springer.

Lutz Prechelt. 1998b. Automatic early stopping using cross validation: quantifying the criteria. *Neural Networks*, 11(4):761–767.

Yunchen Pu, Zhe Gan, Ricardo Henao, Xin Yuan, Chunyuan Li, Andrew Stevens, and Lawrence Carin. 2016. Variational autoencoder for deep learning of images, labels and captions. In *Advances in neural information processing systems*, pages 2352–2360.

Chris Quirk, Chris Brockett, and William Dolan. 2004. Monolingual machine translation for paraphrase generation. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 142–149.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Saurabh Rathor. 2018. Simple RNN vs GRU vs LSTM :- Difference lies in More Flexible control. https://medium.com/@saurabh.rathor092/simple-rnn-vs-gru-vs-lstm-difference-lies-in-more-flexible-control-5f33e07b1e57. Accessed: 2020-05-13.

Irina Rish et al. 2001. An empirical study of the naive Bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46.

Sebastian Ruder. 2016. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A Neural Attention Model for Abstractive Sentence Summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389.

Nathaniel Schenker and Jane F Gentleman. 2001. On judging the significance of differences by examining the overlap between confidence intervals. *The American Statistician*, 55(3):182–186.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Aliaksei Severyn and Alessandro Moschitti. 2013. Automatic feature engineering for answer selection and extraction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 458–467.

Shikhar Sharma, Layla El Asri, Hannes Schulz, and Jeremie Zumer. 2017. Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. *arXiv preprint arXiv:1706.09799*.

Dou Shen, Jian-Tao Sun, Hua Li, Qiang Yang, and Zheng Chen. 2007. Document Summarization Using Conditional Random Fields. In *IJCAI*, volume 7, pages 2862–2867.

Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. 2018. Leveraging context information for natural question generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 569–574.

Yu Su and Xifeng Yan. 2017. Cross-domain Semantic Parsing via Paraphrasing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1235–1246.

Fu Sun, Linyang Li, Xipeng Qiu, and Yang Liu. 2018. U-Net: Machine Reading Comprehension with Unanswerable Questions. *arXiv preprint arXiv:1810.06638*.

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. 2013. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2017. Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1171–1181.

Igor V Tetko, David J Livingstone, and Alexander I Luik. 1995. Neural network studies. 1. Comparison of overfitting and overtraining. *Journal of chemical information and computer sciences*, 35(5):826–833.

Tijmen Tieleman and GE Hinton. 2012. Neural networks for machine learning. *Coursera (Lecture 65-RMSprop)*.

Lisa Torrey and Jude Shavlik. 2010. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI global.

Aäron Van Den Oord, Sander Dieleman, and Benjamin Schrauwen. 2014. Transfer learning by supervised pre-training for audio-based music classification. In *Conference of the International Society for Music Information Retrieval (ISMIR 2014)*.

Ashish Vaswani, Samy Bengio, Eugene Brevdo, Francois Chollet, Aidan Gomez, Stephan Gouws, Llion Jones, Łukasz Kaiser, Nal Kalchbrenner, Niki Parmar, et al. 2018. Tensor2Tensor for Neural Machine Translation. In *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Papers)*, volume 1, pages 193–199.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017a. Attention is All you Need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017b. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 2*, pages 2692–2700. MIT Press.

Liangguo Wang, Jing Jiang, Hai Leong Chieu, Chen Hui Ong, Dandan Song, and Lejian Liao. 2017. Can syntax help? Improving an LSTM-based sentence compression model for new domains. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1385–1393.

Lu Wang, Hema Raghavan, Vittorio Castelli, Radu Florian, and Claire Cardie. 2016a. A sentence compression based framework to query-focused multi-document summarization. *arXiv preprint arXiv:1606.07548*.

Wei Wang, Ming Yan, and Chen Wu. 2018. Multi-granularity hierarchical attention fusion networks for reading comprehension and question answering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1705–1714.

Zhiguo Wang, Haitao Mi, Wael Hamza, and Radu Florian. 2016b. Multi-perspective context matching for machine comprehension. *arXiv preprint arXiv:1612.04211*.

John Wieting and Kevin Gimpel. 2017. Pushing the Limits of Paraphrastic Sentence Embeddings with Millions of Machine Translations. *arXiv preprint arXiv:1711.05732*.

