MODEL COPYING AND REWRITING IN NEURAL ABSTRACTIVE SUMMARIZATION

CAO ZIQIANG
PhD

The Hong Kong Polytechnic University
2019

The Hong Kong Polytechnic University
Department of Computing

Model Copying and Rewriting in Neural Abstractive Summarization

CAO Ziqiang

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

July, 2018

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

Hong Kong, July, 2018

CAO Ziqiang

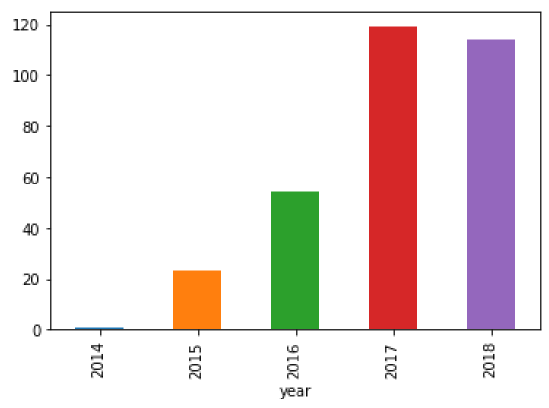
Abstract

Copying and Rewriting are two core writing behaviors in human summarization. Traditional automatic summarization approaches basically follow these two styles. For example, extractive summarization copies source sentences, compressive summarization copies source words, and template-based summarization utilizes handcrafted rules to rewrite from pre-defined templates. Since 2016, sequence-to-sequence (seq2seq) neural networks have attracted increasing attention from abstractive summarization researchers. Compared with traditional summarization approaches, seq2seq models generate summaries end-to-end and require less human efforts. However, most existing seq2seq approaches focus more on how to learn to generate the summary text, but overlook the previously mentioned two essential summarization skills, i.e., copying and rewriting. These approaches suffer from two major problems. First, summarization has to start almost from scratch, discarding the **prior knowledge** accumulated during the past half a century research. The data scale thus becomes the most significant bottleneck for performance improvement. Second, the neural network architecture lacks explanation and is hard to evaluate. To address these problems, we explore to explicitly model copying and rewriting in seq2seq summarization by utilizing the prior knowledge learned from traditional summarization approaches.

Our research consists of three parts. In the work to be presented in Chapter 3, we leverage the popular attention mechanism to copy and rewrite **words** in the source text. Our model fuses a copying decoder and a rewriting decoder. The copying decoder finds out words to be copied in the source text based on learned attentions. The rewriting decoder produces other necessary summary words limited in the source-specific vocabulary, which is also derived from the attention mechanism. Extensive experiments show that our model is able to generate informative summaries efficiently. In

Chapter 4, we investigate an important but neglected problem, i.e., the faithfulness problem in abstractive summarization. Abstractive summarization has to fuse different parts of the source text, which inclines to create fake facts. We call this issue summary faithfulness. Our preliminary study reveals nearly one third of the outputs from a state-of-the-art neural abstractive summarization system suffer from fake generation. To copy **facts** in the source text, we leverage open information extraction and dependency parsing techniques to extract true facts from the source text. Note that these techniques are also widely-used in compressive summarization. We propose a dual-attention seq2seq summarization model to force the summary generation conditioned on both the source text and the extracted facts. Experiments demonstrate that our model greatly reduces fake summaries by 55%, and at the same time achieves significant improvement on informativeness. Inspired by template-based summarization, we propose to use existing summaries as **soft templates** to guide the seq2seq model, which will be elaborated in Chapter 5. To this end, we use a popular information retrieval tool to retrieve appropriate existing summaries as candidate templates. We extend the seq2seq model by jointly learning template reranking and template-aware summary generation. Essentially, the model learns to rewrite the selected template (i.e., the summary pattern) according to the source text. Experiments show that our model significantly outperforms the state-of-the-art methods in terms of informativeness, and even soft templates themselves demonstrate high competitiveness. More importantly, the import of high-quality "external" summaries improves the stability and readability of output summaries and provides potential in generation diversity. As one of the few large-scale studies of copying and rewriting in seq2seq models, our work is expected to advance a more in-depth research in core writing behavior driven neural abstractive summarization.

Publications Arising from the Thesis



Google Scholar citations. Total: 312, h-index: 10.

1. **Cao Ziqiang**, Li Wenjie, Li Sujian and Wei Furu. 2018. Retrieve, Rerank and Rewrite: Soft Template Based Neural Summarization. *In Proceedings of ACL.*

2. **Cao Ziqiang**, Wei Furu, Li Wenjie and Li Sujian. 2018. Faithful to the Original: Fact Aware Neural Abstractive Summarization. *In Proceedings of AAAI.*

3. **Cao Ziqiang**, Luo ChuWei, Li Wenjie and Li Sujian. 2017. Joint Copying and Restricted Generation for Paraphrase. *In Proceedings of AAAI.*

4. **Cao Ziqiang**, Li Wenjie, Li Sujian and Wei Furu. 2017. Improving Multi-Document Summarization via Text Classification. *In Proceedings of AAAI.*

5. Li Yanran, Su Hui, Shen Xiaoyu, Li Wenjie, **Cao Ziqiang** and Niu Shuzi. 2017. DailyDialog: A Manually Labelled Multi-turn Dialogue Dataset. *In Proceedings of IJCNLP.*

6. **Cao Ziqiang**, Li Wenjie and Wu Dapeng. 2016. Polyu at cl-scisumm 2016. *In Proceedings of the Joint Workshop on BIRNDL.*

7. **Cao Ziqiang**, Li Wenjie, Li Sujian, Wei, Furu and Li Yanran. 2016. AttSum: Joint Learning of Focusing and Summarization with Neural Attention. *In Proceedings of COLING.*

8. **Cao Ziqiang**, Chen Chengyao, Li Wenjie, Li Sujian, Wei Furu and Zhou Ming. 2016. TGSum: Build Tweet Guided Multi-Document Summarization Dataset. *In Proceedings of AAAI.*

9. **Cao Ziqiang**, Li Sujian, Liu Yang, Li Wenjie and Ji Heng. 2015. A Novel Neural Topic Model and Its Supervised Extension. *In Proceedings of AAAI.*

10. **Cao Ziqiang**, Wei Furu, Li Sujian, Li Wenjie, Zhou Ming and Wang Houfeng. 2015. Learning summary prior representation for extractive summarization. *In Proceedings of ACL.*

Patent

1. Wei Furu, Zhou Ming, Liu Yang, **Cao Ziqiang**, Huang Shaohan, Dong Li, Cui Lei. 2018. SYSTEMS AND METHODS FOR PROVIDING A COMMENT-CENTERED NEWS READER. *US Patent App. 15/578,203.*

2. Wei Furu, Zhou Ming, Liu Yang, **Cao Ziqiang**, Huang Shaohan, Dong Li, Cui Lei. 2018. Comment-centered news reader. *US Patent App. 15/578,195.*

Under Review

1. Li Yanran, Zhang Ruixiang, Li Wenjie and **Cao Ziqiang**. 2018. Hierarchical Prediction and Adversarial Learning For Conditional Response Generation. *Under Review of TKDE.*

2. **Cao Ziqiang**, Li Wenjie, Wei Furu and Li Sujian. 2018. Towards Summary Faithfulness in Abstractive Summarization. *Under Review of CL.*

Acknowledgements

I would like to express my sincere thanks to all those who gave me help during the completion of this thesis. Firstly, my deepest gratitude goes to Dr. LI Wenjie, my supervisor, for her constant encouragement and guidance. In July 2015, I came to Hong Kong with limited systematic understanding of Natural Language Processing. Dr. LI had a cordial talk with me and indicated a promising research focus, i.e., neural abstractive summarization, for me. Since then, we have kept discussing each week. Her suggestions and writing guidance largely promoted my publications. To promote my research, she even gave the first GPU of our group to me. Without her patient instruction, insightful criticism and kind encouragement, I would never have the chance of writing down these words.

In addition, I must thank our department for their support. In 2016, our department assigned me a chance to apply Microsoft Research Asia Fellowship. Theretofore, I did not believe that my research accomplishment deserved such a famous award. Without the supervision and urgency from the department, I would be likely to miss this fellowship which played a significant role in job application.

I owe a special debt of gratitude to my wife for all her support and sacrifice during the full three years. She shouldered the family burdens these years. Meanwhile, she kept comforting and encouraging me, which were the most important relief in my frustrated and lonely periods. Heartfelt appreciation to my wife. It made me far more delightful to share my achievement with her.

The colleagues and visitors of our group have offered constant support and cooperation throughout the years. I would like to thank all of you: Ms. CHEN Chengyao, Mr. LEI Yu, Mr. WANG Zhitao, Ms. LI Yanran, Dr. CHEN Qiang, Mr. LUO Chuwei, Mr. TAN Haihui, Ms. LV Ziyu, Ms. ZHANG Ke, and Mr. WU Zongheng. I would especially acknowledge my co-authors: Ms. LI Yanran, Mrs. CHEN Chengyao and Mr. LUO Chuwei.

Last but not the least, two mentors deserve my sincere gratitude. Dr. LI Sujian, my supervisor during the master degree, has been supporting and encouraging my work for more than 7 years. Dr. WEI Furu, mentored me during my internship in MSRA, inspired me to explore a plenty of insightful ideas.

Table of Contents

Chapter 1

Introduction

More matter with less art.

— William Shakespeare
(Poet, playwright and actor)

1.1 A Brief Tour of Text Summarization

Automatic text summarizationis a typical application of machine learning and text mining[1]. The fundamental idea of summarization is to find out a subset of text, which can condense the "information" of the overall document. The study of text summarization spans over a half century. Since [Luhn, 1958] firstly brought us the concept of automatic text summarization, the age of information explosion has largely necessitated the development of effective text summarization systems. Currently, text summarization has become a research focus of Natural Language Processing (NLP) area and been widely applied in the industry today, for example, used by search engines.

In this section, I review the important aspects to conduct summarization research, including its taxonomies, available datasets, evaluation and existing systems.

[1]`https://en.wikipedia.org/wiki/Automatic_summarization`

1.1.1 Taxonomies

Over decades of research, numerous terms and theories have been developed to refine this task. As a result, text summarization can be classified based on different criteria. According to the constituents of summaries, two automatic summarization approaches are **extraction** and **abstraction**. Extractive summarization approaches pick up a subset of existing sentences in the source document to form a summary. It is easy to implement, able to preserve the original information and maintain the readability at the sentence-level. In contrast, abstractive summarization approaches adopt Natural Language Generation (NLG) techniques to generate a summary, which is expected to be similar to a human-written summary. Such a summary usually requires content fusion and paraphrasing. Traditional abstractive summarization approaches include **compressive** and **template-based** approaches (see Section 2 for more details). According to the learning approaches, summarization models can be either **supervised** or **unsupervised**. A large amount of human-written summaries are necessary for a supervised approach to learn model parameters. Various machine learning techniques have be applied in the past, including Support Vector Machine (SVM) [Ouyang *et al.*, 2011], Logistic Regression (LR) [Li *et al.*, 2013b] and Neural Network (NN) [Cao *et al.*, 2015a]. Unsupervised approaches, in contrast, do not need any labeled data. They usually apply heuristic rules (e.g., tfidf and sentence position) to extract salient sentences to form a summary. They can produce summaries by accessing merely the input documents. Thus, unsupervised approaches can be quickly adapted to a new domain. One of the most popular unsupervised summarization models is the graph-based models like LexRank [Erkan and Radev, 2004].

According to the number of documents, summarization can be classified into **single-document** and **multi-document** summarization. Compared with single-document summarization, it is much more hard to summarizing multiple documents due to the serious redundancy problem. Recently, **Sentence** summarization [Rush *et al.*, 2015a], which aims to shorten an input sentence while preserving the main meaning, becomes popular.

It can be used to design or refine appealing headlines. Thus, it is often called headline generation. Sentence-level summarization is different from document-level summarization since it is hard to apply the common extractive techniques [Over and Yen, 2004]. That is, selecting existing sentences to form the sentence summary is impossible. In nature, sentence summarization is abstractive.

There are also a series of summarization tasks. **Generic** summarization does not care about a particular query while **query-focused** summaries are expected to response to a query. In **update** summarization, readers are supposed to already know some information about a topic and they need to be updated with the latest news regarding the topic. The above-mentioned summarization systems all provide the same summary to different users. [Paice and Jones, 1993] finds that users usually pick up the pieces of documents, which are more closely associated with their interests. **Personalized** summarization aims to identify the user's interest and extract the important information that the user specifically desires to generate a summary. In other words, different users will receive different summaries from the same document(s).

Language is also important criterion, which divide summarization into **multi**-lingual, **mono**-lingual and **cross**-lingual summarization. If documents and summaries are written in the same language, it is defined as a mono-lingual summarization scenario. In comparison, if summaries are expected to be written in a language rather than the source language, it is cross-lingual summarization. When source documents are mixed in multiple languages, e.g., English, Chinese and Japanese, and summaries are written in the same languages, it is known as multi-lingual summarization.

Referring to [Gambhir and Gupta, 2017], Table 1.1 presents the different kinds of summarization along with the determinant factors.

	Category	Factor
Approach	Extractive and Abstractive	Sentence Selection or Sentence Generation
	Supervised and Unsupervised	Training Data Available or Not
Source	Single-document, Multi-document and Sentence	One Document, Multiple Documents or One Sentence
Task	Generic	Document(s) Alone
	Query-focused	An Additional Given Query
	Update	Current Updates Regarding previous Summaries
	Personalized	Target to Particular Users
Language	Mono, Multi and Cross-lingual	Languages in Source Document(s) and Target Summaries

Table 1.1: Taxonomies of summarization.

1.1.2 Datasets

The most popular benchmark datasets for multi-document summarization are published by Document Understanding Conferences (DUC[2]) and Text Analytics Conferences (TAC[3]). The documents in DUC/TAC are news from a wide range of categories, such as Natural Disaster, Politics and Biography. A set of documents coming from the same topic are grouped together. Each document set contains a number of reference summaries written by human experts. DUC 01, 02 and 04 focus on multi-document generic summarization, whereas DUC 03, 05, 06 and 07 are query-focused. The subsequent two datasets TAC 08 and 09 are devoted to update summarization. Note that DUC 03 and 04 provide additional datasets on sentence summarization.

Although originally built for machine reading comprehension, DuReader [He *et al.*, 2017] can be regarded as a typical Chinese multi-document query-focused summarization dataset. It collects user logs of Baidu Search and Baidu Zhidao as queries, and the corresponding search results as the documents. The summaries (called answers in DuReader) are generated manually. In addition, DuReader provides rich annotations of query types, in particular including yes-no and opinion questions. The scale of this

[2]`https://duc.nist.gov/`
[3]`https://tac.nist.gov/`

corpus is quite large, containing 200K queries, 420K summaries and 1M documents. A similar English dataset is MS-MARCO [Nguyen *et al.*, 2016], which is created through Bing Search.

For single-document summarization, the New York Times Annotated corpus [Sandhaus, 2008] is a good resource. It contains more than 1.8 million news documents published by this newswire. The documents span from January 1987 to June 2007, and are accompanied by metadata such as dates, authors and taxonomies. Among them, over 650,000 articles contain summaries written by library scientists. Another large scale dataset is CNN/Daily Mail crawled by [Hermann *et al.*, 2015]. The documents are news from CNN[4] and Dail Mail[5]. The highlights provided by the newswires are regarded as summaries. In all, the dataset contains 287k training pairs, 13k validation pairs and 11k test pairs.

There are two small sentence summarization datasets provided in DUC 03 and 04. Later, [Rush *et al.*, 2015a] pair the headline and the leading sentence of a news document as the labeled data for sentence summarization. From English Gigaword, they pick 3.8M training instances. Afterwards, [Hu *et al.*, 2015a] build a large-scale Chinese dataset for sentence summarization, called LCSTS. It is collected from Sina Weibo by regarding a Weibo news post as the source text and its title as the summary. LCSTS contains over 2 million actual Chinese (short text, summary) pairs. On this dataset, the authors manually annotate the qualities of 11k pairs.

The basic information of each summarization dataset is presented in Table 1.2.

1.1.3 Evaluation Criteria and Automatic Evaluation Metrics

DUC/TAC has developed sophisticated evaluation criteria for the summarization challenges. Two important quality criteria are as follows:

[4]`https://www.cnn.com/`
[5]`http://www.dailymail.com`

Dataset	Source	Task
DUC 01, 02, 04	News	Multi-document Generic Summarization
DUC 03, 05∼07	News	Multi-document Query-focused Summarization
DuReader	Web doc/CQA	
MS-MARCO	Web doc	
TAC 08, 09	News	Multi-document Update Summarization
New York Times	News	Single-document Generic Summarization
CNN/Daily Mail	News	
DUC 03, 04	News	Sentence Summarization
LCSTS	Weibo	
Gigaword	News	

Table 1.2: Popular datasets for summarization.

Informativeness A summary must reflect the main meaning of the source document. It is the essential requirement of summarization. Informativeness is also called saliency. Most current automatic summarization evaluation tools focus on the measurement of informativeness.

Readability It is the ease with which a reader can understand a written text. A set of keywords is obviously not an ideal summary. Although perplexity somewhats reflects readability, in most cases, it is still inspected manually.

In addition to the above two, we propose the third one, summary faithfulness, which is equally important if not more important, but largely neglected in previous work. Our preliminary study reveals nearly one third of the outputs from a state-of-the-art neural abstractive summarization model create fake facts. We argue that a summary must accord with the semantic meaning of the source document. Summary faithfulness should be the precondition of a practical summarization system.

Observing the datasets provided by DUC/TAC as well as the conclusion from [Rath *et al.*, 1961], it is notated that human summaries can vary largely. Therefore, a summarization system, which is able to provide **diverse** candidate summaries, should be more welcome.

Summarization evaluation is a necessary step in summarization research. Manual evaluation is time-consuming and subjective. Therefore, researchers concentrate on the development of automatic evaluation tool. TAC has even established competitions, i.e., Automatically Evaluating Summaries of Peers (AESOP), to advance this study.

Automatic summarization evaluation can be classified into two types, **extrinsic** evaluation and **intrinsic** evaluation. Extrinsic approaches evaluate a summarization system by leveraging external tasks and comparing the performance based on the summaries against the original documents. A typical example is [Morris *et al.*, 1992]. They use generated summaries and original documents respectively as the source text to solve reading comprehension questions. Other tasks include topic judgment [Tombros and Sanderson, 1998], text classification and question answering [Mani *et al.*, 1999].

Intrinsic approaches evaluate a summarization system through cross-summary comparisons. Usually, the system-generated summary is compared with human-written summaries. Considering human-produced summaries often vary largely [Rath *et al.*, 1961], a number of human-written summaries can be used to enhance objectivity in evaluating a single system summary. This is a common practice in DUC/TAC. Inspired by the machine translation evaluation tool BLEU [Papineni *et al.*, 2002], [Lin, 2004] develop the content-oriented automatic evaluation metrics ROUGE. ROUGE measures the quality of a candidate summary by computing the overlapping lexical units between it and the golden summaries. The extensively used ROUGE variants include ROUGE-1 (uni-gram), ROUGE-2 (bi-gram), ROUGE-L (longest common subsequence) and ROUGE-SU4 (skip-bi-gram plus uni-gram). ROUGE scores have become the official ranking basis for DUC/TAC competitive tasks since 2004. [Passonneau *et al.*, 2005] propose a more accurate evaluation method, called Pyramid. It measures the matching of Summarization Content Units (SCUs). However, the acquirement of SCUs requires extensive human annotation efforts.

Currently, these automatic evaluation tools mainly evaluate the **informativeness** performance. Since we find the prevailing neural abstractive summarization tends to create fake facts, we are the first to explore the automatic evaluation of summary faithfulness.

1.1.4 Text Summarization Systems

Numerous summarization systems have been developed throughout the decades of research. In this section, I briefly introduce some important publicly available systems. A well-known open domain platform for multi-document summarization is MEAD [Radev *et al.*, 2004a]. It implements a number of summarization algorithms (e.g., Centroid [Radev *et al.*, 2004b] and LexRank [Erkan and Radev, 2004]) and provides popular classifiers for supervised content selection. Recently, the prevailing topic modeling Python library gensim[6] has embedded a series of summarization functions. Based on a variation of TexRank [Barrios *et al.*, 2016], the summarization package of gensim is able to conduct generic summarization and keyword extraction at the same time. OpenNMT[7] is an open source sequence-to-sequence [Bahdanau *et al.*, 2014] platform. Since its launch in December 2016, OpenNMT has become a collection of implementations targeting both academia and industry. Although initially designed for neural machine translation, there are pre-trained OpenNMT models for sentence summarization and single-document generic summarization.

There are also some real-life summarization systems. For example, Columbia Newsblaster[8] is a system that assists users to find out interesting news. The system automatically crawls news from several websites (CNN, Reuters[9], etc.) every day. This system process the news through clustering, classification and summarization to provide users a flexible interface to browse the results. Newsblaster is online in 2001 [McKeown *et al.*, 2001] and also has a multilingual version [Evans *et al.*, 2004]. iResearch Reporter[10], a text-mining solution, allows you to quickly and easily analyze online sources or local documents and receive informative report on virtually any subject. Ul-

[6]`https://radimrehurek.com/gensim/`
[7]`http://opennmt.net/`
[8]`http://newsblaster.cs.columbia.edu/`
[9]`https://www.reuters.com/`
[10]`http://iresearch-reporter.com`

timate Research Assistant[11] is an advanced internet knowledge mining technology. It summarizes and visualizes the results of search engines to make them more understandable.

1.2 Research Motivation

Copying and Rewriting are two core writing behaviors in human summarization. Traditional automatic summarization approaches basically follow these two styles. As shown in Figure 1.1, extractive summarization [Ouyang *et al.*, 2011] copies source sentences, compressive summarization [Li *et al.*, 2013a] copies source words, while template-based summarization [Zhou and Hovy, 2004] utilizes handcrafted rules to rewrite from pre-defined templates. Since 2016, the seq2seq model [Bahdanau *et al.*, 2014] has attracted increasing attention from abstractive summarization researchers [Rush *et al.*, 2015a; Chopra *et al.*, 2016; Nallapati *et al.*, 2016a]. With such a neural network structure, the source text is encoded by the encoder as a context vector. Then, a decoder decodes the semantic information in the context vector and outputs the target text. Compared with the traditional summarization approaches, a seq2seq model generates summaries end-to-end and requires less human efforts.

However, since the seq2seq model is originally developed for machine translation [Kalchbrenner and Blunsom, 2013; Cho *et al.*, 2014a; Sutskever *et al.*, 2014], it is hard to exactly fit the need of summarization. Most existing seq2seq approaches focus more on how to learn to generate the summary text, but overlook the previously mentioned two essential summarization skills, copying and rewriting. These approaches suffer from two weaknesses. First, summarization has to start almost from scratch, discarding the **prior knowledge** during the past half a century research, which makes the data scale usually become the most significant bottleneck for performance improvement. There are plentiful models and theories in traditional summarization approaches. For

[11] `http://www.ultimate-research-assistant.com`

example, extractive summarization proposes the idea of sentence ranking to estimate the importance of a sentence; compressive summarization designs a series of models to find the stem of a sentence; and template-based summarization develops a pipeline from template construction to rule-based summary generation. In fact, a well-designed traditional summarization system can also achieve competitive performance in many scenarios. Therefore, we argue that the import of prior knowledge from the previous summarization research could greatly advance abstractive summarization research. Second, the neural network architecture lacks explanation and is hard to evaluate. For example, the attention mechanism [Bahdanau *et al.*, 2014], i.e., a crucial mechanism in the seq2seq model, is expected to learn the word alignment between source language and target language for machine translation. Thus, it can be evaluated explicitly or utilize the predefined alignment table. However, for summarization, there is no specific alignments between summary words and document words, which heavily weakens the effect of the attention mechanism. We show the attention learning results of the typical work [Rush *et al.*, 2015a] in Figure 1.2. As can be seen, “calls” is linked to “called” and “against” is associated with “combating”. These learning results seem reasonable. Nevertheless, the word that matches “Russia” best is “creation”. What does it mean? Due to the ambiguous interpretation, it is indeed hard to evaluate the effect of attention mechanism in summarization, and thus there is no chance to guide the learning of attention mechanism. Therefore, it is crucial to endow neural network architecture of seq2seq with explicit meaning. We believe that modeling core summarization behaviors should be an ideal choice.

1.3 Research Overview

In this dissertation, I am devoted to modeling the core writing behaviors of summarization, i.e., copying and rewriting. Our work is unique in two aspects. On the one hand, we introduce the prior knowledge of traditional summarization research into the seq2seq model to instruct summary generation. On the other hand, we design the novel

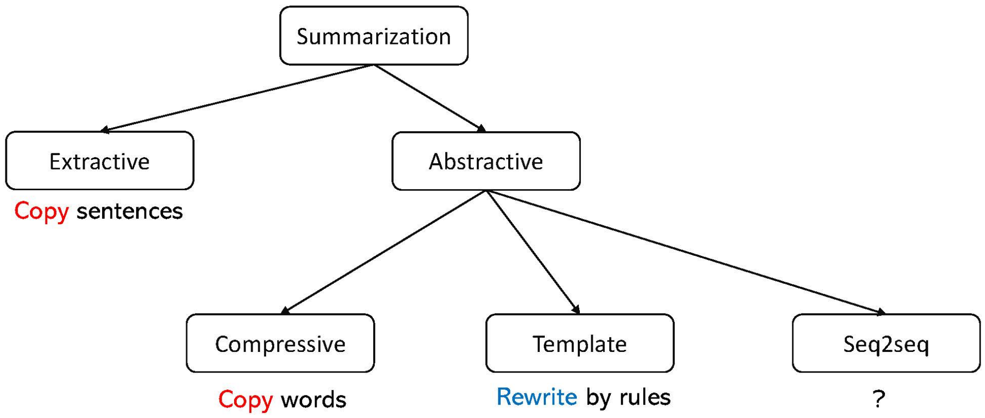


Figure 1.1: Copying and Rewriting in different summarization approaches.

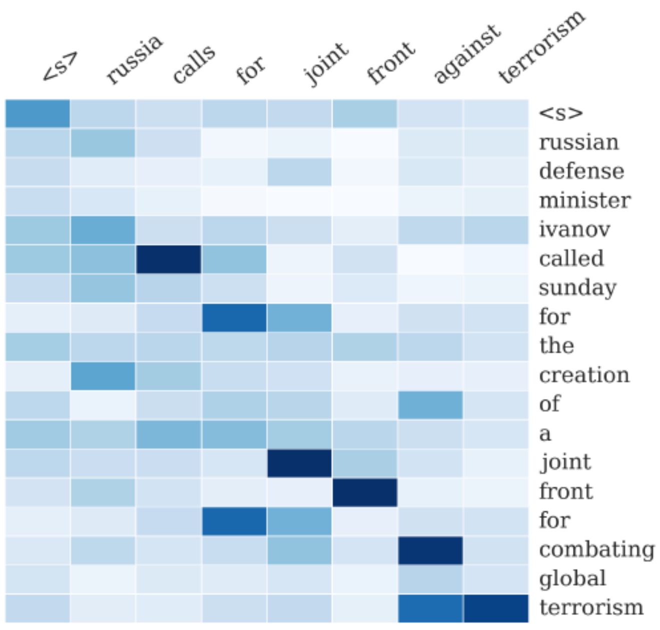


Figure 1.2: Learning result of attention mechanism in summarization.

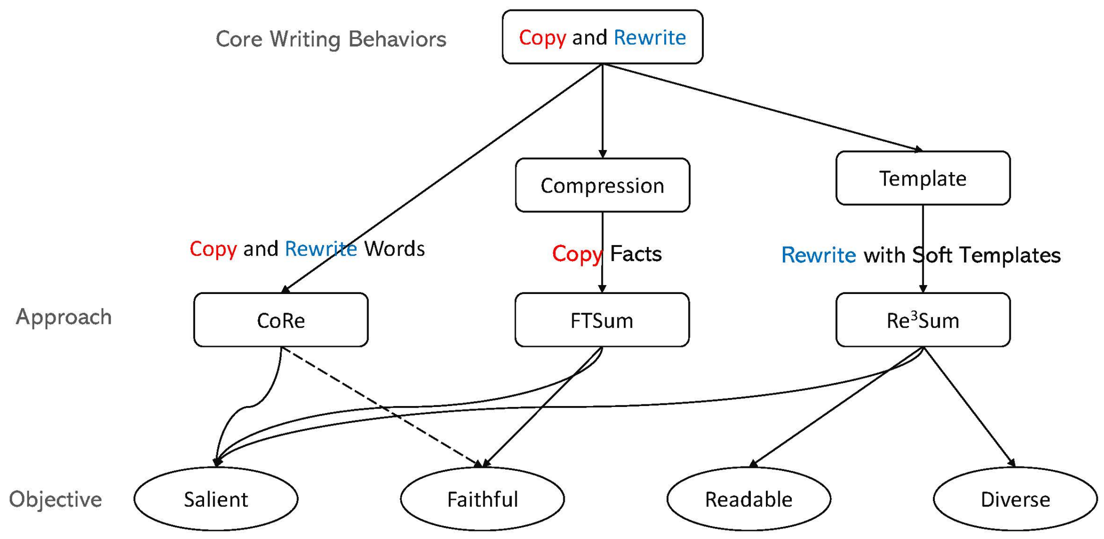


Figure 1.3: Outline of the technical chapters. We use the dash line to link CoRe to Faithful since the rewriting decoder suppresses the generation of irrelevant words, which partially enhances faithfulness.

neural network architecture to explicitly capture the copying and rewriting modes in the source text. Figure 1.3 depicts our research focuses. The summaries of technical chapters are as follows.

In Chapter 3, we leverage the attention mechanism to copy and rewrite words in source text. Our system, called CoRe, fuses a copying decoder and a rewriting decoder. The copying decoder locates the words to be copied from the source text according to the attention weights. The rewriting decoder produces other necessary words from the source-specific vocabulary derived from learned alignments. To combine the two decoders and determine the final output, we train a predictor to predict the actual writing modes. This predictor can be learned from the actual writing modes in the training dataset. Extensive experiments show that our model is capable of generating informative summaries efficiently. In addition, compared with the standard seq2seq model, CoRe better balances the proportions of the summary words generated via copying and rewriting.

In Chapter 4, we investigate the an important but neglected problem, i.e., the faithfulness problem in abstractive summarization. As we know, abstractive summarization inevitably needs to fuse different parts of the source text. Consequently, the fused summaries often misrepresent the original meaning and yield fake facts. Thus, we propose a fact-aware summarization system, called FTSum, to copy source facts. We employ OpenIE and dependency parsing tools to extract the facts in the source sentence. We propose the dual-attention seq2seq model to force summary generation conditioned on both source sentence and extracted facts. Manual and automatic evaluations demonstrate that our summarization system greatly outperforms state-of-the-art models on both informativeness and faithfulness. We also develop an automatic faithfulness evaluation tool, called FTEval, which reveals high correspondence with the manual judgment of faithfulness.

In Chapter 5, inspired by template-based summarization, we propose a system, called Re^{3}Sum, that uses soft templates as rewriting references to guide seq2seq summarization. We use the mainstream IR tool Lucene to retrieve proper existing summaries as candidate soft templates. Then, we extend the seq2seq model to jointly learn template saliency (i.e., Rerank) and final summary generation (i.e., Rewrite). Rerank measures the informativeness of a candidate template, similar to sentence ranking in extractive summarization. In Rewrite, the summary is generated according to both sentence and salient template. Experiments show that our model can generate informative, readable and stable summaries. In addition, our model demonstrates promising prospect in generation diversity.

Our work covers three fundamental representations of copying and rewriting in summarization, namely copying and rewriting source words, copying source facts and rewriting with soft templates. We believe that the three models can work closely with each other to yield a more powerful neural summarization system. For example, in Chapter 5, we propose to embed the rewriting decoder into Re^{3}Sum, which considerably restrains the generation of new named entities derived from soft templates.

Our experiments are conducted on sentence summarization based on the following considerations.

- The output of sentence summarization is relatively short, which benefits human evaluation.
- As mentioned in Section 1.1.2, the sentence summarization benchmark Gigaword [Rush *et al.*, 2015b] is almost the largest abstractive summarization dataset available.
- Sentence summarization is a growing attractive task, with numerous models developed recently (refer to Section 2.3). Thus we can conduct plentiful comparisons.
- Sentence summarization is the foundation of document summarization. Abstractive summarization is still in its infancy. Without the success of sentence summarization, document summarization cannot come into being.

