EXPLORING TEXT SUMMARIZATION BEYOND NEWS ARTICLES

RUIFENG YUAN

PhD

The Hong Kong Polytechnic University

2024

The Hong Kong Polytechnic University
Department of Computing

Exploring Text Summarization Beyond News Articles

Ruifeng Yuan

A thesis submitted in partial fulfillment of the requirements for
the degree of Doctor of Philosophy
January 2024

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 acknowledgment has been made in the text.

Signature: _____

Name of Student: Ruifeng Yuan

Abstract

Text summarization has been an important task for natural language processing. It aims to compress the source document(s) into a more concise version that covers the most important information. In recent years, with the development of the neural-based language model, text summarization has made great progress. In this process, news summarization is undoubtedly the most important research topic in this field. On the one hand, news summaries have inherent application scenarios in real life. On the other hand, a set of large-scale news summarization datasets has been proposed to meet the data requirement of neural models. Therefore, for a considerable period, researching general summarization models on news data has become the mainstream paradigm in text summarization. With the continuous advancement of text summarization models and techniques, researchers are no longer confined to such a paradigm but are further exploring or rediscovering more diverse text summarization problems. These problems often have their unique characteristics, which means the general approaches cannot be blindly applied. Meanwhile, they still share similarities with it in many ways. In this thesis, we aim to investigate text summarization problems beyond the "news data + general model" mainstream paradigm. More specifically, we identify three research problems to be addressed: 1. How to use the natural features of news articles to improve the general summarization model on news summarization? 2. How to extend the current general summarization models to summarization tasks/domains where these models cannot be directly applied? 3. How to utilize the large-scale data in news summarization to assist summarization tasks/domains with

insufficient data?

To address the aforementioned problems, we aim to develop summarization approaches for specific domains or tasks. Based on the three proposed research questions, the thesis is naturally divided into three parts.

In the first part (work 1 and work 2), to enhance the news summarization with its unique characteristics, we propose to incorporate a typical kind of extra information into the summarization model, the event-level information. The research target here is to investigate what role event-level information plays in both extractive and abstractive news summarization, and how to make good use of them. In work 1, we propose to extract event-level semantic units for better extractive news summarization. We also introduce a hierarchical structure, which incorporates the multi-level of granularities of the textual information into the model. In work 2, we explore the effective sentence fusion approach that can fuse extracted salient information to abstractive summary sentences. We propose to build an event graph from the input sentences to effectively capture and organize related events in a structured way and use the constructed event graph to guide the summarization. In addition to making use of the attention over the content of sentences and graph nodes, we further develop a graph flow attention mechanism to control the fusion process via the graph structure for the faithfulness of the fused summaries. The experiments and further ablation studies on news datasets demonstrate the effectiveness of event-level information in news summarization.

The second part (work 3) aims to explore text summarization problems that can not directly apply general summarization models, where the most representative one is long-input summarization. As general summarization models struggle with long-length input because of their high memory cost, these models cannot directly apply to documents with thousands of tokens. The main challenge is how to effectively extend mature summarization techniques and efficiently handle the difficulties brought by the long input. In work 3, we present a context-aware extract-generate framework

(CAEG) for long-input text summarization. It focuses on preserving both local and global context information in an extract-generate framework with little cost. CAEG generates a set of context-related text spans called context prompts for each text snippet and uses them to transfer the context information from the extractor and generator. To find such context prompts, we propose to capture the context information based on the interpretation of the extractor, where the text spans having the highest contribution to the extraction decision is considered as containing the richest context information. The experiments show the effectiveness and efficiency of our model in capturing and preserving the context information in the long-input summarization.

The third part (work 4 and work 5) delves into problem 3 and investigates how to effectively transfer the knowledge of summarization learned from news data to tasks or domains with insufficient data. Work 4 explores this problem from the perspective of task knowledge transferring in the context of query-focused summarization. In this work, we investigate the idea of whether we can integrate and transfer the knowledge of news summarization and question answering to assist the few-shot learning in query-focused summarization. Here, we propose prefix-merging, a prefix-based pre-training strategy for few-shot learning in query-focused summarization. We integrate the task knowledge from text summarization and question answering into a properly designed prefix and apply the merged prefix to query-focused summarization. In addition to task knowledge transfer, we also investigate domain transfer of extractive summarization in work 5. In text summarization, context information is considered as a key factor. Meanwhile, there also exist other pattern factors that can identify sentence importance, such as sentence position or certain n-gram tokens. In this work, we attempt to apply disentangled representation learning on extractive summarization, and separate the two key factors for the task, context and pattern, for a better generalization ability in the low-resource setting. The experiments suggest the great potential of the knowledge contained in the large-scale news summarization data in

improving the summarization system in other tasks or domains.

As a conclusion, we study our proposed research problems of text summarization in a systematic way. We illustrate the importance of these problems and demonstrate the effectiveness of our approaches on various datasets. This shows the potential of our works to benefit real-world applications, such as news summarization, academic research and medical records collection.

Publications Arising from the Thesis

1. Ruifeng Yuan, Zili Wang and Wenjie Li, "Fact-level Extractive Summarization with Hierarchical Graph Mask on BERT", accepted in *The 28th International Conference on Computational Linguistics (COLING 2020).*

2. Ruifeng Yuan, Zili Wang and Wenjie Li, "Event Graph based Sentence Fusion", accepted in *The 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP 2021).*

3. Ruifeng Yuan, Zili Wang, Ziqiang Cao and Wenjie Li, "Few-shot Query-oriented Summarization with Prefix-merging", accepted in *The 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP 2022).*

4. Ruifeng Yuan, Zili Wang, Ziqiang Cao and Wenjie Li, "Preserve Context Information for Extract-Generate Long-Input Summarization Framework", accepted in *The Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI 2023).*

5. Ruifeng Yuan, Shichao Sun, Zili Wang, Ziqiang Cao and Wenjie Li, "Separating Context and Pattern: Learning Disentangled Sentence Representations for Low-Resource Extractive Summarization", accepted as findings in *The 61st Annual Meeting of the Association for Computational Linguistics (ACL 2023).*

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my supervisor, Prof. Li Wenjie, Maggie, for her invaluable guidance, patience, and support throughout this research journey. Her insightful critiques and expertise have been crucial in shaping this work. Her precious advice and assistance on life issues also help me a lot. It is an honor to be her student.

I would also like to thank all our lab members and my collaborators, Wang Zili, Li Yongqi, Sun Shichao, Mu Feiteng, Wang Jiashuo, Wang Jian, Xu Kaishuai, Hou Wenjun, Lin Peiqing, Cheng Yi, Cao Ziqiang and etc, for their valuable feedback and suggestions on my research. I also want to express my appreciation to my office mates and roommate, who provided a friendly and fun environment in which to learn and grow. I feel lucky to meet so many interesting and friendly people.

Last but not the least, I would like to acknowledge the support and encouragement I received from my family. Their unwavering confidence in me and their love have been a constant source of strength throughout my Ph.D. journey.

Table of Contents

Abstract **i**

Publications Arising from the Thesis **v**

Acknowledgments **vi**

List of Figures **xiv**

List of Tables **xvii**

1 Introduction **1**

1.1 Background 1

1.2 Research Problems 3

1.3 Research Overview and Contributions 5

1.4 Structure of Thesis 10

2 Literature Review **12**

2.1 Text Summarization System Overview 12

2.1.1 Extractive Summarization 15

2.1.2 Abstractive Summarization 16
2.1.3 Datasets and Evaluation 17
2.2 Existing Approaches Based on Neural Model 20
2.2.1 Modification on Network Structure 21
2.2.2 Modification on Encoder 23
2.2.3 Modification on Extractor and Generator 27
2.2.4 Modification on Optimization 28
2.2.5 Hybird Model 30
2.3 Sub-Tasks for Text Summarization 31
2.3.1 Multi-document Summarization 31
2.3.2 Query-Focused Summarization 32
2.3.3 Dialogue Summarization 34
2.3.4 Long-Input Summarization 35

I News Summarization via Event Information 37

3 Event-Level Extractive Summarization with Hierarchical Graph Mask on BERT 38

3.1 Introduction 38
3.2 Related Work 41
3.3 Extraction of Events and Alignment with Gold Summary 42
3.4 Summarization Model 45
3.4.1 Model Framework 45

3.4.2 BERT-Based Encoder 45
3.4.3 Hierarchical Graph Mask on BERT 46
3.4.4 Classifier 48
3.4.5 Learning 49
3.5 Experiments 49
3.5.1 Dataset and Evaluation Metric 49
3.5.2 Details 50
3.5.3 Automatic Evaluation 50
3.5.4 Human Evaluation 51
3.5.5 Model Analysis 53
3.5.6 Case Study 55
3.6 Conclusion 57

4 Event Graph based Sentence Fusion for Abstractive Summarization 58

4.1 Introduction 58
4.2 Related Work 61
4.3 Method 62
4.3.1 Event Graph Construction 62
4.3.2 Encoder 63
4.3.3 Decoder 64
4.3.4 Training 68
4.3.5 Faithful Beam Search 69

4.4 Experiments 70
4.4.1 Experimental Set-Up 70
4.4.2 Automatic Evaluation 71
4.4.3 Ablation Study 72
4.4.4 Human Evaluation 74
4.4.5 Application in Text Summarization 76
4.5 Conclusion 77

II Long-Input Summarization via Context-Aware Extract-Generate Framework 78

5 Preserve Context Information for Extract-Generate Long-Input Summarization Framework 79

5.1 Introduction 79
5.2 Related Work 82
5.3 Method 83
5.3.1 Extract-Generate Framework 84
5.3.2 Local Context Information 86
5.3.3 Global Context Information 89
5.3.4 Application 89
5.4 Experiment 90
5.4.1 Dataset 90
5.4.2 Baseline 91

5.4.3 Implementation Details 92
5.4.4 Experiment Results 93
5.4.5 Analysis and Discussion 94
5.5 Conclusions 99

III Low-Resource Summarization via Knowledge Transferring 100

6 Few-shot Query-Focused Summarization with Prefix-Merging 101

6.1 Introduction 101
6.2 Related Work 104
6.3 Method 105
6.3.1 Problem Statement 105
6.3.2 Prefix-tuning 106
6.3.3 Intuition for Prefix-merging 107
6.3.4 Prefix-merging 108
6.3.5 Self-adaptive Prefix-merging 109
6.3.6 Applying the Merged Prefix to the Target Task 109
6.4 Experiment 110
6.4.1 Datasets 110
6.4.2 Experiment Setting 111
6.4.3 Result 112
6.4.4 Ablation Study 118

6.4.5 Prefix Visualization 118

6.5 Discussion and Conclusion 120

7 Separating Context and Pattern: Learning Disentangled Sentence Representations for Low-Resource Extractive Summarization 122

7.1 Introduction 122

7.2 Related Work 125

7.3 Model 126

7.3.1 Problem Statement 128

7.3.2 Extractive Summarization Model 128

7.3.3 Learning Context Representation 129

7.3.4 Learning Pattern Representation 130

7.3.5 Learning Disentangled Representation 131

7.3.6 Training Strategy 133

7.3.7 Application in Low-Resource Setting 134

7.4 Experiment 134

7.4.1 Experiment Details 134

7.4.2 Comparison 135

7.4.3 Experiment Results 136

7.4.4 Visualization 140

7.5 Discussion and Conclusion 141

8 Conclusions and Future Work 143

8.1 Summary of Contributions 144

8.2 Future Work 145

References **148**

List of Figures

1.1 The research path of this thesis. 4

2.1 Classifications of the text summarization systems. 14

2.2 Framework for text summarization approaches based on neural model. 21

3.1 A dependency tree example. We extract the following two event descriptions: Ahmadinejad essentially called Yukiya Amano, the director general of the IAEA, a U.S. puppet ||| said the U.N.A has no jurisdiction in Iran and Irap 44

3.2 The framework of the summarization model 47

4.1 Examples of two types of sentence fusion in text summarization. . . . 59

4.2 The framework of our proposed sentence fusion model. The various colors in the left refer to nodes and corresponding event components, while dotted lines represent how information disseminates in the BERT attention layer. In the middle part, different gray scales stand for different levels of attention on the tokens or nodes. 63

4.3 Calculation process of graph flow attention. 66

5.1 The extract-generate framework for long-input summarization. The grey strips stands for text snippets such as sentences or utterances from the source document. This figure shows that in the extract generate summary framework, both local and global context information is lost. 80

5.2 The framework of our proposed CAEG framework. The extractor we used is RoBERTa-base, and the generator is BART-large. Here, we use the blocks with various colors to represent the different types of information in the model and adopt the dot lines to emphasize the information flow added for preserving the context information. To obtain a clear observation, we simplified the structure of the generator in the figure. 84

5.3 The generation of context prompts. The yellow squares are the [cls] tokens and the grey squares denote the text tokens. The darker red stands for a higher attention rollout score. 87

6.1 Focusing on the encoder layer of BART, the figure shows annotated examples and comparison between the prefix-merging (top, mid) on the two auxiliary tasks (summarization and QA) and applying the merged prefix on query-focused summarization with prefix-tuning (bottom). . 106

6.2 The attention score for query-focused summarization in both encoder and decoder of model "Unq(20)+Sha(10)". 119

7.1 Comparison of position distribution of oracle sentences in news summarization dataset CNN/DailyMail and science paper summarization dataset arXiv. The X-axis refers to 1 to 100 sentence position and the Y-axis represents its proportion. 123

7.2 The framework of our proposed model. Part (1) shows the model based on the adversarial objective, while part (2) displays the one based on the MI minimization objective. The blue blocks refer to different sentence representations, the green blocks stand for the model components and the yellow blocks represent the target features. The solid lines represent normal classification loss, and the dashed lines stand for the discriminator loss plus the adversary loss. 127

7.3 The predicted sentences position distribution on arXiv when using context/pattern representation. 139

7.4 Visualization of context and pattern representations. 141

List of Tables

1.1 Overview of research works in this thesis. 6

3.1 The average unit number and unit length in train set of CNN/DM dataset. 45

3.2 Results on oracle summary on CNN/DM test set. 50

3.3 Results on CNN/DM test set. BERTSUM+event refers to the BERTSUM baseline which simply changes the primary textual unit for extraction from sentences to events. The last three models represent three variants of our method. For example, d+f means that we use document-level information and event-level information for summarization 52

3.4 Human evaluation results on CNN/DM test set. 53

3.5 Proportion of extracted events according to their position in the original text. 54

3.6 Results for disentangling test on CNN/DM test set. 55

3.7 Example from the CNN/DailMail test dataset 56

4.1 Statistic of CNN/DaliyMail Fusion dataset and Multi-News Fusion dataset. Multi/Single indicates whether the source sentences are from multiple documents or a single document. 70

4.2 Automatic evaluation with Rouge, faithfulness(Fai), fusion rate(Fus), and generated sentence length (Len) on CNN/DaliyMail Fusion dataset and Multi-News Fusion Dataset. 73

4.3 The results of ablation study on Multi-News Fusion test set. 74

4.4 Examples from the Multi-News Fusion test dataset. The different colors represent equivalent content between the source and the generated sentence. 75

4.5 The results of the human evaluation on Multi-News Fusion test set. . 75

4.6 The result of sentence fusion application on CNN/DaliyMail test set. 76

5.1 The statistics and comparison of datasets in the experiment. The Source Len and Target Len stand for the token number of the source document and the summaries. 90

5.2 Results on QMSum. 92

5.3 Results on arXiv. 93

5.4 Analysis of the number of context prompts from K=1 to K=4 on QMSum Dataset. 95

5.5 Analysis of the number of context prompts from K=1 to K=4 on arXiv Dataset. 95

5.6 The comparison between different approaches in preserving local context information on QMSum dataset. 96

5.7 The case study of the context prompts on QMSum. For each extracted snippet, we display two generated context prompts. For a better observation, we use various colors to label the salient context information, and adopt the underline to emphasize the position of the generated context prompts in the source text. 97

5.8 The case study of the context prompts on arXiv. 98

6.1 Evaluation result for query-focused summarization on PubMedQA. We compare the result on three different training data size: 50, 150, 300. Here, we also provide result of BART-base on the full-size training for better comparison. 111

6.2 .Standard Deviation of the Results on PubMedQA 112

6.3 Evaluation result for query-focused summarization on DUC. Ext_Oracle refers to oracle extractive result taking the query-related sentences as input, which can be seen as the upper bound of this experiment. . . . 114

6.4 Evaluation result for query-focused summarization on Debatepedia. In the upper and middle part, we display the result of few-shot learning with 50 data samples on standard division of Debatepedia. In the lower part, we show the result of BART training with full-size data but with different data division. BART(full) represents the standard division and BART(full)_redivided refers to a new division that do not have the data leakage problem (we achieve this by redivide all data samples by an alphabetical sort, where similar data samples tend to gather together rather than scatter in both training and testing set). . 115

6.5 The comparison between prefix-merging and fine-tuning with a training data size of 50. 116

6.6 The comparison between using different auxiliary tasks with a training data size of 50. 116

6.7 The experiment result for ablation study with a training data size of 50.118

7.1 Examples about the high-frequency n-grams in oracle sentences from CNN/DailyMail and arXiv. 124

7.2 The statistics and comparison of the datasets. The Source Len and Target Len stand for the token number of the source document and the summaries. 134

7.3 The results of models trained on CNN/DM in zero-shot setting. . . . 137

7.4 The results of models trained on arXiv in zero-shot setting. 138

7.5 The results on CNN/DM when using context/pattern representation. 139

7.6 The results on arXiv and CNN/DM in few-shot setting. 140

7.7 The ablation study in the zero-shot setting. 140

Chapter 1

Introduction

1.1 Background

In today's digital age, the volume of online news articles, blogs, research papers, and various other types of documents is growing at an unprecedented rate. Every day, countless new pieces of information are added to the vast digital repository, the internet. This explosion of information has led to a situation where people are overwhelmed with more data than they can possibly consume or even comprehend, making it difficult for individuals to sift through and identify the information that is most relevant or important to them. This presents a significant challenge: how can people efficiently organize, categorize, and summarize this vast amount of information or how can people distill the key points from a lengthy document into a concise summary that is easy to understand and digest?

To address this challenge, researchers have proposed automatic summarization models. These models leverage natural language processing techniques and advanced machine learning models to automatically generate summaries for documents. This allows people to alleviate the burden of manual summarization, freeing up human labor for more complex tasks that require human judgment and creativity. The pri-

mary goal of the automatic summarization models is to generate a condensed version or highlight of a given document. This involves extracting the most salient and relevant information from the source text and presenting it in a concise and coherent manner. The challenge here is to capture all the key points while ensuring that the summary faithfully represents the original content. It is a complex task that requires sophisticated algorithms and models, but the potential benefits of saving time and effort are enormous.

Text summarization has been a classic Natural Language Process (NLP) task for many years, and it is still in development. There are two types of widely investigated approaches for text summarization: extractive summarization and abstractive summarization. Extractive models aim to directly select text spans or sentences from source documents and aggregate them as a summary. Researchers usually consider it as a ranking problem, where each extracted text span is given a score that represents its importance in the input document(s). Abstractive models attempt to understand the input document and then reorganize the important information into new summary sentences, which is often regarded as a natural language generation task.

Throughout decades of research, numerous summarization systems have been developed. In recent years, neural model based text summarization approaches have become increasingly popular. On the one hand, the neural language models such as LSTM [138] or Transformer [145] have great potential in text understanding and generation. On the other hand, a set of large-scale summarization datasets [105, 47, 106, 34] have been proposed to meet the data requirement of these models. As news summarization has always been an important sub-topic and the related data is easily accessible, the majority of these datasets are from the news domain. Therefore, for a considerable period, researching general summarization models on news data has become the mainstream paradigm in the field.

With the improvement of text summarization systems, researchers are not satisfied and have proposed higher and more refined expectations for the task. Meanwhile, text

summarization is no longer limited itself to news articles. From different summarization domains to various summarization sub-tasks, the diverse application scenarios bring further challenges to the development summarization system. In this case, it is easy for mainstream research to overlook these increasingly new demands in the field of text summarization. This thesis aims to explore text summarization beyond the "news data + general model" research paradigm and fill the gaps left by the mainstream research with further investigation into specific summarization tasks or domains.

1.2 Research Problems

The major goal of the thesis is to explore or rediscover the text summarization problems beyond the traditional mainstream research paradigm, and develop summarization models for specific domains or tasks. The investigation is expected to continue exploring the gaps in previous mainstream research, addressing the increasingly diverse summarization needs in real-world applications. Three main research problems in the thesis are shown as follows:

- Problem 1: How to use the natural features of news articles to improve the general summarization model on news summarization?
- Problem 2: How to extend the current general summarization models to summarization tasks/domains where these models cannot be directly applied?
- Problem 3: How to utilize the large-scale data in news summarization to assist summarization tasks/domains with insufficient data?

The motivation and relationships of the three research problems are shown in Figure 1.1.

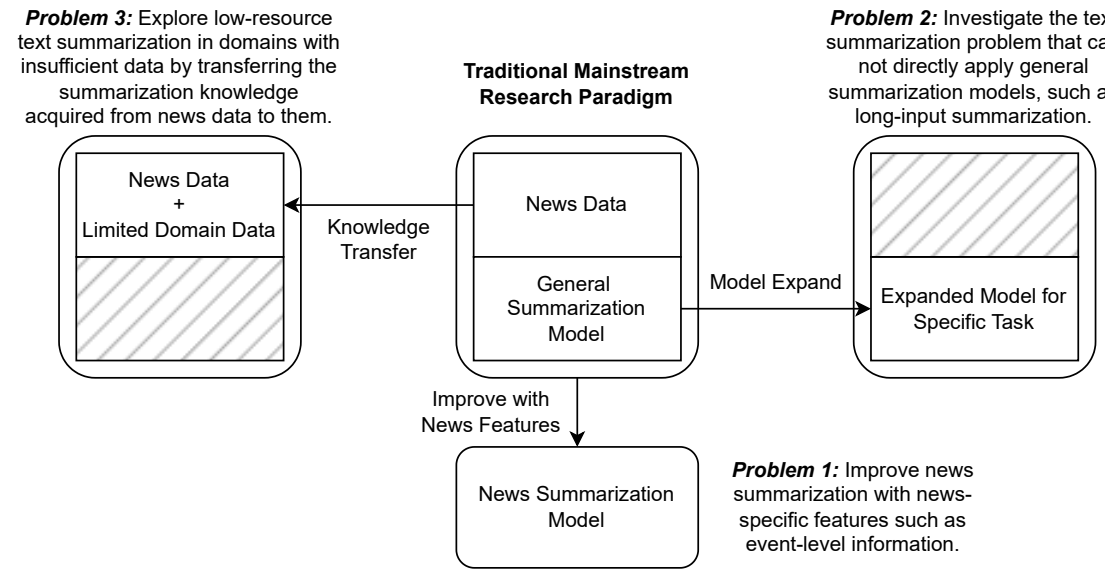

Figure 1.1: The research path of this thesis.

- The first problem pays attention to improving news summarization with news-specific features. Although general summarization models can accomplish this task, models focused on news summaries can be improved by capturing the unique characteristics of news articles, which could potentially enhance the quality of the generated news summaries. The main challenge of this problem is what are the unique characteristics of new summarization and how to effectively combine them with the summarization model. Considering a major characteristic of news articles is that they are often composed of multiple events, we propose to enhance news summarization using event-level information on this problem.

- The second problem focuses on investigating text summarization problems that can not directly apply general summarization models, where the most representative one is long-input summarization. Due to the exponential increase in memory consumption caused by excessively long inputs, current state-of-the-art

transformer-based summarization models cannot directly apply to documents with thousands of tokens. How to extend the existing general models into a framework that can be used for long-input summarization becomes the key challenge. This framework is expected to effectively integrate mature summarization techniques and efficiently handle the difficulties brought by the long input.