Wen Xiao and Giuseppe Carenini. 2019. Extractive Summarization of Long Documents by Combining Global and Local Context. 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)*, pages 3002–3012.

Jiacheng Xu and Greg Durrett. 2019. Neural Extractive Text Summarization with Syntactic Compression. 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)*, pages 3283–3294.

Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir Radev. 2017. Graph-based Neural Multi-Document Summarization. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 452–462.

Naitong Yu, Jie Zhang, Minlie Huang, and Xiaoyan Zhu. 2018. An Operation Network for Abstractive Sentence Compression. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1065–1076.

Jianmin Zhang, Jiwei Tan, and Xiaojun Wan. 2018. Adapting neural single-document summarization model for abstractive multi-document summarization: A pilot study. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 381–390.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J Liu. 2019a. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. *arXiv preprint arXiv:1912.08777*.

Xingxing Zhang, Furu Wei, and Ming Zhou. 2019b. HIBERT: Document Level Pre-training of Hierarchical Bidirectional Transformers for Document Summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5059–5069.

Zhilu Zhang and Mert Sabuncu. 2018. Generalized cross entropy loss for training deep neural networks with noisy labels. In *Advances in neural information processing systems*, pages 8778–8788.

Sanqiang Zhao, Rui Meng, Daqing He, Andi Saptono, and Bambang Parmanto. 2018a. Integrating Transformer and Paraphrase Rules for Sentence Simplification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3164–3173, Brussels, Belgium. Association for Computational Linguistics.

Shiqi Zhao, Cheng Niu, Ming Zhou, Ting Liu, and Sheng Li. 2008. Combining multiple resources to improve SMT-based paraphrasing model. In *Proceedings of ACL-08: HLT*, pages 1021–1029.

Yang Zhao, Zhiyuan Luo, and Akiko Aizawa. 2018b. A Language Model based Evaluator for Sentence Compression. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 170–175.

Yang Zhao, Hajime Senuma, Xiaoyu Shen, and Akiko Aizawa. 2017. Gated neural network for sentence compression using linguistic knowledge. In *International Conference on Applications of Natural Language to Information Systems*, pages 480–491. Springer.

Qingyu Zhou, Nan Yang, Furu Wei, Shaohan Huang, Ming Zhou, and Tiejun Zhao. 2018. Neural Document Summarization by Jointly Learning to Score and Select Sentences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 654–663.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer Learning for Low-Resource Neural Machine Translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575.

Appendix A

Sample system extractive summaries from CNN/DM dataset

Here, we show some examples of our system extractive summaries using our single document extractive summarization model described in Chapter 2 and human-written reference summaries from CNN/DM dataset.

System output: three former managers of the aids healthcare foundation filed a suit last week alleging the company paid employees and patients kickbacks for patient referrals in an effort to boost funding from federal health programs . employees were paid 100dollar bonuses for referring patients with positive test results to its clinics and pharmacies . the lawsuit alleges kickbacks started in 2010 at the company's california headquarters and spread to programs in florida and several other locations .
Human-written Reference summary: three former managers of the aids healthcare foundation filed a suit . they alleged the company paid employees and patients kickbacks for patient referrals in an effort to boost funding from federal health programs . employees were paid 100dollar bonuses for referring patients with positive test results to its clinics and pharmacies . the lawsuit alleges kickbacks started in 2010 at the company's california headquarters and spread to programs in florida and several other locations.

System output: zachariah fike head of vermont-based purple hearts reunited , says the military id belonged to world war ii veteran cpl. william benn , who lost them in 1939 at a coastal artillery placement on salisbury beach . metal detector enthusiast bill ladd found them after a storm last year.
Human-written Reference summary: zachariah fike head of vermont-based purple hearts reunited , says the military id belonged to world war ii veteran cpl. william benn . benn lost them in 1939 at a coastal artillery placement on salisbury beach . discovered last year by metal detector enthusiast following a storm.'