1.4 Research Contributions

The contributions of our research are summarized in this section. Generally, our work is the one among the few large-scale studies of copying and rewriting in seq2seq models. We are the precursor to promote the development of summarization-specific seq2seq models. Actually, following our work, researchers (e.g., [Song *et al.*, 2018; Han *et al.*, 2017]) have developed a series of models focusing on copying or rewriting. In addition, our work builds a bridge between traditional summarization approaches and the seq2seq neural network models. We make better use of prior knowledge from extractive, compressive and template-based summarization to advance seq2seq models for generating more informative, faithful and fluent summaries. We believe that the prior knowledge largely accumulated during the past half a century research could highly broaden the mind of researchers in seq2seq summarization.

The specific technical contributions are listed as below.

CoRe We distinguish the learned weights of the attention mechanism to explicitly represent copying and rewriting of the source words, respectively. In copying, the attention mechanism directly indicates the current copied position. In rewriting, the attention mechanism helps to construct the source-specific vocabulary, which empower the decoder to generate relevant words efficiently. Our model greatly enhances the interpretability of the neural network architecture, which is a crucial issue in research of deep neural networks. In addition, CoRe provides a uniform framework to handle paraphrasing-related tasks as well as many other types of generation. For example, we can easily adapt CoRe to the the increasingly attractive task of sentiment style transfer [Li *et al.*, 2018a; Xu *et al.*, 2018], where the copying decoder focuses on preserving semantic content while the rewriting decoder generates words restricted in the opposite sentiment.

FTSum To the best of our knowledge, we are the first to explore the faithfulness problem in abstractive summarization. We want to arouse attentions in researchers that faithfulness, in addition to informativeness, is a vital precondition for a practical abstractive summarization system. To address this issue, we propose the first fact-aware seq2seq model. Experiments show that the proposed model greatly reduces fake summaries by 55%, and at the same time achieves significant improvement on informativeness, compared with the standard seq2seq model. We publish the first labeled faithfulness dataset and believe that this dataset will promote the research on faithfulness in summarization. We also develop a tool, called FTEval, for automatic faithfulness evaluation. FTEval highly corresponds to the manual judgment. Thus, we expect that FTEval could become a benchmark tool for summary evaluation. In addition, FTEval gives explanations to all identified fake summaries, which benefits researchers to fix the errors and develop more reliable summarization systems.

Re^{3}Sum We incorporate the idea of template-based summarization into the seq2seq model to make it more stable and generate more fluent summaries. We are the pioneers to integrate IR-based ranking techniques and seq2seq-based generation techniques in a creative way by fully taking the advantages of both sides. The abstraction-after-extraction framework is deserves more attention in the summarization area. Finally, to the best of our knowledge, we initiate the research of generation diversity in seq2seq summarization and provide a simple yet effective solution. It is the basis of personalized summarization, another fascinating task.

1.5 Structure of Dissertation

Now, let me introduce the skeleton of the thesis. Chapter 1, this chapter, describes the background of automatic text summarization and provides the overview of our research. We explain the motivations of our work and summarize our contributions. Chapter 2 is the review of the literature of text summarization, including traditional extractive summarization, compressive summarization and template-based summarization. It also introduces the existing research on generation-based seq2seq summarization, covering its application and adaptation in text summarization and other relevant areas. The next three chapters construct the technical body of this thesis. Chapter 3 illustrates the copying and rewriting behaviors in summarization, where I propose to explicitly model them within the seq2seq framework. Chapter 4 is absorbed in the faithfulness problem of neural abstractive summarization. To reduce fake generation, I modify seq2seq to simulate compressive summarization, that is, using source facts as additional input. Chapter 5 is engaged with the stability requirement of summarization. Borrowed the idea from template-based summarization, I force seq2seq to learn to rewrite from an existing summary. In Chapter 6, I point out the potential extension of our work in personalized summarization. In Chapter 7, the last chapter, I review the major findings, technical results , and reiterate the major contributions in this thesis.

Chapter 2

Literature Review

Automatic text summarization was born in 1958 [Luhn, 1958]. Since then, researchers on text summarization have proposed numerous theories, models and algorithms, and applied this technique to various scenarios. The past decades have witnessed text summarization gradually becomes one of the most significant research focuses in the Natural Language Processing (NLP) area.

I will firstly introduce the widely-studied extractive summarization in Section 2.1. Various approaches are discussed, including shallow feature-based, graph-based, learning-based and the recently proposed deep neural network-based approaches. It is followed by the review of two types of traditional abstractive summarization approaches, namely compressive summarization and template-based summarization, in Section 2.2. Finally, Sequence-to-sequence (seq2seq) neural summarization models, which are the focus of this thesis, will be discussed and analyzed in Section 2.3.

2.1 Extractive Summarization

Extractive summarization [Over and Yen, 2004] selects important sentences in the source text to form a summary. It is easy to implement, able to preserve the original information and maintain the readability at the sentence-level. Thus, extractive summarization becomes most common approaches to automatic summarization. Many actual scenarios can be regarded as the application of extractive summarization like the snippets provided by a search engine. Basically, there are two crucial processes in extractive summarization: *sentence ranking* and *sentence selection*. The former measures the saliency of a sentence, while the latter aims to generate a non-redundant summary. Sentence ranking plays a core role in extractive summarization. Research on it spans a large range of approaches. Based on the intermediate representation, four important sentence ranking approaches are introduced in the rest of this section. They are shallow features-based, graph-based, learning-based and deep neural network-based ranking approaches.

Sentence selection is an also important step for extractive summarization, especially for multi-document summarization. Since the strategies for sentence selection are relatively fixed, I go through a quick overview here. Maximum Marginal Relevance (MMR) [Carbonell and Goldstein, 1998] is one of the most well-known sentence selection approaches. MMR, rooted in Information Retrieval (IR) systems, used a greedy approach to consider the trade-off between sentence saliency and redundancy among existing summary sentences. Better performance could be achieved by formulating sentence selection as an integer linear programming (ILP) problem where the optimal solution exists [McDonald, 2007]. Instead of directly measuring redundancy between two sentences, some studies, such as [Gillick and Favre, 2009], modeled redundancy based on the n-gram overlap. Then, the task is transformed to the weighted set cover problem which can be solved by ILP as well. Since ILP is NP-hard, researchers (e.g., [Sipos *et al.*, 2012; Dasgupta *et al.*, 2013]) converted it into a submodular maximization problem that can be handled by efficient approximation algorithms. Without specification, the extractive summarization approaches mentioned below choose one of the sentence selection strategy (MMR, ILP, submodular) to handle redundancy.

2.1.1 Shallow Feature-based Approaches

This type of summarization approaches take advantages of shallow text features, such as word or n-gram frequency, sentence length and sentence position, to detect salient sentences. Most models are developed heuristically or empirically. One early work on this study was [Luhn, 1958], who measured the saliency of a word by its frequency and relative position in sentences. Then, the summary sentence were expected to be those containing the most frequent words in certain important positions. Later, [Edmundson, 1969] extended [Luhn, 1958]'s work with additional consideration of cue phrases, title, and sentence position in the document. It is noted that the proposed two features, title similarity and sentence position, have been widely used in subsequent research work. [Lin and Hovy, 1997] conducted rigorous experiments to assess the importance of different positions. [Nenkova and Vanderwende, 2005] simply used word frequency to measure saliency. Their summarization system, called SumBasic, achieved competitive performance in DUC 2004. Later, its extended version, named SumFocus [Vanderwende *et al.*, 2007], was also a top system on query-focused summarization in DUC 2007. [Radev *et al.*, 2004a] developed a famous summarization platform MEAD. It allowed people to employ several important features to measure sentence saliency, including Centroid [Radev *et al.*, 2004b], the sentence length and LexRank scores [Erkan and Radev, 2004]. MEAD required manually defined weights to merge these features to derive the final saliency score. Recent years, concept coverage based summarization approaches also demonstrated competitive performance. For example, [Gillick and Favre, 2009] used bi-grams to represent concepts and weighed them with document frequency. Then, the Integer Linear Programming (ILP) formulas were proposed to find the summary sentences covering the most important bi-grams. [Schluter and Søgaard, 2015] further tested the work of [Gillick and Favre, 2009] by using other text units to convey concepts. They introduced a series of syntactic and semantic concepts, including syntactic dependencies, semantic frames and named entities. The conclusion was that using such concepts could lead to significant improvements in text summarization performance on legal judgments and Wikipedia articles.

2.1.2 Graph-based Approaches

The graph-based approaches have flourished increasingly in Natural Language Processing (NLP) as well as Information Retrieval (IR) areas [Mihalcea and Radev, 2011]. This technique has also attracted growing attentions in extractive summarization these years. Normally, a graph-base model regards documents as a text graph where a node represents a text unit such as a sentence or a word, and an edge stands for relations between text units. Common relations include similarity, association and relevance. Then, the importance of a node can be computed through graph-based ranking algorithms which are capable of considering the global information from the entire graph. Common ranking algorithms include PageRank [Page *et al.*, 1999] or HITS [Kleinberg, 1999]. A popular stochastic graph-based summarization approach was LexRank [Erkan and Radev, 2004]. It built a sentence graph where edges represented sentence similarities measured by the tf-idf cosine similarity between two sentences.. Developed on the PageRank algorithm, LexRank measured sentence saliency based on eigenvector centrality in this graph. A similar and equally representative model was TextRank [Mihalcea and Tarau, 2004], where sentence similarities were simply computed from term overlaps. LexRank and TextRank served generic summarization. Later, [Otterbacher *et al.*, 2005] extended LexRank to handle query-focused summarization. They treated the relevance to the query as the initial probabilities to construct a biased PageRank ranking.

The above-mentioned approaches ignore the difference among documents and simply combine multiple documents into a single document. A series of following work have tried to handle this problem. For example, [Wan *et al.*, 2006] built two independent graphs, connecting inter-document sentences and intra-document sentences, respectively. Separate graph-ranking algorithms were then run on the two graphs. Finally, the saliency score of a sentence was the weighted sum of the inter-document ranking score and intra-document ranking score. Interestingly, their experiments showed that using only intra-document relationships between sentences outperformed other approaches, including using both the intra-document graph and inter-document graph. Later, [Wei *et al.*, 2010] presented a document-sensitive graph model called DsR. It distinguished sentence relations in sentence similarities calculation.

Apart from representing sentences as nodes, researchers have attempted other text units to build the graph. [Zha, 2002] constructed a weighted bipartite graph on both the words and the sentences. If a word appeared in a sentence, an edge would connect them. Their motivation was that a salient word should appear in many salient sentences, and vice versa. By applying the mutual reinforcement principle, their system extracted both salient sentences and terms at the same time. [Wan and Yang, 2008] introduced themes, represented by sentence clusters, as additional nodes in the text graph. They argued that themes usually held various sizes and saliency for users to understand the source text. While [Wan and Yang, 2008] used simple clustering algorithms, such as k-means and agglomerative clustering, [Cai *et al.*, 2010] extended this work by integrating clustering and ranking simultaneously to update each other so as to improve both performances.

Many graph-based approaches rank sentences according to their independent saliency. Then, a sentence selection step as mentioned above is necessary to avoid or to redundant sentences. Some researchers considered the diversity issue when developing the graph ranking algorithms. For example, [Zhu *et al.*, 2007] adopted the absorbing random walk on graph, which ranked nodes with an emphasis on diversity. At each iteration, the node representing the sentence selected in the last iteration became the absorbing node, preventing the redundant sentences from receiving a high ranking score. In DivRank [Mei *et al.*, 2010], the diversity was guaranteed by using the reinforced random walk on graph. The nodes competed with each other so that the node with a high ranking score would progressively absorb the ranking scores from its neighbors.

2.1.3 Learning-based Approaches

In contrast to the aforementioned unsupervised approaches, there are also numerous efforts on supervised learning for summarization. That is, with human summaries as labeled data, supervised machine learning models are introduced to train a summarization system. A classical learning-based summarization system was KPC [Kupiec *et al.*, 1995]. The authors built a corpus containing 188 (document, summary) pairs as the training dataset. Then, they defined a set of features, such as sentence length, fixed-phrase list, paragraph-related features (whether leading or ending paragraphs, and sentence position in the paragraph), topic word frequency and uppercase word frequency, to learn a Bayesian classifier. [Larsen, 1999] experimented combinations of different features. Likewise, they found that the summaries produced by supervised learning significantly outperformed those by unsupervised learning. Many early studies followed the same idea although examined different classification models, such as decision tree [Chuang and Yang, 2000] and Support Vector Machine (SVM) [Hirao *et al.*, 2002]. Some researchers converted single-document summarization into a sequence labeling task. It was thus nature to apply maximum entropy [Osborne, 2002], conditional random field (CRF) [Galley, 2006] or hidden Markov model (HMM) [Conroy *et al.*, 2004]. Later, learning-to-rank models were exploited. [Fisher and Roark, 2006] built a perceptron-based ranking system and automatically constructed a corpus to train this system. [Toutanova *et al.*, 2007] proposed a system called PYTHY, which learned a log-linear sentence ranking function to combine features. They also developed different metrics to measure the actual saliency of a sentence. With two sophisticated post-processing processes, their system achieved the second best performance in DUC 2007. Recently, regression models have attracted growing attention in extractive summarization. [Ouyang *et al.*, 2007] presented the pioneering work of applying Support Vector Regression (SVR) to learn sentence saliency. Later, many researchers [Li *et al.*, 2013b; Hong and Nenkova, 2014] adopted logistic regression to find out important n-grams in the documents. Some works were also devoted to the performance evaluation of different learning-based models. For example, To build a text summarizer, [Fattah and Ren, 2009] implemented and compared five learning models, including mathematical regression (MR), Gaussian mixture model (GMM), genetic algorithm (GA), probabilistic neural network (PNN) and feed forward neural network (FFNN). The result showed that GMM were the most promising. [Ouyang *et al.*, 2011] conducted direct comparisons of regression, classification and learning-to-rank models, and indicated that regression models were usually preferable.

2.1.4 Deep Neural Network-based Approaches

Recently, the application of deep neural networks has become increasingly popular in the summarization area. [He *et al.*, 2015] trained their summarization model based on data reconstruction. They treated a summary as a compressed representation of documents. Using a term frequency vector to represent a sentence, they aimed to find the proper subset which could reproduce the input documents. Nevertheless, their system failed to achieve satisfactory performance. Later, [Yao *et al.*, 2015] improved the work of [He *et al.*, 2015] by introducing an additional sentence dissimilarity term in the loss function. Their system showed competitive performance on DUC 2006 and DUC 2007 benchmarks. A few researches have explored to directly measure saliency based on similarity between distributed representations. [Yin and Pei, 2015] trained a language model on the basis of Convolutional Neural Networks (CNN) to map sentences into embedding representations. Afterwards, they use cosine similarity of embeddings to measure the sentence similarity. Based on the sentence similarity, they adopted PageRank to get the sentence saliency score and designed a submodular function to consider both prestige and diversity. Others like [Kobayashi *et al.*, 2015] just summed up existing word embeddings to represent a document or sentence. [Li *et al.*, 2017b] used Variational Auto-Encoders (VAEs) to represent sentences. Then, an unsupervised data reconstruction framework was followed to jointly consider the reconstruction in both latent semantic space and observed term vector space. They regarded attention as saliency and performed sentence ranking and keyword extraction at the same time. Supervised methods have also been explored. For example, [Cheng and Lapata, 2016] used binary sequence labeling to model single-document summarization and proposed a RNN solution. Some researchers based on the neural network representation of documents to develop multi-task learning algorithms. For example, [Isonuma *et al.*, 2017] jointly learned sentence ranking and document classification.

The summarization approaches mentioned above are purely data-driven. Since numerous significant features have been defined for summarization (e.g., TF-IDF) in the past, some researchers also investigated the combination of hand-crafted features and learned features. For example, we [Cao *et al.*, 2015a] applied recursive neural networks to automatically learn the combination of hand-crafted features. Subsequently, regressions were conducted at different text granularities based on the learned representations as well as the input features. Later, we [Cao *et al.*, 2015b] proposed to use convolutional neural networks to learn document-independent features. These features mainly stood for the summary prior quality hard to be manually defined. Together with four simple document-dependent features, the model of [Cao *et al.*, 2015b] reached a new state of the art.

Some researchers also attempted to use deep learning for summary evaluation. For example, [Genest *et al.*, 2011] used unsupervised auto-encoders to represent both manual and system summaries for the task of summary evaluation. The difference of these two representations were defined as features for regression, with regard to the Pyramid scores [Passonneau *et al.*, 2005]. Their method, however, did not surpass ROUGE.

2.2 Traditional Abstractive Summarization

Although extractive approaches are relatively easy to implement, the quality of extracted summaries is rather restricted. For example, suppose there is a long yet partly relevant sentence. Without this sentence, the summary may lose important information.

While extracting this sentence, the conciseness of the summary is likely to worsen seriously. In addition, experiments in [Murray *et al.*, 2010] showed that users highly expected summaries to be more abstractive. Thus, abstractive summarization has drawn growing attentions in recent years. Here I will review two typical kinds of traditional abstractive summarization approaches, i.e., compressive summarization and template-based summarization.

2.2.1 Compressive Approaches

Compressive summarization[Almeida and Martins, 2013] does not restrict a summary sentence to be the exact same as a original sentence in the documents. Instead, it allows the extraction of compressed sentences [Knight and Marcu, 2000] where some words can be deleted. In comparison, sentence fusion [Barzilay and McKeown, 2005] aims to construct a new sentence which covers the main information of sentences in a given set of similar sentences. To simplify, we call both techniques compressive approaches.

[Lin, 2003] conducted a pilot study on compressive summarization. Their results showed that pure compressive approaches failed to outperform extractive approaches. However, reranking using an oracle demonstrated a potential significant improvement. Bascially, there are two strategies for compressive summarization, with either the joint or pipeline approaches. Joint approaches attempted to conduct compress and extract sentences at the same time. [Gillick and Favre, 2009], though using an unsupervised model, explored to combine extraction and compression within the Integer Linear Programming (ILP) formulation. Specifically, they extended the ILP formulation used for pure extraction with additional constraints to incorporate sentence compression. Compared with pure extraction, compression showed a marginal increase in the ROUGE evaluation, but a decline in the Pyramid evaluation. [Daume, 2006] proposed a joint framework to learn model parameters for extraction and compression in the same time, but his system could not compete with purely extractive approaches.

The lack of proper training datasets largely hinders the development of joint compressive summarization. Moreover, joint optimization is rather computationally expensive. As a result, numerous researchers opted for pipeline approaches, which conducted sentence compression in advance and then formed summaries from the compressed sentences. For example, [Zajic *et al.*, 2006] applied heuristic rules to yield additional compressive candidates for each sentence. Then, a simple greedy sentence selection step was used to pick summary sentences from these candidates. Their system, however, did not achieve the state-of-the-art performance. [Li *et al.*, 2013a] trained a supervised sentence compression model beforehand using a set of word-, syntax-, and document-level features. Then, they fed multiple compressed sentences into Integer Linear Programming (ILP) to select salient summary sentences. Experiments showed that their compressive model was simple yet effective. [Bing *et al.*, 2015] proposed a fusion-based summarization framework with a particular focus on noun and verb phrases. Specifically, their first extracted phrases from the source documents to resemble a pool of concepts and facts. Then they generated new sentences by choosing and merging informative phrases. The objective of the generated summary was to maximize the saliency of the selected phrases and also obey the restraints of sentence construction. Deep learning technologies have also been employed in compressive summarization. For instance, [Filippova *et al.*, 2015] regarded compression as a binary sequence labeling task which was modeled by Long Short Term Memory (LSTM) networks. They also built a large scale compression dataset through automatically paralleling (sentence, headline) pairs.

All the above-mentioned approaches rely on parse trees for compression and/or fusion. Researchers have also proposed to utilize word graphs [Barzilay and Lee, 2003] to fuse important content. A word graph is represented by a directed graph. If word X and word Y are adjacent in a source sentence, then an an edge from X to Y will be added into the graph. The technique of word graphs has been widely used in NLP such as language modeling and paraphrasing. [Filippova, 2010] introduced this idea for fusion-based summarization. They used various graph and semantic features to measure the weight for an edge. Then, the shortest path algorithm was adopted to find the concise and informative sentence to represent a word graph.

2.2.2 Template-based Approaches

One serious problem that all the above-mentioned summarization approaches suffer is that there might be no appropriate sentence to extract or compress. It hence promotes a more abstractive approach called template-based summarization [Moratanch and Chitrakala, 2016]. In general, template is a common natural language generation technique that matches its nonlinguistic input directly (i.e., without intermediate representations) to the linguistic surface structure [Reiter and Dale, 1997]. This structure may contain “gaps” which are filled in during output. Defining templates usually requires a lot of human efforts and domain knowledge. Thus, template-based natural language generation approaches are often applied in industry to produce fluent and relevant text. For example, the prevailing chatting robot XiaoIce[1] contains numerous templates manually created in real time to fix bad cases. In summarization research, [Harabagiu and Lacatusu, 2002] developed an automatic summarization system GISTEXTER to produce both extracts and abstracts from the documents. It leveraged the CICERO IE system [Surdeanu and Harabagiu, 2002] to generate templates. Thereafter, these templates were mapped into text snippets from documents, in order to generate coherent, informative multi-document summaries. [Zhou and Hovy, 2004] investigated template-based headline summarization. Their system consisted two phases. Utilizing a serious of manually defined keywords, the keyword clustering phase found out title phrases in the front of the document. The template filter phase subsequently populated pre-specified headline templates with headline phrases to produce the resulting headlines. [Oya *et al.*, 2014] presented a template-based abstractive meeting summarization system. It first automatically extracted templates from human-written summaries through clustering and fusion. Afterwards, the system picked up the best templates for

[1] `http://www.msxiaoice.com/`

the current summary generation based on the relationship between summaries in the training data and the source meeting transcripts.

2.3 Seq2seq in NLP

The sequence-to-sequence (seq2seq) framework (a.k.a encoder-decoder architecture) is a newly emerging approach for natural language generation. With a seq2seq model, the source text is encoded by the encoder as a context vector. Then, a decoder decodes the semantic information in the vector and outputs the target text. The seq2seq framework, as Neural Machine Translation (NMT), was originally proposed by [Kalchbrenner and Blunsom, 2013; Cho *et al.*, 2014a; Sutskever *et al.*, 2014]. Compared with the traditional statistical machine translation (SMT) approaches (e.g., [Koehn *et al.*, 2007]), seq2seq models required less human efforts and achieved better performance. Later, [Bahdanau *et al.*, 2014] developed the attention mechanism which largely promoted the applications of the seq2Seq models. In addition to machine translation, seq2seq models have reached the competitive performance in many other areas like dialog generation [Shang *et al.*, 2015]. Recently, the application of the attentional seq2seq model has also attracted growing attention in abstractive sentence summarization. [Rush *et al.*, 2015a] firstly attempted the encoder-decoder architecture. In their model, the encoder was attentional Convolutional Neural Network (CNN), and the decoder was a simple neural network language model like [Bengio *et al.*, 2003]. Later, the authors proposed to replace the decoder with Recurrent Neural Network (RNN) [Chopra *et al.*, 2016]. Finally, in [Nallapati *et al.*, 2016a], both encoder and decoder were RNN, yielding a complete attentional seq2seq model. Since then, there have been a series of variants of seq2seq summarization models. In what follows, I will introduce five important types of modifications, including changes in encoder, attention, decoder, neural network architecture and loss function. Since these changes can usually be employed universally, I will survey work on different generation tasks (e.g., machine translation and dialog generation), where the study on abstractive summarization will be highlighted.

2.3.1 Encoder Modification

The original encoders [Sutskever *et al.*, 2014] only read source text either forward or backward, which makes a word ignore its subsequent context. [Sundermeyer *et al.*, 2014] proposed a bidirectional architecture, which can take the full source sentence into account for all word predictions. Since then, the bidirectional LSTM has become the standard configuration in seq2seq.

Syntactic information benefits a series of traditional generation models, including syntax-based SMT and compressive summarization. Thus, the syntax-aware seq2seq framework has been proposed in the area of machine translation. [Li *et al.*, 2017a] linearized a phrase parse tree into a structural label sequence and used the standard encoder to read this mixed word and label sequence. [Hashimoto and Tsuruoka, 2017] argued that a syntactic parser trained by supervised learning in advance might not be suitable for the current generation. Therefore, the authors proposed to learn a parser as part of the encoder. Different from the common dependency parse tree, they defined a parse graph where a word could hold multiple head words. In such way, they added an attention layer over the encoder to learn the dependency weights of each source word on the source sentence. Experimental results demonstrated that their model was able to learn meaningful dependency by itself, while using initialization of a pre-trained dependency tree could further improve the performance.

The representations of the RNN encoder may be insufficient for a more complete understanding of the source text, such as its style or theme. Inspired by the variational auto-encoder (VAE) [Kingma and Welling, 2013; Rezende *et al.*, 2014], many researchers (e.g., [Zhang *et al.*, 2016]) have proposed to learn latent variables from the encoder output. These latent variables, fed into each decoder state, are expected to handle the diversity in the source text. Generative Adversarial Net (GAN) [Yu *et al.*, 2017] has also been applied for the similar purpose. Their system outperformed other approaches on three tasks, i.e., poem generation, speech language generation and music generation.

Many researches on summarization have attempted to inject more semantics into the encoder. For example, [Nallapati *et al.*, 2016a] enriched the encoder with lexical and statistic features such as position, named entity and part-of-speech tags. These features played important roles in traditional feature based summarization approaches. Their experiments verified the effect of these features. [Zhou *et al.*, 2017b] added a selective gate network over the encoder hidden states. This network constructed a second level sentence representation by controlling the information flow from encoder to decoder. Experiments showed that their model was able to tailor source text according to saliency.

2.3.2 Attention Modification

Recently, the attention mechanism has nearly become a standard module in seq2seq models, due to its strong ability to learn alignments between different modalities [Chorowski *et al.*, 2014; Mnih *et al.*, 2014; Xu *et al.*, 2015]. the attention mechanism can be classified into hard and soft types [Xu *et al.*, 2015]. The soft attention works globally by measuring alignment weights over all the source positions. In comparison, the hard attention only concentrates on a small part of the source content at a time.

[Bahdanau *et al.*, 2014] firstly introduced the attention mechanism into neural machine translation to learn the alignment between source and target words. They used multi-layer perceptrons (MLP) to measure the weights of attentions. [Vaswani *et al.*, 2017] proposed the scaled dot-product attention, which added a scaling factor on the dot product of two hidden states. The authors believed this attention could run much faster and achieve competitive performance against MLP attention.

In addition to source-target word alignment learning, researchers also explored the other uses of attentions. For instance, attentions were accumulated to measure which source words had already been translated in [Tu *et al.*, 2016b]. [Vaswani *et al.*, 2017] proposed a more complicate multi-head attention, which computed attentions on all the encoder-encoder, encoder-decoder, decoder-decoder positions. Their machine translation system, Transformer, could be trained far more quickly than common architectures, and outperformed state-of-the-art approaches on two translation benchmarks. In the summarization area, [Gu *et al.*, 2016] made use of attentions to measure the copying mechanism . As mentioned above, copying is one of the core writing behavior in summarization, that is, most keywords in the source text should be reserved by the summary. [Nallapati *et al.*, 2016a] developed hierarchical attentions to measure the alignment of a summary word from source sentences to source words. Inspired by the graph-based extractive summarization approach LexRank [Erkan and Radev, 2004], [Tan *et al.*, 2017b] proposed the graph-based attention mechanism to measure the saliency of sentences.

2.3.3 Decoder Modification

When the seq2seq framework was first applied in abstractive summarization in [Rush *et al.*, 2015a], the decoder was a neural network language model. Most subsequent work applied the RNN decoder as to generate a sentence word by word. The general decoder packs all linguistic granularities (e.g., words, phrases and clauses) in the same timescale of RNN. [Zhou *et al.*, 2017a], in contrast, proposed a hierarchical RNN decoder for machine translation. It split the hidden states of the decode into two phases and updated them in different time steps. The bottom state was used for phrase modeling, and the word states were generated on it. While the encoder in a seq2seq framework is usually bi-directional, the decoder often works in a forward manner, leaving the target-side contexts generated from right to left unexploited. [Zhang *et al.*, 2018] equipped the conventional seq2seq framework with a backward decoder to enable bi-directional machine translation. Specifically, the backward decoder first learned to generate the target-side hidden state sequence from right to left. Then, the forward decoder performed generation in the forward direction, which simultaneously applied two attention neural networks corresponding to the source-side and the reverse target-side hidden states, respectively. [He *et al.*, 2016; Wang *et al.*, 2017] found that NMT generally produced fluent but inadequate translations, in contrast to conventional SMT. To handle this problem, they incorporated the SMT model into the NMT framework where SMT offered additional recommendations of generated words for NMT. [Kiyono *et al.*, 2017] proposed the Source-side Prediction Module (SPM) which was put on the decoder to directly estimate the correspondence between source and target words. In this way, SPM enabled to jointly estimate the probability distributions over source and target vocabularies. In abstractive summarization, [Kikuchi *et al.*, 2016] argued that length control was crucial. They added the desired length information into the decoder through two ways: *LenEmb and* LenInit. The first one inputted the remaining length to the decoder at each decoding step, while the second one embedded the expected length as the additional initialization parameters of the decoder. Results showed that both methods were able to control length and keep summary quality.

To enhance efficiency of learning, researchers have also tried to modify the output dimension of the decoder. [Cho *et al.*, 2015] restricted the decoder to generate the words from the actual target words together with a sampled word set during training of NMT. In dialog generation, [Wu *et al.*, 2017] proposed a dynamic vocabulary seq2seq model which allowed each input to possess their own vocabulary in decoding. Vocabulary construction and response generation were jointly learned during training. In summarization, [Nallapati *et al.*, 2016b] extended the work of [Cho *et al.*, 2015] by supplementing the output vocabulary with the 1-nearest-neighbors of words in the source text, as measured by the similarity in the word embedding space. We believe these neighbors of source words can partially reflect the rewriting behavior in summarization.

2.3.4 Neural Network Architecture Modification

In NLP, [Mikolov *et al.*, 2010] was the first to use the vanilla RNN to train a language model. However, with the increase of network depth, the vanilla RNN often suffers from vanishing gradient or exploding gradient. To handle this problem, earlier work [Kalchbrenner and Blunsom, 2013; Cho *et al.*, 2014a; Sutskever *et al.*, 2014] opted to use Long Short-Term Memory (LSTM) [Gers *et al.*, 1999] as the recurrent unit. LSTM is a special type of recurrent neural network. It usually outperforms vanilla recurrent neural network due to the import of various gates to control the input and update processes. Later, another competitive architecture called Gated Recurrent Unit (GRU) [Cho *et al.*, 2014b] was also widely applied in seq2seq (e.g., [Bahdanau *et al.*, 2014]). Compared with LSTM, GRU has been shown to exhibit better performance on smaller datasets [Chung *et al.*, 2014]. Meanwhile, it contains fewer parameters with the removed output gate. However, both LSTM and GRU are still far slower than Convolutional Neural Network (CNN) due to the intrinsic difficulty in parallelizing their state computations. Recently, [Lei and Zhang, 2017] have proposed Simple Recurrent Unit (SRU), i.e., a recurrent unit that simplified the computation and exposed more parallelism. m. In SRU, the majority of computation for each step was independent of the recurrence and could be easily parallelized. SRU was reported to be as fast as a convolutional layer and 5-10 times faster than an optimized LSTM implementation. [Li *et al.*, 2018b] argued that construction and training of a deep LSTM or GRU based RNN network was practically difficult because of the use of the hyperbolic tangent or the sigmoid functions as the activation function. In addition, all neurons in an RNN layer were entangled together and their behaviors were hard to interpret. Thus, they developed the Independently Recurrent Neural Network (IndRNN), where neurons in the same layer were independent of each other and connected across layers. Experimental results showed that IndRNN was able to process very long sequences (over 5000 time steps), and could be used to construct very deep networks.

Some studies even attempted to substitute the RNN architecture in seq2seq. For example, [Gehring *et al.*, 2017] used the Convolutional Neural Network (CNN) for both encoder and decoder. Compared to RNN, computations over CNN can be fully parallelized, which makes full use of the power of GPUs. In addition, the optimization of CNN is more convenient due to the fixed number of non-linearities. [Vaswani *et al.*, 2017] proposed Transformer which totally depended on attention mechanisms, with neither recurrence nor convolution.