- The last problem emphasizes on exploring text summarization in tasks/domains with insufficient data by transferring the summarization knowledge acquired from news data to them. Not all summarization tasks or domains, like news summarization, have a large amount of high-quality annotated data to train a neural-based model. Considering there exists common knowledge in all types of summarization, a feasible approach is to use the knowledge from news summarization to assist other summarization tasks/domains. The major challenge here is how to extract and represent the summarization knowledge for transferring and how to apply it to the target tasks/domains.

1.3 Research Overview and Contributions

In this thesis, we target at exploring the gaps in previous mainstream research and develop summarization approaches for specific domains or tasks. Based on the three research problems, the studies in the thesis are divided into three parts. The first part concentrates on enhancing news summarization itself with event-level information, and the second part attempts to develop a summarization framework for long-input summarization, while the third part focuses on transferring summarization knowledge from news summarization to other summarization tasks/domains. In the Table 1.1, we display the overview of these works. As for enhancing news summarization, we develop two event-aware summarization models (work 1 and work 2), one extractive summarization model and one abstractive summarization model, as the response for

Research Work	**Method Category**	**Research Problem**	**Publication Venue**
Work 1: Event-Level Extractive Summarization with Hierarchical Graph Mask on BERT	Extractive	Problem 1	COLING
Work 2: Event Graph based Sentence Fusion for Abstractive Summarization	Abstractive	Problem 1	EMNLP
Work 3: Preserve Context Information for Extract-Generate Long-Input Summarization Framework	Abstractive	Problem 2	AAAI
Work 4: Few-shot Query-Focused Summarization with Prefix-Merging	Abstractive	Problem 3	EMNLP
Work 5: Separating Context and Pattern: Learning Disentangled Sentence Representations for Low-Resource Extractive Summarization	Extractive	Problem 3	ACL

Table 1.1: Overview of research works in this thesis.

research problem 1. As for long-input summarization, we establish a context-aware extract-generate framework (work 3) to address the research problem 2. As for knowledge transferring of text summarization, we divide it into two situations, knowledge fusion and knowledge detachment, and develop two approaches (work 4 and work 5) to explore solutions for research problem 3. A brief overview of these works is summarized below.

Work 1 & 2: News Summarization via Event Information

To enhance the news summarization with its unique characteristic, we propose to incorporate a typical kind of extra information into the summarization model, the event-level information. The research target here is to investigate what role event-level information plays in both extractive and abstractive summarization, and how to make good use of them. In work 1, compared with sentence-level extractive summarization, we discover that using event-level semantic units for extractive summarization better aligns with the reference summary and reduces over extraction. To obtain the smaller granularity of the event semantic unit, we design a rule-based splitting algorithm based on the dependency tree. As for the model, we adopt BERT as the encoder and improve it with a hierarchical graph mask that incorporates the multi-level of granularities of the textual information into the model. In this case, it better captures the salience of different events with the help of structural information. Following work 1, work 2 proposes a generation model that aims to fuse multiple sentences or events into abstractive summary sentences. The fusion model is based on the event graph where the event elements (subject, predicate, object, time, location, etc) are represented as nodes. To effectively utilize the graph structure, we introduce graph flow attention to guide the generation process. Rather than just selecting the important information, this model focuses on integrating information, which brings a higher compression rate for text summarization.

Contributions: In work 1, we propose to extract event-level semantic units for extractive summarization to fill the gap between the gold summary and oracle labels, reducing redundancy and over-extraction. We develop a rule-based algorithm on the dependency tree to split sentences into events. To remedy the information loss after splitting sentences into facts, we propose a graph-based mask algorithm to impose the hierarchical structure information on a transformer-based encoder directly. This work is accepted by Computational Linguistics (COLING) as a conference paper [171].

In work 2, we propose a model to address the sentence/event fusion problem, which is critical for abstractive summarization. We establish an event graph to guide the fusion, which allows our model to utilize the structural event information and various cross-sentence relations. We innovatively propose a graph flow attention to control the fusion process via the graph structure. This work is accepted by Empirical Methods in Natural Language Processing (EMNLP) as a conference paper [172].

Work 3: Long-Input Summarization via Context-Aware Extract-Generate Framework

As the general transformer-based summarization models struggle with long-input summarization for their high memory cost, our research goal is to extend these models to address this problem. Inspired by the classic hybrid summarization models, an extractive to abstractive framework that adopts a divided and conquer strategy seems a solution. The source document(s) are truncated into multiple parts and important information is extracted separately. Finally, all the information is aggregated to generate the final summary. However, the cost of its effectiveness in dealing with long-input summarization is the loss of context information. In this work, we present a context-aware extract-generate framework (CAEG) for long-input text summarization, which aims to preserve both local and global context information with little cost. CAEG generates a set of context-related text spans called context prompts for each part of extracted information and uses them to transfer the context information from the extractor and generator. To find such context prompts, we propose to capture the context information based on the interpretation of the transformer-based extractor, where the text spans having the highest contribution to the extraction decision is considered as containing the richest context information. This model is flexible and potential to be applied to most long-input summarization models.

Contributions: In work 3, we first investigate the influence of context information

loss for extract-generate framework in long-input summarization and propose the CAEG for this problem. We introduce a new approach for capturing the context information based on the interpretation of the extractor. The experiment result shows that CAEG is capable of effectively preserving the context information from the extractor to the generator without largely increasing the complexity or memory cost of the model. This work is accepted by the AAAI Conference on Artificial Intelligence (AAAI) as a conference paper [170].

Work 4 & 5: Low-Resource Summarization via Knowledge Transferring

Although a large amount of high-quality news data has greatly promoted the development of general summarization models, not every summary task/domain has sufficient data. For current neural-based summarization models, the lack of data makes it difficult for these models to fully release their capabilities. Hence, we aim to utilize the summarization knowledge learned in news summarization to assist the other tasks/domains with insufficient data. In work 4, we investigate knowledge integration in the context of query-focused summarization, which can be regarded as the combination of general summarization and question answering. To address this problem, we propose prefix-merging, a prefix-based pretraining strategy for few-shot learning in query-focused summarization. We integrate the task knowledge from text summarization and question answering into a properly designed prefix and apply the merged prefix to query-focused summarization. Furthermore, we explore knowledge detachment in domain transferring for extractive summarization in work 5. In text summarization, context information has been considered one of the key factors for this task. Meanwhile, there also exist other pattern factors that affect importance, such as sentence position or certain n-gram tokens. Such pattern information is only effective in specific datasets or domains and can not be generalized. In this case, we attempt to apply disentangled representation learning on extractive summarization, and separate the two key factors for the task, context and pattern, for a better

generalization ability in the other domains.

Contributions: In work 4, we provide a new solution for few-shot query-focused summarization by decoupling it into two basic tasks with large-scale training data, text summarization and question answering. We propose prefix-merging that integrates the task-specific knowledge from basic tasks to assist the learning of a more complex task, which provides a new solution to many-to-one parameter-transfer learning. We further expand the application of prompt-based approaches by applying the prefix to multi-task situations, exploring the interaction between different task knowledge through prefix. This work is accepted by Empirical Methods in Natural Language Processing (EMNLP) as a conference paper [169].

In work 5, we propose to detach the common summarization knowledge from news summarization and apply it to the other low-resource summarization tasks. We discover that pattern information like sentence position or common n-gram tends to be domain-specific knowledge, while context information is more generalized. Hence, we build a disentanglement learning-based framework that aims to detach the context representation and pattern representation in extractive summarization. This work is accepted by the Association for Computational Linguistics (ACL) as a findings paper [168].

1.4 Structure of Thesis

The overall blueprint of the thesis is as follows. In Chapter 1, we first describe the background of the research on text summarization. This chapter also introduces the key research problems, research overview and contribution of this thesis. In Chapter 2, we provide a systematic survey on text summarization including an overview of the text summarization task and its related dataset, a detailed introduction to current neural model based summarization systems, and some key sub-topics requiring addi-

tional attention. Based on the three research problems proposed in Chapter 1, the following chapters can be divided into three parts. Part 1 (Chapter 3 and Chapter 4) mainly introduces the enhancement of news summarization models with event-level information. Chapter 3 presents an extractive summarization approach that takes events as basic semantic units. Following the previous work, chapter 4 investigates the event fusion problem for abstractive summarization that aims to compress multiple events into one sentence. In Part 2 (Chapter 5), we tend to focus on establishing a context-aware extract-generate framework for long-input summarization by extending the general summarization techniques in new application scenarios. Part 3 (Chapter 6 and Chapter 7) considers the knowledge transferring from news summarization to other summarization tasks/domains. Chapter 6 explores knowledge fusion in the context of query-focused summarization, and Chapter 7 focuses more on knowledge detachment to adapt the knowledge in news summarization to various other domains like science papers or dialogue records. In Chapter 9, the last Chapter, we conclude our proposed approaches, their contribution and our findings. We further discuss the potential directions and suggestions for future works.

Chapter 2

Literature Review

In this chapter, we survey the studies that are relevant to our research works in this thesis. We mainly focus on summarizing text summarization from the perspectives of tasks, evaluation, and existing approaches including both extractive approaches and abstractive approaches. We also give a wide review of some important sub-tasks in the field of text summarization and discuss their unique characteristics and solutions.

2.1 Text Summarization System Overview

Text summarization has been one of the classic NLP tasks and it is in development. Formally in existing text summarization systems, the user takes document(s) (such as articles, documents, or news) as input, and the system returns a concise and coherent version of the document(s). The purpose of text summarization is to extract and condense the most important information and key points of a given document, allowing users to quickly grasp the main content without having to read the entire text.

According to different approaches to composition summaries, text summarization can be categorized into two main types: extractive summarization and abstractive

summarization. The former extractive summarization systems directly utilize the existing sentences from the source document(s) as components of the summaries. The latter abstractive summarization systems choose to use the natural language generation models to generate the summaries token by token. The very essence of the two types of approaches is the same, which is to understand the input document(s) and find its crucial information. Therefore, hybrid approaches combine both the extractive and abstractive approaches.

With the development of text summarization, these approaches have garnered a lot of research attention due to their numerous applications in various fields, including information retrieval, document categorization, news aggregation, and even chatbots and virtual assistants. It can help users quickly scan a large amount of text and make informed decisions based on the summarized information. With the advancements in deep learning and NLP techniques, automatic text summarization systems are continuously improving, becoming an essential tool for information management and consumption. In Figure 2.1, we display the different classifications of existing text summarization systems.

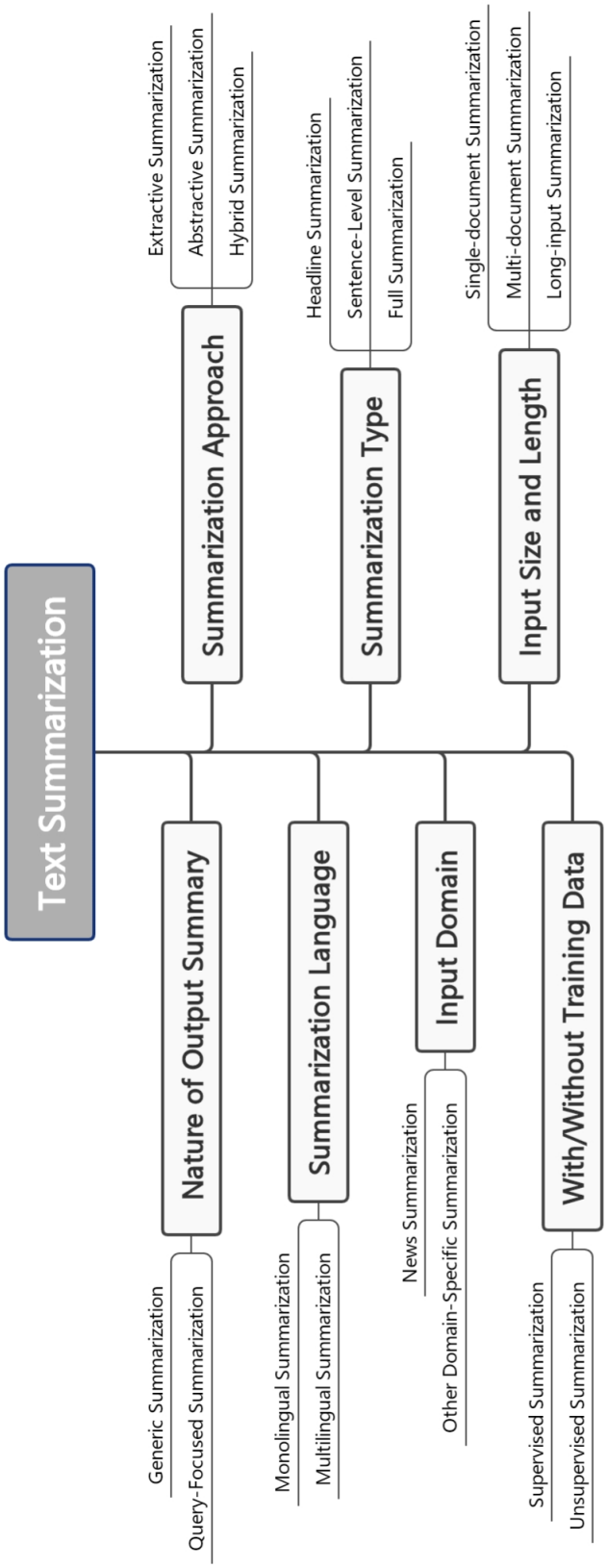

Figure 2.1: Classifications of the text summarization systems.

2.1.1 Extractive Summarization

Extractive models convert the summarization problem into a ranking problem which ranks the sentences in the source document based on their importance and concatenates the top-k sentences as the final summary. This approach is able to maintain the salient information while ensuring its faithfulness to the original text. Therefore, extractive models have been the primary approaches before the appearance of neural models. Unsupervised models like TextRank [101] apply structure information in the graph for importance ranking, which shows its great ability to identify salient information. Other extractive models [154] rely on features such as keyword/phrase extracted by TF-IDF, position of the sentences and similarity with the document topic. With the development of neural models, researchers aim to obtain sentence representations and formulate this problem as a binary classification problem. As one of the earliest neural summarization model, SummaRuNNer [104] generate sentence representation with an RNN encoder and applies an MLP to decide whether a sentence will be included in the final summary. Some other researchers improve the result with more complicated models. [192] presents a decoder that jointly learns to select and score sentences. [159] combine information from both sentence and paragraph for importance modeling. In recent years, pretrained models have shown their ability in representation training, and extractive models based on pretrained models [92, 2, 179], especially BERT have made a step forward. [186] further enhances the quality of sentence representation by converting the extraction process into a text-matching problem.

However, extractive models also face some limitations. Firstly, for most summarization datasets, gold summarization is usually abstractive and there are no oracle labels for extraction. And the oracle labels are generated with a greedy algorithm that maximizes the rouge score between oracle and gold. Therefore, there exists a gap between the oracle summary and the gold summary. Some extractive models try to fill this gap through directly learning from gold summary with a reinforcement model

[107] or a latent variable model that maximizes the likelihood of human summary [178]. Secondly, reputation information between sentences has been a key problem for summarization, especially extractive summarization. For most current models, the Trigram Blocking algorithm and Maximal Marginal Relevance [97] are applied to reduce redundancy. One recent research [108] proposed a step-wise extraction model to model the redundancy problem using a sequence decoder. Thirdly, extractive summarization fails to cope with highly abstract summarization tasks. Since the sentences in the extractive summary are copied from the source document, it is impossible to fuse information from various sentences into one summary. For example, when the position and time of a certain event are described in two different sentences in the original text, the extractive model will either cause redundancy by selecting both or lose part of the necessary information, by selecting one of them. Under this situation, abstractive summarization will have a better performance.

2.1.2 Abstractive Summarization

Abstractive summarization has attracted more and more attention with the development of neural language models. Compared with the extractive model, abstractive models tend to understand the input document and generate the summary through compression, paraphrasing and fusion. This allows abstractive summarization to have a higher upper bound and the ability to generalize to more real-life applications. Before the age of the neural model, previous researchers tend to use heuristic approaches and human-written rules/templates for text generation. [155] use integer linear programming for abstractive multi-document summarization. This framework allows it to utilize the output of the expert learners and to generate aspect-focused summaries by predefined objective settings, with soft or hard constraints. [141] also adopt integer linear programming to investigate sentence fusion in abstractive summarization. [16] combine the dependency tree from the source document to reconstruct the summary sentences. [116] proposes a summarization framework based on event templates

extracted from the training corpus.

Fortunately, the advancement of the neural model brings a new paradigm for natural language generation. Data-driven sequence-to-sequence generation allows abstractive summarization to generate fluent language compared to humans. [124] uses a large corpus of documents and corresponding headlines to train a neural summarization model. [18] further extends it with an attentive recurrent neural network framework. [105] enhance the performance of RNN s2s model with various techniques. Pointer-generator [130] adds a pointer mechanism that allows to copying words directly from source documents, and applies a coverage algorithm to avoid reputation problems. Abstractive models begin to show their power with the revolution of the pretrained model. Bart [78], a pretrained Transformers finetuned on a supervised summarization dataset, has achieved state-of-art performance on all abstractive summarization models. In addition to pure S2S models, researchers tend to use external information to assist in better understanding the input document. However, one limitation of approaches based on pretrained models is that it is inherently difficult to control and prone to unfaithful description [76, 9]. This happens because the model struggles to generate salient words and maintain the fact description.

2.1.3 Datasets and Evaluation

In early ages, researchers establish text summarization systems using feature-based methods and human-written rules/templates. Despite the simplicity, these methods require tedious human efforts on feature selection or creating abundant rules/templates to foster the model. More importantly, the performance of these methods is far from satisfactory, especially when it comes to abstractive summaries. To address this concern, researchers have been exploring data-driven approaches and creating multiple datasets. Before this decade, the widely-known datasets in this research field include DUC 2003-2006 [23] and TAC 2010 [57]. For example, the Document Under-

standing Conference (DUC) dataset is a collection of news articles and corresponding summaries that were created by the National Institute of Standards and Technology (NIST) from 2001. It includes various genres and is a popular benchmark for both extractive and abstractive summarization tasks. However, these datasets are often limited in size and the systems trained on them are constrained to specific domains.

Within the fast development of the neural model based text summarization system, various large-scale datasets have been proposed. Among them, datasets for news article summarization has garnered the most attention due to such natural application scenario. CNN/DaliyMail [105] is the most representative benchmark for both extractive and abstractive summarization tasks. This dataset consists of news articles from CNN and Daily Mail sources along with summaries written by human editors. Its summaries contain 3 to 5 sentences, many of which are directly aligned with sentences in the source article. Other classic news summarization datasets such as Newsroom [47] or New York Times corpus are also similar to CNN/DaliyMail. XSum [106] takes a step further and focuses on highly abstractive summarization of news articles. In this dataset, each source article only has a one-sentence summary that is highly compressed, which brings more challenges to the information fusion ability of summarization models. Multi-News [34] is also in the domain of news articles. Instead of summarizing a single document, this dataset aims to summarize multiple news articles related to one topic. Gigaword [130] pays attention to sentence-level summarization, aiming to generate a headline from the first sentence of news articles. Multi-lingual news summarization is another important topic. Datasets such as MlsumDataset [129] and XLSum [48] are proposed to foster the study on this topic.

Although news summarization has been considered as a key content in the field of text summarization, researchers do not limit themselves to the news domain, more datasets are proposed for specific domains and application scenarios. ArXiv [19] and PUBMED [19] are datasets for scientific article summarization. The articles are from the famous preprint website ArXiv.org and the datasets take the abstract section for

each article as the pseudo summaries. Billsum [66] focuses on the summarization of US Congressional and California state bills, which extend the edges of text summarization to the legislation domain. GovReport [54] aims to generate summaries for government reports that contain thousands of words. SummScreen [13] concentrates on summarizing TV series transcripts, and it utilizes the concatenation of human written recaps as summaries. Wikihow [67] is an instruction summarization dataset based on the question-answering website www.wikihow.com, where users ask for step-by-step instructions on certain topics. Dialogue summarization is another rising research topic in the field. SAMsum [46] is a dialogue summarization dataset built on open-domain dialogue, and its summaries are written by human linguists. QMSum [189] is another dialogue summarization dataset, but for long dialogue such as meeting records. Query-focused summarization requires the model to generate the summary based on a user-given query, which plays an important role in information retrieval. DUC2005-2007 [23] first introduces a query-focused summarization task and provides a human-generated dataset containing question-summary pairs. Unfortunately, since it is not a large-scale dataset and only contains hundreds of examples, it only can be used as a test set for neural-based summarization models. Debatepdia [109] also has been used to train a query-focused summarization model. It is a QA dataset consisting of yes/no questions and for each yes/no answer it provides an explanation that supports its answer. The researchers consider the explanation as the summary. Similar to Debatepdia, PubMedQA [60] also provides a long answer and a short yes/no answer for a question. Unfortunately, few of them are specifically built for this task. In the era of language models, the input limitation for language models becomes another challenge for text summarization for long documents. Hence, research on long-input summarization also become popular, and the datasets that can be used for this task include ArXiv [19], PUBMED [19], QMSum [189], GovReport [54] and etc.

Evaluation is an inevitable problem in the development of text summarization. Gener-

ally, automatic measurements such as N-gram matching are the most commonly used criteria when evaluating the summary quality. Metrics like ROUGE [86] or BLEU [110] aims to evaluate the overlapping of the words between the candidate summary and the reference one on different granularity levels. The most widely used metrics include ROUGE-1 (unigram overlap), ROUGE-2 (bigram overlap), and ROUGE-L (a metric based on the longest common sequence). It is worth noticing that the N-gram-based metrics only represent word-level similarity and can not capture the hidden semantics in the comparison. Nevertheless, summaries can be expressed in various forms and do not necessarily strictly follow the reference. It is unreliable to evaluate the quality of the summarization solely based on the n-gram similarity of a pre-defined ground truth. Therefore, traditional N-gram metrics often fail to show a high correlation with human judgment. In order to better evaluate such text generation problems, metrics based on neural models have been proposed. BERTSCORE [176] uses the similarity between the context embedding generated by a language model to replace the hard N-gram matching. MOVERSCORE [182] adopts pretrained word embeddings to calculate the semantic similarity through Word Mover's Distance. Except for the similarity-based evaluation, another crucial requirement for a good summary is faithfulness to the source document.

2.2 Existing Approaches Based on Neural Model

Neural-model-based text summarization models have dominated the field of text summarization for its performance, scalability and versatility. Neural models, especially those based on pretrained models, are capable of understanding the context of the text better than traditional methods. They can capture long-range dependencies in the text, which is crucial for generating high-quality summaries. In this case, neural-model-based text summarization becomes a main stream approach that attracts lots of attention. In the following section, we will introduce four important types of mod-

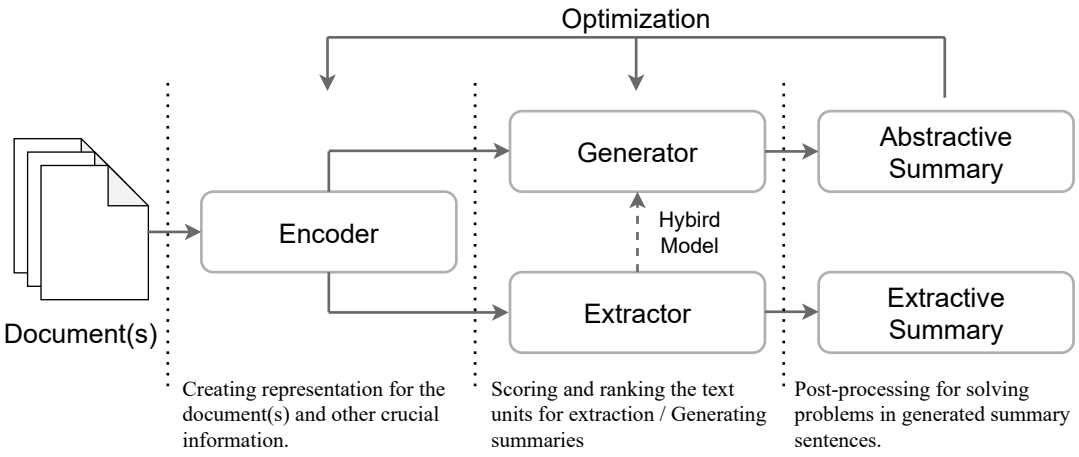

Figure 2.2: Framework for text summarization approaches based on neural model.

ifications on a basic neural model framework shown in Figure 2.2, including changes in fundamental network structure, encoder, decoder/extractor.