System output: a consumer update revised by the fda on tuesday said that the tests should be used for medical reasons only and only be performed by licensed staff . ultrasounds are used to help doctors determine the health , size , and age of the child and can detect birth defects . while there is no proof that ultrasounds are harmful to the baby , it heats the baby 's tissues slightly and can also cause small bubbles in tissue.
Human-written Reference summary: a consumer update revised by the fda on tuesday

said that ultrasounds should be performed for medical reasons only and by licensed staff . while there is no proof that ultrasounds are harmful to the baby , it heats the baby 's tissues slightly and can also cause small bubbles in tissue . ultrasounds help doctors determine the health , size , and age of the child and can detect birth defects and should n't be used for emotional reasons.

System output: a contestant on friday night 's episode of jeopardy left a lasting impression with viewers for all the wrong reasons after giving a highly inappropriate answer to a question about puberty . host alex trebek asked : ' in common law , the age of this , signaling adulthood , is presumed to be 14 in boys and 12 in girls ? ' the first contestant to press his buzzer was tom , a freemason , who inexplicably answered : ' what is the age of consent?'
Human-written Reference summary: a contestant called tom left a lasting impression with viewers for all the wrong reasons on friday night . host alex trebek asked : ' in common law , the age of this , signaling adulthood , is presumed to be 14 in boys and 12 in girls ? ' . the first contestant to press his buzzer was tom who inexplicably answered : ' what is the age of consent ? ' . the correct answer was puberty and tom 's inappropiate reply left him trending on twitter.

System output: hollywood auction extravaganza xvii took place on saturday , and collectors had a chance to bid on some memorable parts of movie history . with props from 2001 : a space odyssey and star trek to a wide selection of beatles merchandise , there was something for everyone . 2001 a space odyssey hero screen used aries 1b trans-lunar space shuttle .
Human-written Reference summary: a few of hollywood 's most famous props have gone up on the auction block . hollywood auction extravaganza xvii took place on saturday , and collectors had a chance to bid on some memorable parts of movie history . with props from 2001 : a space odyssey and star trek to a wide selection of beatles merchandise , there was something for everyone .

System output: paypal is developing a new generation of edible passwords which stay lodged in your stomach to let you log in . jonathan leblanc , the company 's top developer , said that the devices would be powered by stomach acid and include mini computers . he said that technology had become so advanced that it allowed ' true integration with the human body ' .
Human-written Reference summary: company developing a password that stays lodged in your stomach . jonathan leblanc , the company 's top developer , said that the devices would be powered by stomach acid and include mini computers . added that technology had become so advanced that it allowed ' true integration with the human body '

Appendix B

Sample system extractive summaries from NEWSROOM dataset

Here, we show some examples of our system extractive summaries using our single document extractive summarization model described in Chapter 2 and human-written reference summary from NEWSROOM dataset.

System output: Former Ohio State star Ezekiel Elliott , the top-rated running back in next month's NFL Draft , is embracing speculation linking him to the Giants , who hold the No . 10 overall pick , with the loftiest possible prediction.
Human-written Reference summary: Zeke in New York ? Former Ohio State star Ezekiel Elliott , the top-rated running back in next month's NFL Draft, is embracing speculation linking him to the Giants , who hold the No.10 overall.

System output: Baldwin has been named the new president and chief operating officer of BJ ' s Wholesale Club . Baldwin will begin working for BJ ' s on Sept .
Human-written Reference summary: Christopher J . Baldwin has been named the new president and chief operating officer of BJ ' s Wholesale Club . Baldwin will begin working for BJ ' s on September 8.

System output: DETROIT — BMW is recalling nearly 49,000 motorcycles in the U . S and Canada because flanges that hold the rear wheel can crack if bolts are too tight.
Human-written Reference summary: BMW is recalling motorcycles in the US and Canada because flanges that hold the rear wheel can crack if bolts are too tight.

System output: School principals are pushing for a standardised age for children to start school . It would mean children would have to be at least five years old before entering the classroom .
Human-written Reference summary: School principals want a standardised age for children to start school , meaning they would have to be at least five years old before entering the classroom .

Appendix C

Sample system paraphrases from MSCOCO corpus

Here, we show some examples of our system generated paraphrases using our paraphrasing model described in Chapter 3 and reference paraphrases from MSCOCO dataset.