2.3.5 Loss Function Modification

The dominant approach [Bahdanau *et al.*, 2014] to training a seq2seq model is equivalent to training a conditional language model. The learning objective is to maximize the likelihood of each actual target word given the source text and the actual history of target text. Usually, the word-level loss function, called Negative Log-Likelihood (NLL), is adopted during training. However, in test, seq2seq models are expected to fluent target text similar to what humans write. It is a global objective and is impossible to evaluate word-by-word. [Ranzato *et al.*, 2015] argued that NLL loss in seq2seq models would suffer from two serious problems, i.e., exposure bias and loss-evaluation mismatch. Later, [Wiseman and Rush, 2016] pointed out another important issue, i.e., label bias [Lafferty *et al.*, 2001], which deteriorates the performance of a seq2seq model quickly with the increase of the length of generation [Koehn and Knowles, 2017]. In view of these weaknesses, some summarization researchers proposed to use a global loss function during training. For instance, [Ayana *et al.*, 2016] employed the Minimum Risk Training (MRT) strategy, which could tune model parameters directly according to the summarization evaluation metrics of ROUGE. Specifically, during training, their model also used the beam search to generate a summary. Then, according to MRT, the loss function was based on the difference of ROUGE scores between the generated summary and actual summary. As such, the training process was exactly the same as the test process. Similar idea was adopted by [Wiseman and Rush, 2016], except that they computed the ROUGE loss during each decoding step to accelerate convergence.

Although these methods were able to achieve high ROUGE scores on test datasets, the generated summaries were often only composed of key phrases and ungrammatical. Later, [Paulus *et al.*, 2017] applied the reinforcement learning algorithm to learn a mixed objective function of likelihood and ROUGE scores.

Other auxiliary losses have also been introduced in summarization. [Cao *et al.*, 2017a] added a 0/1 label for each summary word to predict whether it was copied from the source text. These labels provided additional supervision for this task. Likewise, [Zhou *et al.*, 2017a] used the similar binary labels to indicate the boundary of a phrase for machine translation. Inspired by the idea of input reconstruction in extractive summarization [He *et al.*, 2012], [Miao and Blunsom, 2016] firstly used a seq2seq model to generate a summary from the source sentence, and then recovered the input conditioned on this summary. Since the generated summary was discrete, they used the reinforcement learning algorithm to tune model parameters. Notably, their approach can work in the semi-supervised and even unsupervised scenarios. Later, [Tu *et al.*, 2016a] further developed the reconstruction concept for NMT. Differently, the input of the reconstruction process was the hidden layer at the target side, rather than the output words. As such, model became derivable, bringing huge convenience and efficiency for training.

Chapter 3

Copy and Rewrite Words in Source Text

You can't do better design with a computer, but you can speed up your work enormously.

— **Wim Crouwel**
(Graphic designer and typographer)

3.1 Introduction

Summarization is to use a condensed description to summarize the main idea of a document. As a result, copying and rewriting are two main writing styles in summarization. Copying means repeating keywords in the source text while in rewriting a concept is paraphrased or generalized using other words. Recently, the encoder-decoder structure (aka. seq2seq framework) has become more and more popular in many language generation tasks [Bahdanau *et al.*, 2014; Shang *et al.*, 2015]. Studies such as [Rush *et al.*, 2015b; Hu *et al.*, 2015b] have applied the seq2seq models to the sentence summarization task. Despite the competitive performance, these models seldom take into account the two major writing modes in summarization.

On the one hand, due to the nature of the task, keywords in the source text are usually reserved in the target text. However, with only one decoder generating over the entire vocabulary, a typical seq2seq model fails to reflect the copying mode. As a result, many keywords provided in the source text may be overlooked in the target text. In addition, certain keywords like named entities are often rare words and masked as unknown (UNK) tags in seq2seq models, which unavoidably causes the decoder to generate a number of UNK tags. Although not aiming to explicitly explore the copying mechanism, the work of [Rush *et al.*, 2015b] finds that it largely improves the performance to add the input-related hand-crafted features to guide generation.

On the other hand, rewriting also plays a significant role in summarization. With this writing mode, although the target words are not the same as the source words, there are semantic associations between them. For example, “seabird” is possibly generalized as “wildlife”, and “rise strongly” can be paraphrased into “boom” for short. The decoders in most previous work generate words by simply picking the likely target words that fit the contexts out of a large vocabulary. This common practice suffers from two problems. First, the computation complexity is linear to the vocabulary size. In order to cover enough target words, the vocabulary size usually reaches 10^4 or even 10^5. Consequently, the decoding process becomes quite time-consuming. Moreover, the decoder sometimes produces the named entities that are common but do not exist in or even irrelevant to the input text, leading to an unfaithful summary.

In this chapter, we propose a novel seq2seq model named CoRe, which captures the two **core** writing modes in summarization, i.e., **Co**pying and **Re**writing. CoRe fuses a copying decoder and a rewriting decoder. Inspired by [Vinyals *et al.*, 2015], the copying decoder finds the position to be copied based on the existing attention mechanism. Therefore, the weights learned by the attention mechanism have the explicit meanings in the copying mode. Meanwhile, the rewriting decoder produces the words restricted in the source-specific vocabulary. This vocabulary consists of a (source, target) word alignment table together with a small collection of frequent words. The alignment table is trained in advance, and updated according to the attention mechanism. With the supplement of a few frequent words, experiments (see Table:3.2) show that more than 90% of the target words have been covered by our decoders. While the output dimension of our rewriting decoder is just one tenth of the output dimension used by the common seq2seq models, it is able to generate highly relevant words concerning rewriting. To combine the two decoders and determine the final output, we develop a predictor to predict whether the writing mode is copying or rewriting. Since we know the actual writing mode at each output position in a training instance, we introduce a binary sequence labeling task to guide the learning of this predictor, which takes advantages of the supervision derived from the writing modes.

To our knowledge, the work most relevant to ours is the COPYNET [Gu *et al.*, 2016], which also explores the copying mechanism. However, COPYNET adds an additional attention-like layer to predict the copying weight distribution. This layer then competes with the output of the rewriting decoder. Therefore, it is not easy for COPYNET to explain the contributions of copying and generation. Compared with our model, COPYNET introduces a lot of extra parameters and ignores the supervision derived from the writing modes. Moreover, the rewriting decoder of COPYNET is only allowed to produce frequent words. As a result, the rewriting patterns are discarded to a large extent.

We conduct a wide range of experiments on three different datasets. The result shows that both informativeness and sentence quality of our model outperform the standard seq2seq models.

The rest of this chapter is organized as follows. Section 3.2 surveys the previous work in sentence summarization. Section 3.3 describes the proposed method. Experiments are presented in Section 3.4. Finally, we summarize this chapter in Section 3.5.

3.2 Related Work

The seq2seq model is a newly emerging approach. It was initially proposed by [Kalchbrenner and Blunsom, 2013; Cho *et al.*, 2014a; Sutskever *et al.*, 2014] as NMT. Compared with the SMT approaches (e.g., [Koehn *et al.*, 2007]), seq2seq models require less human efforts. Later, [Bahdanau *et al.*, 2014] developed the attention mechanism which largely promoted the applications of seq2seq models. In addition to machine translation, seq2seq models achieved competitive performance in many other applications such as response generation [Shang *et al.*, 2015] Some researches (e.g., [Rush *et al.*, 2015b; Hu *et al.*, 2015b]) have directly applied the general seq2seq model [Bahdanau *et al.*, 2014] to abstractive sentence summarization. However, the experiments of [Rush *et al.*, 2015b] demonstrated that the introduction of hand-crafted features significantly improved the performance of the original model. Consequently, the general seq2seq model used for machine translation seemed not suitable for summarization, which involves both copying and rewriting.

Limited work has explored the copying mechanism. [Vinyals *et al.*, 2015] proposed a pointer mechanism to predict the output sequence directly from the input. In addition to the different applications, their model cannot generate items outside of the set of input sequence. Later, [Allamanis *et al.*, 2016] developed a convolutional attention network to generate the function name of the source code. Since there are many out-of-vocabulary words, they used another attention model in the decoder to directly copy a source code token. In seq2seq generation, the most relevant work we find is COPYNET [Gu *et al.*, 2016], which has been explained in the introduction.

Some existing work has tried to modify the output dimension of the decoder to speed up the training process. In training, [Cho *et al.*, 2015] restricted the decoder to generate the words from the actual target words together with a sampled word set. [Nallapati *et al.*, 2016b] supplemented the 1-nearest-neighbors of source words, as measured through the similarity in the word embedding space. Notice that, these models still decoded on the full vocabulary during test. In comparison, our rewriting decoder always produces the words in a small yet highly relevant vocabulary.

3.3 Method

3.3.1 Background

Seq2seq models have been extensively applied to a series of natural language generation tasks, including machine translation [Bahdanau *et al.*, 2014], response generation [Shang *et al.*, 2015] as well as abstractive summarization [Rush *et al.*, 2015b]. With these models, the source sequence $X = [x_1, \cdots, x_n]$ is converted into a context vector $\mathbf{c}$. The common practice is to adopt a Recurrent Neural Network (RNN) encoder:

$$\mathbf{h}_\tau = f(x_\tau, \mathbf{h}_{\tau-1}) \tag{3.1}$$

$$\mathbf{c} = \phi(\mathbf{h}_1, \cdots \mathbf{h}_n) \tag{3.2}$$

where $\{\mathbf{h}_\tau\}$ are the RNN states, f is the recurrent function, and ϕ is used to summarize the hidden states. For example, some early work simply pick up the last state $\mathbf{h}_n$.

The decoder unfolds the context vector $\mathbf{c}$ into the target RNN state $\mathbf{s}_t$ through the similar dynamics in the encoder:

$$\mathbf{s}_t = f(y_{t-1}, \mathbf{s}_{t-1}, \mathbf{c}) \tag{3.3}$$

Then, the predictor is followed to generate the final sequence, usually using a softmax classifier:

$$p(y_t | y_{<t}, \mathbf{X}) = \frac{\exp(\psi(y_{t-1}, \mathbf{s}_t, \mathbf{c}_t)\mathbf{w}_t)}{\sum_{y_{t'} \in \mathbf{V}} \exp(\psi(y_{t-1}, \mathbf{s}_t, \mathbf{c}_t)\mathbf{w}_{t'})} \tag{3.4}$$

where y_t is the predicted target word at the state t, $\mathbf{w}_t$ is the corresponding weight vector, and ψ is an affine transformation. $\mathbf{V}$ is the target vocabulary, and it is usually as large as 10^4 or even 10^5.

To release the burden of summarizing the entire source into a single context vector, the attention mechanism [Bahdanau *et al.*, 2014] uses a dynamic context $\mathbf{c}_t$ to replace

$\mathbf{c}$ in Equation 3.3. A common practice is to use the weighted sum of $\{\mathbf{h}_\tau\}$ to compute the context vector:

$$\alpha_{t\tau} = \frac{e^{\eta(\mathbf{s}_{t-1},\mathbf{h}_\tau)}}{\sum_{\tau'=1}^{n} e^{\eta(\mathbf{s}_{t-1},\mathbf{h}_{\tau'})}} \quad, \forall\tau \in [1, n] \tag{3.5}$$

$$\mathbf{c}_t = \sum\nolimits_{\tau=1}^{n} \alpha_{t\tau}\mathbf{h}_\tau \tag{3.6}$$

where $\alpha_{t\tau}$ reflects the alignments between source and target words, and η denotes the function to measure the correspondence weight of the attention.

3.3.2 Overview of CoRe

As illustrated in Figure 3.1, CoRe is an advanced seq2seq framework. The source sequence is transformed by a RNN **Encoder** into the context representation. Then we develop two **Decoders** to model copying and rewriting, respectively.

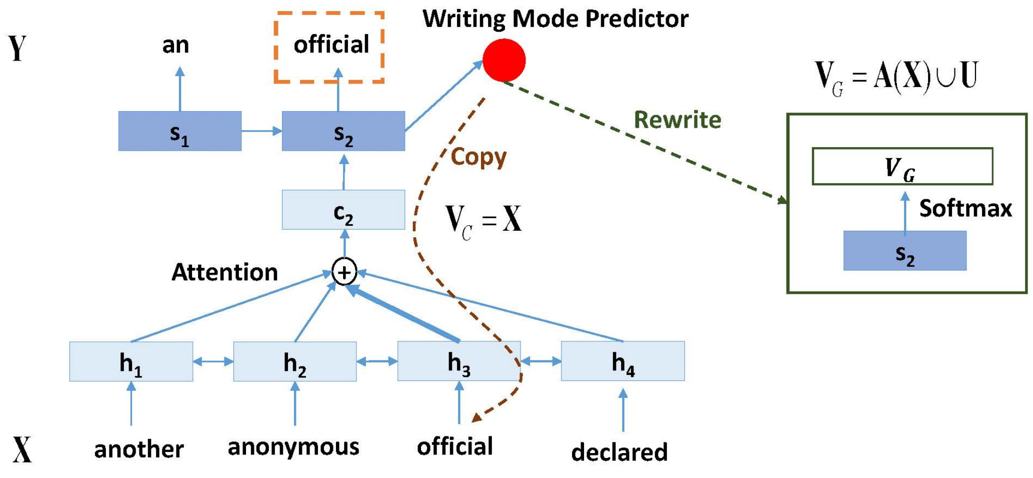


Figure 3.1: Overview of CoRe.

Encoder

We follow the work of [Bahdanau *et al.*, 2014] to build the encoder. Specifically, the bi-directional RNN (BiRNN) is introduced to make the hidden state $\mathbf{h}_\tau$ aware of the entire context from both sides. Given a source word x_τ, its hidden state of the forward

RNN can be represented by:

$$\overrightarrow{\mathbf{h}}_\tau = \mathrm{RNN}(x_\tau, \overrightarrow{\mathbf{h}}_{\tau-1}) \tag{3.7}$$

Here we use the Gated Recurrent Unit (GRU) [Cho *et al.*, 2014b] as the recurrent unit, which often performs much better than the vanilla RNN. The detailed computation of GRU is as follows:

$$\mathbf{z}_\tau = \sigma([\overrightarrow{\mathbf{h}}_{\tau-1}, \mathbf{x}_\tau]\mathbf{W}_z) \tag{3.8}$$

$$\mathbf{r}_\tau = \sigma([\overrightarrow{\mathbf{h}}_{\tau-1}, \mathbf{x}_\tau]\mathbf{W}_r) \tag{3.9}$$

$$\mathbf{l}_\tau = \tanh([\mathbf{r}_\tau \odot \overrightarrow{\mathbf{h}}_{\tau-1}, \mathbf{x}_\tau]\mathbf{W}_l) \tag{3.10}$$

$$\overrightarrow{\mathbf{h}}_\tau = (\mathbf{1} - \mathbf{z}_\tau) \odot \overrightarrow{\mathbf{h}}_{\tau-1} + \mathbf{z}_\tau \odot \mathbf{l}_\tau \tag{3.11}$$

where $\mathbf{z}_\tau$, $\mathbf{r}_\tau$ are two gates, $\mathbf{W}_.$ stands for model weights and $\odot$ means element-wise multiplication. We use bold font $\mathbf{x}_\tau$ to represent the word embeddings of the word x_τ. The BiRNN reads text both forward and backward. Suppose the corresponding outputs are $\{\overrightarrow{\mathbf{h}}_\tau\}$ and $\{\overleftarrow{\mathbf{h}}_\tau\}$, respectively. Then we concatenate two RNN representations to finally represent a word, i.e., $\mathbf{h}_\tau = [\overrightarrow{\mathbf{h}}_\tau; \overleftarrow{\mathbf{h}}_\tau]$.

Then, according to Equation 3.2, we apply the attention mechanism to compute the context vector. Here we adopt the common form of η as [Luong *et al.*, 2015]:

$$\eta(\mathbf{s}_{t-1}, \mathbf{h}_\tau) = \mathbf{v}_a \tanh(\mathbf{s}_{t-1}\mathbf{W}_a + \mathbf{h}_\tau \mathbf{U}_a)^\top \tag{3.12}$$

where $\mathbf{v}_a$, $\mathbf{W}_a$ and $\mathbf{U}_a$ are model parameters. However, unlike most previous work, we re-use the alignment learned by Equation 3.5 in the decoders.

Decoder

Instead of using the canonical RNN-decoder like [Bahdanau *et al.*, 2014], we develop two distinct decoders to simulate the copying and rewriting behaviors, respectively.

The **Copying Decoder** (C) picks the words from the source text. In summarization, most keywords from the original document will be reserved in the output. This decoder reflects this fact. Since the attention mechanism is supposed to provide the focus of source text during generation, for the copying behavior, the weights learned by Equation 3.5 can be interpreted as the copying probability distribution. Thus, the output of the copying decoder is:

$$p_C(y_t|y_{<t}, \mathbf{X}) = \begin{cases} \alpha_{t\tau}, & \text{if } y_t = x_\tau \\ 0, & \text{otherwise} \end{cases} \tag{3.13}$$

Most previous work uses the attention mechanism as a module to build the context vector. In contrast, our copying decoder provides the explicit meanings (i.e., the copying probability distribution) to the learned alignment. Notice that, this decoder only generates words in the source, i.e., the vocabulary for this decoder is $\mathbf{V}_C = \mathbf{X}$. We observe that quite a number of low-frequency words in the actual target text are extracted from the source text. Hence, the copying decoder largely reduces the chance to produce the unknown (UNK) tags.

The **Rewriting Decoder** (G), on the other hand, restricts the output in a small yet highly relevant vocabulary according to the source text. At first, we use a GRU decoder like [Bahdanau *et al.*, 2014] to compute the current generation state $\mathbf{s}_t$, that is:

$$\mathbf{z}_t = \sigma([\mathbf{s}_{t-1}, \mathbf{y}_{t-1}, \mathbf{c}_t]\mathbf{U}_z) \tag{3.14}$$

$$\mathbf{r}_t = \sigma([\mathbf{s}_{t-1}, \mathbf{y}_{t-1}, \mathbf{c}_t]\mathbf{U}_r) \tag{3.15}$$

$$\mathbf{l}_t = \tanh([\mathbf{r}_t \odot \mathbf{s}_{t-1}, \mathbf{y}_{t-1}, \mathbf{c}_t]\mathbf{U}_l) \tag{3.16}$$

$$\mathbf{s}_t = (\mathbf{1} - \mathbf{z}_t) \odot \mathbf{s}_{t-1} + \mathbf{z}_t \odot \mathbf{l}_t \tag{3.17}$$

where $\mathbf{U}_.$ denotes model parameters. The output of this decoder is formulated by Equation 3.18, which is similar to Equation 3.4, except for the reduced vocabulary $\mathbf{V}_G$:

$$p_G(y_t|y_{<t}, \mathbf{X}) = \frac{\exp(\psi(y_{t-1}, \mathbf{s}_t, \mathbf{c}_t)\mathbf{w}_t)}{\sum_{y_{t'} \in \mathbf{V}_G} \exp(\psi(y_{t-1}, \mathbf{s}_t, \mathbf{c}_t)\mathbf{w}_{t'})} \tag{3.18}$$

To obtain $\mathbf{V}_G$, we train a rough alignment table $\mathbf{A}$ based on the IBM Model [Dyer *et al.*, 2013] beforehand. This table is able to capture many representative rewriting patterns, such as "sustain $\rightarrow$ injury", and "seabird $\rightarrow$ wildlife". In addition, during training, we utilize the learned attentions to supplement more rewriting alignments. Specifically, for each target word y_t, if it is not copied, we find out the word with the maximal attention weight as its alignment. To formulate, the computation is:

$$\mathrm{A}(y_t) = \arg\max_{\tau}(\alpha_{t\tau}), \quad \forall y_t \notin \mathbf{X} \tag{3.19}$$

Our pilot experimental results show that the alignment table covers most target words which are not extracted from the source. To further increase the coverage, we introduce an additional frequent word table $\mathbf{U}$[1]. Putting together, the final vocabulary for this decoder is limited to:

$$\mathbf{V}_G = \mathbf{A}(\mathbf{X}) \cup \mathbf{U} \tag{3.20}$$

Compared with generation with the large vocabulary $\mathbf{V}$, the restricted decoder not only runs much faster but also produces more relevant words.

To combine the two decoders, we introduce a binary sequence labeling task to decide whether the current target word should come from copying or rewriting. Specifically, for each hidden state $\mathbf{s}_t$, we compute a predictor λ_t to represent the probability of copying at the current generation position:

$$\lambda_t = \sigma(\mathbf{s}_t \mathbf{w}_C) \tag{3.21}$$

where σ represents the sigmoid function and $\mathbf{w}_C$ is the model parameters. λ_t measures the contributions of the two decoders. The final combined prediction probability is:

$$p(y_t|y_{<t}, \mathbf{X}) = \lambda_t p_C + (1 - \lambda_t) p_G \tag{3.22}$$

It is noted that λ_t has the following actual supervision in the training set:

$$\lambda_t^* = \begin{cases} 1, & \text{if target word at } t \text{ exists in the source} \\ 0, & \text{otherwise} \end{cases} \tag{3.23}$$

Therefore, we can utilize this supervision to guide the writing mode prediction.

The common canonical RNN-decoder outputs the probability distribution over the reserved target word vocabulary $\mathbf{V}$. Since the computation complex of a seq2seq model

[1]We add the UNK tag in this table

is linear to the output dimension of the decoder, a large amount of infrequent target words have to be discarded to ensure a reasonable vocabulary size. As a result, a target sentence may contain many UNK tags, and thus unreadable. By contrast, the output dimension of our rewriting decoder is totally independent on the preserved target word vocabulary. Therefore, we opt to reserve all the training target words. Experiments demonstrate that our model runs efficiently and well covers the actual target words.

3.3.3 Learning

The cost function ϵ in our model is the sum of two parts, i.e.,

$$\epsilon = \epsilon_1 + \epsilon_2 \tag{3.24}$$

The first one ϵ_1 is the difference between the output $\{y_t\}$ and the actual target sequence $\{y_t^*\}$. As the common practice, we use Negative Log-Likelihood (NLL) to represent the generation error:

$$\varepsilon_1 = -\sum_t \ln(p(y_t^*|y_{<t}, \mathbf{X})) \tag{3.25}$$

In most existing seq2seq models, ϵ_1 is the final cost function. However, in our model, we include another cost function ϵ_2 derived from the prediction of writing modes. As shown in Equation 3.21, a binary sequence labeling process in our model predicts whether or not the current target word is copied. ϵ_2 measures the performance of this task,

$$\varepsilon_2 = -(\sum_t (\lambda_t^* \ln(\lambda_t) + (1 - \lambda_t^*) \ln(1 - \lambda_t))) \tag{3.26}$$

ϵ_2 utilizes additional supervision of the training data. The experiments show that this cost function accurately balances the proportion of the words derived from copying and generation.

3.4 Experiment

3.4.1 Datasets

We test CoRe on various datasets, including

Gigaword Gigaword Headline Generation

CNN/DM CNN/Daily Mail Highlight Generation

Wikipedia Wikipedia Text simplification

The first two are sentence summarization datasets, while the last one focuses on sentence simplification. Text simplification modifies a document in such a way that the grammar and vocabulary is greatly simplified, while the underlying meaning remains the same. It is able to make the scientific documents easily understandable for outsiders. This task is analogous to text summarization where copying and rewriting are also dominant writing modes. All the raw corpora are publicly available.

Gigaword: We conduct experiments on the Annotated English Gigaword dataset, as with [Rush *et al.*, 2015b]. This parallel dataset is produced by pairing the first sentence in the news article and its headline as the summary with heuristic rules. The whole dataset is publicly available[2]. This dataset has already been tokenized, lowercased and masked rare words with an out-of-vocabulary (OOV) tag. We replace the OOV tag "<unk>" with "UNK" in the training and validation datasets, which accords with the practice in the test set.

CNN/DM: We modify a machine reading comprehension [Hermann *et al.*, 2015] corpus to get the <document,highlight> pairs. In this dataset, a collection of news documents and the corresponding highlights are downloaded from CNN and Daily Mail websites. For each highlight, we reserve the original sentences that have at least one

[2]`https://github.com/harvardnlp/sent-summary`

Statistics	Gigaword	CNN/DM	Wikipedia
Training#	3.8M	987k	132k
Validation#	189k	52k	7k
Test#	1951	42k	3393
Source Length	31.4	71.0	24.3
Target Length	8.3	12.6	20.9
Vocab Size	124k	116k	123k

Table 3.1: Statistics of the three datasets.

word overlap with the source text. Therefore, if a document holds multiple highlights, the source text for each highlight can be different.

Wikipedia: Simple English Wikipedia[3] articles represent a simplified version of traditional English Wikipedia articles. [Kauchak, 2013] built a <Wikipedia text, Simple English Wikipedia text> corpus according to the aligned articles. We eliminate the non-English words in the corpus, and remove the pairs where the source and the target are exactly the same.

The overall information of the experimental datasets is shown in Table 3.1. As can be seen, each dataset has a large vocabulary size. In the summarization datasets, the target length is much shorter than the source length, while in the simplification dataset, their lengths are similar.

In addition, we compute the target word coverage ratio based on different vocabulary sets, as shown in Table 3.2. For CNN/DM and Wikipedia, it appears that both datasets hold a high copying ratio. When we restrict the rewriting decoder to produce the source alignments, more than 85% target words can be covered. When combined with 2000 frequent words, the coverage ratio of our model is already close to that using the vocabulary of 30000 words. For Gigaword, due to its relatively high abstraction, the alignment table can only cover 80% target words. Therefore, we introduce 3000 frequent words.

[3]`http://simple.wikipedia.org`

Vocabulary	Gigaword	CNN/DM	Wikipedia
$\mathbf{X}$	58.4	79.2	78.1
$\mathbf{X} \cup \mathbf{A}(\mathbf{X})$	79.5	89.2	85.8
$\mathbf{X} \cup \mathbf{A}(\mathbf{X}) \cup \mathbf{U}$	92.4	95.3	96.0
$\lvert\mathbf{V}\rvert = 30000$	96.5	96.3	95.4

Table 3.2: Target word coverage ratio (%) on the test set.

3.4.2 Evaluation Metrics

Informativeness is evaluated using ROUGE[4] [Lin, 2004], which has been regarded as a standard automatic summarization evaluation metric. ROUGE counts the overlapping units between the candidate text $\mathbf{Y}$ and the actual target text $\mathbf{T}$ to evaluate the quality of $\mathbf{Y}$. As the common practice, we report ROUGE-1 (uni-gram), ROUGE-2 (bi-gram) and ROUGE-L (longest common subsequence) scores. Since we do not try to control the length of the generated sentences, we use the f-score rather than the recall for comparison. The detailed formulas are presented as follows:

$$\mathrm{ROUGE}-1_{\text{f - score}} = \frac{2 \times \sum_{u \in \mathbf{Y}} \min\{N_{\mathbf{Y}}(u), N_{\mathbf{T}}(u)\}}{\sum_{u \in \mathbf{Y}} N_{\mathbf{Y}}(u) + \sum_{b \in \mathbf{T}} N_{\mathbf{T}}(u)} \tag{3.27}$$

$$\mathrm{ROUGE}-2_{\text{f - score}} = \frac{2 \times \sum_{b \in \mathbf{Y}} \min\{N_{\mathbf{Y}}(b), N_{\mathbf{T}}(b)\}}{\sum_{b \in \mathbf{Y}} N_{\mathbf{Y}}(b) + \sum_{b \in \mathbf{T}} N_{\mathbf{T}}(b)} \tag{3.28}$$

$$\mathrm{ROUGE}-\mathrm{L}_{\text{f - score}} = \frac{2 \times LCS(\mathbf{Y}, \mathbf{T})}{|\mathbf{Y}| + |\mathbf{T}|} \tag{3.29}$$

where u, b denote a uni-gram or bi-gram, $\mathbf{Y}$ is a generated summary and $\mathbf{T}$ means the actual summary, $N(.)$ is the number of an item, and LCS is the length of a longest common subsequence.

3.4.3 Model Settings

Turned on the validation dataset, we choose 256 as word embedding dimension and 512 as the RNN dimension. The initial learning rate is 0.05 and the batch size is 32. Our implementation is on the basis of the standard attentional seq2seq framework dl4mt[5]

[4]ROUGE-1.5.5 with the following parameters: -p 0.5 -t 0 -n 2 -x -m -u -c 95 -r 1000 -f A.
[5]`https://github.com/nyu-dl/dl4mt-multi`

under the Theano framework[6]. We use the RmsProp [Tieleman and Hinton, 2012] optimizer with mini-batches to tune the model weights. RmsProp is a popular method to train recurrent neural networks. We leverage the popular tool Fast Align [Dyer *et al.*, 2013] to construct the initial source-target word alignment table $\mathbf{A}$. Then, this table is automatically updated according to the learned attentions. The vocabulary of our rewriting decoder is restricted to the top 10 alignments of each source word plus 2000 frequent words (3000 for Gigaword). Although our model is more complex than the standard attentional seq2seq model, it only spends about three quarters of the time in training. To be consistent with Chapter 5, for Gigaword, we also test a recently published seq2seq implementation called OpenNMT[7] which is built on the deep learning framework Pytorch[8]. All the settings are the same as Chapter 5. Thus, this practice also provides chances to compare the effect of deep learning frameworks. It is noted that copying mechanism has already been equipped in OpenNMT. To distinguish the results from dl4mt and OpenNMT, we add suffixes "D" and "O", respectively.

3.4.4 Baselines

We compare the proposed model CoRe with various typical methods. At first, we introduce the standard baseline called "LEAD" that simply outputs the "leading" words of the source text. According to the averaged target length in Table 3.1, we choose the first 20 words for CNN/Daily Mail, 25 for Wikipedia and 8 for Gigaword. In addition, since statistical machine translation (SMT) has been used in sentence summarization (e.g., [Banko *et al.*, 2000]) and we adopt a popular SMT aligner Fast Align, we introduce the prevailing SMT platform Moses [Koehn *et al.*, 2007] as the baseline. For fair comparison, when implementing Moses, we also employ the alignment tool Fast Align as its aligner. We implement the standard attentional seq2seq model s2s-att [Bahdanau *et al.*, 2014] with dl4mt. Note that, we would like to but fail to take COPYNET [Gu *et*

[6]`http://deeplearning.net/software/theano/`
[7]`http://opennmt.net/`
[8]`https://pytorch.org/`

al., 2016] into comparison. Its source code is not publicly available.

Since Gigaword is a widely studied dataset, we introduce the following published results cited from corresponding papers:

ABS The first neural abstractive summarization system proposed by [Rush *et al.*, 2015a].

ABS+ [Rush *et al.*, 2015a] further tuned the ABS model with additional features to balance the abstractive and extractive tendency.

RAS-Elman As the extension of the ABS model, it changed the decoder with RNN [Chopra *et al.*, 2016].

Featseq2seq [Nallapati *et al.*, 2016a] used a full seq2seq model and added the handcrafted features such as POS tag and NER, to strengthen the encoder representation.

Luong-NMT [Chopra *et al.*, 2016] implemented the neural machine translation model of [Luong *et al.*, 2015] for summarization. This model contained two-layer LSTMs for both encoder and decoder.

s2s-att$_O$ We also implement the standard attentional seq2seq model with OpenNMT. All the settings are the same as our system.

3.4.5 Performance on CNN/DM and Wikipedia

The ROUGE performance is shown in Table 3.3. As can be seen, CoRe achieves the highest ROUGE scores on both datasets. In contrast, the standard attentional seq2seq model s2s-att is slightly inferior to Moses. It even performs worse than the simple baseline LEAD in terms of ROUGE-2 in CNN/DM. Apparently, introducing the copying and restricted generation mechanisms is critical for summarization.

Then, we check the text quality of the generation result. We learn a 3-gram Language Model (LM) on the entire target datasets using the tool SRILM [Stolcke *et al.*,

Data	Model	ROUGE-1(%)	ROUGE-2(%)	ROUGE-L(%)
CNN/DM	LEAD	28.13	14.12	25.82
	Moses	27.81	14.15	24.91
	s2s-att	28.16	12.42	26.50
	CoRe	**30.51**	**16.27**	**28.61**
Wikipedia	LEAD	66.42	49.47	62.45
	Moses	70.95	52.14	67.56
	s2s-att	68.46	50.33	66.12
	CoRe	**72.73**	**55.30**	**69.23**

Table 3.3: ROUGE performance on CNN/DM and Wikipedia.