2.2.1 Modification on Network Structure

As for the application of the neural model in NLP, [102] is the first to use the vanilla RNN. Unfortunately, suffering from the vanishing gradient or exploding gradient problems, vanilla RNN fails to achieve real success in this field. The later work [138] adopts Long Short-Term Memory (LSTM) as the recurrent unit to solve this problem. However, these models only read the source text in a unidirectional manner, either forward or backward, which means each word can not focus on its subsequent context on both sides. To address this, bidirectional LSTM [137] proposes a bidirectional architecture that considers the entire source sentence for all word predictions, which has become a classic configuration in seq2seq generation. Except for the LSTM recurrent unit, other researchers also propose a competitive structure called Gated Recurrent Unit (GRU) [17]. Compared with LSTM, it has fewer parameters by removing the output gate from the LSTM. This makes such structure have a higher training and

inference speed and perform better on some small-scale datasets. In conclusion, the bidirectional RNN with LSTM or GRU recurrent unit and its variants become the first generation of network structure of neural language model. They are widely used as encoders for representation generation and decoders for sequential text generation. Other researchers attempted to explore the substitution of the RNN architecture. For instance, [42] adopts the Convolutional Neural Network (CNN) for the encoder. Compared to RNN, CNN allows the model to parallelize all elements during training and better exploit the GPU hardware.

Transformer is a type of deep learning model for NLP introduced by [145]. Transformer is based on the attention mechanism, which allows the model to focus on different parts of the input sequence during the encoding and generation. This is in contrast to traditional RNN models, which process the input sequence in a fixed order. The transformer achieves a comprehensive interaction between tokens by stacking multiple attention layers. Although the performance of transformer-based models do not largely exceed traditional RNN initially, the more scalable structure of the transformer provides it with the capacity to accommodate more parameters. With the introduction of pre-training approaches for language model [27, 119, 78], transformer soon becomes the most suitable structure for the pretrained language model. Language model pretraining is a technique used in natural language processing (NLP) to train a language model on a large corpus of text data with a set of unsupervised tasks such as next token prediction. It allows the model to learn the statistical patterns and relationships within the language. BERT [27] first proposes a pretrained deep bidirectional transformer encoder by joint conditioning on both left and right context in all layers. It can be directly used in downstream tasks by fine-tuning with the output layer, and achieves state-of-the-art performance for a wide range of tasks. BART [78] introduce a pretrained sequence-to-sequence model containing both encoder and decoder with an arbitrary noising function. It is widely used in various natural language generation tasks such as translation, question answering, and text

summarization. GPT [119] proposes a left-to-right transformer that predicts tokens auto-regressively. This this case, the encoder and decoder is combined in the same neural model structure. Pegasus [174] further extend the pretraining tasks specifically for text summarization task. It removes the important sentences from the input documents and aims to generate the masked sentences during the pretraining from the remaining sentences, which makes it stand out in many downstream summarization tasks. T5 [121] explores the idea of transferring learning techniques for NLP by constructing a unified framework that converts all text-based language problems into a text-to-text generation format. Overall, the pretrained transformer model and its attention mechanism have been highly influential in the field of NLP, enabling significant advances in tasks like text summarization.

2.2.2 Modification on Encoder

The modification on the encoder has two main types: the modification on the network structure itself and the injection of external information such as semantic, topic and graph. Considering we have introduced the development of neural model structure in NLP, we would like to describe how different types of context information are combined in the text summarization system through the encoder.

Syntactic-Based Model

Fine-grained information like syntactic information and semantic information has been used in a set of traditional natural language generation models such as syntax-based SMT and compressive summarization. Therefore, the sequence-to-sequence model with syntax information is initially proposed in the field of machine translation. [79] converts a phrase parse tree as a label sequence and extends the standard encoder to take both word sequence and label sequence as input. [49] believe that a parser trained in advance is not effective enough for the generation in the current task.

Hence, this work suggests to train a parser together with the encoder. Instead of using a common dependency parse tree, it designs a parse graph where a single word can hold several headwords. An attention layer is added to the encoder to learn the dependency between the source sentence and each word.

Semantic-Based Model

Text summarization models attempt to combine semantic information with the encoder to better select the salient information. For instance, [105] enhances the encoder with various lexical and statistic features that play important roles in traditional text summarization, including part-of-speech tags, position and named entity. In this case, the traditional feature verifies its effectiveness in the neural-based summarization model. [193] proposes a selective gate network and adds it over the encoder's hidden states. It generated a higher-level sentence representation by limiting the information transfer to the decoder, which tailors the source text based on its importance. In transformer-based model [27], a pretrained position embedding is added to the input token embedding for the crucial position information.

Graph-Based Model

Graph structures have long been exploited for text summarization. Early works such as Textrank [101] and Lexrank [32] adopt similarity-based graphs to find salient information. As for neural models, graph neural network (GNN) [127] is usually adopted to model highly structured data. [140] designs a graph-based attention in the sequence-to-sequence framework to identify the important factor of summarization. [38] enhances a sequence-based encoder with GNN to take token-level entity types into consideration. It aims to conduct reasoning about long-distance dependency from the source text. Its experiment shows it outperforms both pure sequence-based models and pure graph-based models. Other researchers focus on making use of more

complicated graphs that contain richer structure semantic information. [90] use the Abstract Meaning Representation (AMR) in summary generation. The source text is first parsed to multiple AMR graphs and the graphs are further compressed into a summary graph. Through graph-to-text generation, it transformed the summary graph into an abstractive summary. [153] transfer the summarization problem into a latent tree learning in the encoder and propose a reinforcement learning method to optimize the model. [55] transfer event triples extracted from OpenIE to an event graph to acquire semantic interpretation over input and propose a semantic-driven reward to train the model. [85] further extend this idea and propose an event-level discourse graph for abstractive summarization. Other researchers use event information for the generation of more faithful summaries.

Event-Based Model

Currently, most of the summarization tasks focus on the news domain, which is strongly correlated with events, since most news article is composed of a series of events. Therefore, how to leverage event information in summarization models becomes a key problem. One advantage of event information is that it helps the model understand the SVO relations between sentences and fuse them as one final event description. This problem is especially crucial in multi-document summarization, since a single event may have various descriptions with multi-source. In fact, it is still challenging for current state-of-art models [74, 71]. Another advantage is that event information contributes to generating faithful summarization. Faithfulness in summarization refers to whether the components of an event (subjects, predicates, objects, time, position and number, etc.) follow the source document. In this way, event information can provide additional constraints for the generation process [9]. Inspired by the advantage of event information, the event-based summarization model has been well-researched before neural language models. [24] conduct a pioneer hand-based experiment on event-based multi-document summarization. Other researchers [175, 82]

propose pattern-based approaches that enrich the representation of sentences using an event key term and other related entities. [80] propose a clustering-based approach to automatic generation of aspect-oriented summaries from multi-document. [161] utilize Element Discourse Unit (EDU) as components to conduct extractive summarization. Each EDU can be regarded as a smaller event-level granularity compared to sentences. [9] propose an additional event relation encoder for event triples and allow the decoder to balance the informativeness and faithfulness during the generation. [71] tries to solve the sentence fusion problem in summarization by linking representations of shared components of an event under a transformer encoder.

Topic-Based Model

Utilizing topic information to assist summarization is another approach that has been explored for a long time. The LDA topic model is most commonly used in topic-based summarization. [11] presents an extractive summarization model that utilizes hierarchical representation of words, sentences and documents in a corpus. It enhances the model with the distributions for latent topics in word level and sentence level with the LDA topic model. [158] first introduces an extractive summarization model based on topic words selected by a topic model. [148] proposes a deep learning approach to tackle the text summarization task by incorporating topic information into the convolutional sequence-to-sequence model and using the self-critical sequence training for optimization. [106] utilizes a convolutional neural network conditioned on the article's topics to capture the summarization patterns under different types of articles. [151] proposed a novel topic assistant on a transformer-based summarization model to learn explicit document semantics. The topic assistant is a plug-and-play model that can be directly inserted into the original transformer framework and it only contains a small number of parameters. Aspect-based summarization has attracted more and more attention in recent years. [35] suggests that one topic in given source documents can be described by various aspects and different topics prefer different aspects.

The work aims to exploit appropriate priors to generate aspect-based summarization based on group selection. [156] investigate aspect-based summarization on reviews on particular products. To consider both aspect mapping and sentiment classification, the paper proposes two CNN networks to model the information. [26] proposed a Wikipedia-based dataset that contains topic-level supervision. [41] believe that new document should not be treated as 1 to 1 from document to summary, since there are multiple aspects of information. The model first learns to accurately segment documents by several predefined topics and then generates multiple summaries based on the segment. [139] is not limited to a small set of predefined topics but searches for potential topics with external knowledge sources such as ConceptNet and Wikipedia with a weak supervision approach, which significantly expands the application of the task in practice.

2.2.3 Modification on Extractor and Generator

Extractors for extractive approaches are classifiers that determine whether a sentence will be included in the final summary. For neural models, they are usually multi-layer perceptron (MLP) that conduct binary classification on sentence representations [104, **?**]. To identify each sentence, a start token is usually inserted to the beginning of each sentence and its hidden state is considered to represent the whole sentence. The summary is the most important sentences with the top-k prediction score. Its length depends on a predefined sentence number or a compression rate using a length cutoff. In this case, few works focus on modifications on the extractor. Instead of simply converting extractive summarization into a ranking problem, [192] presents a decoder that extracts sentences sequentially by jointly scoring and selecting sentences with an RNN.

Generator for abstractive approaches usually follows the same neural structure as the encoder in a sequence-to-sequence framework. For example, RNN-based models

adopt the RNN decoder to generate the summary token by token. The decoder usually packs all the additional information from different granularities in the same timescale of RNN. [191] introduces a hierarchical RNN decoder that splits the hidden states of the decode into phrase state and word state. The phrase state is first updated and then the word state is generated on it. Although the encoder in RNN is bi-directional, the decoder only works in a left-to-right manner. To fully utilize information from both directions, [177] extends the RNN decoder with a backward decoder that enables bi-directional text generation. Another important factor for summary generation is the length control. [63] propose a method that adds the expected length information to the decoder by transferring it into additional initialization parameters of the decoder. For transformer-based models, the decoder applies the cross-attention, an attention mechanism that aims to pass information from the encoder to the decoder. [30] propose to use multiple layers of cross-attention to transfer information from different sources to the decoder. Generally, the decoder predicts each word by conducting a multi-label classification of all vocabulary. Some researchers believe it is too large for the generation and choose to modify the output dimension of the decoder to enhance the efficiency of learning. [56] limits the decoder to generate the words only from the parallel token and a sampled word set in machine translation. In text summarization, following a similar idea, [105] extend the output vocabulary with the neighboring words from the source text, which are measured by the similarity between the word embedding.

2.2.4 Modification on Optimization

There are four types of optimization approaches for text summarization models: binary classification loss, maximum likelihood estimation (MLE) loss, reinforcement learning and contrastive learning. Binary classification loss focuses on is used in extractive summarization, where the classifier determines whether a sentence will be included in the final summary. MLE loss, a traditional text generation loss, is used in

abstractive summarization. It helps in achieving this by maximizing the probability of the next word in the sequence given the previous reference words. However, MLE loss still faces an "exposure bias" problem. It refers to the discrepancy between the way a model is trained and the way it is used at inference time. During training, the model is typically trained to predict the next token in a sequence given the true previous tokens. However, during inference, the model generates each token based on its own previously generated tokens. This can lead to errors propagating through the generated sequence. Reinforcement learning aims to solve this problem by directly aligning the optimization target with the summary quality. [113] combines likelihood and ROUGE score in the loss function by applying a reinforcement learning algorithm. [112] proposes two novel reward functions: ROUGESal and Entail based on the previous work. ROUGESal enhances the ROUGE metric by assigning greater weight to important phrases or words identified through a keyphrase classification model. Entail rewards the summaries that logically follow the original text, as determined by an entailment classifier. Reinforcement learning also works for extractive summarization, [165] presents a model based on a Deep Q-Network. It predicts the importance and redundancy of sentences with Q-value approximation and aims to learn a policy that maximizes the ROUGE score with respect to the reference summary. Contrastive learning achieves a similar goal with reinforcement learning but with a totally different method, which transfers the training target into a ranking problem. [186] gets rid of the traditional extractive framework of extracting sentences individually, and transfers the problem into a semantic text-matching problem. Training with a contrastive learning objective, the closest candidate summary to the source document in the semantic space is the final summary. [95] proposes a new training paradigm by assigning different candidate summaries with probability mass according to their quality in abstractive summarization. Such contrastive learning objective allows it to maximize the probability of generating high-quality summaries.

2.2.5 Hybird Model

The hybrid models combine both abstractive and extractive approaches. The two complementary approaches allow the model to take advantage of both sides and improve the overall performance of text summarization. There are mainly two types of hybrid models: extractive to abstractive models and extractive to shallow abstractive models. Extractive to abstractive methods start by using extractive models to obtain salient sentences and then use these sentences for abstractive summarization. This type of model is also widely used in the field of long-input summarization, where the input length is too long for existing pretrained language models. These works [150, 117, 183, 4, 180] usually adopt a divide and conquer strategy by firstly extracting salient information from truncated source documents and then aggregating them to final abstractive summaries. DYLE [98] extends this approach by proposing a dynamic latent extraction approach that is capable of optimizing the extractor based on the summary generation loss. Other works tend to use extracted content to assist the summary generation. [125] extracts important information including keywords or crucial sentences and feeds them to the summarization model. [30] uses extractive summarization results as guidance for abstractive summarization. Together with the source document, it allows the model to generate more faithful summaries. Instead of generating completely abstractive summaries, extractive to shallow abstractive models tend to modify the extracted sentences while maintaining their original meaning. [96] use information compression and fusion techniques on the extracted sentences. [74] extracts a set of sentence clusters containing similar sentences and then uses a generation model to fuse each cluster to an independent summary sentence.

2.3 Sub-Tasks for Text Summarization

To fulfill the requirements of real-life problems, text summarization is not limited to generating a summary for the document(s). Variant tasks for text summarization are also considered important problems in the text summarization community. In this section, we mainly introduce three popular text summarization variant tasks and their unique problems and solutions compared with common text summarization.

2.3.1 Multi-document Summarization

Despite generating the summary from a single document, multi-document summarization (MDS) is another major topic in the field. Compared to single-document summarization (SDS), multi-document summarization has two major problems: overlap between documents and cross-document relation modeling. Under a similar topic, a single fact can be described by multiple documents several times and these descriptions may share some overlapping content but also contain unique information. Fusing this unique information while avoiding overlap is one challenge for multi-document summarization models. Since there is no sequential relationship between the input documents, understanding the ways to use semantic-level correlation to build up cross-document relations is important. Moreover, the total length of the input of the MDS model is usually far longer than its summary, which means there exists a large amount of unnecessary information in the input. For current state-of-the-art transformer-based models, how to perform effective preprocessing becomes a rising problem.

To reduce the redundancy in MDS, [75] adopts the maximal marginal relevance method to extract important sentences from multi-document input, and utilize a generation model to fuse these sentences to the final summary. [34] introduces a MDS dataset called Multi-News, and proposes a model combining the traditional extractive

summarization model with the single document summarization model. [81] develops a MDS model that leverages graph structures. The graph can be the similarity graph or the discourse graph, and it allows the model to capture the cross-document relations from multiple input documents. [59] introduces a multi-granularity interaction network for MDS. It aims to jointly learn the semantic representation of different granularity including documents, sentences and words. An attention mechanism is employed to achieve interaction between these semantic representations. [147] proposes a heterogeneous graph-based neural network for extractive summarization in MDS. Similarly, this work also adopts multi-granularity information, but uses the nodes in the graph to represent it. In this case, this work is flexible in both SDS and MDS.

2.3.2 Query-Focused Summarization

Different from traditional text summarization, query-focused summarization aims to summarize the source document with a specific topic or query. Since the source documents on the Internet may contain various aspects of information, query-focused summarization is capable of providing the user with a concise highlight on the aspect he/she would like to know. Therefore, we believe this task better meet the requirement of the users during the implementation of text summarization. Query-focused summarization has attracted more and more attention with the development of the question answering task. Although the question-answer pair is relatively easy and the answer often refers to a specific entity in the field of QA, understanding how to answer a complex question like how or why is an important challenge. Under such a situation, query-focused summarization is able to summarize the evidence retrieved by the QA model and generate a summary to answer the question. Therefore, we believe query-focused summarization is strongly related to QA and is essential to further improve the current QA system.

Query-focused summarization is similar to Question Answering but also different in some aspects. Compared to the development of QA, query-focused summarization has not been fully explored. Before the invention of the neural model, [152] propose a probabilistic approach to model the topic distribution of both the query and the source document and extract the related sentences as summaries based on the given query. [109, 11] first applies a sequence-to-sequence model on the query-focused summarization task. The researchers propose a query attention model that learns to focus on different portions of the query at different time steps. [160] utilize models in information retrieval and QA to extractive query-aware evidence from coarse to fine and proposes an unsupervised graph-based summarization model to generate the final summary. [25] presents a novel conditional self-attention that limits the information diffusion among the source tokens based on the relevance between the tokens and the query. [50] believe that the intermediate output is only treated as the middle steps in non-factoid QA, but they are important to form a complete answer. Therefore, a query-focused summarization model is supposed to build up a bridge between the multi-hop inference and summarize them to form the final answer. Multi-document query-focused summarization is another problem that is worth exploring. Different from QA, the model needs to collect information from multiple documents and generate a summary as a compressed highlight. In such a process, the role of summarization should be emphasized, which is a major difference between QA. This also makes researchers believe that multi-document query-focused summarization is one of the most potential problems in the field. [163] adopts a mutual reinforcement chain and incorporates the query influence into the mutual reinforcement chain to cope with the need for query-focused multi-document summarization. [190] proposes a theoretical connection between sentence retrieval and the extraction of a transversal in a hypergraph, and uses the approach to capture the sentences that are related to main query-relevant topics.

2.3.3 Dialogue Summarization

Dialogue summarization is a newly emerged task that has attracted a lot of attention. Dialogue summarization extracts useful information from a dialogue. It helps people quickly capture the highlights of a dialogue without going through long and sometimes twisted utterances. Compared with pure text summarization, dialogue summarization require the model to face some unique features in dialogue situation such as different speakers and informal text. We conclude them as four main challenges: Multiple participants, conversations with more speakers are harder to be summarized since it may require models to accurately differentiate both language styles and content from different speakers. Role change, the role of a speaker may shift from a questioner to an answer, requiring the summarization model to dynamically deal with speaker roles and the associated language. Scattered Information, information is generally scattered throughout the whole conversation, and speakers may interrupt each other, reconfirm, back channeling or repeat themselves. Unstructured text, unlike news reports or encyclopedia articles, dialogue data do not have natural paragraph/document-level segment. Meanwhile, the absolute position information is not effective in dialogue summarization compared to document summarization. These features make summarization model more difficult to identify the importance of a sentence.

Facing different real-life situations, previous works have proposed different types of datasets including meeting record [68], customer service and daily conversation [46]. Considering the utterance nature of dialogue summarization, researchers first consider a hierarchical encoder to generate the representation for both word-level and utterance-level. Moreover, [184] adopts an addition encoder to learn the semantic representation of conversation with adaptive conversation segmentation at a higher level, where each segment represents a series of utterances under one major topic. Since different people in a conversation may play different roles, especially in meeting dialogue and customer service, [184] propose to use a role vector to model different

people in the dialogue. Facing the limitation of data shortage, it also proposes an approach to generate pseudo data from text summarization dataset. [12] aims to leverage multiple views including topic and stage to better understand the conversational structures of the dialogue, while [37] aims to use a graph structure to capture the dependency between the multiple utterances.

2.3.4 Long-Input Summarization

Long-input summarization is a sub-topic of text summarization that focuses on generating concise summaries from long pieces of text. These could be lengthy articles, research papers, books, or even transcripts of speeches or interviews. Traditional summarization models often struggle with long inputs due to memory constraints and the difficulty of maintaining coherence over a long span of text. Long-input summarization is a challenging task, as it requires the ability to efficiently process thousands of tokens while preserving the long-term dependency across the input. More importantly, the high efficiency usually leads to the loss of long-term dependency, which makes this task even more difficult.

Potential solutions involve the utilization of transformers with a sparse attention mechanism. Longformer [6] integrates sliding windows with global attention patterns, which leads to much less memory consumption. Similarly, BigBird [173] follows the same idea and replaces global attention with random blocks. Reformer [64] uses locality-sensitive hashing as the replacement of the dot-product attention. This enhancement reduces the computational cost of the model. Despite focusing on self-attention, some researchers [54] suggest the use of head-wise positional strides to enhance the efficiency of cross-attention. Some hierarchical models is another solution for long-input summarization. HAT-Bart [123] introduces a transformer with hierarchical attention that leverages information from multiple granularity including sentence and paragraph. HMNet [194] proposes a hierarchical structure that cap-

tures discourse-level information and speaker roles for long dialogue summarization. [117, 183] apply the extract-generate framework to the long-input summarization to overcome the capability limitation of transformer models such as BERT or PEGASUS. with the generalization ability of the pretrained language model, [4] focuses on low-resource long-input summarization. [180] conduct an exploratory study on what techniques are effective for long dialogue summarization.

Part I

News Summarization via Event Information

Chapter 3

Event-Level Extractive Summarization with Hierarchical Graph Mask on BERT

3.1 Introduction

In extractive summarization models, sentences are regarded as the basic textual units making up the final summaries in most extractive summarization models. One of the problems with sentence-level extractive summarization is that there exists a gap between the human-written gold summary and the training objective. Since most datasets do not contain sentence-level labels, oracle summaries are normally generated by a greedy algorithm, which maximizes the ROUGE score between the sentences in the source text and in the gold summary [104, 92]. As pointed out by [107], sentences with high ROUGE scores may not necessarily lead to an optimal summary. Such discrepancy may be due to the overlapping contents and over extraction. Therefore, some researches try to perform extraction at a lower level such as words or phrases [8, 14, 43]. Although these models are able to learn which phrases or words are more

important directly from the gold summary, it is hard to maintain semantic integrity when information is scattered. For the news article, as stated by [10], it usually can be decomposed into a series of crucial events. For example, a long sentence in a news article may contain two events, one in the main clause and one in the subordinate clause. In this case, we believe event can be a proper granularity between phrases and sentences for news summarization.

In this chapter, we propose to extract event-level information as the primary textual units to generate summaries in news summarization. Here, we define an event as the smallest unit that contains complete subject verb object information, which is a granularity smaller than sentences and bigger than words. In other words, one sentence can contain multiple events. In this work, we first develop a heuristic algorithm to split a sentence into one or more smaller units, where each unit is considered as a single event description. Then we apply a one-to-one strategy to match each event in the gold summary to one event in the source text to obtain the oracle label. In this way, we smooth the gap between the gold summary and oracle labels and further reduce redundancy and over-extraction. Meanwhile, event units can still well keep semantic integrity and thus are able to provide faithful summaries. Considering events are relatively independent semantic units, in order to maintain rich contextual information and to capture potential relationships among events, we believe the sentence-level information still plays an important supporting role. Moreover, based on the assumption that the selected events should be semantically close to the majority of source documents, we also import the document-level information as part of our model. We will show that such a event-sentence-document hierarchical text modeling facilitates to capture more structured contextual information for effective event extraction.