Original sentence: A person is riding a motorcycle down a country road.
System paraphrase: a person riding a motorcycle on a road street
Reference paraphrase: a person riding a motorcycle on a road with trees

Original sentence: A young man riding a skateboard down a ramp.
System paraphrase: a man riding a skateboard a skate
Reference paraphrase: A man riding a skateboard on a ramp.

Original sentence: A motorcycle is parked outside of a cafe.
System paraphrase: a motorcycle is parked in front
Reference paraphrase: A motorcycle is parked in front of a cafe.

Original sentence: A motorcycle is parked outside of a cafe.
System paraphrase: a motorcycle is parked in front
Reference paraphrase: A motorcycle is parked in front of a cafe.

Original sentence: A group of elephants walking across a river.
System paraphrase: a group of elephants walking in water
Reference paraphrase: many elephants walking in waters in a group

Original sentence: A bathroom sink beneath a very large mirror reflecting a roll of toilet paper.
System paraphrase: a bathroom with a sink and a mirror
Reference paraphrase: A bathroom sink with a large mirror over it.

Original sentence: A yellow and red bus traveling past a tall building.
System paraphrase: a red and yellow bus on a street city
Reference paraphrase: A red and yellow city bus driving down the street

Appendix D

Sample system paraphrases from QUORA corpus

Here, we show some examples of our system generated paraphrases using our paraphrasing model described in Chapter 3 and reference paraphrases from QUORA dataset.

Original sentence: Which is the best laptop to buy above 25000 and below/upto 30000?
System paraphrase: which is the best laptop under 30k
Reference paraphrase: Which is the best laptop under INR 30,000?

Original sentence: Which is the best laptop for gaming under 60k INR?
System paraphrase: which is the best gaming laptop under 60000
Reference paraphrase: What are the best gaming laptops under INR 60000?

Original sentence: What are some places to spend the weekend near Pune?
System paraphrase: what are the best places to visit in Pune
Reference paraphrase: What are the best places to hangout in Pune?

Original sentence: What are some of the best smartphones technology gadgets?
System paraphrase: what are some of the best gadgets
Reference paraphrase: What are some of the best phones tools and gadgets?

Original sentence: Which is the most inspiring movie for you?
System paraphrase: is are the most inspirational movies ever
Reference paraphrase: Which is the most inspirational movie ever?

Original sentence: How can I impress a teenage girl?
System paraphrase: How can I get a girl?
Reference paraphrase: How do I impress a girl on chat?

Original sentence: What is the difference between percentile and percentage?
System paraphrase: What is the difference between percentage and pecentile?
Reference paraphrase: What is the difference between percentage and pecentile?

Appendix E

Sample sentence compressions from GOOGLENEWS corpus

Here, we show some examples of the outputs of our compression models (BI-TRANS, BI-TRANS+F1, BI-TRANS+COSSIM) described in Chapter 4 and reference compressions from GOOGLENEWS dataset.

Original sentence: John Abraham has been prohibited from using the title Hamara Bajaj for his home production
BI-TRANS+COSSIM*:* John Abraham has been prohibited from using the title Hamara
BI-TRANS+F1*:* John Abraham has been prohibited from using his home production
BI-TRANS*:* John Abraham has been prohibited from his home
Reference compression: John Abraham has been prohibited from using the title Hamara Bajaj .

Original sentence: The mobile phone services that remained suspended throughout the day as part of the government's security plan on the occasion of 9th of Muharram has restored in Karachi and other cities, Geo News reported .
BI-TRANS+COSSIM*:* The mobile phone services has restored in Karachi and other cities .
BI-TRANS+F1*:* government's security plan restored in Karachi
BI-TRANS*:* mobile security plan restored in
Reference compression: The mobile phone services has restored in Karachi and other cities.

Original sentence: Thus, four of the nation's biggest institutional landlords – among them Scottsdale-based Colony American Homes – have teamed up to form a nonpartisan trade group that will advocate and educate the public, lawmakers and business leaders on their growing industry, according to a statement today .
BI-TRANS+COSSIM*:* Four of the nation's biggest institutional landlords have teamed up to
BI-TRANS+F1*:* Thus, four of American Homes – have teamed up
BI-TRANS*:* Thus, four of the nation's biggest Homes – have teamed up to
Reference compression: Four of the nation's biggest institutional landlords have teamed up to form a trade group .