Data	Model	PPL	LENGTH	UNK(%)	COPY(%)
CNN/DM	LEAD	176	19.9	0	100
	Moses	214	73.0	0*	99.6
	s2s-att	113	13.7	0.88	92.0
	CoRe	**95**	14.0	0.14	88.6
Wikipedia	LEAD	66.5	20.8	0	100
	Moses	70.3	24.4	0*	97.6
	s2s-att	69.5	22.7	5.6	87.7
	CoRe	**60.9**	19.6	2.3	85.9

Table 3.4: Text quality performance on CNN/DM and Wikipedia. *Moses simply ignore the unknown words.

2011], and calculate the perplexity (PPL) of the generated text. The lower PPL usually means higher readability. We also perform the statistical analysis on the mean length of the output text, the UNK ratio and the copy ratio. We assume that the good sentences ought to have the similar length and copying ratio to the answers. As shown in Table 3.4, according to PPL, the sentences produced by CoRe resemble the target language the most. It is interesting that LEAD extracts human-written text in the source. Nevertheless, its PPL is considerably higher than CoRe on both datasets. It seems that CoRe indeed captures some characteristics of the target language, such as diction. We also find that the PPL of Moses is the largest, and its generated length reaches the length of the source text. Moses seems to conduct word-to-word translation. This practice totally violates the summarization requirement. Although not manually controlled, the lengths of the outputs in s2s-att and CoRe are both similar to the actual one, which demonstrates the learning ability of seq2seq models. In addition, Table 3.4 shows that

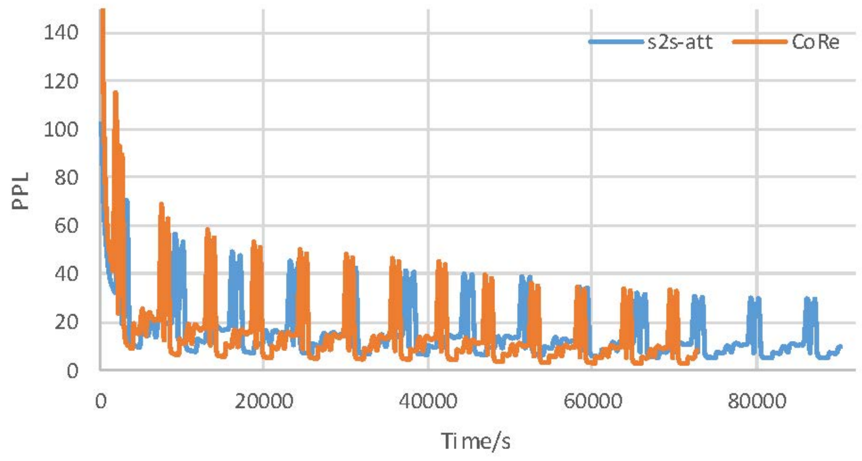


Figure 3.2: PPL changes during training.

compared to s2s-att, CoRe generates far fewer UNK tags and its copying ratio is closer to the actual one. The former verifies the power of our two decoders, while the latter may be attributed to the supplement of the supervision of writing modes.

3.4.6 Performance on Gigaword

Model	ROUGE-1(%)	ROUGE-2(%)	ROUGE-L(%)
LEAD	22.37	7.90	21.12
Moses	28.90	12.12	26.56
ABS†	29.55	11.32	26.42
ABS+†	29.78	11.89	26.97
Featseq2seq†	32.67	15.59	30.64
RAS-Elman†	33.78	15.97	31.15
Luong-NMT†	33.10	14.45	30.71
s2s-att$_D$	34.23	15.52	31.57
s2s-att$_O$	35.01	16.55	32.42
CoRe$_D$	36.22	16.54	33.25
CoRe$_O$	**36.72**	**17.12**	**33.37**

Table 3.5: ROUGE performance on Gigaword. We use † to indicate the results from citation.

Now let us look at the results on the Gigaword dataset. From Table 3.5, we observe that CoRe achieves larger ROUGE values than all the other approaches. Note that ABS+ and Featseq2seq have utilized a series of hand-crafted features, but our model is

Model	Copy		Rewrite		
	COPY(%)	RIGHT$_C$	RIGHT$_R$	NEW_NE	NEW_UP
s2s-att	80	42	26	0.34	0.19
CoRe	74	48	25	0.26	0

Table 3.6: Copying and rewriting performance on Gigaword.

entirely data-driven. Even though, our model surpasses Featseq2seq by 10% and ABS+ by 44% in terms of ROUGE-2. We also observe that OpenNMT always works better than dl4mt. Therefore, in the following part we only display the results of OpenNMT.

Next, we compare the training processes of CoRe$_O$ and s2s-att$_O$. The result is shown in Figure 3.2. Obviously, PPL of CoRe drops faster than s2s-att, and the entire training process of CoRe is shorter. It is interesting that PPL surges at the beginning of each epoch. Maybe it is caused by the strategy of the learning rate decay.

Finally, we use series of metrics to evaluate the detailed copying and rewriting performance, as described below:

COPY The copying ratio mentioned previously.

RIGHT$_C$ The proportion of the copied words which also appear in actual summaries.

RIGHT$_R$ The proportion of the rewritten words which also appear in actual summaries.

NEW_NE The number of the named entities that appear in neither the source sentence nor the actual summary. Intuitively, the appearance of new named entities in the summary is likely to bring unfaithfulness. We use Stanford CoreNLP [Manning *et al.*, 2014] to recognize named entities.

NEW_UP The number of the uppercase words that appear in neither the source sentence nor the actual summary. We use the "trucase" annotator of CoreNLP to find these words. An uppercase word usually represent a named entity or some adjectives like English or Chinese. The appearance of a new uppercase word in the summary is improper in most cases.

The results are presented in Table 3.6. Similar to Table 3.4, our model contains less coying words, but its correct copying proportion increases 6%, verifying the effect of explicit application of the attentions for copying. Although the correctness in rewriting is almost the same, we can see that CoRe generates less new named entities and blocks the appearance of new uppercase words. Thus we believe that rewriting modeling benefits the improvement of summary faithfulness to some extent. This concept will be illustrated in details in the next chapter.

We also investigate the n-gram level copying behaviors of different systems. As shown in Figure 3.3, CoRe makes more accurate prediction than s2s-att in all n-gram copying except tri-grams. It largely verifies the effectiveness of the proposed method. In addition, we analyze the proportion of lengths in sequence copying. The results of actual summaries, CoRe and s2s-att are displayed in Figure 3.4, Figure 3.5 and Figure 3.6, respectively. As can been seen, long-sequence copying accounts for a half in actually summaries. However, it is seriously biased in the seq2seq framework. For example, the copying of a 5-gram or more is quite rare (5%) in actual summaries. By contrast, this number is 10% in our model and 12% in s2s-att. Therefore, it requires a mechanism to alleviate the long-sequence copying tendency in seq2seq. We leave it as our future work.

3.4.7 Case Study

We further manually inspect what our model actually generates. We observe that the paraphrase rules of Simple Wikipedia are relatively fixed. For example, no matter how the article in Wikipedia illustrates, Simple Wikipedia usually adopts the following pattern to describe a commune:

$\boxed{\text{#NAME is a commune . it is found in #LOCATION .}}$

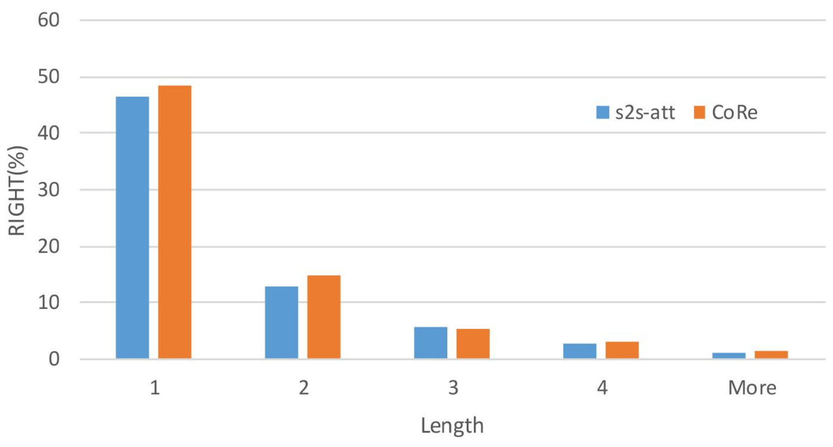


Figure 3.3: Correct copying in different sequence lengths.

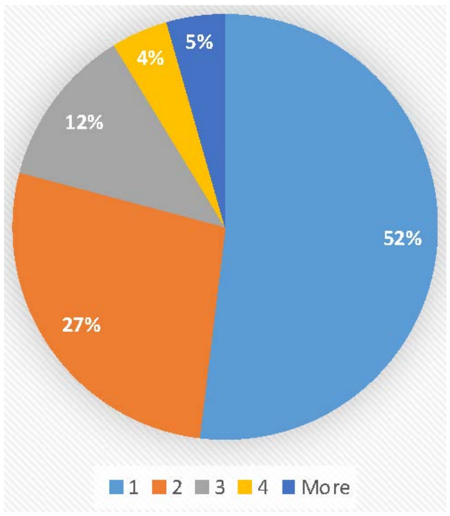


Figure 3.4: Proportion of lengths of sequence copying in actual summaries.

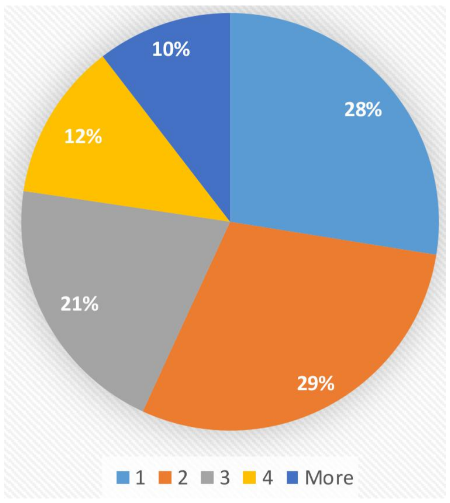


Figure 3.5: Proportion of lengths of sequence copying in CoRe.

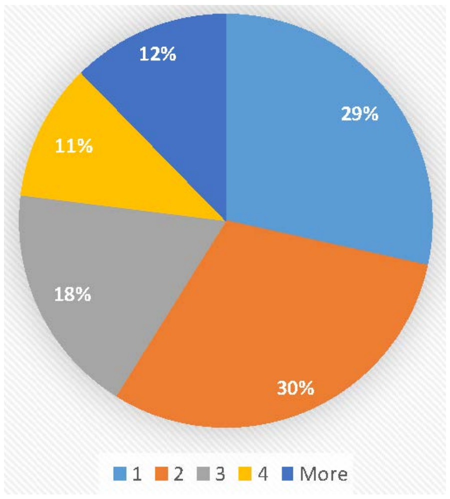


Figure 3.6: Proportion of lengths of sequence copying in s2s-att.

Source	another @entity34 military official who spoke on the condition of anonymity told @entity3 that the fall of @entity11 is not imminent
Target	a @entity34 military official tells @entity3 the fall of @entity11 is not imminent
s2s-att	the @entity34 military official spoke on the condition of anonymity
CoRe	a @entity34 military official said the fall of @entity11 is not imminent

Figure 3.7: Generation example in CNN/DM. We use colors to distinguish the word source, i.e., copying , alignment or common words .

Source	german chemical giant hoechst group announced plans wednesday to invest ### million dollars in china next year , with the idea of getting a strong foothold in the fast growing economy , xinhua news agency reported .
Target	hoechst to invest total ### million us dollars in chinese operation
s2s-att	UNK to invest ### million dollars in china
CoRe	hoechst to put ### million us dollars in chinese operation
Source	india won the toss and chose to bat on the opening day in the opening test against west indies at the antigua recreation ground on friday .
Target	india win toss and elect to bat in first test
s2s-att	india wins toss bats in opening test
CoRe	india win toss and elect to bat in opening test

Figure 3.8: Generation examples in Gigaword.

CoRe grasps many frequent paraphrase rules, and there are more than 130 cases where the generation results of CoRe exactly hit the actual target sentences. Therefore, we focus more on the analysis of the summarization results next.

In summarization, although most target words come from the copying decoder, we find CoRe tends to pick keywords from different parts of the source document. By contrast, the standard attentional seq2seq model often extracts a large part of continuous source words. Meanwhile, the restricted generation decoder plays the role to "connect" these keywords, such as to change the tenses, or to supplement article words. Figure 3.7 and Figure 3.8 present some generation examples on CNN/DM and Gigaword. We find that the sentence generated by CoRe is fluent and satisfies the need of summarization. On CNN/DM, the only difference from the actual summary is that CoRe does not assume "told @entity3" is important enough and simplifies it to "said". It also accords with the common sense. Notably, CoRe changes the starting word from "another" to "a", which is actually more preferred for an independent highlight. Looking at other models, Moses almost repeats the content of the source text. As a result, the summary generated by Moses is the longest one and fails to catch the main idea. The attentional seq2seq framework indeed compresses the source text. It however focuses on the wrong place, i.e., the attributive clause. Therefore, its output does not even form a complete sentence. Moreover, in the first case on Gigaword, s2s-att wrongly changes “hoechst” to UNK, which is avoided in our rewriting module.

3.5 Summary

In this chapter, we present a novel seq2seq model called CoRe to simulate the two core writing modes in summarization, i.e., copying and rewriting. CoRe fuses a copying decoder and a rewriting decoder. The copying decoder finds the position to be copied based on the existing attention mechanism. The rewriting decoder produces words limited in the source-specific vocabulary. To combine the two decoders and determine the final output, we train a predictor to predict the writing modes. Experiments on three datasets demonstrate that our model outperforms the state-of-the-art approaches in terms of both informativeness and language quality. Our model has board application prospects. For example, it currently focuses on producing a single sentence. We plan to extend it to generate multi-sentence documents. Meanwhile, we are going to develop mechanisms to model the frequent long-sequence copying.

Chapter 4

Copy Facts in Source Text

Word of truth can be heavier than the whole world.

— **Aleksandr Isayevich Solzhenitsyn**
(Novelist)

4.1 Introduction

Abstractive sentence summarization [Rush *et al.*, 2015a] focuses on shortening a given sentence while keeping its main idea. This task is different from document-level summarization since it is hard to apply the common extractive techniques [Over and Yen, 2004]. That is, selecting existing sentences to form the sentence summary is impossible. Recently, the application of the attentional sequence-to-sequence (seq2seq) framework has attracted growing attention in this area [Rush *et al.*, 2015a; Chopra *et al.*, 2016; Nallapati *et al.*, 2016a]. To evaluate the performance of a summarization system, by far the most extensively applied automatic evaluation tool is ROUGE [Lin, 2004].

As we know, sentence summarization inevitably needs to fuse different parts of the source sentence and is abstractive in nature. Consequently, the fused summaries often misrepresent the original meaning and yield fake facts. Look at an illustrative example of the generation result using the state-of-the-art seq2seq model [Nallapati *et al.*, 2016a]

Source	the repatriation of at least #,### bosnian moslems was postponed friday after the unhcr pulled out of the first joint scheme to return refugees to their homes in northwest bosnia .
Target	**repatriation** of bosnian moslems postponed
seq2seq	**bosnian moslems** postponed after unhcr pulled out of **bosnia**

Table 4.1: An example of fake summaries generated by the state-of-the-art seq2seq model. "#" stands for a digit masked during preprocessing.

in Table 4.1. The actual subject of the verb "postponed" is "repatriation". Nevertheless, probably because the entity "bosnian moslems" is closer to "postponed" in the source sentence, the summarization system wrongly regards "bosnian moslems" as the subject and counterfeits a fact "bosnian moslems postponed". In addition, the seq2seq system generates another fake fact: "unhcr pulled out of bosnia" and puts it into the summary. As a result, although the informativeness and readability of this summary are high, its meaning departs far from the original.

Our preliminary study reveals that nearly one third of the outputs from the state-of-the-art seq2seq system suffer from this problem. While previous researches are mainly devoted to increasing summary informativeness, we argue that, it is one of the most essential prerequisites for a practical abstractive summarization system that its generated summaries must accord with the facts expressed in the source. We refer to this aspect as **summary faithfulness**. A fake summary may greatly misguide the comprehension of the original text.

To evaluate the system performance, ROUGE counts the overlapping lexical units (e.g., n-grams), which mainly reflects informativeness. It is not originally defined to evaluate faithfulness. Our experiments (refer to Fig 4.6) show that although the summaries with very large ROUGE values are usually faithful, it is hard to judge the faithfulness of a summary with relatively low ROUGE scores. Unfortunately, the average performance of the state-of-the-art seq2seq summarization models is located in the indistinguishable area. As such, ROUGE scores fail to indicate the level of faithfulness.

In this chapter, we aim to handle both the above issues. Specifically, we propose a faithful summarization system and a faithfulness evaluation tool, both based on the import of the facts in the source sentence. To this end, the first process is to extract the facts mentioned in the source sentence. In the relatively mature task of Open Information Extraction (OpenIE) [Banko *et al.*, 2007], a fact is usually represented by a relation triple consisting of (subject; predicate; object). However, the relation triples are not always extractable, e.g., from the imperative sentences. Hence, we further adopt a dependency parser to supplement with the (subject; predicate) and (predicate; object) tuples identified from the parse tree of the sentence. This is inspired by the work of parse tree based sentence compression (e.g., [Clarke and Lapata, 2008]). Using both source sentence and facts as input, we extend the standard attentional seq2seq model [Nallapati *et al.*, 2016a] to fully leverage their information. Since our summarization system encodes **FacTs** to enhance **FaiThfulness**, we call it **FTSum**. On the other hand, we develop a tool called **FTEval** to automatically evaluate faithfulness by matching facts between a summary and its source sentence.

We test our summarization system and evaluation tool extensively on a popular sentence summarization dataset. The results show that FTSum greatly reduces the fake summaries by 55% with respect to the state-of-the-art seq2seq framework. Due to the compression nature of facts, the use of them also brings the significant improvement on informativeness. Meanwhile, FTEval demonstrates high correspondence with the manual judgment of faithfulness.

4.2 Related Work

4.2.1 Abstractive Sentence Summarization

Abstractive sentence summarization benefits headline design and is able to refine the selected sentences in extractive summarization. Early work of sentence summarization covered a wide range of approaches, such as statistical machine translation [Banko *et al.*, 2000], parsing tree-based sentence compression [Knight and Marcu, 2002] and template-based summarization [Zhou and Hovy, 2004]. Recently, the application of the attentional sequence-to-sequence (seq2seq) models [Rush *et al.*, 2015a], has attracted growing attention in this area. Experiments on the Gigaword test set [Rush *et al.*, 2015a] show that seq2seq models achieve state-of-the-art performance.

In addition to the direct application of the general seq2seq framework, researchers attempted to integrate various properties of summarization. For example, [Nallapati *et al.*, 2016a] enriched the encoder with manually defined features such as TF-IDF and POS tags. These features have played important roles in traditional feature based summarization systems. [Gu *et al.*, 2016] proposed CopyNet which considered the copying mechanism during generation. Recently, [See *et al.*, 2017] used the coverage mechanism to discourage repetition. There were also studies to modify the loss function to fit the evaluation metrics. For instance, [Ayana *et al.*, 2016] applied the Minimum Risk Training strategy to maximize the ROUGE scores of generated summaries. [Paulus *et al.*, 2017] applied the Reinforcement Learning (RL) algorithm to learn a mixed objective function of likelihood and ROUGE scores.

Although focused on the document level, we believe the event-based extractive summarization [Filatova and Hatzivassiloglou, 2004] are relevant to our work. This technique leverages the events described in the sentences to select and order sentences to form the summary. A general representation of an event is "[Who] did [What] to [Whom] [When] and [Where]" [Liu *et al.*, 2007]. Usually "did [What]", i.e., the action, acts as the key element of an event. Apparently, the facts defined in this chapter can be regarded as the core part of an event. In addition, OpenIE is also widely used for event extraction [Pighin *et al.*, 2014].

Previous researches usually focused on the improvement of summary informativeness. To our knowledge, we firstly explore the faithfulness issue in abstractive summarization.

4.2.2 Summary Evaluation

Automatic evaluation is crucial to advance the summarization research. [Lin, 2004] proposed the widely-used summarization metric ROUGE. Motivated by BLEU [Papineni *et al.*, 2002], ROUGE measures the quality of summary by computing the overlapping lexical units between the candidate summary and reference summaries. [Passonneau *et al.*, 2005] proposed a more accurate evaluation method, called Pyramid, based on the matching of Summarization Content Units (SCUs). However, the annotation of SCUs requires heavy human efforts. While ROUGE and Pymarid mainly measure the informativeness of the candidate summary, some studies also inspect the readability. For example, [Pitler *et al.*, 2010] used a set of syntactic features to judge the grammaticality, and [Lin *et al.*, 2012] applied the discourse relations to examine the coherence of a summary with multiple sentences. Although our experiments show that fake generation is a serious problem in neural abstractive summarization, as far as we know, no previous work has focused on the evaluation of faithfulness.

4.3 Fact Extraction

Based on our observation, about one third of the summaries generated by a state-of-the-art seq2seq model suffer from the fact counterfeit problem, which is mainly caused by mismatching the predicate with its subject or object. Therefore, we propose to explicitly use the facts conveyed in the source sentence for summarization and evaluation. We leverage the popular tools of Open Information Extraction (OpenIE) and dependency parser to identify the facts. OpenIE refers to the extraction of entity relations from the open-domain text. In OpenIE, a fact is typically interpreted as a relation triple consisting of (subject; predicate; object). For example, given the source sentence in Table 4.1, OpenIE [Angeli *et al.*, 2015] generates two relation triples. They are *(repatriation; was postponed; friday)* and *(unhcr; pulled out of; first joint scheme)*. Obviously, these triples can help rectify the mistakes made by the seq2seq model. It is also noted

Sentence	I saw a cat sitting on the desk
Triples	(I; saw; cat)
	(I; saw; cat sitting)
	(I; saw; cat sitting on desk)

Table 4.2: Examples of OpenIE triples in different granularities. We extract the following fact: *I saw cat sitting on desk*

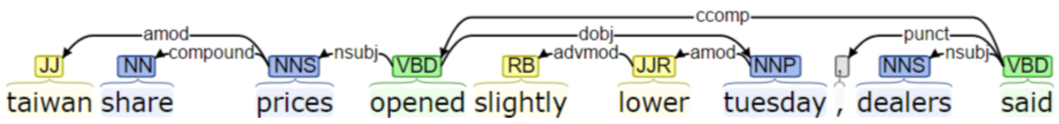


Figure 4.1: A dependency tree example.

that OpenIE may extract multiple triples to reflect an identical fact in different granularities, as shown in Table 4.2. In some extreme cases, one relation can yield over 50 triple variants, which brings high redundancy and burdens the computation cost of the model. To balance redundancy and fact completeness, we remove a relation triple if all its words are covered by another. For example, only the last fact, i.e., *(I; saw; cat sitting on desk)*, in Table 4.2 is reserved. When using these triples, we join all the items in a triple, i.e., (subject + predicate + object), considering it usually acts as a concise sentence. Thus, the final extracted fact of Table 4.2 is *(I saw cat sitting on desk)*. When multiple facts are extracted at the end, we use a special separator "|||" to concatenate them to accelerate the encoding process, which will be explained in Section 4.4.2.

OpenIE is able to give a complete description of the entity relations. However, it is worth noting that the relation triples are not always extractable, e.g., from the imperative sentences. In fact, about 15% of the OpenIE outputs are empty on our experimental dataset. These empty instances are likely to damage the robustness of our model. As observed, although the complete relation triples are not always available, the (subject; predicate) or (predicate; object) tuples are almost present in every sentence. Therefore, we leverage the dependency parser to dig out the appropriate tuples to supplement the extracted triple facts. A dependency parser converts a sentence into the labeled (governor; dependent) tuples. We extract the predicate-related tuples according to the labels of *nsubj*, *nsubjpass*, *csubj*, *csubjpass*, *dobj* and *iobj*. To acquire more complete facts, we also reserve the important modifiers such as the adjectival (*amod*), numeric (*nummod*) and noun compound (*compound*). The details of the tags we use are presented in Figure 4.2. We then merge the tuples containing the same words, and order words according to their positions in the original sentence to form the facts. For example, a dependency tree is shown in Figure 4.1. The output of OpenIE is empty for the corresponding sentence. Based on the dependency parser, we firstly pick up the following predicate-related tuples: *(prices; opened) (opened; tuesday) (dealers; said)* and the modify-head tuples: *(taiwan; price) (share; price) (lower; tuesday)*. These tuples are then merged to form two facts: *taiwan share prices opened lower tuesday* ||| *dealers said*.

In the experiments, we employ the popular NLP pipeline Stanford CoreNLP [Manning *et al.*, 2014][1] to handle OpenIE and dependency parsing. We combine the facts derived from both parts. The details of the redundancy control during combination is presented in Algorithm 1. Thereafter, we screen out the facts with the pattern "somebody said/declared/announced", which are usually meaningless and insignificant. The extracted facts actually form the skeletons of sentences. Referring to the copy ratios in Table 4.3, words in facts are 40% ($0.17/0.12 - 1$) more likely to be used in the summary than the words in the original sentence. It indicates that facts truly condense the meaning of sentences to a large extent. The above statistics also supports the practice of dependency parse based compressive summarization [Knight and Marcu, 2002]. However, the length sum of extracted facts is shorter than the actual summary in 20% of the sentences, and 4% of the sentences even contain no fact. It is clear that, we cannot reply on facts alone to generate summaries without reference to the source sentence.

[1]Use the annotator property: "tokenize,depparse,openie"

Function	Tag	Name	Description
Core	nsubj	nominal subject	a noun phrase acting as the syntactic subject of a clause
	nsubjpass	passive nominal subject	a noun phrase acting as the syntactic subject of a passive clause
	csubj	clausal subject	a clausal syntactic subject of a clause
	csubjpass	clausal passive subject	a clausal syntactic subject of a passive clause
	dobj	direct object	the noun phrase acting as the accusative object of the verb
	iobj	indirect object	the noun phrase acting as the dative object of the verb
Auxiliary	amod	adjectival modifier	an adjectival phrase used to modify the NP
	nmod	nominal modifier	a noun functioning as a non-core argument or adjunct
	nummod	numeric modifier	any number phrase used to modify the meaning of the noun with a quantity
	compound	compound	noun compounds
	conj	conjunct	the relation between two elements connected by a coordinating conjunction

Figure 4.2: Dependency tags for fact extraction. Refer to `http://universaldependencies.org/docsv1/en/dep/all.html`.

Source:	Sentence	Fact
AvgLen	31.4	18.2
Count	1	2.7
Copy%	12	17

Table 4.3: Comparisons between source sentences and relations. AvgLen is the average number of tokens. Copy% means the proportion of source tokens can be found in the summary.

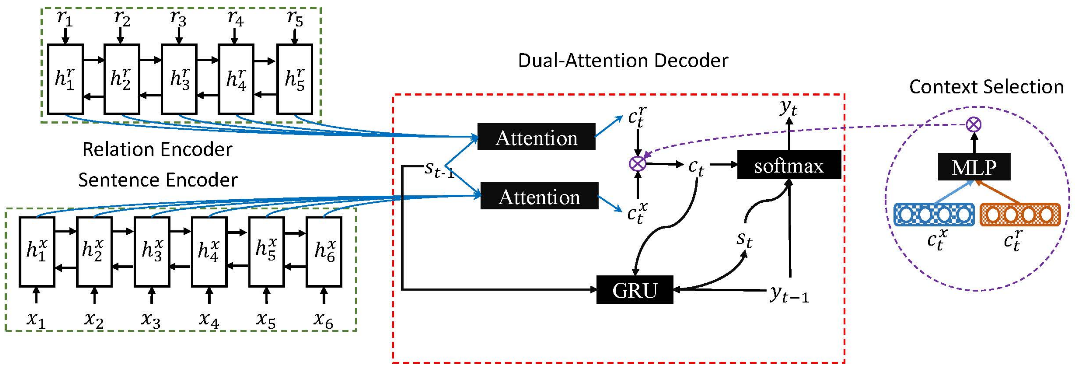


Figure 4.3: Model framework

4.4 Fact-Aware Neural Summarization

4.4.1 Overview

As shown in Figure 4.3, our model consists of three modules, including two encoders and a dual-attention decoder equipped with a context selection gate network. The sentence encoder reads the input words $\mathbf{x} = (x_1, \cdots x_n)$ and builds its corresponding representation $(\mathbf{h}_1^x, \cdots \mathbf{h}_n^x)$. Likewise, the relation encoder converts the facts $\mathbf{r} = (r_1, \cdots r_k)$ into hidden states $(\mathbf{h}_1^r, \cdots \mathbf{h}_k^r)$. With the respective attention mechanisms, our model computes the sentence and relation context vectors ($\mathbf{c}_t^x$ and $\mathbf{c}_t^r$) at each decoding time step t. The gate network is followed to merge the context vectors according to their relative associations with the current generation. The decoder produces summaries $\mathbf{y} = (y_1, \cdots y_l)$ word-by-word conditioned on the tailored context vector which embeds the semantics of both source sentence and facts.

4.4.2 Encoders

The input to the encoders includes the source sentence $\mathbf{x}$ and the extracted facts $\mathbf{r}$. For each sequence, we employ the bidirectional Recurrent Neural Network (BiRNN) encoder [Cho *et al.*, 2014b] to construct its semantic representation, similar to Chapter 3. Take the sentence $\mathbf{x}$ as an example. Given a source word x_τ, its hidden state of the forward RNN is represented by:

$$\overrightarrow{\mathbf{h}}_\tau = \mathrm{RNN}(x_\tau, \overrightarrow{\mathbf{h}}_{\tau-1}) \tag{4.1}$$

In experiments, we use he Gated Recurrent Unit (GRU) [Cho *et al.*, 2014b] as the recurrent unit, which often performs much better than the vanilla RNN. Its computation is as follows:

$$\mathbf{z}_\tau = \sigma([\overrightarrow{\mathbf{h}}_{\tau-1}, \mathbf{x}_\tau]\mathbf{W}_z) \tag{4.2}$$

$$\mathbf{r}_\tau = \sigma([\overrightarrow{\mathbf{h}}_{\tau-1}, \mathbf{x}_\tau]\mathbf{W}_r) \tag{4.3}$$

$$\mathbf{l}_\tau = \tanh([\mathbf{r}_\tau \odot \overrightarrow{\mathbf{h}}_{\tau-1}, \mathbf{x}_\tau]\mathbf{W}_l) \tag{4.4}$$

$$\overrightarrow{\mathbf{h}}_\tau = (\mathbf{1} - \mathbf{z}_\tau) \odot \overrightarrow{\mathbf{h}}_{\tau-1} + \mathbf{z}_\tau \odot \mathbf{l}_\tau \tag{4.5}$$

where $\mathbf{z}_\tau$, $\mathbf{r}_\tau$ are two gates, $\mathbf{W}_\cdot$ stands for model weights and $\odot$ means element-wise multiplication. We use bold font $\mathbf{x}_\tau$ to represent the word embeddings of the word x_τ. The BiRNN encodes text both from left to right and from right to left. Suppose the forward outputs are $[\overrightarrow{\mathbf{h}}_1; \cdots; \overrightarrow{\mathbf{h}}_{-1}]$ and backward outputs are $[\overleftarrow{\mathbf{h}}_1; \cdots; \overleftarrow{\mathbf{h}}_{-1}]$, where the index "$-1$" stands for the last element. Then, BiRNN concatenate the two RNN representations to build the composite hidden state of a word, i.e., $\mathbf{h}_\tau = [\overrightarrow{\mathbf{h}}_\tau; \overleftarrow{\mathbf{h}}_\tau]$. To distinguish source sentence and fact sequence, we add superscripts "x" and "r" on the hidden states, respectively.

For the fact sequence $\mathbf{r}$, since it contains multiple independent facts, we introduce boundary indicators γ to separate their hidden states. Specially, the value of γ is defined as follows:

$$\gamma_i = \begin{cases} 0, r_i \text{ is "|||"} \\ 1, \text{otherwise} \end{cases} \tag{4.6}$$

Then, γ is used to reset the GRU state in Equation 4.1:

$$\mathbf{h'}_i = \gamma_i \mathbf{h}_i \tag{4.7}$$

In this way, all the facts start with the same zero vector. In other words, they are encoded *independently*. Finally, both sentence hidden states $\{\mathbf{h}_i^x\}$ and relation hidden states $\{\mathbf{h}_i^r\}$ are fed into the decoder.