Recent works [92, 2, 179] have demonstrated that it is highly beneficial to apply pretrained language models such as BERT to extractive summarization models. Following this trend, we adopt a BERT-based encoder to generate contextualized repre-

sentations for further extraction. It is challenging to impose the structure information when fine-tuning the BERT model in a downstream task. [36, 179] separately encode representations for each granularity with BERT and then capture the structure information with a graph network stacked upon the encoder. However, this method not only results in a large complex model but also cannot fully utilize the advantage of pretrained language models. Based on the idea that BERT can be regarded as a full connected graph, we propose to directly impose the structure information on BERT with a graph-based mask to jointly learn contextual representations of different text granularities within a single BERT. On the one hand, we significantly reduce the time cost of running BERT several times plus running a graph neural network to running BERT once. On the other hand, we are able to utilize the complex structure of BERT to capture the structure information.

We conduct experiments on the CNN/DailyMail dataset, a well-known extractive summarization benchmark dataset for news summarization. The results show that using events as extractive units improves the summarization quality and our model achieves better performance than the state-of-art model. The contributions of our work can be summarized as follows. (1) We propose to extract event-level semantic units for extractive news summarization to fill the gap between the gold summary and oracle labels, reducing redundancy and over-extraction. (2) We import a hierarchical structure to remedy the information loss after splitting sentences into events. (3) We propose a graph-based mask algorithm to impose the structure information on BERT directly. To the best of our knowledge, we are the first to combine the pretrained language model and the structure information without increasing the scale of the model, which provides a new idea for fine-tuning pretrained models on downstream tasks.

3.2 Related Work

Extractive Summarization Extractive summarization attempts to select the most important sentences from the source text and subsequently concatenate them as a summary. With neural network models, researchers usually formulate it as a sentence binary classification problem. SummaRuNNer [104], one of the earliest neural summarization models, applies an RNN-based encoder to generate sentence representations and a neural classifier to determine which sentences should be included in the summary. [107] further extends SummaRuNNer with a reinforcement model, which optimizes the summary-level ROUGE metric. Some other works achieve better performance through more complex models. [192] proposes a decoder that jointly learns to score and select sentences, while [178] presents a latent variable extractive summarization model, which directly maximizes the likelihood of human summaries. In recent works, models based on pretrained models [92, 2, 179], especially BERT, have made a step forward.

Graph Based Summarization Graph-based summarization methods aim to utilize the structure information of the text. Based on the assumption that important sentences are often linked with each other, unsupervised models like TextRank [101] and LexRank [32] show great ability to identify the salient information. [90] proposes an abstractive summarization framework based on the Abstract Meaning Representation (AMR) graph, which captures the propositional semantic information. [65] presents a graph transformer to generate one-sentence summaries from a knowledge graph. Meanwhile, other researches focus on learning latent tree structures while optimizing summarization models. [153] regards the tree structure learning problem as a reinforcement learning problem and [93] generates a multi-root dependency tree where each root is a summary sentence.

Pretrained Language Models Pretrained language models [119, 27] have been proved to be extremely successful for making great progress in various nature language tasks. These models are originated from the idea of word embeddings [114] and further extend it by pretraining a sentence encoder on the huge unlabeled corpus using language modeling objectives. Bidirectional Encoder Representations from Transformers (BERT) [27], one of the state-of-art language models, is trained with a masked language model and a next-sentence-predicting task. Recently, pretrained language models have been widely used to improve performance in language understanding tasks [28]. As for the extractive summarization task, it provides the powerful sentence embeddings and the contextualized information among sentences [187], which have been proven to be critical to extractive summarization.

3.3 Extraction of Events and Alignment with Gold Summary

We propose to explicitly align a gold summary with its corresponding events descriptions in the source text. For this purpose, we develop a heuristic algorithm to break up the source text and the summary into smaller granularities as introduced below.

When splitting sentences into smaller semantic units, we need to ensure each unit has a proper size while maintaining the integrity of information. An intuitive idea might be to split sentences with commas and other conjunctions. But this straightforward strategy is not capable of handling complicated sentences in documents. Therefore, we leverage the dependency parser to handle this issue. To begin with, we adopt a dependency parser to convert a sentence into the labeled tuples in the form of (word1, word2, label), where the label denotes a grammatical relation between a pair of words. After that, the sentence is split using the labels representing punctuation marks, conjunctions and clauses, including punct (punctuation marks), cc (coordina-

tion relationship) and mark (finite clause). To acquire a more complete semantic unit, we merge units that are connected by some special labels such as the relative clause modifier (acl:relcl), the adverbial clause modifier (advcl), the appositional modifier (appos) and the clausal complement (ccomp). Furthermore, we use the conjunct (conj) to identify whether a conjunction connects two sentences or two phrases. A conjunct is a relation between two elements connected by a coordinating conjunction. When the distance between the two elements is less than a threshold, we regard the two coordinated elements are words or phrases rather than clauses. If so, the two split parts are merged back as one unit.

Moreover, we predefine a minimum unit length and a maximum clause length. If the length of a unit is smaller than the minimum unit length, this unit will be merged with the preceding unit. If a clause is longer than the maximum clause length, we regard it as an independent semantic unit. Take the dependency trees in Figure 3.1 as an example. The sentence is split into five parts based on the above-mentioned rules. We merge the first three parts back together since they are connected by the appos label while not exceeding the threshold length for being an independent clause. As for the last two parts, although there is a coordination relationship, it turns out that they are two coordinate entities based on the conj label. Hence, we also merge the last two parts as a event.

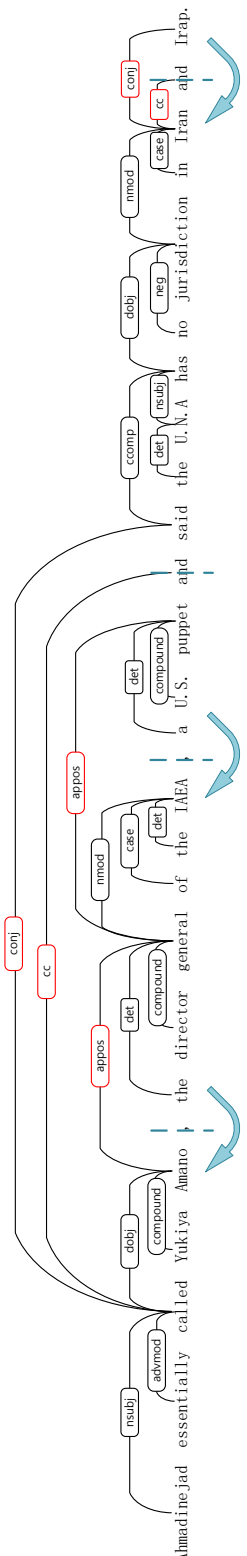

Figure 3.1: A dependency tree example. We extract the following two event descriptions: Ahmadinejad essentially called Yukiya Amano, the director general of the IAEA, a U.S. puppet ||| said the U.N.A has no jurisdiction in Iran and Irap

In the experiments, we adopt the well-known NLP pipeline Stanford CoreNLP [51] for dependency parsing. Table 3.1 presents the statistics on the train set of CNN/DaliyMail. On average, a sentence contains 1.6 events. It should be noted that some sentences remain as single events, while others are split into 2 or 3 events.

granularity	**num**	**len**
Sentence	34.3	24.7
event	51.5	14.8

Table 3.1: The average unit number and unit length in train set of CNN/DM dataset.

Once events are extracted, the next step is to obtain oracle labels for them from the gold summary. We match each event in the summary with one corresponding event that has the maximum ROUGE score from original text. Such an alignment allows each part of the summary to be accurately mapped to a semantic unit in the source text.

3.4 Summarization Model

3.4.1 Model Framework

As illustrated in Figure 3.2, our model consists of three components: a BERT-based encoder, a hierarchical graph mask, which incorporates the structure constraint on BERT and a classifier taking the multi-granularity information as input to extract salient events to form the summary.

3.4.2 BERT-Based Encoder

We use a BERT-based encoder to generate contextualized representations of the semantic units with different granularities. Since the outputs of BERT are grounded

to tokens, we adapt the strategy that is similar to [92] to modify the embeddings and the input sequence of BERT.

To obtain representations of the events, the sentences and the document, we add a special token at the begining of each semantic unit. As shown in Figure 3.2, [cls_d], [cls_s] and [cls_f] are inserted at the beginning of the document, the head of each sentence and the event representing the embeddings of different granularities, respectively. Initially, these three kinds of [cls] tokens share the same pre-trained [cls] embedding. We also add a [seq] token at the end of each event to separate the smallest semantic units. In addition, segment embeddings are used to identify multiple granularity levels within a document. For the ith granularity level, we assign the segment embedding E_A or E_B conditioned on whether i is odd or even. For example, in Figure 3.2, we first assign the [cls_d] segment embedding E_A and then for the next granularity level, sentence, we allocate all [cls_s] with E_B. With two different segment embeddings that separate adjacent granularity levels, the model is aware of the hierarchical structure among different granularity levels.

3.4.3 Hierarchical Graph Mask on BERT

A document is composed of a collection of sentences, while each sentence may contain multiple events. Such a hierarchical structure motivates us to generate representations for different granularity levels with hierarchical constrains on BERT. We propose a hierarchical graph mask to restrict information dissemination in BERT. To build a hierarchical graph, we add three kinds of edges to a token in the input sequence, including the bi-directional edges connecting to all other tokens at the same granularity level, the edges from the tokens belonging to them at the smaller granularity level and the edges to their corresponding tokens at the larger granularity level. For example, as shown in Figure 3.2, an event token [cls_f1] can disseminate its information to other event tokens and the sentence token [cls_s1], while receiving the information

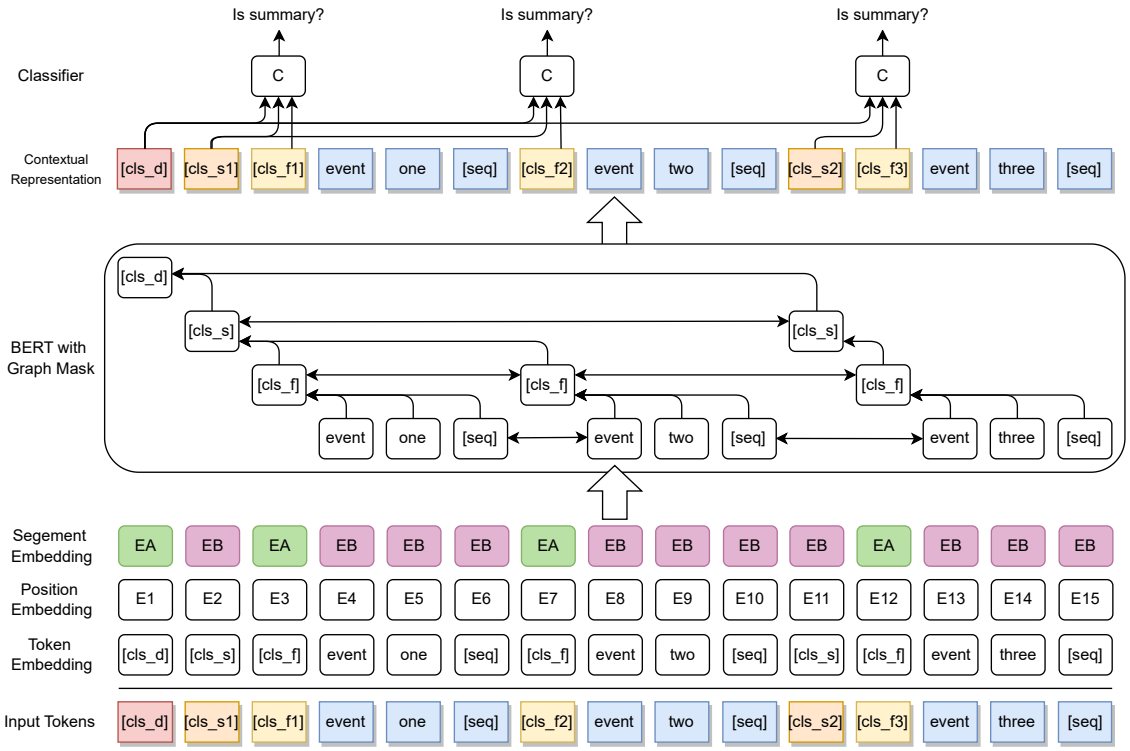

Figure 3.2: The framework of the summarization model

from the word tokens "event", "one" and [seq]. The tokens at each granularity level thus capture semantics from different sources. As a result, the hierarchical graph can effectively utilize all levels of structural information with different granularities.

To implement this graph structure on BERT, we assign a mask vector to each token based on the hierarchical graph H. Given an input sequence with n tokens $[t_1, t_2, ..., t_n]$, the mask vector of t_i is denoted by $[h_{i1}, h_{i2}, ..., h_{in}]$, where h_{ij} refers to whether there is a directed edge from the token j to the token i in the graph H. Taking [cls_f1] in Figure 3.2 as an example, the mask vector of this token is $[0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0]$. After obtaining the mask vector for each token, we stack them to form a $n \times n$ mask matrix M and calculate attentions with the equation below. For simplification, we write it in the one-head form.

$$Attention(Q, K, V) = Softmax(\frac{QK^T M}{\sqrt{d^k}})V$$

where Q refers to the query matrix, K refers to the key matrix, V refers to the value matrix and $\sqrt{d^k}$ represents a scaling factor.

3.4.4 Classifier

After applying the hierarchical graph mask on BERT from words to document, we obtain representations of all levels of granularities. Although our focus is to extract the salient events from a given document, we believe that the sentence-level information also contributes to identifying the importance among events. Considering events are rather independent semantic units, the sentence information is able to build up connections from a higher-level perspective and help locate the important events more accurately. Moreover, we postulate that the selected events should be semantically representative of the source document. Therefore, we also import the document representation as part of input to the classifier.

Given a event representation f_i from the last layer of BERT, we concatenate it with the document representation d and its corresponding sentence representation s_j. Then we employ a sigmoid classifier to predict whether or not it should be selected into the summary:

$$y_i^{'} = \sigma(W_c(d||s_j||f_i) + b_c)$$

where $||$ represents concatenation, W_c and b_c are the parameters of the classifier.

3.4.5 Learning

We apply a binary classification entropy of prediction $y_i^{'}$ against gold label y_i as the training loss. Similar to [92, 145], we use the Adam optimizer with $lr = 2e^{-5}$, $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and adopt a learning rate schedule with warming-up=10000:

$$lr = 2e^{-3} \cdot min(step^{-0.5}, step \cdot warmup^{-1.5})$$

For testing, we first predict the score y_i for each event using our model and then rank these events with their scores to select the top-4 events as the summary, considering the average number of events contained in gold summaries is four. We also apply the Trigram Blocking algorithm [113] to reduce redundancy. In particular, given the current summary s, a candidate event c will not be included in s if there is a trigram overlapping between c and s. In this way, we minimize the similarity between the current summary and the candidate sentence to increase information diversity and richness.

3.5 Experiments

3.5.1 Dataset and Evaluation Metric

We conduct experiments on the non-anonymized version of the CNN/DailyMail dataset [51]. The dataset is composed of news articles and their corresponding highlights that give brief overviews of articles. We apply the standard splits for training, validation, and testing (CNN: 90,266/1,220/1,093 and DailyMail: 196,961/12,148/10,397). Due to the length limitation of BERT, we follow the common practice to truncate all input documents to 512 tokens.

To evaluate the summarization quality, we apply ROUGE [87] for automatic evalua-

tion. Following the convention, we report ROUGE-1 (unigram), ROUGE-2 (bi-gram) and ROUGE-L (longest common subsequence) F1-score in the following experiments. Moreover, we manually evaluate the event-level recall, precision and F1-score between system-generally summaries and human-written gold summaries.

3.5.2 Details

We build our model using PyTorch and the BERT-base-uncased version of BERT. All the input documents are tokenized by BERT's sub-words tokenizer. We train the model for at most 50000 steps and the batch size in each step is 32. After training for 10000 steps, the model is saved and evaluated for every 1500 steps. With the three best checkpoints on the validation dataset, we record best model on the test dataset among the three.

3.5.3 Automatic Evaluation

We first examine the ROUGE results of the oracle summaries matched with different granularities. As shown in Table 3.2, by aligning events with gold summaries it achieves significant improvement on all three ROUGE metrics. It shows that our idea of building oracle summaries based on events is able to generate more accurate oracle labels for training.

Granularity	**R-1**	**R-2**	**R-L**
event	57.96	34.93	54.69
Sentence	52.59	31.24	48.87

Table 3.2: Results on oracle summary on CNN/DM test set.

Table 3.3 summarizes the results of a variety of models on the CNN/DailyMail test set. The first section in the table presents two lead-3 baselines that select the first

three sentences as summaries. One is our own implantation, while the other one is copied from [92]. The slight discrepancy on ROUGE scores may be because of the difference in data preprocessing.

The second section in the table displays performances of the existing extractive models, including the baseline model SummaRunner, two SOTA models with no pretraining and several models based on BERT. All the results are directly taken from their respective papers.

The last section reports the results of our own implementations. In order to avoid interference of other factors, we implement the SOTA model BERTSUM in our training environment. Meanwhile, for better comparison, we propose a event-extraction BERTSUM baseline, which simply changes the primary textual unit for extraction from sentences to events under the same architecture as BERTSUM. Compared to BERTSUM, using events as extraction units improves the performance on ROUGE-2 and ROUGE-L, since it provides more accurate alignment on the salient information and reduces the proportion of meaningless words in summaries. After imposing the structure information on BERT, we further develop the potential of event-level extractive summarization. All three variants of our model achieve improvement on ROUGE scores. Among the variants, the model with additional sentence-level information performs best. This is not surprising at all. In fact, sentence-level relationships contain the rich contextual information. Unexpectedly, we find that it does not lead to the best result by combining all three levels of information. We suspect that it may be due to that the document-level information is not effective enough for single document summarization.

3.5.4 Human Evaluation

As we know, although the ROUGE metric has long been regarded as a classical evaluation metric in summarization, it does not always reflect the true quality of

Model	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**
Lead-3 (our)	40.25	17.40	36.41
Lead-3 [92]	40.43	17.62	36.67
SummaRunner [104]	39.60	16.20	35.30
BANDITSUM [29]	41.50	18.70	37.60
NEUSUM [192]	41.59	19.01	37.98
HIBERT [179]	42.37	19.95	38.83
PNBERT+RL [187]	42.69	19.60	38.85
BERTEXT+RL [2]	42.76	19.87	39.11
BERTSUM [92]	43.23	20.22	39.60
BERTSUM (our)	42.78	19.79	39.03
BERTSUM+event	42.81	19.93	39.84
Our d+f	42.97	20.09	39.98
Our s+f	43.10	20.17	40.10
Our d+s+f	43.01	20.07	40.02

Table 3.3: Results on CNN/DM test set. BERTSUM+event refers to the BERTSUM baseline which simply changes the primary textual unit for extraction from sentences to events. The last three models represent three variants of our method. For example, d+f means that we use document-level information and event-level information for summarization

summaries. Hence, we also conduct human evaluation to check the event-level recall, precision and F1-score from 50 random samples. As shown in Table 3.4, compared with sentence extraction, precision of event extraction increases from 32% to 39%, which demonstrates the ability of our model to reduce the over-extraction problem. As for recall, we also gain some degree of improvement.

Model	**Recall**	**Precision**	**F1**
BERTSUM+event	41.79	38.78	39.52
BERTSUM	40.45	31.79	34.98

Table 3.4: Human evaluation results on CNN/DM test set.

3.5.5 Model Analysis

Position of Extracted Events: In addition to evaluating the models through ROUGE, we further look into the details of our model. As we know, in a document the sentences at the beginning tend to be more important and are much more likely to be included in summaries. That is the reason why lead-3 is such a strong baseline in the CNN/DailyMail dataset. It turns out that current models may rely too much on the positional information. We feel it is important to analyze the position of the selected units in the source text. To ensure fairness in comparison, we convert all sentences into their corresponding event units in this experiment. Table 3.5 shows the proportion of the selected events that appear in the source text at different event positions. 1-5 refers to the first five positions, 6-10 represents the positions 6 to 10, and "Rest" means the rest of the positions except 1-15. The results are obtained from the oracle summaries generated by sentence extraction, the oracle summaries generated by events, those generated by BERTSUM (truncated to four events) and our s+f. We find that the oracle summaries are distributed smoothly across documents , while the summaries generated by both models highly bias towards the beginning text. The difference is that our model shows a flatter curve on the first 10 events, indicating

that our model can achieve better diversity.

Model	**1-5**	**6-10**	**11-15**	**Rest**
Oracle s	33.11	23.15	11.79	31.95
Oracle f	29.34	21.81	11.89	36.96
BERTSUM	59.39	26.65	6.12	7.84
Our s+f	55.66	30.33	8.15	5.86

Table 3.5: Proportion of extracted events according to their position in the original text.

Ablation Study: Since the transformer-based model provides us an effective way to disentangle the information from different sources, we are allowed to design an experiment to investigate what roles these types of information play in our model. As shown in Table 3.6, with our s+f as a base model, we report the ROUGE score after removing segment embeddings or positional embeddings from it. We can observe that the performance of the model without segment embeddings decreases to the same level as BERTSUM+event, since it is challenging for a model to understand the hierarchical structure when there is no difference among the information from different sources. We believe that our segmentation strategy is essential for learning the structural information. As for positional embeddings, it is not a surprise that the performance drops by a large margin. According to [187], current extractive models heavily rely on positional information. The bottom half of Table 6 contains the performance of other models without positional information, including Transformer from [187] and BERTSUM from our experiments. Compared to these models, the hierarchical structure leads to notable improvement, demonstrating that our model has good potential to learn more semantic information rather than simply relying on the positional information.

Model	R-1	R-2	R-L
Original	43.10	20.17	40.10
- Segment	42.91	19.98	39.87
- Position	39.55	17.37	36.52
Transformer	37.90	15.69	34.31
BERTSUM	37.97	15.93	34.66

Table 3.6: Results for disentangling test on CNN/DM test set.

3.5.6 Case Study

Table 3.7 illustrates an example to show why events are more effective than sentences in extractive summarization. Here, we use a semicolon to separate different events in one sentence. The words in italics refer to the sentences selected by the greedy algorithm used in previous models and the colored text represents the events selected by event-level alignment, where each selected event corresponds to one event in a gold summary. Based on our observation, one sentence may contain several events but only one of them is included in summary. Meanwhile, although the first sentence selected by the greedy algorithm has a high ROUGE score against the gold summary, it does not correlate with any important events. On the contrary, our model proves its ability to align the gold summary to its correct location in the source text. The last part of the table contains a summary generated by our model. We can see that our model successfully extracts two gold events out of four. As a contrast, we also mark the sentences generated by BERTSUM using underscore. It is observed that BERTSUM tends to focus on several leading sentences but only one of them actually contains a gold event.

Document
Virtual reality may still seem like a hobby reserved for hardcore gamers; but as headsets drop in price it is on the verge of becoming mainstream. One firm helping to fuel this trend is immerse. *It has created a virtual reality headset that works with any android and ios phone; is compatible with hundreds of virtual reality apps from the respective stores.* The immerse virtual reality headset is available from firebox for 29. It works with any android and ios phone that can run virtual reality apps from the respective stores and play any 3d movie. The maximum size of compatible devices is 3; which means it will work with the iphone 6; not the iphone 6 plus , for example. ... *Immerse calls itself an affordable alternative to rivals such as oculus rift; which is expected to launch a consumer version soon with prices ranging from between 200 and 400. Immerse is available to buy from firebox and can be shipped internationally.*
Gold summary
1: The immerse virtual reality headset is available from firebox for 29.
2: It works with android and ios phones via virtual reality apps and 3d films.
3: The maximum size of the device must be 3.
4: It calls itself an affordable alternative to rivals such as oculus rift.
Our s+f
The immerse virtual reality headset is available from firebox for 29.
The maximum size of the device must be 3.
Is compatible with hundreds of virtual reality apps from the respective stores.
Virtual reality may still seem like a hobby reserved for hardcore gamers.