Original sentence: The changing global scenario calls for postal administrations around the world to think differently .
BI-TRANS+COSSIM*:* changing global scenario calls for postal administrations around the world to think
BI-TRANS+F1*:* global scenario calls for postal administrations
BI-TRANS*:* for postal administrations
Reference compression: The global scenario calls for postal administrations to think differently.

Original sentence: Boaters can become frustrated when a repair or upgrade takes a long time, but delays are often a simple result of supply and demand .
BI-TRANS+COSSIM*:* but delays are often a simple result of supply
BI-TRANS+F1*:* Boaters can often a simple result
BI-TRANS*:* long time, but delays are often a simple result
Reference compression: A repair takes, but delays are often a simple result of supply and demand.

Original sentence: Controversial TV pitchman Kevin Trudeau, who in July was found in contempt for failing to pay a 37.6 million sanction against him for deceptive marketing, was ordered to jail today and remains in federal custody in Chicago
BI-TRANS+COSSIM*:* Kevin Trudeau was ordered to jail
BI-TRANS+F1*:* Kevin failing to pay a 37 . 6 million
BI-TRANS*:* Controversial TV pitchman Kevin to pay a 37 . 6
Reference compression: TV pitchman Kevin Trudeau was ordered to jail .

Original sentence: US stocks dropped as investors turned attention toward lackluster blue-chip earnings reports after the government passed a temporary deal to avert default
BI-TRANS+COSSIM*:* US stocks dropped as investors turned attention toward earnings reports
BI-TRANS+F1*:* US stocks dropped as investors turned attention toward deal
BI-TRANS*:* US stocks dropped as investors turned attention toward lackluster
Reference compression: US stocks dropped as investors turned attention toward earnings reports .

Original sentence: Internet cafes, which were once the communication hub in developing countries, are fast dying out
BI-TRANS+COSSIM*:* Internet cafes are dying out
BI-TRANS+F1*:* Internet are fast dying out
BI-TRANS*:* Internet cafes, are fast dying out
Reference compression: Internet cafes are dying out.

Appendix F

Sample system single-document abstractive summaries from CNN/DM dataset

Here, we show some examples of our system abstractive summaries using our single document abstractive summarization model described in Chapter 5 and human-written reference summaries from CNN/DM dataset.

Source document: barcelona coach luis enrique is optimistic that holding midfielder sergio busquets can recover from injury in time for sunday 's el clasico match against real madrid at the nou camp . enrique said on saturday that busquets ' is n't 100 percent but he will be there ' for the clash against barca 's rivals in what could be a potential decider for the league title . busquets has been sidelined for the last three games since injuring his right ankle against villarreal in the copa del rey . midfielder sergio busquets could make his return to barcelona 's first-team against real madrid on sunday . barcelona coach luis enrique will be hopeful that busquets can return in time for the crucial el clasico match . javier mascherano has filled in for barcelona in the holding midfield position during busquets ' injury . javier mascherano has played in his place and would do so again against madrid if busquets is n't fit enough . barca go into el clasico off the back of six consecutive victories in all competitions , the last of those confirming premier league side manchester city 's exit from the champions league . madrid , meanwhile , arrested their mini slump of three games without a win after a 2-0 triumph over levante , courtesy of a brace from gareth bale . busquets has been sidelined for barcelona 's last three games but could return in time to face real madrid .
System output: enrique said on saturday that busquets ' isn't 100 percent but he will be there ' for the clash against barca 's rivals in what could be a potential decider for the league title . barca has been sidelined games. sergio madrid return real madrid to liga.
Human-written Reference summary: sergio busquets could return for barcelona against real madrid on sunday . he has been sidelined for barca 's last three games with an ankle injury . the el clasico fixture could potentially decide spain 's league title this year.