4.4.3 Dual-Attention Decoder

Previous seq2seq models have developed some task-specific modifications on the decoder, such as to incorporate the copying mechanism [Gu *et al.*, 2016] or the coverage mechanism [See *et al.*, 2017]. As we focus on the faithfulness problem here, we use the most popular decoder, i.e., GRU with attentions [Bahdanau *et al.*, 2014]. At each decoding time step t, GRU reads the previous output y_{t-1} and context vector $\mathbf{c}_{t-1}$ as inputs to compute new hidden state $\mathbf{s}_t$:

$$\mathbf{s}_t = \mathrm{GRU}(y_{t-1}, \mathbf{c}_t, \mathbf{s}_{t-1}) \tag{4.8}$$

Since we have both sentence and relation representations as input, we develop two attentional layers to construct the overall context vector $\mathbf{c}_t$. The context representation $\mathbf{c}_t^x$ of the sentence $\mathbf{x}$ at time step t is computed as [Luong *et al.*, 2015]:

$$\begin{aligned} e_{t,i}^x &= \mathrm{MLP}(\mathbf{s}_t, \mathbf{h}_i^x) \\ &= \mathbf{v}_a \tanh(\mathbf{s}_{t-1}\mathbf{W}_a + \mathbf{h}_i^x \mathbf{U}_a) && (4.9) \\ \alpha_{t,i}^x &= \frac{\exp(e_{t,i}^x)}{\sum_j \exp(e_{t,j}^x)} && (4.10) \\ \mathbf{c}_t^x &= \sum\nolimits_i \alpha_{t,i}^x \mathbf{h}_i^x, && (4.11) \end{aligned}$$

where MLP means multi-layer perceptrons, and $\mathbf{v}_a$, $\mathbf{W}_a$, $\mathbf{U}_a$ are model parameters. The context vector of the relation $\mathbf{c}^r$ is computed in the same way. We combine $\mathbf{c}_t^x$ and $\mathbf{c}_t^r$ to build the overall context vector $\mathbf{c}_t$. We explore two alternative combination approaches. The first one, called "FTSum$_c$", simply concatenates two context vectors:

$$\mathbf{c}_t = [\mathbf{c}_t^x; \mathbf{c}_t^r] \tag{4.12}$$

The second one, denoted as "FTSum$_g$", uses another MLP to build a gate network to combine the two context vectors with the weighted sum:

$$\mathbf{g}_t = \mathrm{MLP}(\mathbf{c}_t^x, \mathbf{c}_t^r) \tag{4.13}$$

$$\mathbf{c}_t = \mathbf{g}_t \odot \mathbf{c}_t^x + (\mathbf{1} - \mathbf{g}_t) \odot \mathbf{c}_t^r, \tag{4.14}$$

Experiments show that FTSum$_g$ significantly outperforms FTSum$_c$, and the gate values apparently reflect the relative reliability of sentence and facts. Given $y_{t-1}, \mathbf{c}_t, \mathbf{s}_{t-1}$, the decoder GRU is computed as follows:

$$\mathbf{z}_t = \sigma([\mathbf{s}_{t-1}, \mathbf{y}_{t-1}, \mathbf{c}_t]\mathbf{U}_z) \tag{4.15}$$

$$\mathbf{r}_t = \sigma([\mathbf{s}_{t-1}, \mathbf{y}_{t-1}, \mathbf{c}_t]\mathbf{U}_r) \tag{4.16}$$

$$\mathbf{l}_t = \tanh([\mathbf{r}_t \odot \mathbf{s}_{t-1}, \mathbf{y}_{t-1}, \mathbf{c}_t]\mathbf{U}_l) \tag{4.17}$$

$$\mathbf{s}_t = (\mathbf{1} - \mathbf{z}_t) \odot \mathbf{s}_{t-1} + \mathbf{z}_\tau \odot \mathbf{l}_\tau \tag{4.18}$$

where $\mathbf{U}_.$ denotes model parameters. Finally, the softmax layer is introduced to generate the next word based on previous word y_{t-1}, context vector $\mathbf{c}_t$ and current decoder state $\mathbf{s}_t$.

$$\mathbf{o}_t = \mathbf{W}_w[y_{t-1}] + \mathbf{c}_t\mathbf{W}_c + \mathbf{s}_t\mathbf{W}_s \tag{4.19}$$

$$p(y_t|y_{<t}) = \mathrm{softmax}(\mathbf{o}_t\mathbf{W}_o) \tag{4.20}$$

where $\mathbf{W}_.$ is a weight matrix.

4.4.4 Learning

The learning objective is to maximize the estimated probability of the actual summary. We adopt the common loss function, i.e., Negative Log-Likelihood (NLL):

$$J(\theta) = -\frac{1}{|\mathbf{D}|}\sum_{(\mathbf{x},\mathbf{r},\mathbf{y})\in\mathbf{D}} \log(p(\mathbf{y}|\mathbf{x},\mathbf{r})), \tag{4.21}$$

where $\mathbf{D}$ denotes the training dataset and θ stands for the model parameters. The optimization algorithm is Adam [Kingma and Ba, 2014] with mini-batches (learning rate

$\alpha = 0.001$ and batch size=32) in our experiments. Similar to [Zhou *et al.*, 2017b], we evaluate the model performance on the validation dataset for every 2000 batches and halve the learning rate if the cost increases for 10 consecutive validations. In addition, we apply gradient clipping [Pascanu *et al.*, 2013] with range $[-5, 5]$ during training to enhance the stability of the model.

4.5 Experiments on Summarization

We conduct experiments on the Annotated English Gigaword dataset [Rush *et al.*, 2015b], as shown in Section 3.4. The statistics of the Gigaword dataset is presented in Table 4.4. We do not examine on the other summarization dataset used in Chapter 3, i.e., CNN/DM, due to the following two reasons. On one hand, the source text of CNN/DM is so long that the manual judgment of the generation faithfulness requires huge human efforts. On the other hand, the generated summaries in CNN/DM often repeat a word or a phrase. Thus the readability becomes a serious issue.

Dataset	Train	Dev.	Test
Count	3.8M	189k	1951
AvgSourceLen	31.4	31.7	29.7
AvgTargetLen	8.3	8.3	8.8
COPY(%)	45	46	36

Table 4.4: Data statistics for English Gigaword. AvgSourceLen is the average input sentence length and AvgTargetLen is the average summary length. COPY means the copy ratio in the summaries (without stopwords).

4.5.1 Evaluation Metric

We adopt ROUGE [Lin, 2004] for automatic evaluation. Following the common practice, we report ROUGE-1 (uni-gram), ROUGE-2 (bi-gram) and ROUGE-L (longest common subsequence) F1 scores in the following experiments. More details can refer to Section 3.4.2.

In addition, we also manually inspect whether the generated summaries accord with the facts in the original sentences. We mark summaries into two categories: FAITHFUL and FAKE. Noted that if a generated summary is too incomplete to judge its faithfulness (e.g., just producing a UNK tag), we also mark it as FAKE.

4.5.2 Implementation Details

Since the dataset has already masked infrequent words with the UNK tag, we reserve all the rest words in the training set. As a result, the sizes of source and target vocabularies are 120k and 69k, respectively. With reference to [Nallapati *et al.*, 2016a], we leverage the popular seq2seq framework dl4mt[2] as the starting point, and set the size of word embeddings to 200. We initialize word embeddings with GloVe [Pennington *et al.*, 2014]. All the GRU hidden state dimensions are fixed to 400. We use dropout [Srivastava *et al.*, 2014] with probability $p = 0.5$. Since the facts work like additional source text, the training time of FTSum is close to the standard seq2seq model.

4.5.3 Baselines

We introduce the following six state-of-the-art neural summarization systems as baselines:

ABS The first neural abstractive summarization system proposed by [Rush *et al.*, 2015a].

ABS+ [Rush *et al.*, 2015a] further tuned the ABS model with additional features to balance the abstractive and extractive tendency.

RAS-Elman As the extension of the ABS model, it changed the decoder with RNN [Chopra *et al.*, 2016].

[2]https://github.com/kyunghyuncho/dl4mt-material

Featseq2seq [Nallapati *et al.*, 2016a] used a full seq2seq model and added the hand-crafted features such as POS tag and NER, to strengthen the encoder representation.

Luong-NMT [Chopra *et al.*, 2016] implemented the neural machine translation model of [Luong *et al.*, 2015] for summarization. This model contained two-layer LSTMs for both encoder and decoder.

s2s-att We implement the standard attentional seq2seq with dl4mt, and call it "s2s-att".

Considering our model is somewhat associated with compressive summarization, we also introduce a state-of-the-art compression system called COMPRESS [Clarke and Lapata, 2008]. We just cite the implementation result from [Rush *et al.*, 2015a].

4.5.4 Informativeness Performance

Let's first look at the cost values on the validation dataset. From Table 4.5 and Figure 4.4, we can see that our model achieves much lower perplexity compared against the state-of-the-art systems. It is also noted that FTSum$_g$ largely outperforms FTSum$_c$, which verifies the importance of context selection. The ROUGE F1 scores are then reported in Table 4.6. Although the focus of our model is to improve faithfulness, the ROUGE scores it achieves are also much higher than the other methods. Note that, ABS+ and Featseq2seq have utilized a series of hand-crafted features, but our model is totally data-driven. Even though, our model surpasses Featseq2seq by 13% and ABS+ by 56% on ROUGE-2. When facts are ignored, our model is equivalent to the standard attentional seq2seq model s2s-att. Therefore, it is safe to conclude that, facts have significant contribute to the increase of ROUGE scores. One possible reason is that facts are much more informative than the original sentence, as shown in Table 4.3. It also largely explains why FTSum$_g$ is superior to FTSum$_c$. FTSum$_c$ treats the source sentence and relations equally, while FTSum$_g$ realizes the facts are often more reliable, as discussed later.

Model	Perplexity
ABS[†]	27.1
RAS-Elman[†]	18.9
s2s-att	24.5
FTSum$_c$	20.1
FTSum$_g$	**16.4**

Table 4.5: Final perplexity on the validation dataset. [†] indicates the value is cited from the corresponding paper. ABS+, Featseq2seq and Luong-NMT do not provide this value.

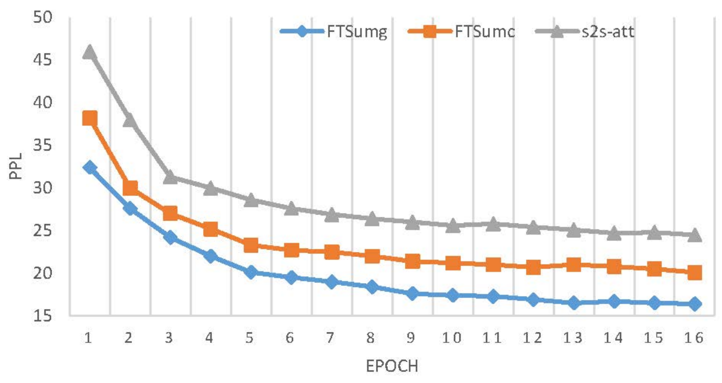


Figure 4.4: Perplexity change on the validation dataset.

Model	ROUGE-1	ROUGE-2	ROUGE-L
COMPRESS[†]	19.63*	5.13*	18.28*
ABS[†]	29.55*	11.32*	26.42*
ABS+[†]	29.78*	11.89*	26.97*
Featseq2seq[†]	32.67*	15.59*	30.64*
RAS-Elman[†]	33.78*	15.97*	31.15*
Luong-NMT[†]	33.10*	14.45*	30.71*
s2s-att	34.23*	15.52*	31.57*
FTSum$_c$	35.73*	16.02*	34.13
FTSum$_g$	**37.27**	**17.65**	**34.24**

Table 4.6: ROUGE F1 (%) performance. "*" indicates statistical significance of the corresponding model with respect to the baseline model on the 95% confidence interval in the official ROUGE script.

4.5.5 Faithfulness Performance

Next, we conduct manual evaluation to inspect the faithfulness of the generated summaries. We randomly select 150 sentences from the test set. Then, we classify the generated summaries as FAITHFUL or FAKE. For the sake of a complete comparison, we present the results of our system FTSum$_g$ together with the attentional seq2seq model s2s-att. As shown in Table 4.7, about one third of the s2s-att outputs give disinformation. This number greatly reduces by 55% in our model, which largely enhances the system practicability. We find that s2s-att tends to copy the words closer to the predicate and regard them as its subject and object. However, this is not always reasonable and may make a counterfeiting message. In comparison, the facts indeed designate the relations between a predicate and its subject and object. As a result, generation in line with the facts is usually able to keep faithfulness.

We illustrate the examples of defective outputs in Table 4.8. As shown, s2s-att often attempts to fuse different parts in the source sentence to form the summary, no matter whether these phrases are relevant or not. For instance, s2s-att treats “bosnian moslems” as the subject of “postponed” and “bosnia” as the object of “pulled out of” in Example 1. By contract, since the fact point out the actual subject and object, the output of our model is faithful. In fact, it is exactly the same as the target summary. In Example 3, neither s2s-att nor our model achieves satisfactory performance. s2s-att again mismatches the object while our model fails to produce a complete sentence. To take a closer look, we find that the target summary of this sentence is somewhat strange – it merely focuses on the prepositional phrase (after taking a ## stoke...), rather than the main clause as usual. Since the main clause is hard to summarize and there is no high-quality fact extracted, our model fails to give a complete summary.

It is also noteworthy that the generation of our model sometimes traps into one item when given multiple long facts. For instance, there are two long facts in Example 4 and our model only utilizes the first one for generation. As a result, despite the

Model	Category	Count
	FAITHFUL	99
s2s-att	FAKE	51
	FAITHFUL	127
FTSum$_g$	FAKE	23

Table 4.7: Faithfulness performance on the test set.

high faithfulness, the informativeness is somewhat damaged. Therefore, it seems more reliable to introduce the coverage mechanism [See *et al.*, 2017] to handle the cases like this one. We leave it as our future work.

Example 1	
Source	the repatriation of at least #,### bosnian moslems was postponed friday after the unhcr pulled out of the first joint scheme to return refugees to their homes in northwest bosnia .
Facts	unhcr pulled out of first joint scheme ||| repatriation was postponed friday ||| unhcr return refugees to their homes
Target	repatriation of bosnian moslems postponed
s2s-att	(FAKE) **bosnian moslems** postponed after unhcr pulled out of **bosnia**
FTSum	(FAITHFUL) repatriation of bosnian moslems postponed
Example 2	
Source	a us citizen who spied for communist east germany was given a suspended jail sentence of ## months here .
Facts	us citizen was given suspended jail sentence ||| who spied
Target	us citizen who spied for east germans given suspended sentence
s2s-att	(FAKE) u.s. citizen gets suspended jail sentence **in germany**
FTSum	(FAITHFUL) us citizen who spied for east germany given suspended jail sentence
Example 3	
Source	davis love said he was thinking of making the world cup of golf a full time occupation after taking a ## stroke lead over japan in the event with us partner fred couples here on saturday .
Facts	making world cup full time occupation ||| taking ## stroke lead
Target	americans lead UNK by ## strokes
s2s-att	(FAKE) davis love says he is thinking of **the world cup**
FTSum	(FAKE) love **in** the world cup of golf
Example 4	
Source	the us space shuttle atlantis separated from the orbiting russian mir space station early saturday , after three days of test runs for life in a future space facility , nasa announced .
Facts	us space shuttle atlantis separated from orbiting russian mir space station ||| us space shuttle atlantis runs after three days of test for line in future space facility
Target	atlantis mir part ways after three-day space collaboration by emmanuel UNK
s2s-att	(FAKE) space shuttle atlantis separated after **#** days of test runs for life
FTSum	(FAITHFUL) space shuttle atlantis separated from mir

Table 4.8: Examples of defective outputs. We use bold font to indicate the problematic parts.

4.5.6 Effect of Facts

We investigate the effects of facts from various aspects. At first, we check the accuracy of copying and rewriting. For comparison, we also present the result of s2s-att. Seen from Table 4.9, the copying proportion and accuracy of FTSum are both higher than s2s-att. A possible reason is due to the sentence skeleton nature of facts. On the one hand, facts are extracted from source sentences. As a result, import of facts seems to "lengthen" the source sentence which brings more chance to copy. On the other hand, facts condense the meaning to guide summary generation. Since s2s-att does not utilize facts, the prediction accuracies of the words in facts and the words not in facts are almost the same.

Next, we manually inspect the correctness of extracted facts. Since the source sentences come from formal news articles, the parsing accuracy is high. In 100 sentences, we only observe one serious error, as shown in Table 4.10. Through common sense we know "arabs" and "sudanese" are coordinate. However, CoreNLP wrongly regards "arabs" as another object of "convicted of" and thus creates a fake facts. As a result, our model is misguided. s2s-att does not achieve satisfactory performance neither. It seems to discard the content before "and".

Finally, we explore the performance of our system when different types of facts are given. The result is present in Table 4.11. We find that the facts derived from dependency parsing outperform the facts derived from OpenIE. A possible reason is that the output of OpenIE is sometimes empty, seriously worsening the stability of the system. As a result, the ROUGE performance of facts solely from OpenIE is close to s2s-att.

4.5.7 Gate Analysis

As shown in Table 4.6, FTSum$_g$ achieves much higher ROUGE scores than FTSum$_c$. Now, we investigate what the gate network (Equation 4.13) actually learns. The chan-

Mode		FTSum	s2s-att
Copy	$\mathbf{r}$	0.71 (0.47)	0.69 (0.38)
	$\mathbf{x}-\mathbf{r}$	0.15 (0.28)	0.12 (0.36)
	$\mathbf{x}$	0.86 (0.43)	0.81 (0.38)
Rewrite		0.14 (0.22)	0.19 (0.23)

Table 4.9: Copy and rewrite performance. Here $\mathbf{r}$ stands for facts and $\mathbf{x}$ means source sentences. The value form is: length proportion (correct proportion).

Source	a sudanese convicted of a series of murders and two other arabs found guilty of drug trafficking were beheaded on friday , the saudi interior ministry said .
Facts	**sudanese** found guilty of drug trafficking
Target	three beheaded in saudi arabia
s2s-att	(FAKE) **two** convicts beheaded in saudi arabia
FTSum	(FAKE) **sudanese** convicted of drug trafficking beheaded in saudi arabia

Table 4.10: An example of extracted fake facts.

ges of the gate values on the validation dataset during training are shown in Figure 4.5. At the beginning, the average gate value exceeds 0.5, which means generation is biased to the source sentence. As training proceeds, the model realizes that facts are more reliable, resulting in a consecutive drop of the gate value. Finally, the average gate value is gradually stabilized to 0.415. Interestingly, the ratio of sentence and relation gate values i.e., $(1-0.415)/0.415 \approx 1.41$, is extremely close to the ratio of copying proportions shown in Table 4.3 i.e., $0.17/0.12 \approx 1.42$. Then, look at the standard deviation of gates. To our surprise, its change is nearly anti-symmetric to the mean value. The final standard deviation reaches about 90% of the mean gate value. Thus, still many sentences can dominate generation. This strange observation urges us to carefully check the summaries with top/bottom-100 gate values in the validation dataset. We find that 10 facts in the top-100 cases are empty, and nearly 60% of the extracted facts contains the UNK tag. Our model believes these facts have less value to guide generation. Instead, there is no empty fact and only 1 UNK tag in the bottom 100 cases. Hence these facts are usually informative enough. In addition, we find the instances with the lowest gate values often hold the following (target summary; fact) pair:

Fact	ROUGE-1	ROUGE-2	ROUGE-L
OpenIE	34.88	15.73	32.02
Parse	36.25	16.32	33.67

Table 4.11: ROUGE F1 (%) performance of FTSum generated with different facts.

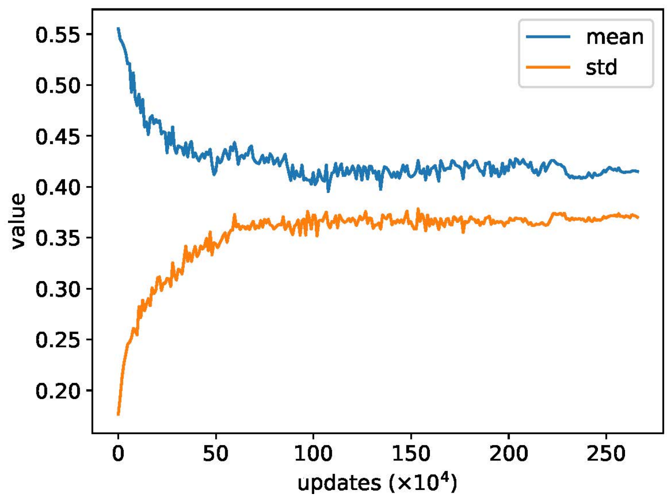


Figure 4.5: Gates change during training.

Target COUNTRY share prices close/open #.# percent higher/lower

Fact COUNTRY share prices slumped/dropped/rose #.# percent

The extracted fact itself is already a proper summary. That is why facts are particularly preferred in generation.

4.6 Automatic Faithfulness Evaluation

4.6.1 Motivation

Metrics	Category	AvgScore
RG-1	FAITHFUL	42.61
	FAKE	22.52
RG-2	FAITHFUL	15.58
	FAKE	3.62

Table 4.12: The Average ROUGE F1 scores (%) in faithful and fake summaries

We investigate the correspondence between faithfulness and ROUGE based on the manually annotated faithfulness data summarized in Table 4.7. The result is shown in Figure 4.6. From the figure, we observe the following points:

- Summaries with very high ROUGE scores are usually faithful.
- Fake summaries usually hold low ROUGE scores. As shown in Table 4.12, the average ROUGE-1 score for faithful summaries is 0.43 while it is only 0.23 for fake ones.
- However, summaries with low ROUGE scores are a mixture of faithful and fake results. For example, even among summaries with ROUGE-2=0, 65% of them are faithful.

Unfortunately, the state-of-the-art performance of the neural abstractive summarization system is still located within the area of low ROUGE scores. Consequently, it is almost impossible to judge the faithfulness performance of the current abstractive summarization system according to ROUGE scores. Thus, we explore to develop an automatic faithfulness evaluation tool, called FTEval.

4.6.2 Tool Overview

A simple solution to faithfulness evaluation is to extract the facts in the summary as what we have done for the source sentence, and then check their fact agreement. However, due to the concise nature of a summary, we observe the following two challenges that OpenIE encounters:

1. About 15% of the OpenIE outputs are empty, which makes it often useless.
2. It is hard to judge the agreement of the relation triples since the granularities in the sentence and summary vary largely.

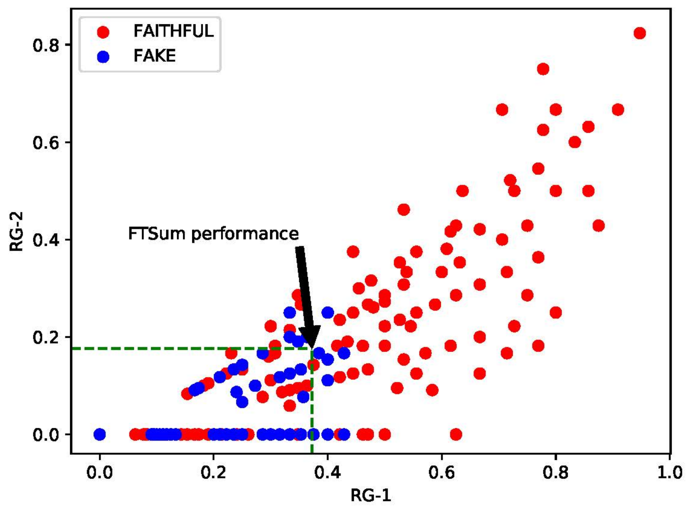


Figure 4.6: The correlation between ROUGE F1 scores and faithfulness. We demonstrate the average performance of our summarization system.

Therefore, we opt to utilize the facts extracted by the dependency parser to measure faithfulness. That is, we compute the matching degree of the predicate-related tuples derived from dependency trees of the sentence and summary. The details will be explained in the rest of this section.

Our tool FTEval also brings an advantage that all the identified fake summaries are explainable. To be specific, the tuples, which cause the faithfulness problem, are highlighted. As a result, we can further inspect whether and how these tuples violate the original facts. The workload is reduced greatly compared with direct manual evaluation.

4.6.3 Text Preprocessing

We conduct the following three tasks on both sentences and summaries, namely lemmatization, POS tagging and dependency parsing. Likewise, we use the CoreNLP pipeline to achieve this goal[3]. As shown in Table 4.13, the first two tasks help to align words

[3]Use the annotator property: "tokenize,pos,lemma,depparse,"

between a source sentence and its summary, while dependency parse is used to measure the predicate agreement.

Label	Task Name	Usage
Lemma	Lemmatization	Section 4.6.4
POS	POS Tagging	Section 4.6.4
Dep	Dependency Parsing	Section 4.6.4 and 4.6.5

Table 4.13: Required NLP Tasks.

4.6.4 Word Alignment

Although the meaning of a summary should be covered by the source sentence, it is sometimes conveyed with different words that express the same concept. As we want to check whether the predicate-related tuples in the summary accords with the source sentence, it is necessary to find the "prototypes" of the summary words. Here we just focus on the alignment of verbs (POS beginning with "VB") and nouns (POS beginning with "NN") which are associated with facts. In addition, we assume each summary word links to *one and only one* sentence word. According to the nature of summarization, we classify the alignment into three types:

COPY The summary word is also found in the source sentence.

LEMMA The lemma of the summary word is the same as the lemma of a word in the source sentence. For example, "stopped" (sentence) $\rightarrow$ "stops" (summary).

OTHER The rest words in the summary. Usually, they are the paraphrases of the source words. For instance, "boost" (sentence) $\rightarrow$ "rise" (summary).

The proportions of different alignment types in the summaries are shown in Table 4.14. As can be seen, in a current neural abstractive summarization system, over 80% summary words are copied from the source sentence.

It is easy to find the first two types of alignment. For the OTHER type, we align a summary word with its most semantically relevant source word. We have tried several approaches for it, such as the alignment tool of statistical machine translation [Dyer *et al.*, 2013] and the attentions learned from the seq2seq model. However, since the proportion of the words belonging to OTHER is quite small, the two alignment approaches do not make much difference on the final performance. To make our tool more general, we choose to use the cosine similarity of pre-trained word embeddings [Pennington *et al.*, 2014] to measure the semantic relevance. Thus, our evaluation tool can be directly used on any other dataset. To formulate, for a source word x and target word y, their similarity is:

$$\text{sim}(x, y) = \frac{\mathbf{e}_x \mathbf{e}_y^T}{||\mathbf{e}_x|| \times ||\mathbf{e}_y||} \tag{4.22}$$

where $\mathbf{e}_.$ stands for a word embedding. The detail of the alignment is presented in Algorithm 2.

Model	COPY	LEMMA	OTHER
s2s-att	8137	991	895
FTSum	8204	627	723

Table 4.14: Statistics of alignment types in different summarization models.

4.6.5 Fact Agreement

We measure the fact agreement between sentence and summary based on the dependency parse tree and word alignment. The detailed algorithm is explained as follows:

Step 1 Extract all predicate-related tuples from the parse tree of a summary, as introduced in Section 4.3.

Step 2 Suppose (s; p) is an extracted subject-predicate tuple. We check whether their aligned words (s'; p') also hold the subject relation in the parse tree of the corresponding source sentence.

Step 3 If all the predicate-related tuples can be found in the source sentence, we label the summary as FAITHFUL, and FAKE otherwise.

In cases when there is no predicate in a summary, we deem this summary as incomplete and label it as FAKE. In addition, based on the findings shown in Figure 4.6, we directly label the summaries whose ROUGE scores exceed a threshold as FAITHFUL. Experiments demonstrate that such practice largely increases the precise of our tool.

4.7 Experiments on Automatic Faithfulness Evaluation

4.7.1 Data and Setting

In this section, we discuss the performance of our faithfulness evaluation tool. We use the 300 summaries with manual faithfulness labels mentioned in Section 4.5.5 as the dataset. Similarly, we make use of CoreNLP to conduct word lemmatization, POS tagging and dependency parsing on both sentences and summaries[4].

Since detecting fake generation is crucial, we adopt three common metrics, i.e., Precise, Recall and F1-measure, to measure the performance of a faithfulness evaluation approach:

$$\text{Precise} = \frac{n_f(\text{correct})}{n_f(\text{predicted})} \tag{4.23}$$

$$\text{Recall} = \frac{n_f(\text{correct})}{n_f(\text{actual})} \tag{4.24}$$

$$\text{F1-measure} = \frac{2 \times n_f(\text{correct})}{n_f(\text{predicted}) + n_f(\text{actual})} \tag{4.25}$$

where $n_f(.)$ stands for the number of fake results.

[4]Use the annotator property: "tokenize,pos,lemma,depparse,"

4.7.2 Baselines

We introduce three baselines for comparison. The first one is based on ROUGE scores. As shown in Figure 4.6, fake summaries usually have small ROUGE scores. Thus, we set a threshold of the ROUGE score and regard all the summaries lower than this threshold as FAKE and higher as FAITHFUL. Through cross validation, we set the threshold to 0.40[5] for ROUGE-1 and 0.15 for ROUGE-2. The two baselines are named RG-1 and RG-2, respectively. The second one is called NO-RG, which is the same as our tool FTEval but without the ROUGE threshold. The last one is derived from the task of textual entailment recognition. Textual entailment judges whether the fact of one text follows from another. Obviously, summary faithfulness is a special kind of textual entailment, where the facts in the source text and the summary are usually the same. We adopt a state-of-the-art textual entailment recognition tool ESIM [Chen *et al.*, 2017]. Notably, this tool is also extended from the framework dl4mt, and uses recurrent neural networks as well as the attention mechanism. We train this tool on the popular textual entailment recognition dataset, called Stanford Natural Language Inference (SNLI) corpus [Bowman *et al.*, 2015]. SNLI contains 570k human-written English sentence pairs. Each pair is manually annotated as neutral, entailment or contradiction. During test, we regard the "entailment" label as FAITHFUL, and the labels of both "neutral" and "contradiction" as FAKE.

4.7.3 Performance Comparison

Table 4.15 presents the performance of different faithfulness evaluation approaches. As can be seen, FTEval greatly outperforms other approaches on precise and F1-measure. Its recall exceeds 90%, meaning that the clear majority of actual fake summaries are detected. The high recall benefits further manual annotation if necessary. The comparison of FTEval and NO-RG reveals that recognition of actual faithful summaries can largely profit from the introduction of ROUGE threshold. The recall of ROUGE is also high,

[5]We also set ROUGE-1=0.4 as the threshold for our evaluation tool FTEval.

Example 1	
Source	some #.# billion people worldwide are expected to watch germany face costa rica on television at the opening match of football 's world cup , german public broadcaster zdf said thursday .
Summary	germany faces costa rica on tv at world cup
Evaluation	Non-existent: (germany; faces)
Example 2	
Source	the united nations condemned saturday an attack on russian embassy employees in baghdad that claimed the life of one russian and resulted in the kidnapping of four others .
Summary	u.n. condemns attack on russian embassy employees in baghdad
Evaluation	Non-existent: (u.n.; condemns)
Example 3	
Source	serbs living in enclaves in northern kosovo have taken a step towards separation from the rest of the UNK province by breaking off ties with the un administration , local media said tuesday .
Summary	serbs in northern kosovo step towards separation
Evaluation	No predicate

Figure 4.7: Faithful summaries misdiagnosed as fake ones.

but it is at the cost of large precise loss. ROUGE regards nearly 60% summaries as fake ones, while the actual proportion is only about 20%. To our surprise, ESIM works terribly on this task, despite its nearly 90% accuracy on the SNLI dataset. It marks 90% summaries as FAITHFUL. One possible reason is the high text-level similarity between sentences and summaries, which much differs from the entailment training data.

Metrics	Precise	Recall	F1-measure
RG-1	0.39	0.89	0.54
RG-2	0.35	0.86	0.50
ESIM	0.25	0.11	0.15
NO-RG	0.32	**0.93**	0.47
FTEval	**0.52**	0.91	**0.66**

Table 4.15: Comparison of different faithfulness evaluation methods.

4.7.4 Error Analysis

In this section, we analyze the mistakes that FTEval makes. Since the recall of FTEval is quite high, we firstly inspect the fake summaries which not detected by our tool. We find the common problems among them are the incomplete facts. Due to the concise nature of summarization, we cannot ensure all the predicate-related tuples reserved in the summary. Nevertheless, sometimes missing a tuple largely obscures the meaning of a summary. An example is shown in Figure 4.8. The complete fact in the source sentence is (deal; cleared; hurdle), but the object is missing in the generated summary. As a result, the meaning of this summary is incomplete and FTEval labels it as fake. However, our evaluation tool only checks the predicate-related tuple in the summary (here is (deal; cleared)), and determines it accords with the source.