Table 3.7: Example from the CNN/DailMail test dataset

3.6 Conclusion

In this paper, we propose to extract event-level semantic units for extractive news summarization. Without increasing the scale of the model, we propose a hierarchical graph mask on BERT to fully utilize the structural information among different semantic levels. Experiments on the CNN/DaliyMail dataset shows that our model achieves state-of-the-art results.

Chapter 4

Event Graph based Sentence Fusion for Abstractive Summarization

4.1 Introduction

In Chapter 3, we discuss the extractive news summarization with the event-level semantic unit. However, extractive models can only select important information and fail to achieve a higher compression rate. Hence, a rewriting process for the extracted sentences is necessary. In this chapter, we investigate sentence fusion that aims to combine several related text spans into a single coherent text. In news summarization, it is a common practice for a proficient editor to fuse the information from several related sentences, however, it remains challenging for a state-of-the-art neural abstractive summarization model to achieve effective sentence fusion. As pointed out in [72], the human-written summaries contain 32% fusion sentences on the CNN/DailyMail dataset, while only 6% of the summary sentences generated by the Pointer-Generator model [130] are shown to fuse the information spread over sentences. Besides, without

Figure 4.1: Examples of two types of sentence fusion in text summarization.

proper guidance, many sentences generated by fusion contain factual errors, which is critical for news summarization. Therefore, it is worthwhile to explore effective sentence fusion methods in the context of news summarization.

In fact, the importance of sentence fusion has long been recognized by researchers in the text summarization community. As shown in Figure 4.1, the researchers have been concerned with two types of sentence fusion task in the past. One is *similar* sentence fusion and the other one is *disparate* sentence fusion. For similar sentence fusion, a word graph or a dependency tree is often explored to find a coherent fusion path [99, 39, 141]. For disparate sentence fusion, the coreference relations are typically considered as the key to tie the sentences together [73, 71]. Although both types of sentence fusion benefit text summarization, especially multi-document summarization, the solutions are rarely proposed to deal with the two types together. In this paper, we propose to apply the structured event information to guide the two types of sentence fusion in a unified framework.

We address the challenge of sentence fusion by building an *event graph* to capture the

semantic relationships among the input text spans (sentences or events). The event graph is a directed graph composed of the nodes representing the predicate and event arguments and the edges that connect these event components together. Compared to the word graph or the dependency tree, the event graph provides more informative event-level (or to say entity-level) information. Meanwhile, it maintains the semantic integrity of each node, which allows us to add additional edges to represent some crucial relationships in disparate sentence fusion like co-reference. Such a structured representation is capable of preserving inherent event information and meanwhile formulating cross-sentence information such as entity interactions and proximity of relevant concepts.

With the target to guide sentence fusion, we develop a decoder that utilizes the information from both the sentence sequence and the event graph equipped with different attention mechanisms. We employ sequence attention and graph attention to determine what information is important to be select to generate the appropriate word token at each decoding step. Note that sentence fusion requires not only selecting the right salient information but also organizing the selected information logically and orderly. Otherwise, the models may tend to randomly combine the key event components or simply copy the most important text span. To this end, we develop a graph flow attention to explore potential fusion paths via the graph structure and control the fusion process. Moreover, how to avoid factual errors in a fused sentence is also a critical issue in sentence fusion. Inspired by [128], we incorporate faithful beam search at the inference stage to reduce possible factual errors. This allows the model to remove the unfaithful candidate output sequence during the generation process by refining the generation probability with a faithful score.

Since there is no available dataset to evaluate the effectiveness of the sentence fusion models in the context of text summarization, following previous work [73], we automatically generate sentence fusion data from summarization datasets including CNN/DaliyMail [51] and Multi-News [34]. The experiments show that our proposed

model indeed improves Rouges and the other metrics like faithfulness and the fusion rate. The contribution of our work can be summarized as follows:

(1) We propose a model to address both similar sentence fusion and disparate sentence fusion, which are critical for abstractive news summarization.

(2) We build an event graph to guide sentence fusion, which allows our model to utilize the structural event information and various cross-sentence relations.

(3) We innovatively apply a graph flow attention to control the fusion process via the graph structure.

4.2 Related Work

Sentence Fusion in Text Summarization Sentence fusion has been considered as an essential step for generating abstractive summaries. Its importance has long been recognized in the traditional text summarization research [5]. The early attempts mainly focus on fusing a set of similar sentences [99, 39, 31, 141]. They often build a dependency graph or a word graph from multiple similar sentences, and then adopt linear programming to generate the fused sentence from the graph. Recently, [72] conducts a comprehensive analysis of sentence fusion in neural abstractive summarization and finds that it remains a challenge for current state-of-the-art models. To address this problem, [71, 73] propose to utilize points of correspondence between sentences to fuse disparate sentences, and develop a transformer enhanced with the links between the co-referred entities. Similar to above-mentioned works, our research also focuses on the research of sentence fusion in the context of text summarization.

Moving beyond sentence fusion alone, [100, 74] discusses the potential application scenarios for enhancing text summarization with sentence fusion. Their models follow a similar framework that first extracts a few related sentences from the source document and then fuses them to obtain a summary sentence. Our model can be

considered as a better replacement of the fusion model in such a framework.

Event-aware Generation Model Currently, in the conditional generation tasks like text summarization and question answering, most of the source documents are usually composed of a series of events. Understanding how to leverage event information in these generation models becomes crucial. [103] learns to generate a fluent sentence with an input subject-verb-object triple that describes an event. [55] transfers event triples extracted with OpenIE to an event graph to acquire semantic interpretation over input to assist text summarization. [185] adopts an event graph to understand the path of multi-hop reasoning in question answering. To control the generation process and avoid factual errors, [9] proposes an additional event relation encoder to produce representations of event triples. Considering the importance of the relations between events in sentence fusion and inspired by the above-mentioned works, we adopt the event graph to guide sentence fusion.

4.3 Method

Our sentence fusion model follows the typical encoder-decoder architecture, as shown in Figure 4.2. It is composed of a joint encoder that produces both source sentences and event graph representations, and a decoder that incorporates the information from the source sentences and the event graph to generate a fused sentence.

4.3.1 Event Graph Construction

The event graph is built to capture the semantic relationships in the source sentences. We utilize AllenNLP-OpenIE [134] to extract the event components, where each event is composed of a predicate and an arbitrary number of arguments. When there is an overlap between two events, only the longer one is retained. These predicates

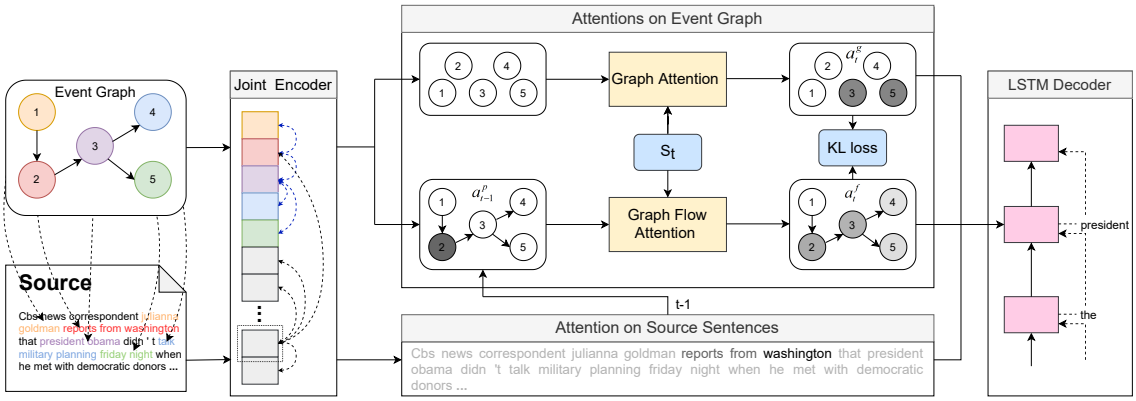

Figure 4.2: The framework of our proposed sentence fusion model. The various colors in the left refer to nodes and corresponding event components, while dotted lines represent how information disseminates in the BERT attention layer. In the middle part, different gray scales stand for different levels of attention on the tokens or nodes.

and arguments are represented as the nodes in the event graph. When two nodes share the same content, we merge them into one. The graph is a directed graph. Two types of edges are considered. (1) Directional edges connect a predicate and its corresponding arguments in an event and the direction follows the order of subject to predicate and predicate to other arguments. (2) Bi-directional edges connect two nodes if they share the same entity or there is a coreference relation between them.

4.3.2 Encoder

We apply a BERT-based encoder to jointly generate contextualized representations of the tokens in concatenated input sentences and the nodes in the event graph. Each node is represented by a special $[cls]$ token and the output representation of this token is considered as the representation of the node. The input of our encoder is the concatenation of sentence tokens and a set of graph node tokens. Since each node

only corresponds to several words in the input sentences, one node token will only be attended by the sentence tokens that belong to this node in the attention layer of BERT. To distinguish the two kinds of tokens, we assign two different segment embeddings to sentence tokens and node tokens. Since there is no sequential relationship between nodes, we initialize the positional embedding for node tokens as a special pad embedding.

We use an additional mask matrix M similar to the one presented in [171] to control the attention of the BERT-based encoder. $M_{ij} = 0$ means token i is allowed to attend to j, while $M_{ij} = -\infty$ prohibits i from attending to j. In our model, three possible situations can happen: (1) a sentence token attends to all other sentence tokens; (2) a sentence token attends to its corresponding graph node token; (3) a node token attends to other adjacent nodes on the event graph. After defining the mask matrix M, we calculate attention with Equation (1) below, where Q, K and V refer to the query matrix, the key matrix and the value matrix, respectively, d^k is a scaling factor.

$$Attention(Q, K, V) = softmax(\frac{QK^T + M}{\sqrt{d^k}})V \tag{4.1}$$

In our preliminary study, we have also considered using the graph neural network as the encoder for the event graph, but we find that the current approach achieves a better result.

4.3.3 Decoder

Overview of the Decoder. The decoder aims to generate the fused sentence utilizing both the (sentences) sequence information and the (event) graph information. We employ a one-layer LSTM as the decoder with the hidden state s_t at step t. The decoder generates tokens recurrently based on three types of attentions, i.e., the sequence attention, the graph attention and the graph flow attention.

Sequence Attention. At each decoding step t, we calculate the context vector c_t^s over a sequence of input sentences using the attention mechanism proposed in [3]. We also employ a coverage mechanism to avoid redundancy.

$$c_t^s = \sum_k a_{t,k}^s h_k \tag{4.2}$$

$$a_{t,k}^s = softmax(W_k tanh(W_1 s_t + W_2 h_k + W_3 Cov)) \tag{4.3}$$

where h_k represents the token representation obtained from the encoder, Cov refers to the coverage vector generated at the last step.

Graph Attention. The graph attention applies the mechanism analogous with the sequence attention but to the node embedding v_i and current hidden state s_t to compute the attention score. The graph vector c_t^g is computed over the node embeddings with attentions.

$$c_t^g = \sum_i a_{t,i}^g v_i \tag{4.4}$$

$$a_{t,i}^g = softmax(W_v tanh(W_4 s_t + W_5 v_i)) \tag{4.5}$$

Graph Flow Attention. When the graph structure is ignored during the decoding process, the graph attention tends to reflect the importance of individual nodes rather than the connections between nodes. We thereby propose a novel graph flow attention to explore potential fusion paths by capturing the content coherence embedded in the graph structure. The graph flow attention is designed to inherit the attention tendency of nodes from the previous decoding step and focuses on neighboring nodes at the current step.

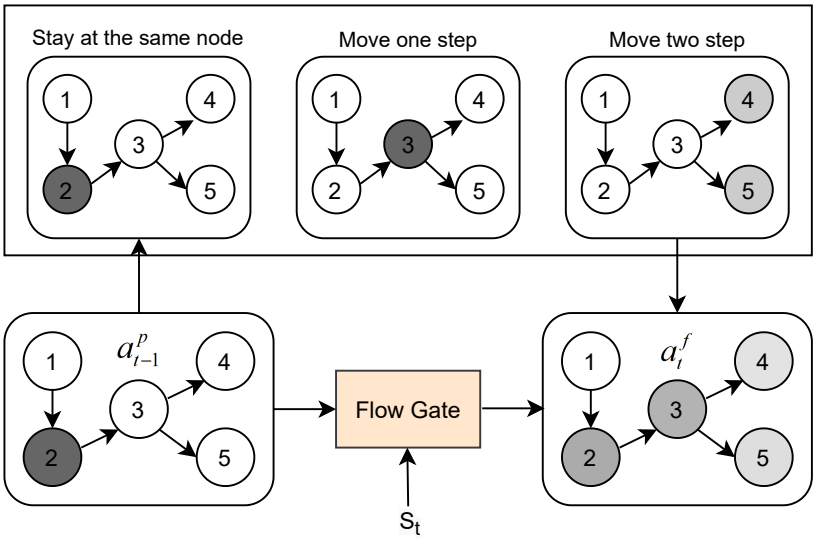

Figure 4.3: Calculation process of graph flow attention.

The attention tendency of nodes is expected to be strongly correlated to the output of the decoder. In this way, the model can maintain the coherence between the generated tokens and the nodes focused by the graph flow attention. Considering the graph attention is not fully synchronized with the decoding process, the following situation may happen. It first focuses on one node, and then teleport to another one far from the current node across the two consecutive decoding steps. Therefore, we choose to compute the distribution of attention tendency of nodes in the last step a^p_{t-1} based on the sequence attention in the last decoding step. Suppose $Map \in i \times j$ is the mapping matrix between tokens and nodes, where $Map_{ij} = 1$ denotes that the i token in the source sequence is in the j node of the event graph. The a^p_{t-1} is then calculated based on the following equation.

$$a^p_{t-1} = softmax(Map^T a^s_{t-1}) \tag{4.6}$$

Given the adjacent matrix A of the event graph, the i row refers to the normalized in-degree of the node i. As shown in Figure 4.3, the graph flow attention transmits

a^p_{t-1} in the following three ways:

(1) Remain in the previous node $f_{t,0} = a^p_{t-1}$. Since one node usually contains multiple tokens, the model may focus on the same node in several steps.

(2) Move one step $f_{t,1} = Aa^p_{t-1}$. For example, the attention moves from one node to its neighbor.

(3) Move two steps $f_{t,2} = A^2a^p_{t-1}$. The attention is allowed to skip a middle connection node.

The graph flow attention is then the weighted sum of the scores of the three flows controlled by a dynamic gate $Gate_t \in 1 \times 3$. And the graph flow vector c^f_t is computed by the following equation.

$$c^f_t = \sum_i a^f_{t,i} v_i \tag{4.7}$$

$$a^f_t = \sum_{h=0}^{2} f_{t,h} Gate_{t,h} \tag{4.8}$$

$$Gate_t = softmax(W_f tanh(W_6 s_t + W_7 \sum_i a^p_{t-1,i} v_i)) \tag{4.9}$$

Token Prediction. After obtaining the three vectors from the input sequence and the graph, we regard them as the representations of the information summarized from different points of view. Then they are concatenated with the decoder hidden state s_t to produce the vocabulary distribution D_{vocab} as follows.

$$D_{vocab} = softmax(W_{out}[s_t; c^s_t; c^g_t; c^f_t]) \tag{4.10}$$

We add a copy mechanism to directly copy words from source text based on the sequence attention. The copy probability is:

$$p_{copy} = sigmoid(W_{copy}[y_{t-1}; s_t; c^s_t; c^g_t; c^f_t]) \tag{4.11}$$

where y_{t-1} denotes the embedding of the token predicted at step $t-1$.

4.3.4 Training

Generation Loss. With the generation loss, the training goal is to maximize the estimated probability of the reference sequence. Following most current works, we adopt the maximum likelihood training objective function that minimizes the following loss.

$$L_{seq} = -\frac{1}{|D|} \sum_{(x,y,g)\in D} logp(y|x,g;\theta) \tag{4.12}$$

where θ represents model parameters and D stands for the training data including source sentences x, reference sequence y, and event graph g.

KL Loss. Our preliminary study reveals that simply concatenating the graph vector and graph flow vector in the decoding process fails to achieve a good performance. We figure out that it is difficult for a model to obtain effective information from two disparate vectors. Therefore, we introduce another training objective that computes the KL loss between the graph attention and the graph flow attention. In this way, the two attentions take advantage of each other. The KL loss is shown below and T is the total number of decoding steps.

$$L_{kl} = -\frac{1}{|D|T} \sum_{D} \sum_{t\in T} KL(a_t^g || a_t^f) \tag{4.13}$$

Node Salience Labeling. We further enhance the node representation via the third objective that models the salience of nodes. The goal of it is to identify whether the non-stop words in a node are mentioned in the reference fused sentence. We incorporate a classification layer over each node v_i above the joint encoder to predict a probability m_i ranged in [0,1]. During training, the gold label n_i is set to 1 if the node contains at least one non-stop word in the reference, and 0 otherwise. The loss function is shown below.

$$L_{node} = -\frac{1}{N_v} \sum_{i} (n_i log(m_i) + (1-n_i) log(1-m_i)) \tag{4.14}$$

where N_v is the number of the nodes in the graph. To summarize, the full training objective function consists of three terms: $L = L_{seq} + L_{kl} + L_{node}$.

4.3.5 Faithful Beam Search

Inspired by [128], we propose faithful beam search to reduce possible factual errors at the inference stage. Given a factual consistency checking model F and a sentence fusion model G, the goal is to re-rank every generated token based on both the generation probability calculated by G and the faithful score derived from F. In our work, we adopt the FactCC model developed by [69], a BERT-based faithfulness checking model, to evaluate faithfulness. The input to FactCC consists of a hypothesis sentence and several source sentences, while the output from FactCC is a probability that refers to whether the hypothesis sentence is faithful to the source sentences. Since what we need here is to verify the faithfulness of an incomplete fused sentence during the decoding process, we made a corresponding change when training FactCC with sentence fusion data. We truncate all the fused sentences in positive samples to random length. For the negative samples, we remove the tokens after the position of the error in fused sentences. At the inference stage, the objective function aims to maximize the cumulative probability of the output tokens. At each decoding step, the top-b sequence with the highest probability is carried into the next step, where b stands for the beam size. We add an additional faithful score to refine the generation probability during beam search, such that:

$$S(y_t) = S(y_{t-1}) + \alpha log F(x, y) + log G(x, y_{1:t-1}) \tag{4.15}$$

where y refers to the generated sequence, x represents the source sentences and α is a weighting factor. F and G stand for the consistency checking model and the sentence fusion model respectively. In the experiments, the α is set to 0.05.

CNN/DaliyMail Fusion	**Train**	**Validate**	**Test**
Number	107347	5948	5100
Source length	53.8	53.5	53.2
Target length	16.3	16.3	16.4
Multi-News Fusion	**Train**	**Validate**	**Test**
Number	19984	2496	2512
Multi / Single	9402/10582	1184/1312	1124/1388
Source length	72.5	71.5	72.4
Target length	28.5	28.6	28.6

Table 4.1: Statistic of CNN/DaliyMail Fusion dataset and Multi-News Fusion dataset. Multi/Single indicates whether the source sentences are from multiple documents or a single document.

4.4 Experiments

4.4.1 Experimental Set-Up

Datasets: We follow the practice of [74] to sample the sentence fusion data from summarization datasets. We choose the well-known single-document summarization dataset CNN/DaliyMail and multi-document summarization dataset Multi-News for the purpose of evaluation. With the CNN/DaliyMail dataset, the fusion data is directly obtained according to the set of heuristics suggested in [71], which we call CNN/DaliyMail Fusion. With the Multi-News dataset, we use a strategy similar to the one proposed in [71] to generate the fusion data, which we call Multi-News Fusion. Note that there is a 60-70% compression rate on both sentence fusion datasets. Hence, they are different from the one proposed by [44] where the compression rate is lower than 5%. This explains why we create the sentence fusion data generated from summarization datasets rather than using the existing one.

Evaluation Metrics: Sentence fusion can be approximately regarded as multi-sentence summarization. Following the common practice, we adopt ROUGE F_1 as the basic evaluation metric. We also apply FactCC [69] to evaluate faithfulness (Fai) automatically. FactCC is trained on the CNN/DaliyMail Fusion dataset and the Multi-News Fusion dataset following the method presented in the original paper. It achieves 90% of accuracy on the test set of two sentence fusion datasets and we believe that it is reasonably good for our evaluation. Note that it is distinct from the one used in our faithful beam search, where the fused sentences are not modified in the training. Besides, we also report the results of another two metrics, including (1) fusion rate (Fus), which is the percentage of the fused sentence that contain at least two unique non-stop words from multiple source sentences; and (2) length (Len), which is the average length of the fused sentences.

Implementation Details: We build the encoder using the BERT-base-uncased version of BERT. We employ the LSTM models with 768-dimensional hidden states as the decoder. We truncate the input sentences to 150 tokens and limit the decoder to a maximum of 60 steps. The batch size is set to 32 and we train the model for 20 epochs. After training, we select top-3 checkpoints on the validation dataset, and report the one with the best record on the test set among the three. For inference, the beam size is set to 5 in CNN/DaliyMail Fusion and 2 in Multi-News Fusion.

4.4.2 Automatic Evaluation

To examine the effectiveness of our model, we compare our model with two widely adopted seq2seq baseline models. **Pointer-Generator** [130] and **BERT+LSTM** are our basic encoder-decoder architecture before integrating the graph information. We also implement the state-of-the-art sentence fusion model for comparisons. **Tranformer-Linking** [71] is a BERT based model proposed for disparate sentence fusion. It utilizes coreference relationships between entities to enhance sentence fu-

sion. Since our data can be approximately regarded as multi-sentence summarization, we also adopt BERT based document summarization model, **BERTSUMABS** [92], for comparisons. Most of these models are trained on the two sentence fusion datasets by ourselves except that the output result of Transformer-Linking is directly obtained from its author.

As shown in Table 4.2, our proposed model obtains the highest Rouge scores on the Multi-News Fusion dataset and the competitive Rouge scores on the CNN/DaliyMail Fusion dataset. Meanwhile, our model achieves the best performance in fusion rate and faithfulness on both datasets. These suggest the effectiveness of our model in fusing sentences and its ability to reduce factual errors. We also notice that the transformer decoder has a clear advantage over the LSTM decoder in fusion rate. One possible reason is that the transformer decoder can generate a more abstractive sentence, which makes fusion a lot easier. Considering our model adopt a LSTM based decoder, we believe the event graph effectively assists the fusion process by providing cross-event connections and reduce the shifting distance between event components.

4.4.3 Ablation Study

To look into more detail, we design an experiment to understand how different components contribute to our model. We remove the KL loss, the graph attention and the graph flow attention independently from the full model and report the results in Table 4.3. On the one hand, we find that the graph flow attention boosts the fusion rate. We believe that the flow attention indeed benefits the fusion process when utilizing the graph structure to find possible fusion paths. On the other hand, the graph attention leads to relatively high Rouge scores but a lower fusion rate. This suggests that although the graph attention does not contribute to sentence fusion, it assists to select important information from source sentences. More importantly, when the KL loss is taken out, the model performance drops more compared to the other two

CNN/DaliyMail Fusion	Rouge-1	Rouge-2	Rouge-L	%Fai	%Fus	#Len
Concat-Baseline	37.29	20.06	28.77	100	100	53.07
Random-Baseline	36.25	17.64	30.72	100	-	26.10
Pointer-Generator	33.37	16.29	29.51	80.13	31.37	13.79
BERT+LSTM	37.56	19.50	33.59	88.77	45.66	16.65
BERTSUMABS	37.96	19.32	33.36	86.17	60.24	16.34
Transformer-linking	**39.79**	**21.08**	**35.35**	90.68	59.42	15.78
Our Model	39.30	21.03	35.12	**91.56**	**61.30**	15.12
Multi-News Fusion	Rouge-1	Rouge-2	Rouge-L	%Fai	%Fus	#Len
Concat-Baseline	48.63	32.95	36.56	100	100	71.28
Random-Baseline	44.60	27.04	37.16	100	-	31.48
Pointer-Generator	49.01	31.57	40.65	81.45	44.39	29.28
BERT+LSTM	50.93	33.99	43.00	85.84	48.30	26.16
BERTSUMABS	51.85	31.60	44.62	78.32	56.48	26.32
Our Model	**53.06**	**36.02**	**45.40**	**89.31**	**59.82**	25.36

Table 4.2: Automatic evaluation with Rouge, faithfulness(Fai), fusion rate(Fus), and generated sentence length (Len) on CNN/DaliyMail Fusion dataset and Multi-News Fusion Dataset.