Source document: the u.s. navy has sent a nuclear aircraft carrier and a guided-missile cruiser to the waters near yemen to help beef up security and join other american ships that are prepared to block iranian shipments . the uss theodore roosevelt and the uss normandy

left the persian gulf on sunday and are steaming through the arabian sea and heading towards yemen . the vessels are believed to be joining other u.s. ships that are poised to intercept any iranian ships carrying weapons to the houthi rebels fighting in yemen . the navy has been beefing up its presence in the gulf of aden and the southern arabian sea amid reports that about eight iranian ships are heading toward yemen and possibly carrying arms . the uss theodore roosevelt , a nuclear-powered aircraft carrier -lrb- background -rrb- , was dispatched to the gulf of aden to blockade an iranian flotilla carrying arms . the carrier is pictured here with the uss vicksburg cruiser - similar to the uss normandy which was also sent to the gulf of aden . the guided missile cruiser uss normandy is pictured here . the cruiser is escorting the roosevelt to the gulf of aden . navy officials said there are about nine u.s. warships in the region , including cruisers and destroyers carrying teams that can board and search other vessels . the officials spoke on condition of anonymity because they were not authorized to discuss the ship movement on the record . but speaking to reuters on monday , a pentagon spokesman denied the ships were on a mission to intercept iranian arms shipments . one u.s. official said the presence of the u.s. warships off yemen give american decision-makers options for action in the event the situation deteriorates . the other u.s. warships in the region include two destroyers , two mine-sweepers and three amphibious ships carrying 2,200 u.s. marines . the shi'ite muslim houthi are battling government-backed fighters in an effort to take control of the country . the houthi fighters sidelined the central government after seizing the capital sana'a in september and expanding across yemen , which borders oil giant saudi arabia . closing in : the uss theodore roosevelt and the uss normandy left the persian gulf on sunday -lrb- seen on the map -rrb- and are heading through the arabian sea towards yemen , according to reports . the u.s. has been providing logistical and intelligence support to a saudi arabia-led coalition , which has been launching airstrikes against the houthis . that air campaign is now in its fourth week . the u.s. navy generally conducts consensual boardings of ships when needed , including to combat piracy around africa and the region . so far , however , u.s. naval personnel have not boarded any iranian vessels since the yemen conflict began . white house spokesman josh earnest would not comment specifically on any navy movements in yemeni waters , but said the u.s. has concerns about iran 's ' continued support for the houthis ' . ' we have seen evidence that the iranians are supplying weapons and other armed support to the houthis in yemen , ' he said . ' that support will only contribute to greater violence in that country . ' these are exactly the kind of destabilizing activities that we have in mind when we raise concerns about iran 's destabilizing activities in the middle east . ' he said ' the iranians are acutely aware of our concerns for their continued support of the houthis by sending them large shipments of weapons ' .

System output: uss roosevelt left persian gulf for yemen. they will join other vessels. the navy has been beefing up its presence in the gulf of aden and the southern arabian sea amid reports that about eight iranian ships are heading toward yemen and possibly carrying arms.

Human-written Reference summary: uss theodore roosevelt and uss normandy left persian gulf on sunday and are steaming through the arabian sea and heading towards yemen . they will join seven other us vessels that are prepared to block iranian ships potentially carrying weapons for houthi rebels fighting in yemen.'

Appendix G

Sample system multi-document abstractive summaries from DUC 04 dataset

Here, we show some examples of our system abstractive summaries using our multi document abstractive summarization model (EXPARCOM) described in Chapter 6 and human-written reference summaries from DUC 04 dataset.

System output: The only other problem was with a Zarya battery; the astronauts took up a replacement part. The 36-foot, 25,000-pound Unity, the first American-made component, will serve as a connecting passageway, or vestibule, for future modules. It was crucial that Zarya and Unity be joined; if they could not be connected with the robot arm, NASA would have sent out two spacewalking astronauts to manually fit them together. In all, three spacewalks are planned for Endeavour's 12-day flight, not only to hook up electrical connections between the two modules but to install handrails and other tools for future crews. It will provide all of the necessary electricity and steering for the fledgling space station until a permanent control module can be launched next summer.
Human-written Reference summary: After discarding a suggested change of orbit, the Russian Space Agency went ahead with plans to launch its Zarya module of the international space station on Nov. 20, 1998. Although delayed for a day, U.S. plans to launch the space shuttle Endeavour carrying the U.S. module Unity and six astronauts were carried out on Dec. 4. The astronauts' job was to connect Unity with the already-orbiting Zarya as the first step in assembling 100 major components of the planned space station. Using the shuttle's 50-foot robot arm, the two modules were joined setting the stage for a spacewalk by two astronauts the next day to attach electrical connectors and cables.