Second, we check the faithful summaries misdiagnosed as the fake ones. Examples are shown in Figure 4.7. A typical error of our tool is rooted from the missing of predicate. Since our evaluation tool focuses on the predicate-related facts, it simply marks any summaries with no predicate as fake. However, we observe that sometimes this type of summaries is still understandable, as shown in Example 3. Thus, the human annotation labels them as faithful, which is impossible for our evaluation tool to distinguish. Another serious problem is caused by the dependency parse error. For instance, the word "face" is wrongly tagged as a noun in the source. As a result, the fact (germany; face; costa rica) cannot be extracted. It is also noted that the alignment errors will occasionally hurt the evaluation performance. As in Example 2, our tool fails to recognize the full name of "u.n." and this makes the fact (u.n.; condemns) unrecognizable. A pre-defined abbreviation table should be a good solution to these problems.

Source	a controversial civilian nuclear energy deal between india and the united states cleared its first major hurdle tuesday , easily winning approval by a us congress committee .
Summary	controversial nuclear energy deal cleared

Figure 4.8: An example of mistake.

4.8 Summary

This chapter resents our work of investigating an important faithfulness issue in abstractive summarization. We employ popular OpenIE and dependency parsing tools to extract facts in the source sentence. We propose the dual-attention seq2seq framework to force the generation conditioned on both source sentence and the facts. Based on the extracted facts, we also develop an automatic tool for faithfulness evaluation. Experiments on the Gigaword benchmark dataset demonstrate that our summarization system greatly outperforms state-of-the-art models on both informativeness and faithfulness. Meanwhile, our evaluation tool reveals high correspondence with the manual judgment of faithfulness.

We think this work can be improved in various aspects in the future. On the one hand, we can improve the decoder of our summarization system with the copying mechanism and coverage mechanism. On the other hand, we plan to test our summarization system and evaluation tool on document summarization datasets.

Algorithm 1 Merge Facts

Input: Fact sets $\mathbf{r}_{ie}$ and $\mathbf{r}_{dep}$ extracted through OpenIE and dependency parsing
Output: Merged facts $\mathbf{r}$

```
1: $\mathbf{r} \leftarrow []$
2: for all $i \in [1, |\mathbf{r}_{dep}|]$ do
3:     $\text{is\_unique} \leftarrow \text{TRUE}$
4:     for all $j \in [1, |\mathbf{r}_{ie}|]$ do
5:         if predicate($\mathbf{r}^i_{dep}$) $\in$ predicate($\mathbf{r}^{\mathrm{j}}_{ie}$) then
6:             $\text{is\_unique} \leftarrow \text{FALSE}$
7:             break
8:         end if
9:     end for
10:    if is_unique then
11:        $\mathbf{r}$.append($\mathbf{r}^i_{dep}$)
12:    end if
13: end for
14: $\mathbf{r}$.extend($\mathbf{r}_{ie}$)
15: Sort $\mathbf{r}$ in descending order based on the length
16: $\mathbf{r}' \leftarrow []$
17: for all $i \in [1, |\mathbf{r}|]$ do
18:     $\text{is\_unique} \leftarrow \text{TRUE}$
19:     for all $j \in [1, |\mathbf{r}'|]$ do
20:         if words($\mathbf{r}_i$) $\subset$ words($\mathbf{r}'_j$) then
21:             $\text{is\_unique} \leftarrow \text{FALSE}$
22:             break
23:         end if
24:     end for
25:     if is_unique then
26:         $\mathbf{r}'$.append($\mathbf{r}_i$)
27:     end if
28: end for
29: $\mathbf{r} \leftarrow \mathbf{r}'$
30: return $\mathbf{r}$
```

Algorithm 2 Get Alignment

```
Input:  Sentence $\mathbf{x} = \{x_i\}$ and Summary $\mathbf{y} = \{y_j\}$
Output:  Alignment Table $a$
 1: for all $j \in [1, |\mathbf{y}|]$ do
 2:     if $\exists i,\ s.t.\quad x_i == y_t$ then
 3:         $a_j \leftarrow i$
 4:         continue
 5:     end if
 6:     if $\exists i,\ s.t.\quad \text{lemma}(x_i) == \text{lemma}(y_t)$ then
 7:         $a_j \leftarrow i$
 8:         continue
 9:     end if
10:     $a_j \leftarrow \arg\max_i \text{sim}(x_i, y_j)$
11: end for
12: return $\mathbf{a}$
```

Chapter 5

Rewrite with Soft Templates

If I have seen further, it is by standing on the shoulders of giants.

— **Isaac Newton**
(Mathematician and physicist)

5.1 Introduction

Most previous seq2seq models purely depend on the source text to generate summaries. However, as reported in many studies such as [Koehn and Knowles, 2017], the performance of a seq2seq model deteriorates quickly with the increase of the length of generation. Our experiments also show that seq2seq models sometimes tend to "lose control" (refer to Table 5.5). For example, 3% of summaries are composed of less than 3 words, while there are 4 summaries repeating a word for even 99 times. These results largely reduce informativeness and readability of the generated summaries. In addition, we find that seq2seq models usually copy source words consecutively, without any actual "summarization". Therefore, we argue that, free generation based on the source sentence is not enough for a seq2seq model.

Template-based summarization (e.g., [Zhou and Hovy, 2004]) is a traditional approach to abstractive summarization. In general, a template is an incomplete sentence that can be filled with the input text using the manually defined rules. An example of

Template	[REGION] shares [open/close] [NUMBER] percent [lower/higher]
Source	hong kong shares closed down #.# percent on friday due to an absence of buyers and fresh incentives .
Summary	*hong kong* shares *close* #.# percent *lower*

Table 5.1: An example of template-based summarization.

template-based summarization is shown in Table 5.1. As can be seen, the generated summary indeed concludes the stock market quotation, and the given template can be easily adapted to sentences describing other stock markets. Since the templates are handcrafted by humans, the produced summaries are usually fluent and informative. However, the construction of templates is extremely time-consuming and requires a plenty of domain knowledge. Moreover, it is impossible to develop all templates for summaries in various domains.

Inspired by retrieve-based conversation systems [Ji *et al.*, 2014], we assume that the golden summaries of the similar sentences can provide a reference point to guide the input sentence summarization process. We call these existing summaries **soft templates** since no actual rules are needed to build new summaries from them. Meanwhile, due to the strong rewriting ability of the seq2seq framework [Cao *et al.*, 2017a], we propose to combine the seq2seq and template-based summarization approaches. We call our summarization system Re^{3}Sum, which consists of three modules: **Re**trieve, **Re**rank and **Re**write. We rely on a widely-used Information Retrieval (IR) platform to find out candidate soft templates from the training dataset. Then, we extend the seq2seq model to jointly learn template saliency measurement (i.e., Rerank) and final summary generation (i.e., Rewrite). Specifically, a Recurrent Neural Network (RNN) encoder is applied to convert the input sentence and each candidate template into hidden states. Rerank measures the informativeness of a candidate template according to its hidden state relevance to the input sentence, similar to sentence ranking in extractive summarization (refer to Section 2.1). The candidate template with the highest predicted informativeness is regarded as the actual soft template. In Rewrite, the summary is generated according to the hidden states of both sentence and template.

We conduct extensive experiments on the popular Gigaword dataset [Rush *et al.*, 2015b]. Experiments show that, Re^{3}Sum significantly outperforms the state-of-the-art seq2seq models with regard to informativeness, and even soft templates themselves demonstrate high competitiveness. In addition, the import of high-quality external summaries improves stability and readability of generated summaries. Last but not least, Re^{3}Sum is able to summarize diversely given different templates.

5.2 Related Work

Retrieve-based summarization, a.k.a extractive summarization, is a popular kind of document summarization approaches [Over and Yen, 2004]. It forms a summary by extracting important source sentences. Basically, there are two major steps in retrieve-based summarization: *sentence ranking* and *sentence selection*. Sentence ranking, the crucial step, measures the saliency of a sentence. It is worth noting that previous researchers [Kobayashi *et al.*, 2015; Cheng and Lapata, 2016] have tried to use neural network representations to conduct sentence ranking. Our Rerank module borrows the idea of sentence ranking. However, the soft templates used in our model are extracted from the summaries in the training dataset, rather than the source text like retrieve-based summarization.

Template-based generation [Reiter and Dale, 1997] is a common natural language generation technique. Defining templates usually requires a lot of human efforts and domain knowledge, which blocks the research of this area. In summarization, [Zhou and Hovy, 2004] investigated template-based headline generation, and [Oya *et al.*, 2014] presented a template-based abstractive meeting summarization system.

Information retrieval (IR) [Larson, 2010] aims to acquire information from a large set of data resources. Information ranges from various types, such as texts [Barry *et al.*, 2007], images [Goodrum, 2000], audio [Foote, 1999] or videos [Lew *et al.*, 2006]. An IR system usually calculates a relevance score between the query and the objects in the stored database. Then the retrial results are ranked according to the scores. Finally, the objects with top ranking scores are returned. Common ranking metrics include BM25 [Robertson *et al.*, 1995] and its extensions [Robertson *et al.*, 2009; Lv and Zhai, 2011].

[Guu *et al.*, 2017] also proposed to encode human-written sentences to improvement the performance of neural text generation. However, they handled the task of Language Modeling and randomly picked an existing sentence in the training dataset. In comparison, we develop an IR system to find proper existing summaries as soft templates. Moreover, [Guu *et al.*, 2017] used a general seq2seq framework while we extend the seq2seq framework to conduct template reranking and template-aware summary generation simultaneously. Some researchers have also explored to combine extraction and generation within the seq2seq framework. For example, in the task of machine reading comprehension, [Tan *et al.*, 2017a] developed an extraction-then-synthesis framework to synthesize answers from extraction results. Specifically, the answer extraction model was first employed to predict the most important sub-spans from the passage as evidence, and the answer synthesis model took the evidence as additional features along with the question and passage to further elaborate the final answers.

5.3 Method

As shown in Figure 5.1, our summarization system is composed of three modules, i.e., Retrieve, Rerank and Rewrite. Given the input sentence $\mathbf{x}$, the Retrieve module picks up candidate soft templates $\mathbf{C} = \{\mathbf{r}_i\}$ from the training dataset. For validation and test, we regard the candidate template with the highest predicted saliency (a.k.a informativeness) score as the actual soft template $\mathbf{r}$. For training, we choose the one with the maximal actual saliency score in $\mathbf{C}$, which speeds up convergence and shows no obvious side effect in the experiments. Similar idea is also used in the work of [Tan

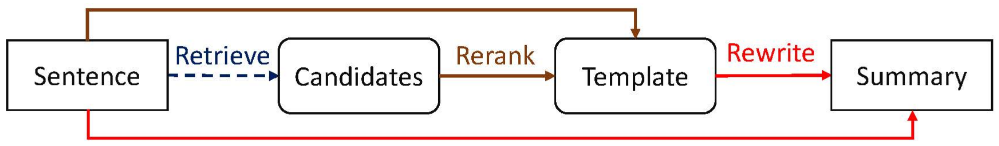


Figure 5.1: Flow chat of the proposed method. We use the dashed line for Retrieve since there is an IR system embedded.

et al., 2017a].

Then, we jointly conduct reranking and rewriting through a shared encoder. Specifically, both the sentence $\mathbf{x}$ and the soft template $\mathbf{r}$ are converted into hidden states via a RNN encoder. In the Rerank module, we measure the saliency of $\mathbf{r}$ according to its hidden state relevance to $\mathbf{x}$. In the Rewrite module, a RNN decoder combines the hidden states of $\mathbf{x}$ and $\mathbf{r}$ to generate a summary $\mathbf{y}$. More details will be described in the rest of this section

5.3.1 Retrieve

The purpose of the Retrieve module in our work is to find out candidate templates from the training dataset. We assume that similar sentences should hold similar summary patterns. Therefore, given a sentence $\mathbf{x}$, we find out its analogies in the dataset and pick up their summaries as the candidate templates. Since the size of the dataset is quite large (over 3M), we leverage the widely-used Information Retrieve (IR) system Lucene[1] to index and search efficiently. There are two fundamental steps in an IR system, i.e.,

Index Construct the index map for the training dataset.

Search Retrieve relevant documents from the index map.

We keep the default settings of Lucene[2]. Pre-processing of a source sentence includes tokenization, lowercasing, stopword removal and stemming. The ranking function we use is BM25 [Robertson *et al.*, 1995].

[1]`https://lucene.apache.org/`

[2]TextField with EnglishAnalyzer

For each input sentence, we select top 30 ranking results as candidate templates. Notice that we remove exact matches when searching for sentences in the training dataset. Otherwise, the model is likely to deem the actual summary the same as the soft template.

5.3.2 Jointly Rerank and Rewrite

To conduct template-aware seq2seq generation (i.e., rewriting), it is a necessary step to encode both the source sentence $\mathbf{x}$ and the soft template $\mathbf{r}$ into hidden states. We propose to jointly conduct reranking and rewriting through a shared encoding step, as shown in Figure 5.2. Specifically, we employ bidirectional Recurrent Neural Network (BiRNN) encoders [Cho *et al.*, 2014b] to read $\mathbf{x}$ and $\mathbf{r}$, similar to previous chapters. For example, given the sentence $\mathbf{x}$, its hidden state of the forward RNN at timestamp i can be represented by:

$$\overrightarrow{\mathbf{h}}_i^x = \text{RNN}(x_i, \overrightarrow{\mathbf{h}}_{i-1}^x) \tag{5.1}$$

In experiments, we use Long Short-term Memory (LSTM) [Gers *et al.*, 1999] as the recurrent unit. Its computation is as follows:

$$\mathbf{f}_t = \sigma([\overrightarrow{\mathbf{h}}_{t-1}, \mathbf{x}_t]\mathbf{W}_f + b_f) \tag{5.2}$$

$$\mathbf{i}_t = \sigma([\overrightarrow{\mathbf{h}}_{t-1}, \mathbf{x}_t]\mathbf{W}_i + b_i) \tag{5.3}$$

$$\mathbf{f}_o = \sigma([\overrightarrow{\mathbf{h}}_{t-1}, \mathbf{x}_t]\mathbf{W}_o + b_f) \tag{5.4}$$

$$\widetilde{\mathbf{l}}_t = \tanh([\overrightarrow{\mathbf{h}}_{t-1}, \mathbf{x}_t]\mathbf{W}_l + b_l) \tag{5.5}$$

$$\mathbf{l}_t = \mathbf{f}_t \odot \mathbf{l}_{t-1} + \mathbf{i}_t \odot \widetilde{\mathbf{l}}_t \tag{5.6}$$

$$\overrightarrow{\mathbf{h}}_t = \mathbf{o}_t \odot \tanh(\mathbf{l}_t) \tag{5.7}$$

where $\mathbf{i}_t$, $\mathbf{o}_t$ and $\mathbf{f}_t$ are gates for input, output and forget, respectively. $\odot$ denotes element-wise multiplication while $b_.$ and $\mathbf{W}_.$ are model weights. The BiRNN is composed of a forward RNN and a backward RNN. Suppose the corresponding outputs are $[\overrightarrow{\mathbf{h}}_1^x; \cdots; \overrightarrow{\mathbf{h}}_{-1}^x]$ and $[\overleftarrow{\mathbf{h}}_1^x; \cdots; \overleftarrow{\mathbf{h}}_{-1}^x]$, respectively, where the index "-1" stands for the last element. Then, the composite hidden state of a word is the concatenation of the

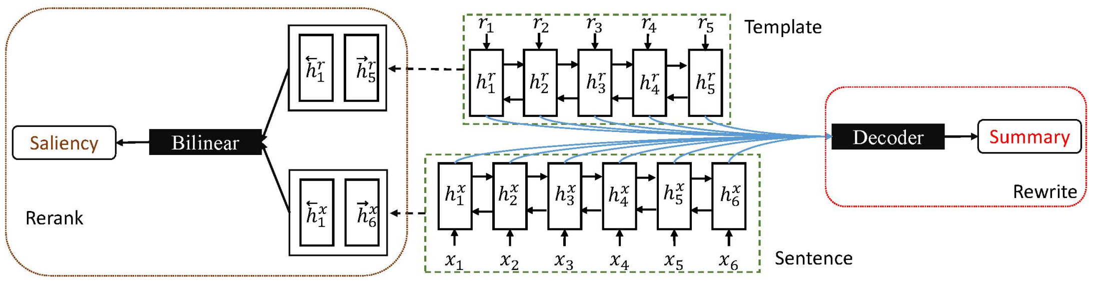


Figure 5.2: Jointly Rerank and Rewrite

two RNN representations, i.e., $\mathbf{h}_i^x = [\overrightarrow{\mathbf{h}}_i^x; \overleftarrow{\mathbf{h}}_i^x]$. The entire representation for the source sentence is $[\mathbf{h}_1^x; \cdots; \mathbf{h}_{-1}^x]$. Since a soft template $\mathbf{r}$ can also be regarded as a readable concise sentence, we use the BiRNN encoder with the same structure to convert it into hidden states $[\mathbf{h}_1^r; \cdots; \mathbf{h}_{-1}^r]$.

Rerank

In Retrieve, the template candidates are ranked according to the text similarity between the input sentence and corresponding indexed sentences. However, for the summarization task, we expect the soft template $\mathbf{r}$ resembles the actual summary $\mathbf{y}^*$ as much as possible. Here we use the widely-used summarization evaluation metrics ROUGE [Lin, 2004] to measure the actual saliency $s^*(\mathbf{r}, \mathbf{y}^*)$ (see Section 5.4.2). We utilize the hidden states of $\mathbf{x}$ and $\mathbf{r}$ to predict the saliency s of the template. Specifically, we regard the output of the BiRNN as the representation of the sentence or template:

$$\mathbf{h}_x = [\overleftarrow{\mathbf{h}}_1^x; \overrightarrow{\mathbf{h}}_{-1}^x] \tag{5.8}$$

$$\mathbf{h}_r = [\overleftarrow{\mathbf{h}}_1^r; \overrightarrow{\mathbf{h}}_{-1}^r] \tag{5.9}$$

Next, we use the Bilinear network to predict the saliency of the template for the input sentence.

$$s(\mathbf{r}, \mathbf{x}) = \text{sigmoid}(\mathbf{h}_r \mathbf{W}_s \mathbf{h}_x^{\mathbf{T}} + b_s), \tag{5.10}$$

where $\mathbf{W}_s$ and b_s are parameters of the Bilinear network. We add the sigmoid activation function to make the range of s consistent with the actual saliency s^*. According to

[Chen *et al.*, 2016], Bilinear outperforms multi-layer forward neural networks in measuring relevance. As shown later, the difference of s and s^* will provide additional supervisions for the seq2seq framework.

Rewrite

The soft template $\mathbf{r}$ selected by the Rerank module has already competed with the state-of-the-art method with regard to ROUGE evaluation (see Table 5.4). However, $\mathbf{r}$ usually contains a lot of named entities which never appear in the source (see Table 5.5). Consequently, it is hard to ensure that the soft templates are faithful to the input sentences. Therefore, we leverage the seq2seq model that has the strong rewriting ability to generate more faithful and informative summaries. Specifically, since the input of our system consists of both sentence and soft template, we use the concatenation function to combine the hidden states of the sentence and the template:

$$\mathbf{H}_c = [\mathbf{h}_1^x; \cdots; \mathbf{h}_{-1}^x; \mathbf{h}_1^r; \cdots; \mathbf{h}_{-1}^r] \tag{5.11}$$

We also attempted complex combination approaches such as the gate network [Cao *et al.*, 2017b] but failed to achieve obvious improvement. We assume the Rerank module has partially played the role of the gate network. The combined hidden states are then fed into the prevailing attentional RNN decoder [Bahdanau *et al.*, 2014] to generate the decoding hidden state at the position t:

$$\mathbf{s}_t = \text{Att-RNN}(\mathbf{s}_{t-1}, y_{t-1}, \mathbf{H}_c), \tag{5.12}$$

where y_{t-1} is the previous output summary word. Specifically, the attention mechanism uses a dynamically changing context $\mathbf{c}_t$ to replace $\mathbf{H}_c$. We follow the common practice to represent $\mathbf{c}_t$ as the weighted sum of $\{\mathbf{H}_c\}$:

$$\alpha_{t\tau} = \frac{e^{\eta(\mathbf{s}_{t-1}, \mathbf{H}_\tau)}}{\sum_{\tau'=1}^{n} e^{\eta(\mathbf{s}_{t-1}, \mathbf{H}_{\tau'})}} \quad , \forall \tau \in [1, n] \tag{5.13}$$

$$\mathbf{c}_t = \sum\nolimits_{\tau=1}^{n} \alpha_{t\tau} \mathbf{H}_\tau \tag{5.14}$$

where $\alpha_{t\tau}$ reflects the alignments between source and target words, and η is the function that shows the correspondence strength for attention. Here we use the formula of [Luong *et al.*, 2015]:

$$\eta(\mathbf{s}_{t-1}, \mathbf{H}_\tau) = \mathbf{v}_a \tanh(\mathbf{s}_{t-1}\mathbf{W}_a + \mathbf{H}_\tau \mathbf{U}_a) \tag{5.15}$$

where $\mathbf{v}_a$, $\mathbf{W}_a$ and $\mathbf{U}_a$ are model parameters. Then the decoder generates the current hidden state $\mathbf{s}_t$ using another LSTM like the one used in the encoder:

$$\mathbf{f}_t = \sigma([\mathbf{s}_{t-1}, \mathbf{y}_{t-1}, \mathbf{c}_t]\mathbf{U}_f + \mathbf{u}_f) \tag{5.16}$$

$$\mathbf{i}_t = \sigma([\mathbf{s}_{t-1}, \mathbf{y}_{t-1}, \mathbf{c}_t]\mathbf{U}_i + \mathbf{u}_i) \tag{5.17}$$

$$\mathbf{o}_t = \sigma([\mathbf{s}_{t-1}, \mathbf{y}_{t-1}, \mathbf{c}_t]\mathbf{U}_o + \mathbf{u}_o) \tag{5.18}$$

$$\widetilde{\mathbf{l}}_t = \tanh([\mathbf{s}_{t-1}, \mathbf{y}_{t-1}, \mathbf{c}_t]\mathbf{U}_l + \mathbf{u}_l) \tag{5.19}$$

$$\mathbf{l}_t = \mathbf{f}_t \odot \mathbf{l}_{t-1} + \mathbf{i}_t \odot \widetilde{\mathbf{l}}_t \tag{5.20}$$

$$\mathbf{s}_t = \mathbf{o}_t \odot \tanh(\mathbf{l}_t) \tag{5.21}$$

where $\mathbf{U}_.$ and $\mathbf{u}_.$ are model parameters.

Finally, a $softmax$ layer is built to predict the current summary word based on $\mathbf{s_t}$, y_{t-1} and $\mathbf{c_t}$:

$$\mathbf{p_t} = softmax([\mathbf{s}_t, \mathbf{y}_{t-1}, \mathbf{c}_t]\mathbf{W}_p) \tag{5.22}$$

where $\mathbf{W}_p$ is a parameter matrix.

5.3.3 Learning

There are two types of costs in our system. For Rerank, we expect the predicted saliency $s(\mathbf{r}, \mathbf{x})$ close to the actual saliency $s^*(\mathbf{r}, \mathbf{y}^*)$. Therefore, we apply the Cross Entropy (CE) between s and s^* as the loss function:

$$\begin{aligned} J_R(\theta) &= \mathrm{CE}(s(\mathbf{r}, \mathbf{x}), s^*(\mathbf{r}, \mathbf{y}^*)) \\ &= -s^* \log s - (1 - s^*) \log(1 - s), \end{aligned} \tag{5.23}$$

where θ stands for the model parameters. For Rewrite, the learning objective is to maximize the estimated probability of the actual summary $\mathbf{y}^*$. We adopt the common Negative Log-Likelihood (NLL) as the loss function:

$$\begin{aligned} J_G(\theta) &= -\log(p(\mathbf{y}^*|\mathbf{x}, \mathbf{r})) \\ &= -\sum\nolimits_t \log(\mathbf{p}_t[y_t^*]) \end{aligned} \tag{5.24}$$

To make full use of supervisions from both sides, we combine the above two costs as the final loss function:

$$J(\theta) = J_R(\theta) + J_G(\theta) \tag{5.25}$$

We use Stochastic Gradient Descent (SGD) with mini-batch to tune model parameters. The batch size is 64. To enhance generalization, we introduce dropout [Srivastava *et al.*, 2014] with probability $p = 0.3$. The initial learning rate is 1, and it decays by 50% if the generation loss does not decrease on the validation dataset.

5.4 Experiments

5.4.1 Dataset

We conduct experiments on the Annotated English Gigaword dataset. The details of this dataset is shown in Section 3.4.

5.4.2 Evaluation Metrics

We adopt ROUGE [Lin, 2004] for automatic evaluation. Following the common practice, we compute ROUGE-1 (uni-gram), ROUGE-2 (bi-gram) and ROUGE-L (LCS) F1 scores in the following experiments. More details can refer to Section 3.4.2. We also measure the actual saliency of a candidate template $\mathbf{r}$ with its combined ROUGE scores given the actual summary $\mathbf{y}^*$:

$$s^*(\mathbf{r}, \mathbf{y}^*) = \text{ROUGE-1}(\mathbf{r}, \mathbf{y}^*) + \text{ROUGE-2}(\mathbf{r}, \mathbf{y}^*) \tag{5.26}$$

ROUGE mainly evaluates informativeness. We also introduce a series of metrics to measure the summary quality from the following aspects:

LEN_DIF The absolute value of the length difference between the generated summaries and the actual summaries. We use mean value$\pm$standard deviation to present this aspect. The average value partially reflects readability and informativeness, while the standard deviation links to stability.

LESS_3 The number of the generated summaries containing less than three tokens. These extremely short summaries are usually unreadable.

COPY The proportion of the summary words (excluding stopwords) copied from the source sentence. A very large copy ratio indicates that the summarization system pays more attention to compression rather than required abstraction.

NEW_NE The number of the named entities which never appear in the source sentence or actual summary. Intuitively, the appearance of new named entities in the summary is likely to bring unfaithfulness. We use Stanford CoreNLP [Manning *et al.*, 2014] to recognize named entities.

NEW_UP The number of the uppercase words in neither the source sentence nor the actual summary. We use the "trucase" annotator of CoreNLP to find them. An uppercase word usually represent a named entity or an adjective related to a country (e.g., Chinese, English). The appearance of a new uppercase word in the summary is improper in most cases.

5.4.3 Implementation Details

We use the popular seq2seq framework OpenNMT[3] as the starting point. To make our model more general, we retain the default settings of OpenNMT to build the network architecture. Specifically, the dimensions of word embeddings and RNN are both

[3]`https://github.com/OpenNMT/OpenNMT-py`

500, and the encoder and decoder structures are two-layer bidirectional Long Short Term Memory Networks (LSTMs). The only difference is that we add the argument "-share_embeddings" to share the word embeddings between the encoder and decoder. This practice largely reduces model parameters for the monolingual task. Since a soft template is introduced as additional source text, the training time of our model approaches the standard seq2seq model.

During test, we use beam search of size 5 to generate summaries. We add the argument "-replace_unk" to replace the generated unknown words with the source word that holds the highest attention weight. Since the generated summaries are often shorter than the actual ones, we introduce an additional length penalty argument "-alpha 1" to encourage longer generation, like [Wu *et al.*, 2016].

5.4.4 Baselines

We introduce the following state-of-the-art neural summarization systems for comparison:

ABS The first neural abstractive summarization system proposed by [Rush *et al.*, 2015a].

ABS+ [Rush *et al.*, 2015a] further tuned the ABS model with additional features to balance the abstractive and extractive tendency.

RAS-Elman As the extension of the ABS model, it changed the decoder with RNN [Chopra *et al.*, 2016].

Featseq2seq [Nallapati *et al.*, 2016a] used a full seq2seq model and added the hand-crafted features such as POS tag and NER, to strengthen the encoder representation.

Luong-NMT [Chopra *et al.*, 2016] implemented the neural machine translation model of [Luong *et al.*, 2015] for summarization. This model contained two-layer LSTMs for both encoder and decoder.

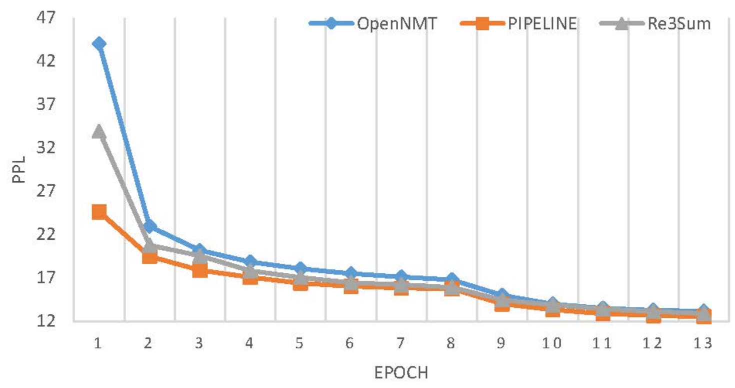


Figure 5.3: Perplexity change on the validation dataset.

OpenNMT We also implement the standard attentional seq2seq model with OpenNMT. All the settings are the same as our system. Noted that Gigaword is a benchmark dataset of OpenNMT. We distinguish the official result[4] and our experimental result with suffixes "O" and "I" respectively.

In addition, to check the power of our joint learning framework, we develop a baseline named "PIPELINE". Its architecture is identical to Re^{3}Sum. However, it trains the Rerank module and Rewrite module in pipeline.

5.4.5 Informativeness Evaluation

Model	Perplexity
ABS†	27.1
RAS-Elman†	18.9
OpenNMT$_I$	13.2
PIPELINE	**12.5**
Re^{3}Sum	12.9

Table 5.2: Final perplexity on the validation dataset. † indicates the value is cited from the corresponding paper. ABS+, Featseq2seq and Luong-NMT do not provide this value.

Let's first look at the cost values (Equation 5.24) on the validation dataset. From Table 5.2 and Figure 5.3, we can see that our model achieves much lower perplexity

[4]`http://opennmt.net/Models/`

Model	ROUGE-1	ROUGE-2	ROUGE-L
ABS[†]	29.55*	11.32*	26.42*
ABS+[†]	29.78*	11.89*	26.97*
Featseq2seq[†]	32.67*	15.59*	30.64*
RAS-Elman[†]	33.78*	15.97*	31.15*
Luong-NMT[†]	33.10*	14.45*	30.71*
OpenNMT$^{\dagger}_{O}$	33.13*	16.09*	31.00*
OpenNMT$_I$	35.01*	16.55*	32.42*
PIPELINE	36.49	17.48*	33.90
Re^{3}Sum	**37.04**	**19.03**	**34.46**

Table 5.3: ROUGE F1 (%) performance.

Type	ROUGE-1	ROUGE-2	ROUGE-L
Random	2.81	0.00	2.72
First	24.44	9.63	22.05
Max	38.90	19.22	35.54
Optimal	52.91	31.92	48.63
Rerank	28.77	12.49	26.40

Table 5.4: ROUGE F1 (%) performance of different types of soft templates.

compared against the state-of-the-art systems. It is also noted that PIPELINE slightly outperforms Re^{3}Sum. One possible reason is that Re^{3}Sum additionally considers the cost derived from the Rerank module.

The ROUGE F1 scores of different methods are then reported in Table 5.3. It is obvious that our model significantly outperforms most other approaches. Note that, ABS+ and Featseq2seq have utilized a series of hand-crafted features, but our model is completely data-driven. Even though, our model surpasses Featseq2seq by 22% and ABS+ by 60% on ROUGE-2. When soft templates are ignored, our model is equivalent to the standard attentional seq2seq model OpenNMT$_I$. Therefore, it is safe to conclude that soft templates have great contribute to guide generation of summaries.

We also examine the performance of directly regarding soft templates as output summaries. We introduce five types of different soft templates:

Random An existing summary randomly selected from the training dataset.

First The top-ranked candidate template given by the Retrieve module.

Max The template with the maximal actual ROUGE scores among the 30 candidate templates.

Optimal An existing summary in the training dataset which holds the maximal ROUGE scores.

Rerank The template with the maximal predicted ROUGE scores among the 30 candidate templates. It is the actual soft template we adopt.