Model	R-1	R-2	R-L	%Fai	%Fus
Our Model	**53.06**	**36.02**	**45.40**	**89.31**	**59.82**
- KL loss	52.63	35.63	45.42	87.10	55.52
- Flow Attention	52.71	35.97	45.37	88.79	51.42
- Graph Attention	52.81	35.94	45.22	86.62	56.13

Table 4.3: The results of ablation study on Multi-News Fusion test set.

reductions. It indicates that the KL loss is essential for our model to take advantage of both attentions.

4.4.4 Human Evaluation

Automatic evaluation results are often not enough to fully reflect the quality of the generated fused sentence. We further conduct human evaluation to analyze unfaithful errors and fusion quality. We randomly extract 50 samples from the Multi-News Fusion test set and invite three fluent English speakers as human judges. Given a sentence fusion instance, the judges are asked to answer yes or no to the following three questions. (1) Fluency: whether the generated sentence is grammatically correct and readable. (2) Fusion: whether the generated sentence is generated through sentence fusion. (3) Faithful: whether the generated sentence is faithful to the source sentences. Table 4.5 shows the percentage of yes on the three questions. We adopt Fleiss' kappa [40] to conduct the inter-annotator agreement test and the result is 0.53. The result shows a similar trend to the automatic evaluation, where our model achieves the best result in both fusion rate and faithfulness. The performance of BERTSUMABS further indicates that sentence fusion will lead to the decline of fluency and more faithful errors if there is no proper guidance.

We illustrate a sentence fusion example that contains both similar and disparate sentence fusion in Table 4.4. As shown, BERT+LSTM tends to fuse sentences by directly

Source: (1) Police identified the rite aid shooter as Snochia Moseley, 26, who lived in the marsh neighborhood of Baltimore. (2) The shooter was found with a self-inflicted gunshot wound and died at an area hospital. (3) The woman died at a nearby hospital after shooting herself in the head.
BERT+LSTM: Police say the shooter as Snochia Moseley, 26, was found with a self-inflicted gunshot wound and died at an area hospital.
BERTSUMABS: The woman, who died at a hospital, was found with a self-inflicted gunshot wound and died at an area hospital.
Our: Snochia Moseley was found with a self-inflicted gunshot wound and died at a nearby hospital after shooting herself in the head.
Reference: Police say the 26-year-old woman, who has not been identified, died of a self-inflicted gunshot wound to the head.

Table 4.4: Examples from the Multi-News Fusion test dataset. The different colors represent equivalent content between the source and the generated sentence.

Model	%Fluency	%Fusion	%Faithful
BERT+LSTM	85.3	51.3	56.7
BERTSUMABS	81.3	56.7	40.0
Our Model	**88.7**	**58**	**58.6**

Table 4.5: The results of the human evaluation on Multi-News Fusion test set.

	Rouge-1	Rouge-2	Rouge-L
Oracle	51.67	29.12	48.06
Oracle_all	48.92	26.43	45.42
Fusion	**52.14**	**29.19**	**48.62**

Table 4.6: The result of sentence fusion application on CNN/DaliyMail test set.

copying the text spans from the source text. BERTSUMABS attempts to utilize the coreference relations between "the shooter" and "the woman" to fuse the last two source sentences, but generates redundancy when merging similar content. On the contrary, our model successfully fuses the information from all source sentences. It shows that our model can effectively handle both types of sentence fusion at the same time.

4.4.5 Application in Text Summarization

We further design an experiment to investigate the effectiveness of the sentence fusion model in text summarization using a framework from [74]. It aims to extract a single sentence (no need for fusion) or a pair of sentences (need fusion), then rewriting them to produce a summary sentence. Each sentence pair consists of a primary sentence and a secondary sentence provides complementary information. We use the oracle extractive results as input to conduct the generation experiment. Table 4.6 shows the summarization results with three different strategies: (1) Oracle: concatenating oracle single sentences and primary sentences in oracle pairs as the summary; (2) Oracle_all: concatenating oracle single sentences and both sentences in oracle pairs as the summary; (3) Fusion: concatenating oracle single sentences and fused sentences as the summary, where the fused sentences are generated by our model using oracle pairs as input. All the summaries are truncated to 100 words. The result shows that the sentence fusion model has the potential to improve the performance of summarization

models by fusing information from multiple sentences.

4.5 Conclusion

In this paper, we investigate the sentence fusion problem in the context of abstractive news summarization by exploring the event graph. Our model captures both node representations and the structural information embodied in the event graph to guide the fusion. We further propose a faithful beam search to reduce the possible faithful errors. The experiment results suggest that event graph is crucial for effective sentence fusion and both node representations and graph structure play important roles in sentence fusion. In the future, we would like to further explore the direct incorporation of event information and the sentence fusion model to text summarization.

Part II

Long-Input Summarization via Context-Aware Extract-Generate Framework

Chapter 5

Preserve Context Information for Extract-Generate Long-Input Summarization Framework

5.1 Introduction

Long-input summarization is a process where a lengthy piece of text, such as a report, science article, or meeting record, is condensed into a shorter version, highlighting the key points and main ideas. Although general approaches based on pretrained language models achieve great success in the domains like new summarization, they are struggling with long-input summarization due to the high memory complexity of full self-attention. In this Chapter, we extend the previous extractive and abstractive approaches to an extract-generate summarization framework to address this problem.

Instead of focusing equally on the whole source document like other long-input summarization models including sparse attention transformers and hierarchical models, the extract-generate summarization framework is based on the assumption that only a part of salient source document is useful for the summarization. This not only

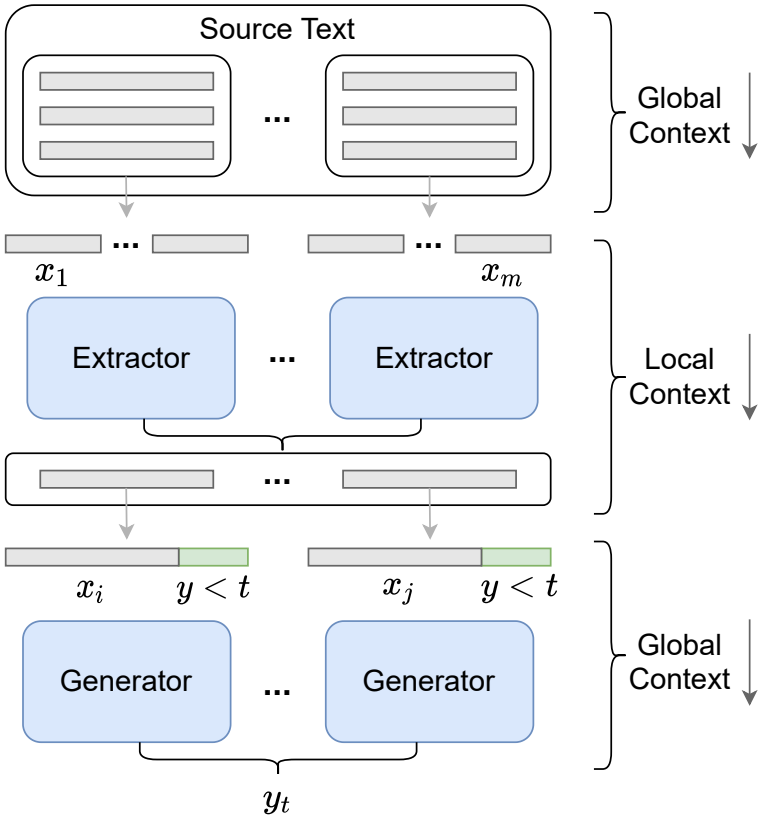

Figure 5.1: The extract-generate framework for long-input summarization. The grey strips stands for text snippets such as sentences or utterances from the source document. This figure shows that in the extract generate summary framework, both local and global context information is lost.

makes the framework follow the human intuition when dealing with long input-summarization, but also allows it to handle the summarization input at any length. Hence, the extract-generate summarization framework achieves a success in long-input text summarization [117, 183, 4, 180, 98]. However, as shown in Figure 5.1, the cost of its effectiveness in handling long-input summarization is the loss of context information. One the one hand, there exist a gap between the extractor and generator, where the context information can not be transferred from the continuous representation in extractor to the dispersed snippets in input of the generator. On the other

hand, the framework leverages a chunking strategy in both extractor and generator to process a longer input, which obstructs the access to global context information between the chunks. Some previous works [159, 22] has noticed this problem and propose context-aware extractive models to capture the global context information in extractor. Unfortunately, they do not solve this problem in the perspective of the whole framework and the context information still can not be effectively transferred from extractor to generator.

In this paper, we aim to investigate the influence of the context information and propose a context-aware extract-generate framework (CAEG) for long-input summarization. It aims to preserve both local and global context information from the extractor to the generator and utilize them to enhance the generation process. To build a bridge between the extractor and generator, CAEG generates a set of context-related text spans called context prompts for each text snippets and use them to transfer the context information. To generate such context prompts, we propose to capture the context information through the interpretation of the extractor. Considering that context information is one of the decisive factors of the extractive summarization, we assume that the text spans having the highest contribution to the extraction decision is considered as containing the richest context information. Here, we adopt a attention-based interpretation approach called attention rollout [1] to generate the context prompts. Then we add the context prompts to the generation process through simple concatenation. In terms of local context information, we concatenate the extracted snippets with their corresponding context prompts. In terms of global context information, each extracted snippet are concatenated with context prompts from its most related snippet to capture the global dependency. In this case, CAEG can be easily applied to most of existing summarization models based on extract-generate framework without largely increasing the complexity or memory cost of the model.

We conducted experiments on two long-input summarization datasets: arXiv [19] for long-document summarization, and QMSum [189] for long-dialogue summarization.

Taking a recent proposed extract-generate summarization model DYLE [98] as the backbone, our approach achieves improvement on both datasets compared to the base model and obtains the state-of-the-art result on QMSum. These experiments suggest the effectiveness of CAEG in preserving the context information. We conclude our contributions as follows:

- We firstly investigate the influence of context information loss for extract-generate framework in long-input summarization and propose the CAEG for this problem.
- We introduce a new approach for capturing the context information based on the interpretation of the extractor.
- The experiment result shows that CAEG is capable of effectively preserving the context information from the extractor to the generator without largely increasing the complexity or memory cost of the model.

5.2 Related Work

Extract-Generate Text Summarization Coarse-to-fine frameworks containing multiple stages are used in many text generation tasks, such as text summarization, dialogue state tracking, and neural storyline generation. In text summarization, the two-stage extract-generate framework is commonly used. This framework first extracts important text snippets from the input, followed by generating an overall summary. Some researchers [100, 74] propose to extract a set of similar sentence clusters and then fuse each cluster to obtain a summary sentence. More researchers [117, 183, 4, 180] apply extract-generate framework to the long-input summarization to overcome the capability limitation of transformer models such as BERT or PEGASUS. DYLE [98] further extends the idea and presents a dynamic latent extraction approach that generate dynamic snippet-level attention weights during de-

coding. However, these models seldom focus on the loss of the context information in the extract-generate framework. Hence, we aim to enhance these existing extract-generate summarization models with the context information using a cost-efficient approach in this paper.

Long-Input Summarization Long-input summarization has attracted more attention recently in different domains such as news, science paper, dialogue and etc. One solution is to adopt transformers with the sparse attention mechanism. Longformer [6] combines sliding windows with global attention patterns, while BigBird [173] uses sliding windows and random blocks. Reformer [64] adopts the locality-sensitive hashing to replace the dot-product attention. Despite focusing on self-attention, some researchers [54] proposes head-wise positional strides to improve the efficiency of the cross-attention. Some hierarchical models also have been proposed for long-input summarization. HAT-Bart [123] presents a transformer with hierarchical attention that utilizes information from both sentence and paragraph-level. HMNet [194] proposes a hierarchical structure that captures discourse-level information and speaker roles for dialogue summarization. As mentioned above, the extract-generate summarization framework also becomes a major solution for handling longer input. Compared with the other two types of models, we believe models based on this framework achieve a good balance between performance and computational cost.

5.3 Method

The extract-generate framework has been widely used in long-input summarization. It is usually composed of two transformer-based models, an extractor that extracts salient snippets from the source document and a generator that compress the extracted snippets into summaries. In this paper, we aim to enhance such framework by preserving previously lost context information. An overview of our proposed ap-

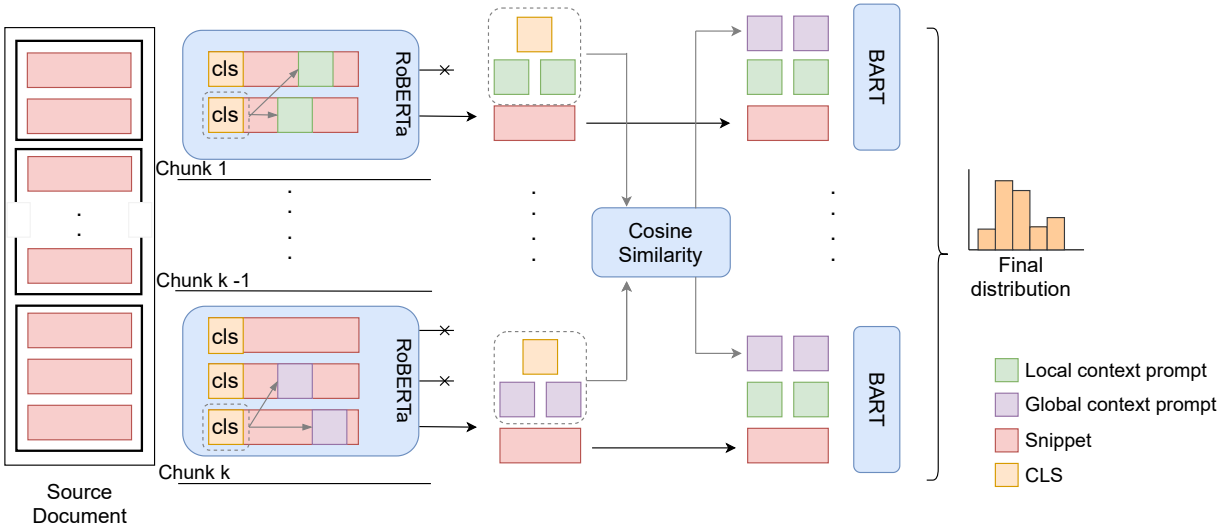

Figure 5.2: The framework of our proposed CAEG framework. The extractor we used is RoBERTa-base, and the generator is BART-large. Here, we use the blocks with various colors to represent the different types of information in the model and adopt the dot lines to emphasize the information flow added for preserving the context information. To obtain a clear observation, we simplified the structure of the generator in the figure.

proach CAEG is shown in Figure 5.2. In the first subsection, we briefly introduce the extractor-generator framework for long-input summarization. In the second subsection, we introduce how we extract the context prompts based on the interpretation of the extractor and use it to transfer the local context information. The context prompt can also be used on preserving the global context information, which we elaborate on in the third subsection. The application of CAEG is shown in the last subsection.

5.3.1 Extract-Generate Framework

In an extract-generate summarization framework, the input is composed of m text snippets, $X = (x_1, ..., x_m)$, and an optional query q if it is a query-focused summa-

rization task. The output is a summary y containing T tokens. In terms of document summarization, we take each sentence as a snippet. In terms of dialogue summarization, dialogue utterances are regarded as snippets. The framework aims to generate a sequence of summary tokens y given the input text X and the generated tokens in previous steps $y < t$ with an extractor E_η and a generator G_θ:

$$P_\theta(y \mid q, X) = \prod_{t=1}^{T} P_{\theta+\eta}\left(y_t \mid q, X, y_{<t}\right) \tag{5.1}$$

The goal for extractor is to output a score s_i for each text snippet x_i taking the source text and the optional query as input. However, limited by GPU memory, it is impractical to encode the full source text within in one language model. Hence, the text snippets are divided into multiple chunks $X = (c_1, ..., c_u)$, each containing consecutive snippets. We feed each chunk and the optional query to the extractor and compute score for each snippet in the chunk. Then top-N snippets X_N with the highest scores are extracted from the document X:

$$X_K = \text{top}N(E_\eta(q, x_i, c_j), x_i \in c_j, c_j \in X) \tag{5.2}$$

Note that even the extracted snippets in long-input summarization may still exceed the capability of a generation model. Some researchers adopt a strategy called fusion in decoder. It allows the generator first encode each extracted snippet independently and then concatenate the hidden states of all snippets as the input of the decoder. DYLE [98] proposes to predict the generation probability and the dynamic weight on each snippet and obtains the final generation probability by marginalizing over all extracted snippets.

In this paper, we adopt the same generator proposed by DYLE. It feeds each extracted snippet to the model and obtains the hidden state h_i^t and generation probability on every decoding time step. An additional MLP is used to map the hidden state to a

scalar weight. The dynamic weight allows the generator to focus on different snippets at different time steps. The final generation probability at time step t is computed by:

$$y_t = \sum_{x_i \in X_N} G_\theta \left(q, x_i, y_{<t}\right) MLP(h_i^t) \tag{5.3}$$

5.3.2 Local Context Information

Local context refers to the neighboring snippets of a target snippet within one chunk. Such information is fully utilized by the extractor to extract the salient snippets. However, due to the extracted snippets are transformed back to discrete representation after the extraction, the local context information can not be transferred to the generator. For example, we assume there is a snippet x_i mentioned entity A and entity B and its local context mentioned entity A multiple times. An extractor can easily extract the snippet x_i, since entity A is considered salient information. When snippet x_i is fed into the generator, the model can no longer identify whether entity A or entity B is the crucial one. In this case, the loss of local context information creates negative effects on the summarization.

The problem lies ahead is how to obtain the local context information from the extractor and apply it to the generator. An intuitive solution is to extend each extracted snippet by concatenating its neighboring snippets with it, which inevitably brings noise and efficiency drop. Another solution is to transfer the snippet representations in the extractor to the generator. However, these representations may not be effective, since the extractor and generator are usually independent models. A wildly accepted idea is that extractive summarization mainly depends on the context information. Inspired by this, we make an assumption that the text spans that contribute the most to the snippet extraction contain the richest context information. We named these text spans context prompts. In CAEG, we generate these context

Figure 5.3: The generation of context prompts. The yellow squares are the [cls] tokens and the grey squares denote the text tokens. The darker red stands for a higher attention rollout score.

prompts through the interpretation of the extractor, and use them to represent the local context information.

A commonly used interpretation approach for a transformer-based model is the attention distribution. However, information originating from the input tokens gets increasingly mixed across layers of the Transformer, which makes attention weights unreliable as explanations. Hence, we adopt attention rollout [1] to interpret the extractor. Its glob is to quantify the flow of information in self-attention layers by simulating the information propagated from the input layer to the higher layers. Attention rollout assumes information propagation in a transformer as a directed acyclic graph, where the nodes refer to the token representations at each layer and edges denote the attention. The attention weight is regarded as the proportion of information transferred between two nodes. In this case, the information propagated between two nodes through a certain path is computed by multiplying the weights of all edges in the path. Considering there exist multiple paths between two nodes, the total amount of information is the sum of all possible paths between two nodes. At the implementation level, the attentions rollout score at i layer is computed by recursively multiplying the attention weight matrices in all layers below:

$$\tilde{A}(l_i) = \begin{cases} A(l_i)\,\tilde{A}(l_{i-1}) & \text{if} \quad i > 0 \\ A(l_i) & \text{if} \quad i = 0 \end{cases} \tag{5.4}$$

where $A = 0.5W_{att} + 0.5I$ refer to the raw attention updated by residual connections, $\tilde{A}$ represents the attention rollout, and the multiplication operation denotes matrix multiplication.

As shown in Figure 5.3, for an extracted snippet x_i, we use the attention rollout score of its [CLS] token to extract K context prompts $L_i = [l_i^1, l_i^2, .., l_i^k]$. Here, we mask the attention rollout scores for all [CLS] tokens, since [CLS] token itself does not contain any effective context information. A sliding window strategy is adopted to extract the text spans containing the highest average attention rollout score, which are considered as the explanation for extracting the snippet. These text spans are used as the context prompts for the extracted snippet x_i. In the application, the window size is set to 8 tokens. One thing that is worth noticing is that the context prompts are not limited to the extracted snippet, but also disperse in its neighboring snippets.

These extracted context prompts are used to enhance the generation process in the same way as prompt-based language models. Given an extracted snippet x_i, we concatenate it with its corresponding context prompts L_i as the input for the generator:

$$y_t = \sum_{x_i \in X_N} G_\theta(q, L_i, x_i, y_{<t}) MLP(h_i^t) \tag{5.5}$$

Note that all the component of the input are not directly concatenated together, instead we use a set of special tokens to separate these components.

5.3.3 Global Context Information

The extract-generate framework split the input of the extractor and the generator into multiple chunks to handle a longer input. The cessation of the information interaction between chunks inevitably leads to the loss of information that captures the long-term dependency, which leads to loss of global context information. Previous works [159, 22] mainly focus on capturing such context information in the extractor. However, similar to the local context information, these information is difficult to be transferred to the generator.

A commonly used method for capturing the global context information of a snippet is to search for its related snippets in the source document. Here, we adopt a similar strategy but in the form of the context prompt. For an extracted snippet x_i, we adopt the context prompts from its most related snippet as its global context prompts G_i. We use the cosine similarity between the snippet representations R in the extractor to select the most related snippet for x_i. Considering a search space of all source snippets inevitably brings noise, in practice, we reduce the search space to top-score snippets that have already been chosen in the snippet extraction stage. Follow the same way used for local context information, global context prompts are used to enhance the generation process:

$$y_t = \sum_{x_i \in X_N} G_\theta(q, G_i, x_i, y_{<t}) MLP(h_i^t) \tag{5.6}$$

5.3.4 Application

Instead of adding more neural network architecture, CAEG preserves context information by adjusting the input and output of the extractor and the generator. This allows it to be easily applied to any trained extract-generate summarization model as long as it has a transformer-based extractor. In this paper, we adopt DYLE [98] as the base model. It is worth noting that we still need to finetune the generator, which

Dataset	Type	Domain	Size	Source Len	Target Len	Query
QMsum	dialogue	meeting	1808	9070	70	√
Arxiv	document	science paper	200000	6030	273	×

Table 5.1: The statistics and comparison of datasets in the experiment. The Source Len and Target Len stand for the token number of the source document and the summaries.

makes it adapt to additional contextual information input. To utilize both local and global context information in the framework, we concatenate the two types of context prompts and add them to the generator.

5.4 Experiment

5.4.1 Dataset

We evaluate our proposed methods in the context of long-input summarization. Two datasets from different domains are adopted as evaluation benchmarks. The detailed comparison is shown in Table 5.1. **arXiv**[19] is a dataset for long-input single-document summarization. It collects scientific articles from arXiv.org and takes the abstracts of these articles as the target summaries. Compared to previous news summarization datasets, it has significantly longer input and output. **QMSum**[189]is a benchmark for query-focused dialogue summarization. The dataset is composed of meeting records from three different domains. Since only a small proportion of the source document is correlated to the query, QMSum has a higher compression rate than other long-input summarization datasets.