System output: For its part, Syria has accused Turkey of forming military alliances with Israel that threaten Arab security and undermine Syria's bargaining position in peace talks with the Jewish state. Syria also has accused Turkey of threatening its supply of water by building dams on the Euphrates River. Iran has offered to mediate between Syria and Turkey in the deepening dispute over Kurdish rebel bases and will dispatch envoys to the two countries, the Tehran Times reported Monday. Greece on Monday warned that mounting tension between Turkey and Syria could lead to "tragic results" if not dealt with in its early stages. "Sources of tension are being created in our region," government

spokesman Dimitris Reppas said.
Human-written Reference summary: In early October 1998 Turkey moved 10,000 troops to the Syrian border accusing its neighbor of harboring Kurdish rebels and their leader Abdullah Ocalan. Syria denied the charges and blamed Turkey's belligerence on its military alliance with Israel. As the dispute threatened to ignite the whole volatile region, Egypt's President Mubarak launched a mediation effort soon joined by Iran and Jordan. Saudi Arabia, Yemen, Sudan, Lebanon and Greece voiced support for Syria, but all called for a diplomatic solution. Israel did not take sides urging diplomatic talks and insisting that Israeli-Turkish military cooperation played no role in the crisis.

System output: Cardoso, under pressure to repair an economy battered by the world financial turmoil, is expected to unveil the full scope of his deficit-cutting plan next week. It is believed to include a spate of new taxes on fuel, income, personal fortunes and bank transactions. The plan is part of a deal with the International Monetary Fund for a rescue package estimated at dlrs 30 billion. Brazil already has agreed to annual targets to sharply reduce its deficit through the year 2001. "And that means he will try to stay as far away as possible from any austerity measure."
Human-written Reference summary: Latin leaders at Ibero-American summit explore ways to avoid economic turmoil and warn of likely global recession. Brazil President Cardoso will announce deficit-cutting austerity measures. Brazil and the IMF move closer to an agreement on a 30 billion rescue package. Cardoso readies his plan for spending cuts and tax increases as part of the IMF deal. He will unveil the full plan next week. The success of his economic efforts may depend on the outcome of upcoming gubernatorial elections. The Commerce Dept. measures the effects of global economic decline on the U.S. economy. The U.S. will give billions of taxpayer dollars to the Brazil-IMF rescue deal.

System output: The attack took place Tuesday near Cailaco in East Timor, a former Portuguese colony, according to a statement issued by the pro-independence Christian Democratic Union of East Timor. Placido dos Santos, a 28-year-old farmer, was tortured and killed by the Indonesian military during the attack, the statement, which cited resistance sources in East Timor's capital, Dili. The statement, released in the Portuguese capital of Lisbon, also said that 22 people were injured and 26 were missing. Formal diplomatic relations, however, will not be resumed. Turmoil has plagued East Timor ever since Indonesian troops invaded in 1975, unleashing a separatist rebel war and the resentment of a population pummeled by human rights abuses.
Human-written Reference summary: Indonesia invaded the former Portuguese territory of East Timor in 1975 and annexed it in 1976. By late 1998 while East Timorese called for independence and accused Indonesian troops of yet another massacre of civilians, Portugal cut off talks with Indonesia. Internationally, Taiwan was timid fearing antagonizing Indonesia, Australians protested against training Indonesian military who might be assigned to East Timor, and fifteen European Union leaders endorsed Portugal's call for a referendum on East Timor's future. A U.N. enjoy saw a peaceful solution as distant, but sensed a "newfound taste for compromise."

Appendix H

Sample system multi-document abstractive summaries from MULTINEWS dataset

Here, we show some examples of our system abstractive summaries using our multi document abstractive summarization model (EXPARCOM) described in Chapter 6 and human-written reference summaries from MULTINEWS dataset.