As shown in Table 5.4, the performance of Random is terrible, indicating it is impossible to use one summary template to fit various actual summaries. Rerank largely outperforms First, which verifies the effectiveness of the Rerank module. However, according to Max and Rerank, we find the Rerank performance of Re^{3}Sum is far from perfect. By comparing Max and Optimal, we believe there is much room to improve the Retrieve module. Notice that Optimal greatly exceeds all the state-of-the-art approaches. This finding strongly supports our practice of using existing summaries to guide the seq2seq models.

5.4.6 Linguistic Quality Evaluation

Item	Template	OpenNMT	Re^{3}Sum
LEN_DIF	2.6±2.6	3.0±4.4	2.7±2.6
LESS_3	0	53	1
COPY(%)	31	80	74
NEW_NE	0.51	0.34	0.30
NEW_UP	0.38	0.19	0.11

Table 5.5: Statistics of different types of summaries.

Here we check the linguistic quality of generated summaries from various aspects, and the results are presented in Table 5.5. Look at the rows "LEN_DIF" and "LESS_3". The performance of Re^{3}Sum is almost the same as that of soft templates. It suggests that the soft templates indeed well guide summary generation. Compared with

Re^{3}Sum, the standard deviation of LEN_DF in OpenNMT is 0.7 times larger, indicating that OpenNMT works quite unstably. Moreover, OpenNMT generates 53 extreme short summaries, which seriously reduces readability. Meanwhile, the copy ratio of actual summaries is 36%. Therefore, the copy mechanism is severely overweighted in OpenNMT. Our model is encouraged to generate according to human-written soft templates, which relatively diminishes copying from the source sentences. Look at the last rows "NEW_NE" and "NEW_UP". A number of new named entities appear in the soft templates, which makes them quite unfaithful to source sentences. By contrast, NEW_NE in Re^{3}Sum is close to NEW_NE in OpenNMT while NEW_UP in Re^{3}Sum is much less than NEW_UP in OpenNMT. It demonstrates the rewriting ability of our seq2seq framework.

5.4.7 Effect of Templates

Type	ROUGE-1	ROUGE-2	ROUGE-L
+Random	32.60	14.31	30.19
+First	36.01	17.06	33.21
+Max	41.50	21.97	38.80
+Optimal	46.21	26.71	43.19
+Rerank(Re^{3}Sum)	37.04	19.03	34.46

Table 5.6: ROUGE F1 (%) performance of Re^{3}Sum generated with different soft templates.

Mode		OpenNMT$_I$	First	Max	Optimal	Rerank
Copy	$\mathbf{r}$	-	0.39 (0.51)	0.43 (0.70)	0.40 (0.85)	0.52 (0.52)
	$\mathbf{r}-\mathbf{x}$	-	0.14 (0.24)	0.13 (0.47)	0.14 (0.70)	0.09 (0.30)
	$\mathbf{r}\cap\mathbf{x}$	-	0.25 (0.66)	0.30 (0.80)	0.26 (0.93)	0.43 (0.57)
	$\mathbf{x}-\mathbf{r}$	-	0.48 (0.33)	0.45 (0.26)	0.48 (0.27)	0.31 (0.31)
	$\mathbf{x}$	0.80 (0.43)	0.73 (0.45)	0.75 (0.48)	0.74 (0.50)	0.74 (0.46)
Rewrite		0.20 (0.26)	0.27 (0.22)	0.25 (0.32)	0.26 (0.45)	0.26 (0.28)

Table 5.7: Copy and rewrite performance. Here $\mathbf{r}$ stands for templates and $\mathbf{x}$ means source sentences. The value form is: length proportion (correct proportion). To compare with OpenNMT, we regard all the words not copied from the source sentences as Rewrite.

Source	grid positions after the final qualifying session in the indonesian motorcycle grand prix at the sentul circuit , west java , saturday : UNK
Target	indonesian motorcycle grand prix grid positions
Template	grid positions for **british** grand prix
OpenNMT	circuit
Re^{3}Sum	grid positions for indonesian grand prix

Figure 5.4: Example 1 of generated summaries. We use Bold font to indicate the crucial rewriting behavior from the templates to generated summaries.

Source	anatoli UNK of kazakhstan shattered a world record of ###.# kilogrammes in the clean and jerk on monday on his way to winning the men 's ##kg class at the asian weightlifting championships .
Target	UNK breaks world weightlifting record
Template	**greek** breaks weightlifting world record
OpenNMT	UNK wins men 's ##kg class at asian weightlifting championships
Re^{3}Sum	kazakh breaks weightlifting world record

Figure 5.5: Example 2 of generated summaries.

Source	india 's children are getting increasingly overweight and unhealthy and the government is asking schools to ban junk food , officials said thursday .
Target	indian government asks schools to ban junk food
Template	**skorean** schools to ban **soda** junk food
OpenNMT	india 's children getting fatter
Re^{3}Sum	indian schools to ban junk food

Figure 5.6: Example 3 of generated summaries.

Source	anny ainge said thursday he had two one-hour meetings with the new owners of the boston celtics but no deal has been completed for him to return to the franchise .
Target	ainge says no deal completed with celtics
Templates	major says no deal with spain on gibraltar
	roush racing completes deal with red sox owner
Re^{3}Sum	ainge **says no deal done** with **celtics**
	ainge **talks** with **new owners**
OpenNMT	ainge talks with **celtics** owners
	ainge talks with **new** owners

Figure 5.7: Example 1 of generation with diversity. We use Bold font to indicate the difference between two summaries

Source	european stock markets advanced strongly thursday on some bargain-hunting and gains by wall street and japanese shares ahead of an expected hike in us interest rates .
Target	european stocks bounce back UNK UNK with closing levels
Templates	european stocks bounce back strongly
	european shares sharply lower on us interest rate fears
Re^{3}Sum	european **stocks bounce back** strongly
	european **shares rise** strongly **on bargain-hunting**
OpenNMT	european stocks rise ahead of **expected** us rate hike **hike**
	european stocks rise ahead of us rate hike

Figure 5.8: Example 2 of generation with diversity.

Source	the sudanese opposition said here thursday it had killed more than ### government soldiers in an ambush in the east of the country .
Target	sudanese opposition says ### government troops killed in ambush
Templates	# government troops killed in taliban ambush in south afghanistan
	sudanese rebel group says it killed ### government troops
Re^{3}Sum	sudanese opposition **claims over** ### government **soldiers** killed **in ambush**
	sudanese opposition **says it** killed ### government **troops**
OpenNMT	sudanese opposition says more than ### soldiers killed **in ambush**
	sudanese opposition says more than ### soldiers killed

Figure 5.9: Example 3 of generation with diversity.

In this section, we show how soft templates affect our model. At the beginning, we feed different types of soft templates (refer to Table 5.4) into the Rewriting module of Re^{3}Sum. As illustrated in Table 5.6, the more high-quality templates are provided, the higher ROUGE scores are achieved. It is interesting to see that,while the ROUGE-2 score of Random templates is zero, our model can still generate acceptable summaries with Random templates. It seems that Re^{3}Sum can automatically judge whether the soft templates are trustworthy and ignore the seriously irrelevant ones. We believe that the joint learning with the Rerank model plays a vital role here. We then examine the copying and rewriting performance of our system. The result is presented in Table 5.7. We find that the most accurate prediction happens in the overlap of source sentences and templates ($\mathbf{r} \cap \mathbf{x}$). We believe that templates reiterate salient words in the source sentences. In addition, the copying behavior in templates is always more accurate than that in source sentences. For Max and Optimal templates, this difference is extremely significant. This fact highlights the effectiveness of high-quality templates. Compared with OpenNMT, our model achieves improvement on both copying and rewriting behaviors. As mentioned above, copying mainly benefits from the overlap part between templates and sentences. For rewriting, the unique words in templates ($\mathbf{r} - \mathbf{x}$) transfer rewriting from source sentences into copying in templates, while the latter usually works better.

Next, we manually inspect the summaries generated by different methods. We find the outputs of Re^{3}Sum are usually longer and more fluent than the outputs of OpenNMT. Some illustrative examples are shown in Figure2 5.4, 5.5 and 5.6. In Example 1, there is no predicate in the source sentence. Since OpenNMT prefers selecting source words around the predicate to form the summary, it fails on this sentence. By contract, Re^{3}Sum rewrites the template and produces an informative summary. In Example 2, OpenNMT deems the starting part of the sentences are more important, while our model, guided by the template, focuses on the second part to generate the summary.

In the end, we test the ability of our model to generate diverse summaries. In practice, a system that can provide various candidate summaries is probably more welcome. Specifically, two candidate templates with large text dissimilarity are manually fed into the Rewriting module. The corresponding generated summaries are shown in Figures 5.7, 5.8 and 5.9. For the sake of comparison, we also present the 2-best results of OpenNMT with beam search. As can be seen, with different templates given, our model is likely to generate dissimilar summaries. In contrast, the 2-best results of OpenNMT are almost the same, and often a shorter summary is only a piece of the other one. To sum up, our model demonstrates promising prospect in generation diversity.

5.4.8 Using Restricted Decoder

Seen from Table 5.5, there are still a number of new named entities and new capitalized word in the generated summaries. It may worsen the faithfulness of our model. As shown next, this problem becomes much more serious when using other types of templates (First, Max or Optimal). Therefore, we try to integrate the restricted decoder (RD) proposed in Chapter 3 to control the vocabulary of the generation. Specifically,

Data		NEW_NE	NEW_UP
Template	First	1.93	1.02
	Max	1.77	0.98
	Optimal	1.26	0.82
	Rerank	0.51	0.38
Summary	First	1.31	0.27
	Max	1.21	0.22
	Optimal	1.08	0.17
	Rerank	0.30	0.11
RD	First	0.35	0
	Max	0.32	0
	Optimal	0.32	0
	Rerank	0.29	0

Table 5.8: Numbers of new named entities and new uppercase words in different data.

Type	ROUGE-1	ROUGE-2	ROUGE-L
First	36.41	17.23	33.52
Max	41.82	22.11	38.98
Optimal	46.44	26.91	43.36
Rerank	37.08	19.04	34.47

Table 5.9: ROUGE F1 (%) performance of generation with restricted decoder.

the permissible output vocabulary is:

$$\mathbf{V}_Y = \mathbf{X} \cup \mathbf{A}(\mathbf{X}) \cup \mathbf{U} \tag{5.27}$$

where $\mathbf{X}$ is the source vocabulary, $\mathbf{A}(\mathbf{X})$ means the alignments of source words (obtained from the attention mechanism) and $\mathbf{U}$ is a frequent word set. In this way, the results of NEW_NE and NEW_UP are shown in Table 5.8. Obviously, after controlling the output vocabulary, the generation of new named entities becomes rare, and the new uppercase words even disappear. Meanwhile, since this restriction aims to reduce surely unreasonable words in the generated summaries, the ROUGE performance also enjoys a consistent increase, as shown in Table 5.9.

5.5 Summary

In this chapter, we propose to introduce soft templates as additional input to guide seq2seq summarization. We use the popular IR platform Lucene to retrieve proper existing summaries as candidate soft templates. Then we extend the seq2seq framework to jointly conduct template reranking and template-aware summary generation. Experiments show that our model can generate informative, readable and stable summaries. In addition, our model demonstrates promising prospect in generation diversity.

We believe our work can be extended in various aspects in the future. Since the candidate templates are far inferior to the optimal ones, we intend to improve the Retrieve module, e.g., by searching both the sentence and summary. We also plan to test our system on the other tasks such as document-level summarization and short text conversation.

Chapter 6

Future Work: Rewrite for Personalization

6.1 Introduction

Chapter 5 has verified that our summarization system is able to generate diverse summaries. We find it is natural to extend this idea to personalized summarization which aims to generate summaries specific to the user's interest. Personalized summarization is not a new idea. The early work on summarization [Luhn, 1958] has already mentioned the possibility of personalizing summaries by adapting them to particular areas of interest. To this end, sentences that contain words related to the reader's interests are given more importance. Experiments on summary user evaluation [Paice and Jones, 1993] show that users tend to select the parts of the text that are more closely related to their interests. As a result, personalized summaries should be far more attractive than a general one. However, the research of personalized summarization is limited largely due to the **lack of labeled data**. As a result, most previous personalized summarization systems (e.g., [Díaz *et al.*, 2005; Díaz and Gervás, 2007; Kumar *et al.*, 2008]) had to simply adopt unsupervised extractive approaches and model the interest of a user merely by the surface features such as given keywords and user's profile. During evaluation, annotators were asked to "imagine" whether the users would like the generated summaries.

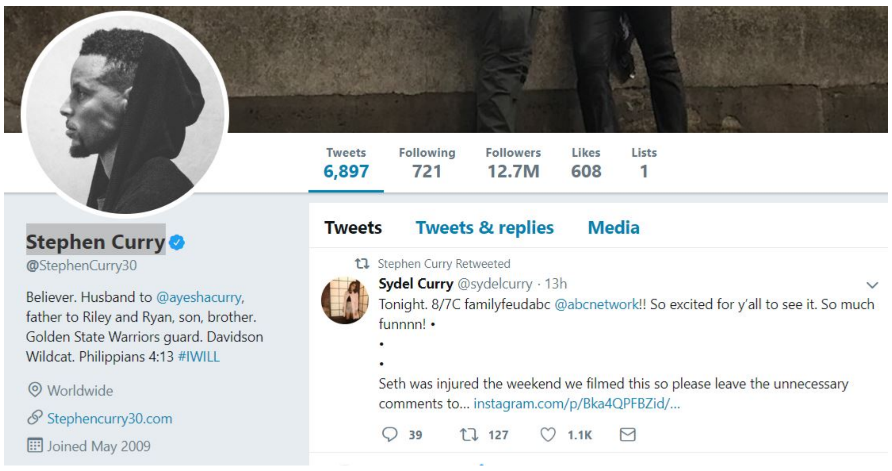
Our future work on personalized summarization is two-fold. On the one hand, we plan to collect a large-scale personalized summarization dataset automatically from social media platforms. In such way, we can conduct experiments on real data. On the other hand, we are going to develop a neural abstractive personalized summarization system by extending our previous work.

6.2 Personalized Summary Dataset Collection

Social media (e.g., Twitter[1] and Weibo[2]) have surged in the recent decade. Numerous tweets have been published every day. Since tweets are written by users, they naturally sculpt their preferences and personal interests. We suggest an effective way to automatically collect large-scales of personalized summaries on Twitter. Previously, a series of NLP tasks have tried to utilize social annotations like followers [Chen *et al.*, 2014], emoticons [Zhao *et al.*, 2012] and replies [Hu *et al.*, 2014] etc. Two kinds of common social labels, i.e., **hyper-link** and **user ID** can be leveraged for our purpose. First of all, our previous work [Cao *et al.*, 2016] has validated that a tweet with a hyper-link can be regarded as the summary of the corresponding document. Take tweets in Table 6.1 as an example. This document describes how Greece's Crisis drowns the life of a sardine fisherman. Since the local fish industry owns the world first robotic sardine processing line, Tweet 1 is interested in how it works. Tweet 2 and 3 both describe the key points of this paper, like "Greece", "fish industry" and "crisis", but Tweet 2 can be used as the title while Tweet 3 tells author's writing experience. As can been seen, each tweet summarizes a part of the document with its own focus and writing style.

Furthermore, given the author ID, we can easily access his/her profile and social behaviors, which largely benefits targeting the preference. For example, the main page of Stephen Curry (user ID: StephenCurry30) on Twitter is shown in Figure 6.1. From this page, we extract plentiful user information, such as:

[1]`https://twitter.com`
[2]`https://weibo.com`



Tweets 6,897 Following 721 Followers 12.7M Likes 608 Lists 1

Stephen Curry
@StephenCurry30

Believer. Husband to @ayeshacurry, father to Riley and Ryan, son, brother. Golden State Warriors guard. Davidson Wildcat. Philippians 4:13 #IWILL

Worldwide

Stephencurry30.com

Joined May 2009

Stephen Curry Retweeted

Sydel Curry @sydelcurry · 13h

Tonight. 8/7C familyfeudabc @abcnetwork!! So excited for y'all to see it. So much funnnn! •

•

•

Seth was injured the weekend we filmed this so please leave the unnecessary comments to... instagram.com/p/Bka4QPFBZid/...

39 127 1.1K

Figure 6.1: An example of user information shown in Twitter

Profile Family, location, occupation and so on.

Relationship Following, followers and likes.

Writing History All the tweets created by the user.

It is noted that user modeling is one of the most fundamental tasks in the area of social network research (e.g., [Abel *et al.*, 2011]), and word embedding-like user representation becomes a growing tendency (e.g., [Elkahky *et al.*, 2015]). Therefore, we plan to apply an existing model to pre-train user embeddings to represent user's preference in our work.

The whole data collection process is described as follows:

1. Find out the tweets containing hyper-links through the Twitter search function to find out tweets containing hyper-links.

2. Retrieve hyper-links to acquire the corresponding documents, for example, using the open source Python package "newspaper"[3].

3. Collect the user information with the Twitter user streaming API.

[3]`https://pypi.python.org/pypi/newspaper`

Tweet	Software determines which are sardines and ... how much they weigh
	A fish tale: How #GreeceCrisis drowned an industry and a way of life
	.@georgikantchev, @movingpicturetv, @vaniabturner and I look inside Greece's fish industry and see crisis
News	The Fisherman's Lament – A Way of Life Drowned by Greece's Crisis (http://t.co/FXGTUY3IBq)

Table 6.1: An example of the tweets linked to the same news. Hyper-links and user IDs in tweets are elided for short.

6.3 Personalized Neural Abstractive Summarization

In Chapter 5, we have demonstrated that our system Re^{3}Sum is able to generate diverse summaries given different templates. Naturally, if we provide a user-specific template, the summary should reflect personalization. We plan to extend Re^{3}Sum by introducing user information to generate personalized summaries. At first, we are considering to make the Retrieve module of our Re^{3}Sum associated with the Rerank and Rewrite modules to improve the recall of salient templates. To achieve this goal, we are going to introduce an additional output vocabulary prediction layer onto the encoder. This layer works in two-fold. On the one hand, the predicted vocabulary can be used in the Retrieve module to conduct composite search. That is, for each retrieved result, its sentence must be similar to the input sentence and its summary must be close to the predicted summary words. We believe this practice is likely to significantly improve the recall of Retrieve. On the other hand, the predicted vocabulary can be naturally adopted in restricted generation describe in Section 5.4.8. Since vocabulary restriction plays a crucial role to reduce unreasonable generation, it should work more accurately to embed it as a learning module of the whole system. The new framework is shown in Figure 6.2. The work flow of the extended system is as follows:

1. Input a source sentence $\mathbf{x}$.

2. Encode it and $\underline{\text{predict the target vocabulary } \mathbf{V}_y}$.

3. The Retrieve module searches candidate templates through $\mathrm{sentence} \approx \mathbf{x}$ and

$\underline{\mathrm{headline} \approx \mathbf{V}_y}$.

4. Ihe Rerank module finds out the most salient template $\mathbf{r}$ within the candidates.

5. The Rewrite module generates a summary based on $\mathbf{x}$ and $\mathbf{r}$, $\underline{\text{restricted in the vocabulary } \mathbf{V}_y}$.

We use the underline to emphasize the extension.

To further import user information, we plan to add pre-trained user embedding to makes all the modules of Retrieve, Rerank and Rewrite personalized. The entire framework is shown in Figure 6.3. We consider to use a state-of-the-art approach to pre-train user embeddings on the collected user information. Then, the work flow of the extended system can be as follows.

1. Input a source sentence $\mathbf{x}$ and $\underline{\text{a user ID}}$.

2. Encode the sentence and project the user ID onto user embeddings $\mathbf{u}$

3. Predict the $\underline{\text{user-specific}}$ target vocabulary $\mathbf{V}_y^u$, based on the sentence representation $\underline{\text{and user embeddings}}$.

4. Retrieve searches candidate templates through $\mathrm{document} \approx \mathbf{x}$ and $\underline{\mathrm{tweet} \approx \mathbf{V}_y^u}$.

5. Rerank finds out the most $\underline{\text{user-specific}}$ salient template $\mathbf{r}^u$ within the candidates.

6. Rewrite generates a summary based on $\mathbf{x}$, $\mathbf{r}$ and $\mathbf{u}$, $\underline{\text{restricted in the vocabulary } \mathbf{V}_y^u}$.

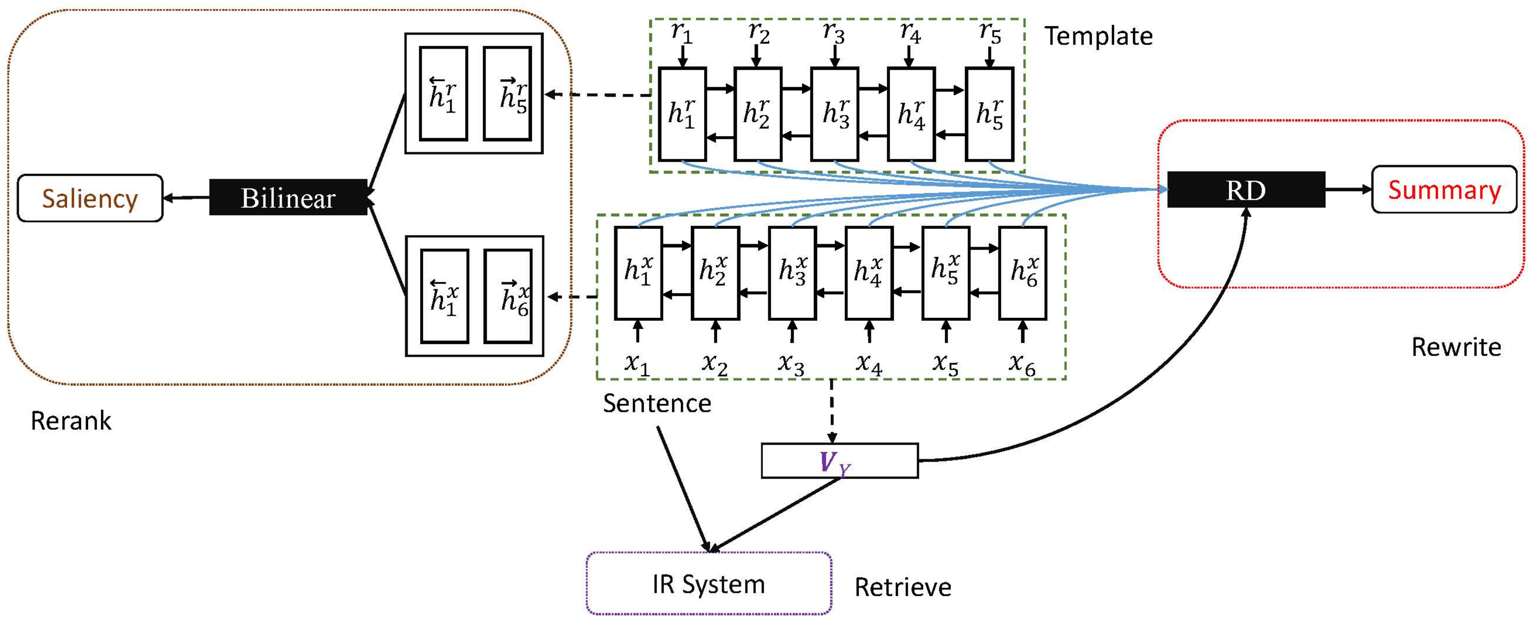
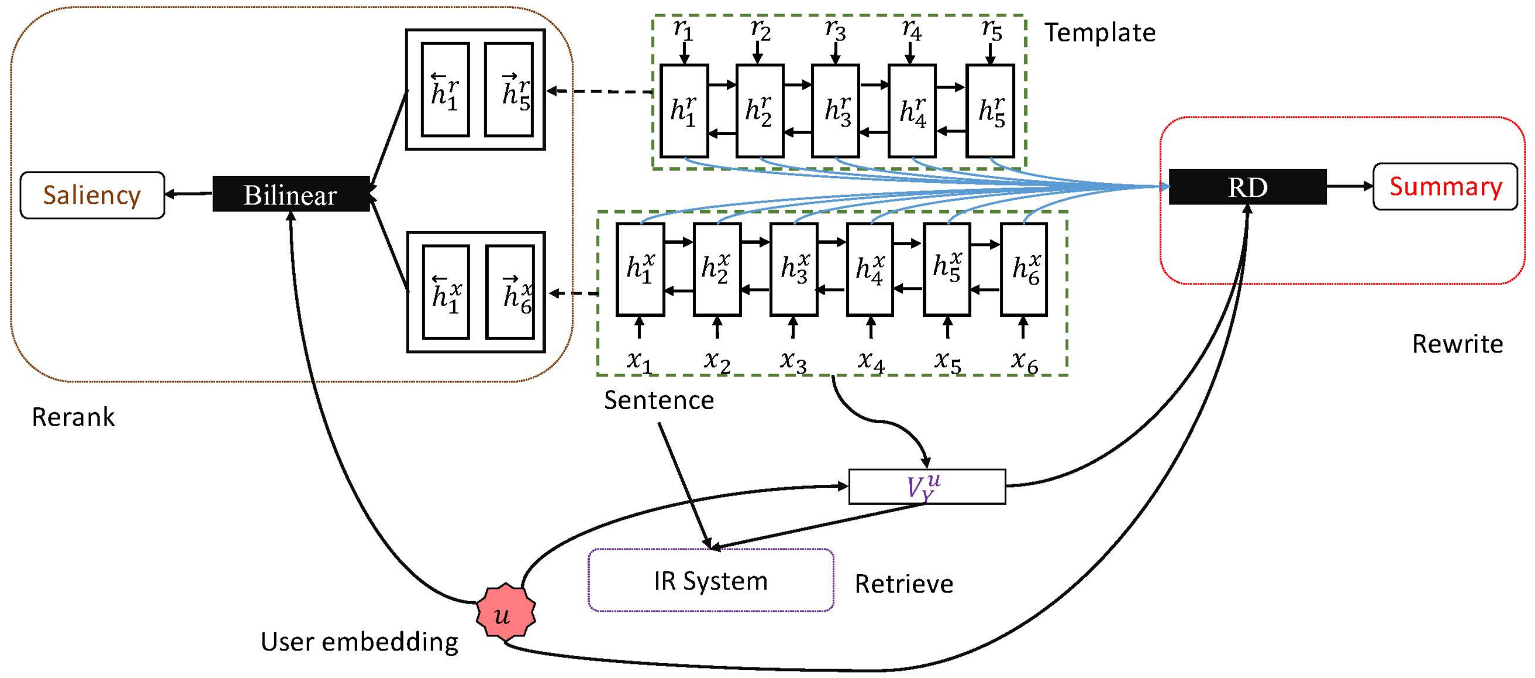


Figure 6.2: Framework of improved template-aware seq2seq summarization.



Template
r_1 r_2 r_3 r_4 r_5
h_1^r h_2^r h_3^r h_4^r h_5^r
h_1^x h_2^x h_3^x h_4^x h_5^x h_6^x
x_1 x_2 x_3 x_4 x_5 x_6
Sentence
$\overleftarrow{h}_1^r$ $\overrightarrow{h}_5^r$
$\overleftarrow{h}_1^x$ $\overrightarrow{h}_6^x$
Saliency
Bilinear
Rerank
RD
Summary
Rewrite
V_Y^u
IR System
Retrieve
u
User embedding

Figure 6.3: Framework of our personalized summarization system.

Chapter 7

Conclusion

In this last chapter, I wrap up the thesis by combining the there technical chapters into a thorough view of copying and rewriting driven neural abstractive summarization.

7.1 Research Summary

We explore two core writing behaviors in summarization, i.e., copying and rewriting, and incorporate them in seq2seq models. Our work is unique in two aspects. On the one hand, we introduce the prior knowledge of traditional summarization research into seq2seq to enable more informative, faithful and diverse generation. On the other hand, we design novel neural network architectures to explicitly capture copying and rewriting behaviors in summarization. The technical details are presented as follows.

In Chapter 3, we leverage the popular attention mechanism to copy and rewrite words in the source text. Our system, called **CoRe**, fuses a copying decoder and a rewriting decoder. The copying decoder finds out the words to be copied in the source text according to the attention weights. The rewriting decoder produces other necessary summary words limited in the source-specific vocabulary derived from learned alignments. To combine the two decoders and determine the final output, we train a predictor to predict the actual writing modes, which is able to utilize additional supervision. Extensive experiments show that our model is capable of generating informative summaries efficiently.

In Chapter 4, we investigates an important but neglected problem, i.e., the faithfulness problem in abstractive summarization. We propose a fact-aware summarization system, called **FTSum**, to copy source facts. We employ open information extraction and dependency parsing tools to extract the facts in the source sentence, which was a common practice in compressive summarization. We develop the dual-attention seq2seq framework to force summary generation conditioned on both the source sentence and the extracted facts. Automatic and human evaluations demonstrate that our summarization system greatly outperforms state-of-the-art models on both informativeness and faithfulness. In addition, we annotate a first dataset with faithfulness labels and propose an effective automatic faithfulness evaluation tool, called **FTEval**.

In Chapter 5, we propose a system, called **Re^{3}Sum**, which uses soft templates as rewriting references to guide seq2seq summarization. This is inspired by template-based summarization. We use Lucene, a popular information retrieval tool, to retrieve appropriate existing summaries as candidate soft templates. We extend the seq2seq framework by jointly learning template reranking and template-aware summary generation. Experiments show that our system can generate informative, readable and stable summaries. Our system also demonstrates promising prospect in generation diversity.

Our future work on personalized abstractive summarization is described in Chapter 6. We plan to collect a large-scale personalized summarization dataset automatically from social media platforms. In such a way, we can conduct experiments on real data, which is usually the most significant hindrance to the previous work. Our aim is to extend the current systems proposed in this thesis to model user-aware rewriting.

The pros and cons of our work are summarized in Table 7.1. Compared with the standard seq2seq model, all the proposed methods require additional pre-processing steps. That is, CoRe initializes an alignment table, FTSum uses OpenIE and dependency parsing to extract facts, and Re^{3}Sum leverages an IR system to find out candidate soft templates. The latter two spend quite a lot of time. In terms of faithfulness, Re^{3}Sum may generate new named entities that appear in the soft template. It is noted that this problem can be solved by combining the rewriting decoder of CoRe (refer to Section 5.4.8).

Method	Pre-processing	ROUGE	Faithfulness
CoRe	−	+	+
FTSum	−−	++	+++
Re^{3}Sum	−−	+++	?*

Table 7.1: Pros and cons of proposed methods compared with the standard seq2seq model. *Unstable and can be addressed by introducing the rewriting decoder.

7.2 Technical Highlights

Our work mainly focuses on modeling the two core writing behaviors in summarization, i.e., copying and rewriting. We have proposed neural network approaches to explicitly model copying and rewriting during summary generation. We also incorporate the prior knowledge from traditional summarization researches into the seq2seq model. As such, the seq2seq model is guided to copy facts (inspired by compressive summarization techniques) and rewrite from templates (inspired by template-based summarization techniques). The specific technical contributions are listed as follows.

CoRe We separate the attention mechanism to explicitly model copying and rewriting source words, respectively. The copying decoder copy the important words from the source text based on the attention mechanism. The rewriting decoder is restricted to generate from the source-specific vocabulary, which is able to produce rewriting-related words efficiently. We introduce a binary sequence labeling task to predict the current writing mode, which utilizes additional supervision.

FTSum To the best of our knowledge, we are the first to approach the faithfulness problem in abstractive summarization. To address this issue, we propose the first fact-aware seq2seq model, which can generate faithful and informative summaries. We develop the dual-attention seq2seq model to force summary generation conditioned on both the source text and the facts extracted from the source text. We also develop the first automatic tool for faithfulness evaluation. FTEval highly corresponds with the manual judgment of faithfulness, and it gives explanations to all identified fake summaries. We expect that FTEval could be a benchmark tool for summary evaluation and awaken the research community to handle the serious fake generation in seq2seq summarization.

Re^{3}Sum We propose to introduce soft templates as additional input to improve readability and stability of seq2seq summarization systems. To achieve this goal, we extend the seq2seq framework to enable simultaneous template reranking and template-aware summary generation. In such a way, Re^{3}Sum fuses the IR-based ranking technique and seq2seq-based generation technique and fully takes advantages of both sides.

Bibliography

[Abel *et al.*, 2011] Fabian Abel, Qi Gao, Geert-Jan Houben, and Ke Tao. Analyzing user modeling on twitter for personalized news recommendations. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 1–12. Springer, 2011.