5.4.2 Baseline

We compare our method with some commonly used pretrained models and previous state-of-the-art methods designed for long-input summarization. The pretrained models include Bart-large [78] and PEGASUS [174]. There are three kinds of methods for long-input summarization: transformers with sparse attention, hierarchical transformers, and models based on the extract-generate framework, which are shown below:

- **Transformers with sparse attention:** We adopt various sparse-attention transformers for comparison including Longformer [6], BigBird [173], and LSH[54]. DialogLM [188] is a pretrained model for dialogue understanding, and we display it with both full attention and sinkhorn sparse attention.
- **Hierarchical transformers:** HMNet [194] designed a hierarchical structure for dialogue summarization, which includes discourse-level information and speaker roles. HAT-BART [123] proposes a Transformer-based model with hierarchical attention. It is capable of capturing information among sentence and paragraph-level.
- **Models based on extract-generate framework:** Dyle [98] is the state-of-the-art summarization model based on extract-generate framework, which is also our base model. BM25+Bart refers to a baseline model taking BM25 as extractor and Bart as the generator, which is drawn from [180]. DANCER [45] propose a divide-and-conquer approaches. Moreover, we also report two extractive summarization models for long-input summarization: ExtSum-LG [159] and SSN-DM [22].

We add the maximum input sequence length of the model in the brackets after the model, and "dyn" represents dynamic denoting there is no limitation for the input length.

Model	R-1	R-2	R-L
Bart-large (3072)	32.16	8.01	27.72
HMNet (8192)	32.29	8.67	28.17
Longformer (8192)	31.60	7.80	20.50
DialogLM (5120)	34.02	9.19	29.77
DialogLM - Sparse (8192)	33.69	9.32	30.01
BM25+Bart (dyn)	32.9	9.0	22.0
DYLE (dyn)	34.42	9.71	30.10
CAEG-local (dyn)	**36.50**	11.15	**30.50**
CAEG-global (dyn)	36.11	11.22	30.24
CAEG-all (dyn)	36.41	**11.41**	30.21

Table 5.2: Results on QMSum.

5.4.3 Implementation Details

Taking the state-of-the-art extract-generate summarization model DYLE as the backbone, we adopt Roberta-base [94] as the extractor and Bart-large [78] as the generator. Both models are initialized by the checkpoint given by DYLE. The implementation of our code is based on transformers from Hugging Face. Adam algorithm is used for optimization and the learning rate is set to $2 - e^{-6}$. The batch size for the training is set to 1, and gradient accumulation steps are set to 8. We conduct the validation for every 100 steps and train the model for a maximum of 20000 steps. The experiments are run on a single V100 GPU. In terms of the inference stage, we adopt a beam search size 4 for arXiv and a beam search size 1 for QMSum. The length limitation is 150 to 450 for arXiv and 50 to 100 for QMSum.

Model	R-1	R-2	R-L
PEGASUS (3072)	44.21	16.95	38.83
BigBird-PEGASUS (3072)	46.63	19.02	41.77
LSH (7168)	**48.24**	**20.26**	41.78
HAT-BART (3072)	46.68	19.07	42.17
ExtSum-LG (dyn)	44.01	17.79	39.09
DANCER-PEGASUS (dyn)	45.01	17.60	40.56
SSN-DM (dyn)	45.03	19.03	32.58
DYLE (dyn)	46.41	17.95	41.54
CAEG-local (dyn)	46.69	18.48	42.04
CAEG-global (dyn)	46.86	18.61	**42.20**
CAEG-all (dyn)	46.81	18.58	42.16

Table 5.3: Results on arXiv.

5.4.4 Experiment Results

The evaluation results are summarized in Table 5.2 and Table 5.3. In the experiment, following previous works, we adopt ROUGE 1.5.5 [86] including Rouge-1 (R-1), Rouge-2 (R-2), and Rouge-L (R-L) as evaluation metrics. There are three types of variants of our approach. (1) CAEG-local: We only add local context prompts to the generator; (2) CAEG-global: We only add global context prompts to the generator; (3) CAEG-all: We add both local context prompts and global context prompts to the generator, and the number of context prompt K is set to 2.

On QMSum, CAEG achieves the new state-of-the-art performance. Compared with the base model DYLE, all three variants of our model yield a clear improvement, which shows that CAEG outperforms previous extract-generate summarization models whose context information is ignored. These results suggest the importance of context information in the long dialogue summarization. CAEG's better performance can be attributed to its effectiveness in preserving context information between the

extractor and the generator.

On arXiv, CAEG outperforms the other models based on the extract-generate framework, but fails to achieve the best result. We believe there are two reasons for this. On the one hand, CAEG is an approach that enhance extract-generate summarization model by utilizing the context information, so the final performance is partly dependent on the base model itself. Hence, even if we achieved improvement on the base model DYLE, it can not fill the natural gap between DYLE and LSH. On the other hand, the performance enhancement brought by context information on arXiv is not as large as QMSum. This might be because the structural patterns also play an important role in identifying salient information in the summarization of science papers. In this case, the context information becomes less useful.

It is a surprise that CAEG-all fails to further improve the performance after adding both local and global context information. A possible reason is that the generator cannot effectively distinguish the two types of context information. It is our future work to find a optimal way for this problem.

5.4.5 Analysis and Discussion

Effect of number of context prompts As suggested in the Method Section, for each extracted snippet, we generate K context prompts to preserve its context information. Here, we vary the value of the hyperparameter K and test it on both QMSum dataset and arXiv dataset in Table 5.4 and Table 5.5. The largest K we show in the table is 4, since a greater value leads to a clear drop in performance.

We can observe that the performance of the model increase when the value of K increase until it reach a upper bound around 3. This suggests that the context information requires multiple context prompts to be effectively represented. Moreover, the model achieves its best performance when k=4 for the local context and k=3 for the global context in QMSum, while the best K value are smaller for arXiv. This

	Local			Global		
	R-1	R-2	R-L	R-1	R-2	R-L
K=1	35.46	10.37	29.56	35.38	10.36	29.34
K=2	35.79	10.77	29.79	36.09	10.88	29.83
K=3	36.05	10.78	30.15	**36.11**	**11.22**	**30.24**
K=4	**36.50**	**11.15**	**30.50**	35.15	10.28	29.15

Table 5.4: Analysis of the number of context prompts from K=1 to K=4 on QMSum Dataset.

	Local			Global		
	R-1	R-2	R-L	R-1	R-2	R-L
K=1	46.56	18.44	41.88	46.69	18.51	42.01
K=2	46.62	18.46	41.94	**46.86**	**18.61**	**42.20**
K=3	**46.69**	**18.48**	**42.04**	46.72	18.50	42.05
K=4	46.56	18.42	41.90	46.47	18.30	41.88

Table 5.5: Analysis of the number of context prompts from K=1 to K=4 on arXiv Dataset.

is expected as the long text snippets (utterance) in QMSum require more context information than the ones (sentence) in arXiv.

Effect of different forms in transferring context information We are interested in the most effective form to transfer the context information from the extractor to the generator. To investigate this, we evaluate the effectiveness of three different forms in transferring the local context information in Table 5.6. (1) Context prompt: We report the best result of our approach using the local context information. (2) Adjacent snippet: We concatenate each extracted snippet with its neighboring snippets and feed them into the generator. (3) Snippet embedding: For each extracted snippet, we can obtain its representation from the its corresponding [cls] token in extractor. The representation are fed to the generator by adding it before the in-

	R-1	R-2	R-L
Context prompt	**36.50**	**11.15**	**30.50**
Adjacent snippet	35.74	10.78	29.46
Snippet embedding	33.26	8.66	27.36
No context(DYLE)	34.42	9.71	30.10

Table 5.6: The comparison between different approaches in preserving local context information on QMSum dataset.

put embeddings. Since the word dimension of the extractor and generator are not the same, we use a MLP to map the snippet embedding from the dimension of the extractor to the generator.

The results show that context prompt outperforms the other two ways in transferring the local context information. Although directly concatenating the neighboring snippets achieves a strong result, it leads to a huge increase in the GPU memory cost of the generator. Meanwhile, due to the noise brought by the large amount of context, it underperforms our proposed context prompt. As for using snippet embeddings, the results suggest that it can not effectively reflect the context information, especially when the distribution between the extractor and generator are not the same. Conducting a end-to-end training may solve this problem, but it will largely increases the computational cost.

Case study of context prompts To have a more clear understanding about how context prompts work, we display the context prompts of some extracted snippets in the Table 5.7 and Table 5.8. Here, we display the extracted snippet, the neighboring snippets of the extracted snippet (Context Snippet), and the generated context prompts (Prompt w Context). For a better comparison, we also show the salient text spans obtained in the same ways as context prompt but without context information (Prompt w/o Context). This is achieved by feeding the extracted snippet solely to the extractor.

QMSum	
Context Snippet	Suzy Davies Am: Do you think it would be better for us as scrutinisers of this act if we could see the draft changes to cps guidance on the public interest test before we make our final decision?
Extracted Snippet	Barry Hughes : I honestly don't think that would necessarily be helpful. I've had some discussions with Kwame, who would have an involvement in this. What we would envisage is that we would simply want to take the present public interest factors, which are set out, in my view, very clearly in the code for crown prosecutors ... And we'd need to work that up as we go along, and I think you'd run a risk of putting the cart before the horse, if I may put it like that.
Context Snippet	Suzy Davies Am: It's just that, personally, I think the public interest test is critical in all this...
Prompt w Context	["see the draft changes to cps guidance", "want to take the present public interest factors"]
Prompt w/o Context	["Barry Hughes : I honestly do", "if I may put it like that"]

Table 5.7: The case study of the context prompts on QMSum. For each extracted snippet, we display two generated context prompts. For a better observation, we use various colors to label the salient context information, and adopt the underline to emphasize the position of the generated context prompts in the source text.

arXiv	
Context Snippet	One may calculate the penetration probability numerically by using the path integral method or the wkb approximation.
Extracted Snippet	However, it is highly desirable to have an analytical expression for the barrier penetrability when one introduces an energy-dependent one-dimensional potential barrier @xcite or barrier distribution functions @xcite.
Context Snippet	In current work, we derived a new barrier penetration formula based on the wkb approximation...
Prompt w Context	["we derived a new barrier penetration formula based", "or barrier distribution functions @xcite"]
Prompt w/o Context	["barrier distribution functions @xcite.", "to have an analytical expression for the barrier"]

Table 5.8: The case study of the context prompts on arXiv.

In terms of the example from QMSum, we find the "public interest test" is the important information among the context. The context prompts successfully capture such information and highlight a related topic "changes to cps guidance", while the prompt with no context information fails to achieve this. In terms of the example from arXiv, "barrier" has been mentioned multiple times in the extracted snippet, which indicate its importance. However, the context prompt can not further focus on "barrier penetration" without the context information. These examples not only suggest the importance of the local context information, but also show the effectiveness of the context information captured through the interpretation of the extractor.

5.5 Conclusions

In this paper, we firstly propose to focus on context information preservation in an extract-generate summarization framework for long input. To address the challenge of context information loss from the extractor to the generator, we generate a set of context-related text spans called context prompts for each extracted snippet and feed them into the generator through concatenation. A novel approach is proposed to generate the context prompts based on the interpretation of the extractor. Hence, our approach can be applied to most extract-generate summarization models at a low cost. The experiments further show its effectiveness in capturing and preserving the context information in the extract-generate summarization framework.

Part III

Low-Resource Summarization via Knowledge Transferring

Chapter 6

Few-shot Query-Focused Summarization with Prefix-Merging

6.1 Introduction

Although the abundance of annotated data for news summarization has ushered in the era of summarization models based on neural models, not all sub-tasks or domains of summarization have enough data. Query-focused summarization is a typical example. As a classic sub-topic for text summarization, it meets that situation that only a specific aspect of information is needed to be summarized. In other words, it aims to generate a summary based on the source content related to a given query. Hence, this task requires not only to locate relevant content in a passage as question answering (QA) but also to summarize and generate a highlight as text summarization. Although text summarization has been widely studied in recent years, there are fewer attempts on exploring query-focused summarization [25, 135, 163, 136] after the age of neural model. One main reason is the lack of generalized large-scale

datasets. Compared with the easily accessible nature reference summaries such as titles or headlines in text summarization, it is hard to collect large-scale data for query-focused summarization. Meanwhile, human-written reference summaries have always been costly.

The rapidly developed few-shot learning techniques provides potential cues to alleviate the problem of lacking large-scale dataset for query-focused summarization, and knowledge transferring is one of them. In fact, when facing unseen tasks, it is natural for human beings to integrate and transfer the knowledge of known tasks to relevant new tasks. Inspired by this, we innovatively propose to decouple the query-focused summarization to two basic tasks, i.e. text summarization and question answering, and transfer the knowledge from these two tasks to query-focused summarization. However, in parameter-based knowledge learning, previous work are usually one-to-one (pre-train then fine-tune [166]) or one-to-many (domain/task adaption [52, 89]), and seldom of them focus on many-to-one (integrate basic tasks to a complex one). In this case, the previous methods may not work well in this task.

In this paper, we propose a pre-trained strategy, prefix-merging, for few-shot learning in query-focused summarization. In recent prompt-based language models, the prompt/prefix is considered as containing the knowledge of the given task, which provides us an explicit way to control the task-specific knowledge previously dispersed in the language model (LM). For example, prefix-tuning [83] achieved a similar result with fine-tuning by training only the task-specific prefix, a sequence of continuous vectors that prepend to the input. Following the framework proposed by prefix-tuning, prefix-merging aims to integrate the task knowledge from text summarization and question answering into a properly designed prefix and apply the merged prefix to the more complex task, query-focused summarization.

Generally, there are two straightforward ideas for merging knowledge from multiple tasks into a prefix: concatenate the separated prefix for different tasks as a whole or adopt a shared prefix for all the tasks. Considering there exist both similarities

and differences across the tasks, a more flexible prefix design composed of both task-specific part and shared part is used in further investigation. Moreover, we propose a self-adaptive prefix merging that allows the basic tasks themselves to decide the prefix design. Drawn the inspiration from [162], we adopt Fisher Information to calculate the importance scores of the prefix embeddings (basic units for the prefix) for each basic task. For one task, only the prefix embeddings with top scores are activated in the following training. Hence, different tasks can adapt to different parts of prefix automatically. After finishing training the merged prefix, it is transferred to a downstream task for few-shot learning. In the experiment, we explore prefix merging in the context of query-focused summarization, taking PubMedQA [**?**] and DUC [23] as the evaluation dataset.

Prefix-merging provides a potential solution for the few-shot learning in complex tasks that can be integrated by the basic tasks. Benefited by the universality of the prompt-based approach, prefix-merging is not limited by the model architecture and can be used in both autoregressive LM and encoder-decoder based LM. We believe this shows a possible direction to the application of prompt-based approaches. Our contribution can be summarized as follow:

- We provide a new solution for few-shot query-focused summarization by decoupling it to two basic tasks with large-scale training data, text summarization and question answering.

- We propose prefix-merging that integrates the task-specific knowledge from basic tasks to assist the learning of a more complex task, which provides a new solution to many-to-one parameter-transfer learning.

- We further expand the application of prompt-based approaches by applying the prefix to multi-task situation, exploring the interaction between different task knowledge through prefix.

6.2 Related Work

Query-Focused Summarization Query-focused summarization aims to generate a concise highlight from the source document(s) according to a specific topic or query, which is considered as a more complex extension of text summarization. Early works [88, 131] focus on extracting query-related sentences as summaries, while further works [149, 84] improve it by rewriting the extracted sentences with sentence compression. [109, 50] propose neural-abstractive models with an additional query attention mechanism to generate the summaries with respect to the given query. [25] consider the relation among the query and source sentences as a multi-hop inference process and generate the summaries by integrating information from different inference steps. Meanwhile, researchers also utilized QA models to find the possible query-related evidence in query-focused summarization. [163, 164] adopts QA models for sentence-level or paragraph-level answer evidence ranking. [136] incorporate answer relevance scores generated by QA model as explicit fine-grained query relevance to a transformer-based abstractive summarization model. Therefore, we believe the text summarization and QA are the foundation for query-focused summarization and choose them as the auxiliary tasks in this work.

Prompt-Based Approaches Prompting originally refers to adding instructions and several examples to a task input and generating the output from the LM. A fundamental idea for prompt-based approaches is that let the tasks adapt to the LM. Some researchers tend to utilize the idea to improve the performance of the model by making the form of the task closer to the LM. A series of works [115, 58, 133] explore the prompt engineering and prompt ensemble in natural language understanding tasks. For instance, instead of manually designing prompt, AutoPrompt [133] automatically search for a sequence of discrete words as prompt to extract knowledge from pre-trained LMs. Other works choose to optimize the prompt in a continuous space. [118, 91] adopt hand-designed prompt as initialization and add learnable perturbation

on the prompt. Other researchers choose to find a parameter-efficient adaption from LM to a specific task. GPT-3 [7] adopts manually designed task-specific prompts to adapt the LM for different generation tasks. Prefix-tuning proposes "prefix tuning" for language generation task: learning a sequence of continuous prefixes that are inserted to every transformer layer. [77] provides a simplified version of "prefix tuning" with fewer parameters and more robust prompt initialization on the SuperGLUE tasks. [181] has recently proposed a prefix-based model that utilize domain words to achieve zero-shot domain adaption on dialogue summarization. In this work, following the framework of prefix-tuning, we aim to integrate basic tasks to a more complex one by merging the task knowledge through the prefix.

6.3 Method

6.3.1 Problem Statement

In this work, we aim to transfer the task-specific knowledge from text summarization and question answering (auxiliary tasks) to query-focused summarization (target task) to assist its learning. In this case, the query-focused summarization model can obtain a fair performance even with limited data. There are mainly two stages to accomplish this. In the first stage, a model is trained on the large-scale data from two auxiliary tasks to obtain the potentially useful knowledge for query-focused summarization. Here, we propose prefix-merging that merges task knowledge from auxiliary tasks into a particularly designed prefix. In the second stage, we train the model with data from query-focused summarization but with the assistance of the trained parameters from the first stage. For prefix-merging, the merged prefix is used to transfer the knowledge from the first stage to the second stage.

Our prefix-merging is considered as an extension of prefix-tuning, so we have a brief introduction about it in the section 3.2 as the background of our method. Then, we

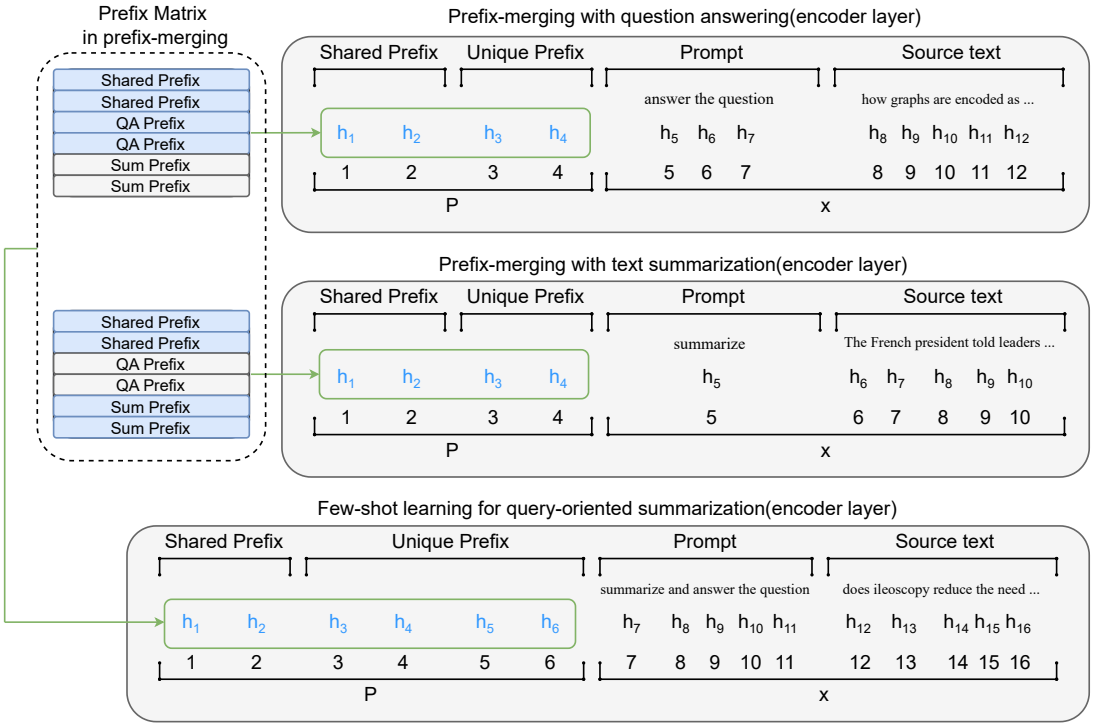

Figure 6.1: Focusing on the encoder layer of BART, the figure shows annotated examples and comparison between the prefix-merging (top, mid) on the two auxiliary tasks (summarization and QA) and applying the merged prefix on query-focused summarization with prefix-tuning (bottom).

introduce our own method from section 3.3 to 3.5, and how we apply the merged prefix on query-focused summarization in 3.6.

6.3.2 Prefix-tuning

Consider there is a transformer-based encoder-decoder LM $p(y|x)$ such as Bart[78] and it is parametrized by ϕ. Taking the encoder layer in transformer as an example, let $z = [x]$ denote its input sequence. We use h_i to represent the concatenation of all activation from all layers at the index i, and each activation consists of a key-value pair. The h_i for all $i \in x$ in encoder layer is a function of z_i and the other activations

in the context based on the LM, as follows:

$$h_i = LM_\phi(z_i, h_{\neq i}) \tag{6.1}$$

Prefix-tuning prepends a prefix for the encoder layer to obtain $z = [prefix; x]$, or prepends prefixes for cross-attention layer or self-attention layer in the decoder to obtain $z = [prefix; x; y]$ or $z = [prefix; y]$. Here, we use P_{idx} to represent the sequence of prefix embedding indices, and $|P_{idx}|$ is used to represent the length of the prefix. A trainable matrix $P_\theta \in |P_{idx}| \times dim(h_i)$ is initialized to store the prefix parameters. Following the recurrence relation in equation (1), h_i is calculated as below in prefix-tuning.

$$h_i = \begin{cases} P_\theta[i,:], & if\ i \in P_{idx} \\ LM_\phi(z_i, h_{\neq i}), & otherwise \end{cases} \tag{6.2}$$

Hence, h_i becomes a function of the trainable P_θ and it allows the prefix parameters to control the model by affecting the activations in every layer of the transformer. During the training in prefix-tuning, the objective maintains the same as normal task, but only the prefix parameters θ are trainable and the parameters of the LM ϕ are fixed. In this case, the prefix parameters contain all the task-specific knowledge learned from the training.

6.3.3 Intuition for Prefix-merging

Intuitively, to merge the knowledge from different tasks into the prefix, the simplest way is to concatenate the individual prefix from these tasks. Another way is to use a shared prefix that is updated by all the tasks. Instead of using either of the two ways, we choose a more flexible prefix design for further investigation of the problem. For each task, its prefix consists of a shared sub-prefix (prefix embeddings shared by all

tasks) and a task-specific sub-prefix (prefix embeddings used for a specific task) whose lengths are controlled by two hyperparameters. We believe the shared sub-prefix tends to represent the similarities between all merged tasks, while the task-specific sub-prefix refers to the uniqueness of each task. Meanwhile, the two mentioned intuitive methods can also be restored when any of the two hyperparameters is set to 0.

6.3.4 Prefix-merging

Similar to prefix-tuning, a trainable matrix P_θ is used to store the prefix parameters. The difference is that there are n different tasks denoted as $[task_1, task_2, .., task_n]$ that share or partly share the whole matrix. For each single task, it corresponds to several prefix embeddings in the prefix metrix, and we separate them into task-specific unique sub-prefix with a length of l_u and a shared sub-prefix with a length of l_s. Figure 6.1 shows an example of training two auxiliary tasks, text summarization and question answering, for prefix-merging. Here, both the shared sub-prefix length and unique sub-prefix length are set to 2. The prefix embedding indices for text summarization is [1,2,3,4], and it changes to [1,2,5,6] for QA.