System output: russell , kan. (ap) a 59-year-old kansas man was killed when the motorcycle he was driving friday night collided with a black cow on a blacktopped road . kansas highway patrol trooper brant birney said there were no witnesses when james zordel hit the cow on a paved rural road about six miles south of interstate 70 near russell . zordel was driving in the roadway when the accident happened and it is not clear if he was speeding or if the cow suddenly appeared from the side of the road . " it was dark . he was driving down a blacktop road and he hit a black cow , " birney said , adding that exactly what caused the accident may never be known .
Human-written Reference summary: a 59-year-old kansas man was killed when the motorcycle he was driving friday night collided with a black cow on a blacktopped road , reports kake . kansas highway patrol trooper brant birney said there were no witnesses when james zordel hit the cow on a paved rural road about six miles south of interstate 70 near russell . it ' s not clear if zordel was speeding or if the cow suddenly appeared from the side of the road , notes the ap . " it was dark . he was driving down a blacktop road and he hit a black cow , " birney said , adding that exactly what caused the accident may never be known . zordel , who was not wearing a helmet , died at the scene .

System output: defuniak springs , fla. (ap) while his mother was preparing food in the kitchen , a 5-year-old florida boy called 911 to invite law enforcement officers over for thanksgiving dinner . monica webster of the walton county sheriff ' s office tells the news herald that with all the bad calls they receive every day , this was a happy call . but young billy nolin ' s family had no idea he ' d invited guests to dinner . mom landi mccormick says she was cooking when billy ' s grandfather noticed him talking to someone on an old cellphone . mccormick reprimanded billy when he admitted calling 911. he was crying when deputy dannon byrd drove up the deputy walked up to mccormick and asked if billy was the young man who had called 911. rather than scold the little boy , byrd knelt down

and thanked billy for his kind invitation .
Human-written Reference summary: while his mother was preparing food in the kitchen , a 5-year-old florida boy called 911 to invite law enforcement officers over for thanksgiving dinner , the ap reports . monica webster of the walton county sheriff ' s office tells the news herald that with all the bad calls they receive every day , this was a happy call . but young billy nolin ' s family had no idea he ' d invited guests to dinner . mom landi mccormick says she was cooking when billy ' s grandfather noticed him talking to someone on an old cellphone . mccormick reprimanded billy when he admitted calling 911. he was crying when deputy dannon byrd drove up . she says the deputy thanked billy for his kind invitation , then reminded him he should only use 911 for emergencies . the deputies gave billy a sheriff ' s badge .

System output: the military operation to wrest mosul from the islamic state group could potentially become the single largest , most complex humanitarian operation in the world in 2016 , a u.n. official said monday . speaking via video-link from iraq , lise grande , the u.n. humanitarian coordinator for iraq , said that in the worst case scenario , some 1 million civilians could flee the city with 700,000 of them requiring shelter overwhelming emergency sites that currently only have the capacity to hold 60,000 people . " our capacity to support 700,000 people in the short-term we couldn ' t do it . and certainly if we had to mount a response over the intermediate-term , if they couldn ' t go back to mosul quickly , if there was too much damage in the city , then it would test us to the breaking point , " grande said . she said that the u.n. was especially concerned about the safety of the estimated 1.2 to 1.5 million civilians inside mosul who may get caught in the fighting . everyone is staying at home because we dont know what else to do . daesh [another name for islamic state] are mostly moving around on motorbike and have small and heavy guns . the planes started bombing mosul around 1am today and they are in the sky constantly and occasionally striking targets ,abu mohammed , a 35-year-old from the east side of the city told the guardian .
Human-written Reference summary: the military operation to wrest mosul from isis could become the single largest , most complex humanitarian operation in the world in 2016 , a un official said monday . lise grande , the un humanitarian coordinator for iraq , said that in the worst case scenario , some 1 million civilians could flee the city , with 700,000 of them requiring shelter overwhelming emergency sites that currently only have the capacity to hold 60,000 people , the ap reports . it can ' t be done right now , and even in " the intermediate-term , if they couldn ' t go back to mosul quickly , if there was too much damage in the city , then it would test us to the breaking point , " grande said . " in the worst-case scenario , we can ' t rule out the possibility that there may be a chemical weapons attack , " grande said , warning that isis " may try and hold civilian populations either as human shields or forcibly expel huge numbers of civilians in the face of an attack by the iraqi security forces knowing the iraqi forces will not fire on their own people . " the guardian reports that american aircraft have started dropping millions of leaflets over mosul , warning residents to stay in their homes , advising them on how to comfort children and avoid flying glass during bombings and advising against trying to flee the city.