[Allamanis *et al.*, 2016] Miltiadis Allamanis, Hao Peng, and Charles Sutton. A convolutional attention network for extreme summarization of source code. *arXiv preprint arXiv:1602.03001*, 2016.

[Almeida and Martins, 2013] Miguel Almeida and Andre Martins. Fast and robust compressive summarization with dual decomposition and multi-task learning. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 196–206, 2013.

[Angeli *et al.*, 2015] Gabor Angeli, Melvin Johnson Premkumar, and Christopher D Manning. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL 2015)*, 2015.

[Ayana *et al.*, 2016] Shiqi Shen Ayana, Zhiyuan Liu, and Maosong Sun. Neural headline generation with minimum risk training. *arXiv preprint arXiv:1604.01904*, 2016.

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

[Banko *et al.*, 2000] Michele Banko, Vibhu O Mittal, and Michael J Witbrock. Headline generation based on statistical translation. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 318–325. Association for Computational Linguistics, 2000.

[Banko *et al.*, 2007] Michele Banko, Michael J Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. Open information extraction from the web. In *IJCAI*, volume 7, pages 2670–2676, 2007.

[Barrios *et al.*, 2016] Federico Barrios, Federico López, Luis Argerich, and Rosa Wachenchauzer. Variations of the similarity function of textrank for automated summarization. *arXiv preprint arXiv:1602.03606*, 2016.

[Barry *et al.*, 2007] Carol Barry, Charles T Meadow, Donald H Kraft, and Bert R Boyce. *Text Information Retrieval Systems.* Academic Press, 2007.

[Barzilay and Lee, 2003] Regina Barzilay and Lillian Lee. Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 16–23. Association for Computational Linguistics, 2003.

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

[Bengio *et al.*, 2003] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155, 2003.

[Bing *et al.*, 2015] Lidong Bing, Piji Li, Yi Liao, Wai Lam, Weiwei Guo, and Rebecca J Passonneau. Abstractive multi-document summarization via phrase selection and merging. *arXiv preprint arXiv:1506.01597*, 2015.

[Bowman *et al.*, 2015] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*, 2015.

[Cai *et al.*, 2010] Xiaoyan Cai, Wenjie Li, You Ouyang, and Hong Yan. Simultaneous ranking and clustering of sentences: a reinforcement approach to multi-document summarization. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 134–142. Association for Computational Linguistics, 2010.

[Cao *et al.*, 2015a] Ziqiang Cao, Furu Wei, Li Dong, Sujian Li, and Ming Zhou. Ranking with recursive neural networks and its application to multi-document summarization. In *Proceedings of AAAI*, 2015.

[Cao *et al.*, 2015b] Ziqiang Cao, Furu Wei, Sujian Li, Wenjie Li, Ming Zhou, and Houfeng Wang. Learning summary prior representation for extractive summarization. *Proceedings of ACL: Short Papers*, pages 829–833, 2015.

[Cao *et al.*, 2016] Ziqiang Cao, Chengyao Chen, Wenjie Li, Sujian Li, Furu Wei, and Ming Zhou. Tgsum: Build tweet guided multi-document summarization dataset. In *Proceedings of AAAI*, 2016.

[Cao *et al.*, 2017a] Ziqiang Cao, Chuwei Luo, Wenjie Li, and Sujian Li. Joint copying and restricted generation for paraphrase. In *AAAI*, pages 3152–3158, 2017.

[Cao *et al.*, 2017b] Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. Faithful to the original: Fact aware neural abstractive summarization. *arXiv preprint arXiv:1711.04434*, 2017.

[Carbonell and Goldstein, 1998] Jaime Carbonell and Jade Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of SIGIR*, pages 335–336, 1998.

[Chen *et al.*, 2014] Chengyao Chen, Dehong Gao, Wenjie Li, and Yuexian Hou. Inferring topic-dependent influence roles of twitter users. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 1203–1206. ACM, 2014.

[Chen *et al.*, 2016] Danqi Chen, Jason Bolton, and Christopher D Manning. A thorough examination of the cnn/daily mail reading comprehension task. *arXiv preprint arXiv:1606.02858*, 2016.

[Chen *et al.*, 2017] Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, Hui Jiang, and Diana Inkpen. Enhanced lstm for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, Vancouver, July 2017. ACL.

[Cheng and Lapata, 2016] Jianpeng Cheng and Mirella Lapata. Neural summarization by extracting sentences and words. *arXiv preprint arXiv:1603.07252*, 2016.

[Cho *et al.*, 2014a] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

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

[Cho *et al.*, 2015] Sébastien Jean Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. On using very large target vocabulary for neural machine translation. 2015.

[Chopra *et al.*, 2016] Sumit Chopra, Michael Auli, Alexander M Rush, and SEAS Harvard. Abstractive sentence summarization with attentive recurrent neural networks. *Proceedings of NAACL-HLT16*, pages 93–98, 2016.

[Chorowski *et al.*, 2014] Jan Chorowski, Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. End-to-end continuous speech recognition using attention-based recurrent nn: First results. *arXiv preprint arXiv:1412.1602*, 2014.

[Chuang and Yang, 2000] Wesley T Chuang and Jihoon Yang. Extracting sentence segments for text summarization: a machine learning approach. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 152–159. ACM, 2000.

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

[Clarke and Lapata, 2008] James Clarke and Mirella Lapata. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, 31:399–429, 2008.

[Conroy *et al.*, 2004] John M Conroy, Judith D Schlesinger, Jade Goldstein, and Dianne P O'leary. Left-brain/right-brain multi-document summarization. In *Proceedings of DUC*, 2004.

[Dasgupta *et al.*, 2013] Anirban Dasgupta, Ravi Kumar, and Sujith Ravi. Summarization through submodularity and dispersion. In *Proceedings of ACL*, pages 1014–1022, 2013.

[Daume, 2006] Harold Charles Daume. *Practical structured learning techniques for natural language processing*. ProQuest, 2006.

[Díaz and Gervás, 2007] Alberto Díaz and Pablo Gervás. User-model based personalized summarization. *Information Processing & Management*, 43(6):1715–1734, 2007.

[Díaz *et al.*, 2005] Alberto Díaz, Pablo Gervás, and Antonio García. Evaluation of a system for personalized summarization of web contents. In *User Modeling 2005*, pages 453–462. Springer, 2005.

[Dyer *et al.*, 2013] Chris Dyer, Victor Chahuneau, and Noah A Smith. A simple, fast, and effective reparameterization of ibm model 2. Association for Computational Linguistics, 2013.

[Edmundson, 1969] Harold P Edmundson. New methods in automatic extracting. *Journal of the ACM (JACM)*, 16(2):264–285, 1969.

[Elkahky *et al.*, 2015] Ali Mamdouh Elkahky, Yang Song, and Xiaodong He. A multi-view deep learning approach for cross domain user modeling in recommendation systems. In *Proceedings of the 24th International Conference on World Wide Web*, pages 278–288. International World Wide Web Conferences Steering Committee, 2015.

[Erkan and Radev, 2004] Günes Erkan and Dragomir R Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *JAIR*, 22(1):457–479, 2004.

[Evans *et al.*, 2004] David Kirk Evans, Judith L Klavans, and Kathleen R McKeown. Columbia newsblaster: Multilingual news summarization on the web. In *Demonstration Papers at HLT-NAACL 2004*, pages 1–4. Association for Computational Linguistics, 2004.

[Fattah and Ren, 2009] Mohamed Abdel Fattah and Fuji Ren. Ga, mr, ffnn, pnn and gmm based models for automatic text summarization. *Computer Speech & Language*, 23(1):126–144, 2009.

[Filatova and Hatzivassiloglou, 2004] Elena Filatova and Vasileios Hatzivassiloglou. Event-based extractive summarization. *Text Summarization Branches Out*, 2004.

[Filippova *et al.*, 2015] Katja Filippova, Enrique Alfonseca, Carlos A. Colmenares, Lukasz Kaiser, and Oriol Vinyals. Sentence compression by deletion with lstms. In *Proceedings of EMNLP*, pages 360–368, 2015.

[Filippova, 2010] Katja Filippova. Multi-sentence compression: Finding shortest paths in word graphs. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 322–330. Association for Computational Linguistics, 2010.

[Fisher and Roark, 2006] Seeger Fisher and Brian Roark. Query-focused summarization by supervised sentence ranking and skewed word distributions. In *Proceedings of the Document Understanding Conference, DUC-2006, New York, USA*. Citeseer, 2006.

[Foote, 1999] Jonathan Foote. An overview of audio information retrieval. *Multimedia systems*, 7(1):2–10, 1999.

[Galley, 2006] Michel Galley. A skip-chain conditional random field for ranking meeting utterances by importance. In *Proceedings of EMNLP*, pages 364–372, 2006.

[Gambhir and Gupta, 2017] Mahak Gambhir and Vishal Gupta. Recent automatic text summarization techniques: a survey. *Artificial Intelligence Review*, 47(1):1–66, 2017.

[Gehring *et al.*, 2017] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122*, 2017.

[Genest *et al.*, 2011] Pierre-Etienne Genest, Fabrizio Gotti, and Yoshua Bengio. Deep learning for automatic summary scoring. In *Proceedings of the Workshop on Automatic Text Summarization*, pages 17–28, 2011.

[Gers *et al.*, 1999] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. 1999.

[Gillick and Favre, 2009] Dan Gillick and Benoit Favre. A scalable global model for summarization. In *Proceedings of the Workshop on ILP for NLP*, pages 10–18, 2009.

[Goodrum, 2000] Abby A Goodrum. Image information retrieval: An overview of current research. *Informing Science*, 3(2):63–66, 2000.

[Gu *et al.*, 2016] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393*, 2016.

[Guu *et al.*, 2017] Kelvin Guu, Tatsunori B Hashimoto, Yonatan Oren, and Percy Liang. Generating sentences by editing prototypes. *arXiv preprint arXiv:1709.08878*, 2017.

[Han *et al.*, 2017] Mengqiao Han, Ou Wu, and Zhendong Niu. Unsupervised automatic text style transfer using lstm. In *National CCF Conference on Natural Language Processing and Chinese Computing*, pages 281–292. Springer, 2017.

[Harabagiu and Lacatusu, 2002] Sanda M Harabagiu and Finley Lacatusu. Generating single and multi-document summaries with gistexter. In *Document Understanding Conferences*, pages 11–12, 2002.

[Hashimoto and Tsuruoka, 2017] Kazuma Hashimoto and Yoshimasa Tsuruoka. Neural machine translation with source-side latent graph parsing. *arXiv preprint arXiv:1702.02265*, 2017.

[He *et al.*, 2012] Zhanying He, Chun Chen, Jiajun Bu, Can Wang, Lijun Zhang, Deng Cai, and Xiaofei He. Document summarization based on data reconstruction. In *AAAI*, 2012.

[He *et al.*, 2015] Liu He, Yu Hongliang, and Deng Zhi-Hong. Multi-document summarization based on two-level sparse representation model. In *Proceedings of AAAI*, 2015.

[He *et al.*, 2016] Wei He, Zhongjun He, Hua Wu, and Haifeng Wang. Improved neural machine translation with smt features. In *AAAI*, pages 151–157, 2016.

[He *et al.*, 2017] Wei He, Kai Liu, Yajuan Lyu, Shiqi Zhao, Xinyan Xiao, Yuan Liu, Yizhong Wang, Hua Wu, Qiaoqiao She, Xuan Liu, et al. Dureader: a chinese machine reading comprehension dataset from real-world applications. *arXiv preprint arXiv:1711.05073*, 2017.

[Hermann *et al.*, 2015] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701, 2015.

[Hirao *et al.*, 2002] Tsutomu Hirao, Hideki Isozaki, Eisaku Maeda, and Yuji Matsumoto. Extracting important sentences with support vector machines. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics, 2002.

[Hong and Nenkova, 2014] Kai Hong and Ani Nenkova. Improving the estimation of word importance for news multi-document summarization. In *Proceedings of EACL*, 2014.

[Hu *et al.*, 2014] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. Convolutional neural network architectures for matching natural language sentences. In *Advances in Neural Information Processing Systems*, pages 2042–2050, 2014.

[Hu *et al.*, 2015a] Baotian Hu, Qingcai Chen, and Fangze Zhu. Lcsts: A large scale chinese short text summarization dataset. In *Proceedings of EMNLP*, pages 1967–1972, 2015.

[Hu *et al.*, 2015b] Baotian Hu, Qingcai Chen, and Fangze Zhu. Lcsts: A large scale chinese short text summarization dataset. *arXiv preprint arXiv:1506.05865*, 2015.

[Isonuma *et al.*, 2017] Masaru Isonuma, Toru Fujino, Junichiro Mori, Yutaka Matsuo, and Ichiro Sakata. Extractive summarization using multi-task learning with document classification. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2101–2110, 2017.

[Ji *et al.*, 2014] Zongcheng Ji, Zhengdong Lu, and Hang Li. An information retrieval approach to short text conversation. *arXiv preprint arXiv:1408.6988*, 2014.

[Kalchbrenner and Blunsom, 2013] Nal Kalchbrenner and Phil Blunsom. Recurrent continuous translation models. In *EMNLP*, volume 3, page 413, 2013.

[Kauchak, 2013] David Kauchak. Improving text simplification language modeling using unsimplified text data. In *ACL (1)*, pages 1537–1546, 2013.

[Kikuchi *et al.*, 2016] Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. Controlling output length in neural encoder-decoders. *arXiv preprint arXiv:1609.09552*, 2016.

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

[Kingma and Welling, 2013] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[Kiyono *et al.*, 2017] Shun Kiyono, Sho Takase, Jun Suzuki, Naoaki Okazaki, Kentaro Inui, and Masaaki Nagata. Source-side prediction for neural headline generation. *arXiv preprint arXiv:1712.08302*, 2017.

[Kleinberg, 1999] Jon M Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.

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

[Knight and Marcu, 2002] Kevin Knight and Daniel Marcu. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139(1):91–107, 2002.

[Kobayashi *et al.*, 2015] Hayato Kobayashi, Masaki Noguchi, and Taichi Yatsuka. Summarization based on embedding distributions. In *Proceedings of EMNLP*, pages 1984–1989, 2015.

[Koehn and Knowles, 2017] Philipp Koehn and Rebecca Knowles. Six challenges for neural machine translation. *arXiv preprint arXiv:1706.03872*, 2017.

[Koehn *et al.*, 2007] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics, 2007.

[Kumar *et al.*, 2008] Chandan Kumar, Prasad Pingali, and Vasudeva Varma. Generating personalized summaries using publicly available web documents. In *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology-Volume 03*, pages 103–106. IEEE Computer Society, 2008.

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

[Lafferty *et al.*, 2001] John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

[Larsen, 1999] Bjornar Larsen. A trainable summarizer with knowledge acquired from robust nlp techniques. *Advances in automatic text summarization*, 71, 1999.

[Larson, 2010] Ray R Larson. Introduction to information retrieval. *Journal of the American Society for Information Science and Technology*, 61(4):852–853, 2010.

[Lei and Zhang, 2017] Tao Lei and Yu Zhang. Training rnns as fast as cnns. *arXiv preprint arXiv:1709.02755*, 2017.

[Lew *et al.*, 2006] Michael S Lew, Nicu Sebe, Chabane Djeraba, and Ramesh Jain. Content-based multimedia information retrieval: State of the art and challenges. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 2(1):1–19, 2006.

[Li *et al.*, 2013a] Chen Li, Fei Liu, Fuliang Weng, and Yang Liu. Document summarization via guided sentence compression. In *EMNLP*, pages 490–500, 2013.

[Li *et al.*, 2013b] Chen Li, Xian Qian, and Yang Liu. Using supervised bigram-based ilp for extractive summarization. In *Proceedings of ACL*, pages 1004–1013, 2013.

[Li *et al.*, 2017a] Junhui Li, Deyi Xiong, Zhaopeng Tu, Muhua Zhu, Min Zhang, and Guodong Zhou. Modeling source syntax for neural machine translation. *arXiv preprint arXiv:1705.01020*, 2017.

[Li *et al.*, 2017b] Piji Li, Zihao Wang, Wai Lam, Zhaochun Ren, and Lidong Bing. Salience estimation via variational auto-encoders for multi-document summarization. In *AAAI 2017: The Thirty-First AAAI Conference on Artificial Intelligence*. AAAI, 2017.

[Li *et al.*, 2018a] Juncen Li, Robin Jia, He He, and Percy Liang. Delete, retrieve, generate: A simple approach to sentiment and style transfer. *arXiv preprint arXiv:1804.06437*, 2018.

[Li *et al.*, 2018b] Shuai Li, Wanqing Li, Chris Cook, Ce Zhu, and Yanbo Gao. Independently recurrent neural network (indrnn): Building a longer and deeper rnn. *arXiv preprint arXiv:1803.04831*, 2018.

[Lin and Hovy, 1997] Chin-Yew Lin and Eduard Hovy. Identifying topics by position. In *Proceedings of the fifth conference on Applied natural language processing*, pages 283–290. Association for Computational Linguistics, 1997.

[Lin *et al.*, 2012] Ziheng Lin, Chang Liu, Hwee Tou Ng, and Min-Yen Kan. Combining coherence models and machine translation evaluation metrics for summarization evaluation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 1006–1014. Association for Computational Linguistics, 2012.

[Lin, 2003] Chin-Yew Lin. Improving summarization performance by sentence compression: a pilot study. In *Proceedings of the sixth international workshop on Information retrieval with Asian languages-Volume 11*, pages 1–8. Association for Computational Linguistics, 2003.

[Lin, 2004] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Proceedings of the ACL Workshop*, pages 74–81, 2004.

[Liu *et al.*, 2007] Maofu Liu, Wenjie Li, Mingli Wu, and Qin Lu. Extractive summarization based on event term clustering. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 185–188. Association for Computational Linguistics, 2007.

[Luhn, 1958] Hans Peter Luhn. The automatic creation of literature abstracts. *IBM Journal of research and development*, 2(2):159–165, 1958.

[Luong *et al.*, 2015] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[Lv and Zhai, 2011] Yuanhua Lv and ChengXiang Zhai. Lower-bounding term frequency normalization. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 7–16. ACM, 2011.

[Mani *et al.*, 1999] Inderjeet Mani, David House, Gary Klein, Lynette Hirschman, Therese Firmin, and Beth Sundheim. The tipster summac text summarization evaluation. In *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*, pages 77–85. Association for Computational Linguistics, 1999.

[Manning *et al.*, 2014] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of ACL: System Demonstrations*, pages 55–60, 2014.

[McDonald, 2007] Ryan McDonald. *A study of global inference algorithms in multi-document summarization*. Springer, 2007.

[McKeown *et al.*, 2001] Kathleen R McKeown, Vasileios Hatzivassiloglou, Regina Barzilay, Barry Schiffman, David Evans, and Simone Teufel. Columbia multi document summarization: Approach and evaluation. Technical report, Columbia University New York United States, 2001.

[Mei *et al.*, 2010] Qiaozhu Mei, Jian Guo, and Dragomir Radev. Divrank: the interplay of prestige and diversity in information networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1009–1018. Acm, 2010.

[Miao and Blunsom, 2016] Yishu Miao and Phil Blunsom. Language as a latent variable: Discrete generative models for sentence compression. *arXiv preprint arXiv:1609.07317*, 2016.

[Mihalcea and Radev, 2011] Rada Mihalcea and Dragomir Radev. *Graph-based natural language processing and information retrieval*. Cambridge university press, 2011.

[Mihalcea and Tarau, 2004] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, 2004.

[Mikolov *et al.*, 2010] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Interspeech*, volume 2, page 3, 2010.

[Mnih *et al.*, 2014] Volodymyr Mnih, Nicolas Heess, Alex Graves, and koray kavukcuoglu. Recurrent models of visual attention. In Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2204–2212. Curran Associates, Inc., 2014.

[Moratanch and Chitrakala, 2016] N Moratanch and S Chitrakala. A survey on abstractive text summarization. In *Circuit, Power and Computing Technologies (ICCPCT), 2016 International Conference on*, pages 1–7. IEEE, 2016.

[Morris *et al.*, 1992] Andrew H Morris, George M Kasper, and Dennis A Adams. The effects and limitations of automated text condensing on reading comprehension performance. *Information Systems Research*, 3(1):17–35, 1992.

[Murray *et al.*, 2010] Gabriel Murray, Giuseppe Carenini, and Raymond Ng. Generating and validating abstracts of meeting conversations: a user study. In *Proceedings of the 6th International Natural Language Generation Conference*, pages 105–113. Association for Computational Linguistics, 2010.

[Nallapati *et al.*, 2016a] Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*, 2016.

[Nallapati *et al.*, 2016b] Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*, 2016.

[Nenkova and Vanderwende, 2005] Ani Nenkova and Lucy Vanderwende. The impact of frequency on summarization. *Microsoft Research, Redmond, Washington, Tech. Rep. MSR-TR-2005*, 101, 2005.

[Nguyen *et al.*, 2016] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*, 2016.

[Osborne, 2002] Miles Osborne. Using maximum entropy for sentence extraction. In *Proceedings of ACL Workshop on Automatic Summarization*, pages 1–8, 2002.

[Otterbacher *et al.*, 2005] Jahna Otterbacher, Güneş Erkan, and Dragomir R Radev. Using random walks for question-focused sentence retrieval. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 915–922. Association for Computational Linguistics, 2005.

[Ouyang *et al.*, 2007] You Ouyang, Sujian Li, and Wenjie Li. Developing learning strategies for topic-based summarization. In *Proceedings of CIKM*, pages 79–86, 2007.

[Ouyang *et al.*, 2011] You Ouyang, Wenjie Li, Sujian Li, and Qin Lu. Applying regression models to query-focused multi-document summarization. *Information Processing & Management*, 47(2):227–237, 2011.

[Over and Yen, 2004] Paul Over and James Yen. Introduction to duc-2001: an intrinsic evaluation of generic news text summarization systems. In *Proceedings of DUC*, 2004.

[Oya *et al.*, 2014] Tatsuro Oya, Yashar Mehdad, Giuseppe Carenini, and Raymond Ng. A template-based abstractive meeting summarization: Leveraging summary and source text relationships. In *Proceedings of the 8th International Natural Language Generation Conference (INLG)*, pages 45–53, 2014.

[Page *et al.*, 1999] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[Paice and Jones, 1993] Chris D Paice and Paul A Jones. The identification of important concepts in highly structured technical papers. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 69–78. ACM, 1993.

[Papineni *et al.*, 2002] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 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, 2002.

[Pascanu *et al.*, 2013] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318, 2013.

[Passonneau *et al.*, 2005] Rebecca J Passonneau, Ani Nenkova, Kathleen McKeown, and Sergey Sigelman. Applying the pyramid method in duc 2005. In *Proceedings of DUC*, 2005.

[Paulus *et al.*, 2017] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*, 2017.

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[Pighin *et al.*, 2014] Daniele Pighin, Marco Cornolti, Enrique Alfonseca, and Katja Filippova. Modelling events through memory-based, open-ie patterns for abstractive summarization. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 892–901, 2014.

[Pitler *et al.*, 2010] Emily Pitler, Annie Louis, and Ani Nenkova. Automatic evaluation of linguistic quality in multi-document summarization. In *Proceedings of the 48th annual meeting of the Association for Computational Linguistics*, pages 544–554. Association for Computational Linguistics, 2010.

[Radev *et al.*, 2004a] Dragomir R Radev, Timothy Allison, Sasha Blair-Goldensohn, John Blitzer, Arda Celebi, Stanko Dimitrov, Elliott Drabek, Ali Hakim, Wai Lam, Danyu Liu, et al. Mead-a platform for multidocument multilingual text summarization. In *LREC*, 2004.

[Radev *et al.*, 2004b] Dragomir R Radev, Hongyan Jing, Małgorzata Styś, and Daniel Tam. Centroid-based summarization of multiple documents. *Information Processing & Management*, 40(6):919–938, 2004.

[Ranzato *et al.*, 2015] Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*, 2015.

[Rath *et al.*, 1961] GJ Rath, A Resnick, and TR Savage. The formation of abstracts by the selection of sentences. part i. sentence selection by men and machines. *Journal of the Association for Information Science and Technology*, 12(2):139–141, 1961.

[Reiter and Dale, 1997] Ehud Reiter and Robert Dale. Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87, 1997.

[Rezende *et al.*, 2014] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

[Robertson *et al.*, 1995] Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. Okapi at trec-3. *Nist Special Publication Sp*, 109:109, 1995.

[Robertson *et al.*, 2009] Stephen Robertson, Hugo Zaragoza, et al. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389, 2009.

[Rush *et al.*, 2015a] Alexander M Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*, 2015.

[Rush *et al.*, 2015b] Alexander M. Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. In *Proceedings of EMNLP*, pages 379–389, 2015.

[Sandhaus, 2008] Evan Sandhaus. The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia*, 6(12):e26752, 2008.

[Schluter and Søgaard, 2015] Natalie Schluter and Anders Søgaard. Unsupervised extractive summarization via coverage maximization with syntactic and semantic concepts. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pages 840–844, 2015.

[See *et al.*, 2017] Abigail See, Peter J Liu, and Christopher D Manning. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*, 2017.

[Shang *et al.*, 2015] Lifeng Shang, Zhengdong Lu, and Hang Li. Neural responding machine for short-text conversation. *arXiv preprint arXiv:1503.02364*, 2015.

[Sipos *et al.*, 2012] Ruben Sipos, Pannaga Shivaswamy, and Thorsten Joachims. Large-margin learning of submodular summarization models. In *Proceedings of EACL*, pages 224–233, 2012.

[Song *et al.*, 2018] Kaiqiang Song, Lin Zhao, and Fei Liu. Structure-infused copy mechanisms for abstractive summarization. *arXiv preprint arXiv:1806.05658*, 2018.

[Srivastava *et al.*, 2014] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[Stolcke *et al.*, 2011] Andreas Stolcke, Jing Zheng, Wen Wang, and Victor Abrash. Srilm at sixteen: Update and outlook. In *Proceedings of IEEE Automatic Speech Recognition and Understanding Workshop*, volume 5, 2011.

[Sundermeyer *et al.*, 2014] Martin Sundermeyer, Tamer Alkhouli, Joern Wuebker, and Hermann Ney. Translation modeling with bidirectional recurrent neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 14–25, 2014.

[Surdeanu and Harabagiu, 2002] Mihai Surdeanu and Sanda M Harabagiu. Infrastructure for open-domain information extraction. In *Proceedings of the second international conference on Human Language Technology Research*, pages 325–330. Morgan Kaufmann Publishers Inc., 2002.

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

[Tan *et al.*, 2017a] Chuanqi Tan, Furu Wei, Nan Yang, Bowen Du, Weifeng Lv, and Ming Zhou. S-net: From answer extraction to answer generation for machine reading comprehension. *arXiv preprint arXiv:1706.04815*, 2017.

[Tan *et al.*, 2017b] Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 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)*, volume 1, pages 1171–1181, 2017.

[Tieleman and Hinton, 2012] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 4(2), 2012.

[Tombros and Sanderson, 1998] Anastasios Tombros and Mark Sanderson. Advantages of query biased summaries in information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 2–10. ACM, 1998.

[Toutanova *et al.*, 2007] Kristina Toutanova, Chris Brockett, Michael Gamon, Jagadeesh Jagarlamudi, Hisami Suzuki, and Lucy Vanderwende. The pythy summarization system: Microsoft research at duc 2007. In *Proc. of DUC*, volume 2007, 2007.

[Tu *et al.*, 2016a] Zhaopeng Tu, Yang Liu, Lifeng Shang, Xiaohua Liu, and Hang Li. Neural machine translation with reconstruction. *arXiv preprint arXiv:1611.01874*, 2016.

[Tu *et al.*, 2016b] Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. Modeling coverage for neural machine translation. *arXiv preprint arXiv:1601.04811*, 2016.

[Vanderwende *et al.*, 2007] Lucy Vanderwende, Hisami Suzuki, Chris Brockett, and Ani Nenkova. Beyond sumbasic: Task-focused summarization with sentence simplification and lexical expansion. *Information Processing & Management*, 43(6):1606–1618, 2007.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010, 2017.

[Vinyals *et al.*, 2015] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700, 2015.

[Wan and Yang, 2008] Xiaojun Wan and Jianwu Yang. Multi-document summarization using cluster-based link analysis. In *Proceedings of SIGIR*, pages 299–306, 2008.

[Wan *et al.*, 2006] Xiaojun Wan, Jianwu Yang, and Jianguo Xiao. Using cross-document random walks for topic-focused multi-document. In *Web Intelligence, 2006. WI 2006. IEEE/WIC/ACM International Conference on*, pages 1012–1018. IEEE, 2006.

[Wang *et al.*, 2017] Xing Wang, Zhengdong Lu, Zhaopeng Tu, Hang Li, Deyi Xiong, and Min Zhang. Neural machine translation advised by statistical machine translation. In *AAAI*, pages 3330–3336, 2017.

[Wei *et al.*, 2010] Furu Wei, Wenjie Li, Qin Lu, and Yanxiang He. A document-sensitive graph model for multi-document summarization. *Knowledge and information systems*, 22(2):245–259, 2010.

[Wiseman and Rush, 2016] Sam Wiseman and Alexander M Rush. Sequence-to-sequence learning as beam-search optimization. *arXiv preprint arXiv:1606.02960*, 2016.

[Wu *et al.*, 2016] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[Wu *et al.*, 2017] Yu Wu, Wei Wu, Dejian Yang, Can Xu, Zhoujun Li, and Ming Zhou. Neural response generation with dynamic vocabularies. *arXiv preprint arXiv:1711.11191*, 2017.

[Xu *et al.*, 2015] Kelvin Xu, Jimmy Ba, Ryan Kiros, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044*, 2015.

[Xu *et al.*, 2018] Jingjing Xu, Xu Sun, Qi Zeng, Xuancheng Ren, Xiaodong Zhang, Houfeng Wang, and Wenjie Li. Unpaired sentiment-to-sentiment translation: A cycled reinforcement learning approach. *arXiv preprint arXiv:1805.05181*, 2018.

[Yao *et al.*, 2015] Jin-ge Yao, Xiaojun Wan, and Jianguo Xiao. Compressive document summarization via sparse optimization. In *Proceedings of IJCAI*, pages 1376–1382, 2015.

[Yin and Pei, 2015] Wenpeng Yin and Yulong Pei. Optimizing sentence modeling and selection for document summarization. In *Proceedings of IJCAI*, pages 1383–1389, 2015.

[Yu *et al.*, 2017] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI*, pages 2852–2858, 2017.

[Zajic *et al.*, 2006] David M Zajic, Bonnie Dorr, Jimmy Lin, and Richard Schwartz. Sentence compression as a component of a multi-document summarization system. In *Proceedings of the 2006 Document Understanding Workshop, New York*, 2006.

[Zha, 2002] Hongyuan Zha. Generic summarization and keyphrase extraction using mutual reinforcement principle and sentence clustering. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 113–120. ACM, 2002.

[Zhang *et al.*, 2016] Biao Zhang, Deyi Xiong, Jinsong Su, Hong Duan, and Min Zhang. Variational neural machine translation. *arXiv preprint arXiv:1605.07869*, 2016.

[Zhang *et al.*, 2018] Xiangwen Zhang, Jinsong Su, Yue Qin, Yang Liu, Rongrong Ji, and Hongji Wang. Asynchronous bidirectional decoding for neural machine translation. *arXiv preprint arXiv:1801.05122*, 2018.

[Zhao *et al.*, 2012] Jichang Zhao, Li Dong, Junjie Wu, and Ke Xu. Moodlens: an emoticon-based sentiment analysis system for chinese tweets. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1528–1531. ACM, 2012.

[Zhou and Hovy, 2004] Liang Zhou and Eduard Hovy. Template-filtered headline summarization. *Text Summarization Branches Out*, 2004.

[Zhou *et al.*, 2017a] Hao Zhou, Zhaopeng Tu, Shujian Huang, Xiaohua Liu, Hang Li, and Jiajun Chen. Chunk-based bi-scale decoder for neural machine translation. *arXiv preprint arXiv:1705.01452*, 2017.

[Zhou *et al.*, 2017b] Qingyu Zhou, Nan Yang, Furu Wei, and Ming Zhou. Selective encoding for abstractive sentence summarization. *arXiv preprint arXiv:1704.07073*, 2017.

[Zhu *et al.*, 2007] Xiaojin Zhu, Andrew Goldberg, Jurgen Van Gael, and David Andrzejewski. Improving diversity in ranking using absorbing random walks. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 97–104, 2007.