In this way, the P_θ has the dimension of $(l_s + l_u * n) \times dim(h_i)$. We use P^n_{idx} to represent the the sequence of prefix embedding indices of $task_n$ and its length $|P^n_{idx}|$ is equal to $l_s + l_u$. As follow, the h_i for $task_n$ is calculated based on the following equation:

$$h_i = \begin{cases} P_\theta[P^n_{idx}[i], :], & if\ i \leq |P^n_{idx}| \\ LM_\phi(z_i, h_{\neq i}), & otherwise \end{cases} \tag{6.3}$$

To distinguish the different tasks during the training, we add a task-specific prompt before the original input tokens following T5 [120]. As shown in Figure 6.1, the prompt is "summarize" for the text summarization and the prompt is "answer the question" for question answering. During the training, we adopt a mixed-task training strategy where instances from different tasks equally exist in the same training batch.

6.3.5 Self-adaptive Prefix-merging

Considering that manual design does not always lead to the best results, we further propose a self-adaptive prefix-merging. Instead of presetting the lengths of shared sub-prefix and unique sub-prefix, we aim to let the auxiliary tasks decide the prefix design. The idea is based on Fisher Information, a evaluation metric that reflects how much the model output changes when its parameters change. It can be considered as the importance of a parameter for the model on a certain set of data [162]. In this way, we can find the most important sub-prefix for each auxiliary task based on Fisher Information with the following equation:

$$F_i = \frac{1}{pq} \sum_{j=1}^{p} \sum_{k=1}^{q} \left(\frac{\partial log(p(y_k|x_k;\theta))}{\partial \theta_j}\right)^2 \tag{6.4}$$

where $F_(i)$ refers to the average Fisher information of the i-th prefix embedding, p denotes the number of parameters in the embedding and q represents the number of data. x and y refer to the input and output data in one auxiliary task.

During the training, we first initialize the prefix as a shared prefix trained by all auxiliary task for one epoch. Taking $task_n$ as an example, we then conduct a complete forward propagation and back propagation (one epoch) for all data in $task_n$, and calculate the Fisher Information for each prefix embedding. Only the top-n prefix embeddings will be used in the later training for $task_n$ and others will be masked. In other words, the P^n_{idx} is the indices of the top-n prefix embeddings. After obtaining the important sub-prefix for each task, naturally, some prefix embeddings are shared by different tasks while others are task-specific. At last, we continue the training of the prefix on the auxiliary tasks with the selected sub-prefix.

6.3.6 Applying the Merged Prefix to the Target Task

After training on the auxiliary tasks, we obtain the prefix parameters that contain task knowledge from text summarization and question answering. We apply the

knowledge to the target task, query-focused summarization, by using the merged prefix as initialization and continue prefix-tuning on it, but with a few differences. As shown in Figure 6.1, all the prefix parameters are used for the target task including the shared sub-prefix and all the unique sub-prefixes. For self-adaptive prefix-merging, only the prefix embedding that is used for at least one auxiliary task is applied for the target task, otherwise it will be masked. We also adopt a new prompt that suggests the relation between the target task and auxiliary tasks. More specifically, we concatenate the prompt of text summarization and question answering as "summarize and answer the question" for query-focused summarization.

6.4 Experiment

6.4.1 Datasets

To evaluate the idea of prefix-merging, we take query-focused summary as the target task, text summarization and question answering as two auxiliary tasks. We focus on commonly used datasets for query-focused summarization: PubMedQA and DUC. We also test our model on Debatepedia [109] and have a discussion about it in the appendix. In terms of the PubMedQA, it requires the model to generate a summary containing 1-3 sentences as an answer to a question based on a medical related document. Since we train the target task under a few-shot situation, only part of the training set is used in the experiment and we test the model on the full testing set containing more than 20000 data samples. In terms of the DUC, it is a multi-document query-focused summarization dataset with hundreds of data samples. Hence, we adopt a extract-generate framework to conduct the experiment. We first adopt BM25 to extract a set of query-related sentences from the source documents and use the concatenation of the query and extracted sentences as the input of our model. The DUC 2006 is used for training and DUC 2007 is used for testing.

Data Size	50			150			300		
Model	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
Random	30.33	9.96	28.00	32.08	11.67	28.97	32.79	11.92	29.51
Unq(30)	30.81	10.97	26.52	32.13	11.73	28.23	32.37	11.86	27.81
Unq(20)+Sha(10)	32.36	11.40	28.30	33.14	12.12	29.10	33.68	12.39	29.81
Unq(10)+Sha(20)	32.64	11.84	**28.60**	33.46	12.34	**29.46**	33.90	12.59	**30.12**
Sha(30)	32.44	11.48	28.17	33.28	12.04	29.11	33.87	12.41	29.83
Self-adaptive	**33.18**	**12.01**	28.45	**33.66**	**12.40**	28.98	**34.19**	**12.65**	29.53
BART(tar)	30.95	10.54	26.87	32.28	11.46	28.23	32.52	11.63	28.33
BART(aux+tar)	31.65	10.75	28.18	32.23	11.27	28.57	32.66	11.62	29.16
BART_base(full)	37.49	14.11	34.45	37.49	14.11	34.45	37.49	14.11	34.45

Table 6.1: Evaluation result for query-focused summarization on PubMedQA. We compare the result on three different training data size: 50, 150, 300. Here, we also provide result of BART-base on the full-size training for better comparison.

In terms of the two auxiliary tasks, we adopt the XSum dataset [106], a highly abstractive single-document summarization dataset, for the text summarization, and we use the classic machine reading comprehension dataset SQUAD 1.1 [122] for question answering. Debatepedia [109] is also one of the commonly used query-focused summarization dataset. However, during our experiment, we find a serious but unintentional data leakage problem between the training set and the testing set in its standard division. Around 64% of summaries in the testing set appear or have similar ones in the training set (difference is lower than 2 words). In this case, the model tends to remember the data samples rather learning to do query-focused summarization.

6.4.2 Experiment Setting

Our implementation is based on the BART-large model from HuggingFace and all the input is truncated to 800 tokens. For the prefix-tuning based method, a default setting is a learning rate of 5×10^{-5} and a prefix length of 30. The batch size is set to 48 when conducting prefix-merging, and for few-shot prefix-tuning, it changes

Data Size	50			150			300		
Model	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
Random	0.20	0.30	0.45	0.30	0.22	0.36	0.08	0.11	0.45
Unq(30)	0.45	0.23	0.40	0.41	0.36	1.02	0.08	0.08	0.11
Unq(20)+Sha(10)	0.50	0.22	0.25	0.04	0.22	0.59	0.17	0.09	0.23
Unq(10)+Sha(20)	0.14	0.01	0.13	0.23	0.15	0.16	0.15	0.02	0.32
Sha(30)	0.19	0.08	0.18	0.25	0.27	0.64	0.23	0.10	0.24
Self-adaptive	0.38	0.16	0.48	0.10	0.20	0.71	0.12	0.08	0.35
BART(tar)	0.53	0.52	0.51	0.21	0.15	0.25	0.32	0.22	0.14
BART(aux+tar)	0.47	0.32	0.43	0.24	0.17	0.21	0.18	0.14	0.16

Table 6.2: .Standard Deviation of the Results on PubMedQA

with the size of the training data. In the experiment, we also use fine-tune based method as a comparison, and the default setting for it is a learning rate of 2×10^{-5} and a batch size of 48. At training time, we adopt the AdamW optimizer with default hyperparameters. At inference time, we use beam search with a beam size of 2. The output length limitation is set from 30 to 75 tokens for PubMedQA and 250 to 300 for DUC. Since few-shot learning is sensitive to the training data, we train the models with three sets of training data and report the average result on PubMedQA.

As for evaluation metric, following previous works, we apply ROUGE [86] including Rouge-1 (R-1), Rouge-2 (R-2) and Rouge-L (R-L) for the query-focused summarization. We adopt a full Python implementation of the ROUGE-1.5.5, to conduct the experiment.

6.4.3 Result

We first evaluate the different prefix designs within three different few-shot learning data sizes (50, 150, 300) for the target task in Table 6.1. To have a better understand-

ing of the experiment results on PubMedQA, we report the standard deviation (std) across multiple runs in the experiment on PubMedQA in Table 6.2. The results of the other two dataset DUC and Debatepedia are also shown in the Table 6.3 and Table 6.4. Here, we only shows the results with a few-shot learning data size of 50. "Unq(n)" stands for the total number of the prefix embeddings in all unique sub-prefix, while "Sha(n)" refer to the shared sub-prefix. For example, "Unq(10)+Sha(20)" represent the merged prefix consists of unique sub-prefix with length 10 (5 for each task) and the shared sub-prefix with length 20. In terms of the self-adaptive prefix-merging, we initialize the prefix length as 40 and select the top-25 prefix embeddings for each tasks. In this case, self-adaptive prefix-merging is more likely to have a comparable parameter numbers with the other prefix designs, which makes a fair comparison. We also add a baseline "random": randomly initialize the prefix and conduct few-shot prefix-tuning on the query-focused summarization dataset. We further compare our model with BART in three training settings:(1) BART(tar) refers to fine-tuning the BART only use the limited data from query-focused summarization; (2) BART(aux+tar) refers to first fine-tuning on the auxiliary tasks then fine-tuning on query-focused summarization, which is similar to some previous approaches [167] and [33]; (3) BART(full) refers to fine-tuning on the large-scale data from query-focused summarization.

In Table 6.5, we compare the prefix-merging with fine-tuning. Since it is a two-stage training process (training on auxiliary tasks then applying on the target task), each stage can adopt prefix-based training (only the prefix parameters are trained and the LM parameters are frozen) or fine-tuning (all parameters are trained). Therefore, we report four variants in total: (1) fine-tuning + fine-tuning (Fine+Fine), which is the same as BART(aux+tar); (2) fine-tuning + prefix-tuning (Fine+Prefix); (3) prefix-merging + fine-tuning (Prefix+Fine); (4) prefix-merging + prefix-tuning (Prefix+Prefix), which is our proposed approach in Section 3. Despite the variant (1), we add a prefix of length 30 to the model. Taking variant (2) as an example, firstly, both the prefix and the LM are updated by the training data from auxiliary tasks

Model	R-1	R-2	R-L
Random	33.96	6.37	23.46
Unq(30)	34.56	6.80	24.40
Unq(20)+Sha(10)	34.54	6.64	23.53
Unq(10)+Sha(20)	34.87	7.23	24.70
Sha(30)	34.53	6.89	23.98
Self-adaptive	**34.99**	**7.47**	**24.74**
BART(tar)	33.83	6.52	22.71
BART(aux+tar)	12.85	4.42	15.17
Ext_Oracle	35.62	9.20	24.00

Table 6.3: Evaluation result for query-focused summarization on DUC. Ext_Oracle refers to oracle extractive result taking the query-related sentences as input, which can be seen as the upper bound of this experiment.

and then only the prefix parameter is trained on the target task.

Table 6.6 displays the result of using different auxiliary tasks for query-focused summarization. "Sum+QA" refers to the best result when using both text summarization and QA; "Only Sum" and "only QA" are designed for ablation study where only one of the two tasks is used in stage one. Moreover, we also import a baseline "Unrelated Task" that takes sentence copying as the auxiliary task, which contains no useful task knowledge for query-focused summarization. We use prefix-tuning to train the model when there is only one auxiliary task.

We summarize the experiment result with the following conclusions.

The self-adaptive prefix-merging achieves a comparable result with the best manually prefix design. It is not a surprise that self-adaptive prefix-merging outperforms most of the prefix designs and achieves the best result in both datasets. One thing that is worth noticing is that the effective length for self-adaptive prefix-

Model	R-1	R-2	R-L
Random	18.57	5.50	17.50
Unq(30)	21.53	6.77	20.07
Unq(20)+Sha(10)	22.20	7.20	**20.59**
Unq(10)+Sha(20)	22.03	7.13	20.29
Sha(30)	21.85	6.92	20.27
Self-adaptive	**22.29**	**7.39**	20.53
BART(tar)	21.60	6.81	19.89
BART(aux+tar)	21.36	6.24	19.00
BART(full)	57.74	43.42	57.03
BART(full)_redivided	24.80	8.06	22.95

Table 6.4: Evaluation result for query-focused summarization on Debatepedia. In the upper and middle part, we display the result of few-shot learning with 50 data samples on standard division of Debatepedia. In the lower part, we show the result of BART training with full-size data but with different data division. BART(full) represents the standard division and BART(full)_redivided refers to a new division that do not have the data leakage problem (we achieve this by redivide all data samples by an alphabetical sort, where similar data samples tend to gather together rather than scatter in both training and testing set).

merging is also around 30 (initialized as 40 and 10 are masked by all tasks), which means the number of parameter maintains equal with other prefix design. Meanwhile, its proportion of shared sub-prefix and unique sub-prefix is similar to the best manual design Unq(10)+Sha(20). This suggests that self-adaptive prefix-merging has the ability to find the best prefix design automatically. Compared with BART, self-adaptive prefix-merging outperforms both BART(tar) and BART(aux+tar), which indicates the effectiveness of prefix-merging. In the experiment on DUC, we notice that BART(aux+tar) drops a lot compared with other results. We believe this is

Model	R-1	R-2	R-L
Fine+Fine	31.65	10.75	28.18
Fine+Prefix	31.64	10.79	27.57
Prefix+Fine	32.03	11.30	28.12
Prefix+Prefix	**33.18**	**12.01**	**28.45**

Table 6.5: The comparison between prefix-merging and fine-tuning with a training data size of 50.

Model	R-1	R-2	R-L
Unrelated Task	31.34	10.77	27.08
Only Sum	32.38	11.56	27.75
Only QA	31.78	11.39	28.43
Sum and QA	**33.18**	**12.01**	**28.45**

Table 6.6: The comparison between using different auxiliary tasks with a training data size of 50.

because the difference between DUC and datasets used in auxiliary tasks is relatively huge and the generalization ability of BART is lost after training on the auxiliary tasks. As for the result of Debatepedia in Table 6.4, we observe a huge gap between the BART(full) and BART(full)_redivided and it can not be explained by the difference of the division. Meanwhile, the result of few-shot learning is much lower than the result of full-size training. Both phenomenon suggest there exist a data leakage problem. The poor performance of BART on the redivided Debatepedia also make us question whether Debatepedia is qualified for query-focused summarization.

Prefix-merging is better than fine-tuning for integrating and transferring task knowledge to the downstream task. In Table 2, prefix-merging outperforms fine-tuning with both downstream training approaches. On the one hand, this is because the generalization ability of the LM is preserved when its parameters are frozen. On the other hand, we believe using prefix as the container of new task knowledge is more similar to the natural form of LM. We believe this shows the potential of prefix-merging in many-to-one knowledge transferring.

The merged prefix contains effective task knowledge from both auxiliary tasks. The initialization of prefix is believed to have a huge effect on the prefix-tuning based approaches. Here, "unrelated task" stands for the performance when the prefix is well-initialized while containing no knowledge for the target task. Compared to it, using one auxiliary task, either text summarization or QA, achieve a better result. This suggests that the two tasks contribute useful knowledge to query-focused summarization. More importantly, prefix-merging gets the best performance. And this can be achieved only when the prefix-merging allows the prefix to integrate effective task knowledge from both tasks.

Model	R-1	R-2	R-L
–Prefix	26.56	8.19	22.16
–Prompt	32.48	11.63	28.57
Unq(10)+Sha(20)	**32.64**	**11.84**	**28.60**
Sha(40)	32.60	11.74	**28.54**
Self-adaptive	**33.18**	**12.01**	28.45

Table 6.7: The experiment result for ablation study with a training data size of 50.

6.4.4 Ablation Study

For more detailed analysis, we design an experiment to explore how different components contribute to our approach. We remove the prefix (-prefix) and the prompt (-prompt) from during the training of the query-focused summarization. The prefix design used here is Unq(10)+Sha(20). We can observe that removing the prompt has a small negative influence on the result. We believe this is because the input form of text summarization and QA is different and the model can distinguish the two tasks even without the given prompt. We also find that the performance drops a lot once the prefix is removed. This indicates that the prompt only plays as guidance, while the prefix is the one containing the task-specific knowledge. For self-adaptive prefix-merging, we compare it with its base prefix design without self-adaption, Sha(40). Even with more trainable parameters, self-adaptive prefix-merging still outperforms it. The result shows that prefix embeddings selected by Fisher Information are crucial for the tasks.

6.4.5 Prefix Visualization

To have a more direct observation, we visualize the attention on the prefix during the inference for query-focused summarization in Figure 6.2. We adopt the attention weights passing through the Softmax layer and further normalize the attention weights

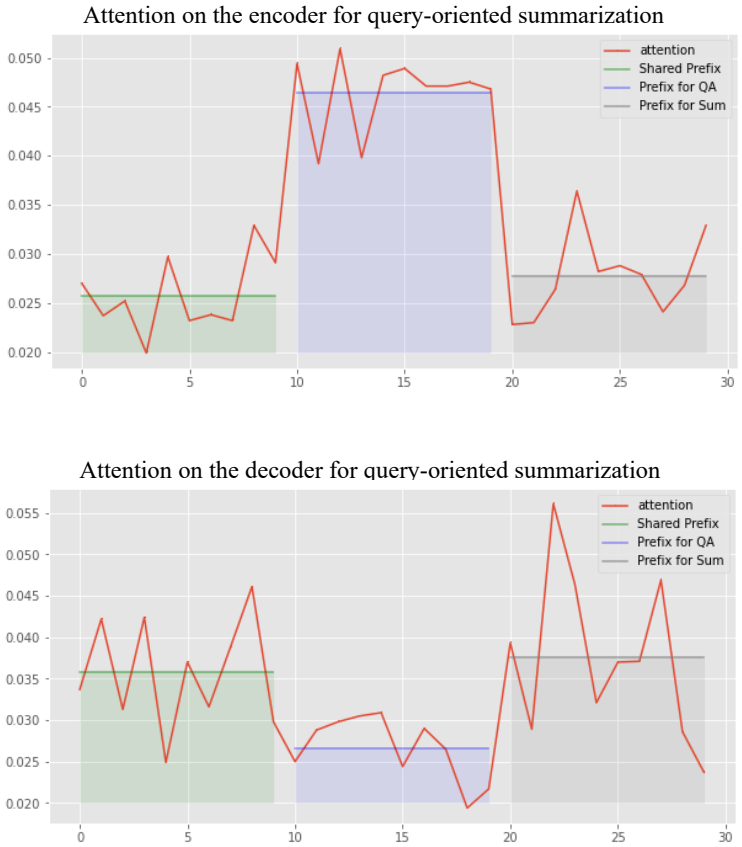

Figure 6.2: The attention score for query-focused summarization in both encoder and decoder of model "Unq(20)+Sha(10)".

only on the prefix embeddings. The final attention score is obtained by averaging attentions from all heads in all layers from 100 random samples. In Figure 6.2, the x axis refers to the indices of the prefix embedding and y axis is the normalized attention score. The straight lines with colors stand for the position of the three types of sub-prefix, shared sub-prefix (0-9), unique sub-prefix originated from QA (10-19) and unique sub-prefix originated from Summarization (20-29), and their heights refer to the average attention score, which can be considered as the prefix's contribution to the query-focused summarization. In this case, it explains how the merged prefix works for query-focused summarization.

For the decoder, we display the attention in the cross-attention layer. In terms of the

encoder, since the model needs to understand the query, we believe it is reasonable that the sub-prefix originated from QA plays the most important role. In terms of the decoder, the sub-prefix originated from QA has little effect on the model, while the shared sub-prefix and sub-prefix originated from summarization dominate. This is because generating the query-focused summaries relies more on generation ability and summarization ability. These findings suggest that the knowledge from QA and summarization is properly used for query-focused summarization through the merged prefix.

6.5 Discussion and Conclusion

Prefix-merging is based on a seq2seq pretrained model, Bart, so it is hard for our model to deal with long input that exceed the input limitation of the pretrained model. Hence, we mainly focus on single-document query-focused summarization. In terms of the experiment, unfortunately, there is seldom few-shot single-document query-focused summarization model. Although there exist some multi-document query-focused summarization models with weak supervision[164, 70, 163], these models all follow a coarse-to-fine framework, which make them hard to directly compared with our model. Hence, we mainly use BART with different training settings as comparison and focus more on the longitudinal comparison. Moreover, we believe that prefix-merging has the potential to be used for other complex tasks that can be integrated from basic tasks. However, we only finish the research in the context of query-focused summarization, which leaves future direction for our work.

In this paper, we show that prefix-merging is an effective approach for transferring and integrating task knowledge from multiple auxiliary tasks to a target task with limited data. In the context of query-focused summarization, integrating text summarization and QA, our approach outperforms the traditional approach fine-tuning. We further discuss the influence of different prefix designs and propose a self-adaptive

prefix-merging. We also provide a visualize explanation for how the merged prefix works. Although this paper focuses on query-focused summarization, we believe these findings suggest a new application for prompt-based approaches in multi-task situation. With the development of large language model, the prompt-level modification has become more and more important. In this case, we believe our work provides guidance for future progress in this field.

Chapter 7

Separating Context and Pattern: Learning Disentangled Sentence Representations for Low-Resource Extractive Summarization

7.1 Introduction

We investigate the knowledge integration between text summarization and question answering for query-focused summarization in Chapter 6. But in text summarization, not every sub-task is as special as query-focused summarization. The uniqueness of most sub-tasks is reflected in their unique domains such as dialogue summarization or science paper summarization. Hence, learning the general summarization knowledge is more important for knowledge transferring. In this chapter, we would like to explore the knowledge transferring across different domains in the context of extractive summarization. The reason why we choose extractive summarization is that it better reflects the very essence of text summarization, that is finding the most crucial

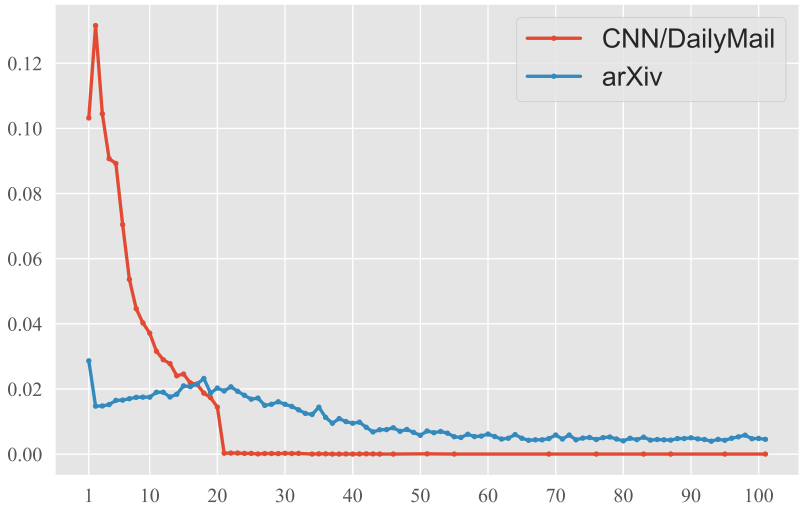

Figure 7.1: Comparison of position distribution of oracle sentences in news summarization dataset CNN/DailyMail and science paper summarization dataset arXiv. The X-axis refers to 1 to 100 sentence position and the Y-axis represents its proportion.

information.

It is widely agreed that extractive summarization is mainly based on context information to select the important sentences. Meanwhile, there also exist other factors that can be used to identify these sentences, such as sentence position or certain n-gram tokens. As shown in Figure 7.1, in the news summarization dataset, lead sentences always have a much higher possibility to become crucial sentences. Meanwhile, Table 7.1 shows that sentences with certain n-gram tokens like "in this paper" or "we find that" are also considered to be important in science paper summarization. Here, we collectively called these factors pattern information, since they are context-independent and can decide the sentence importance solely by themselves. However, as we displayed in Figure 7.1 and Table 7.1, pattern information varies from dataset to dataset. In this case, such information is only effective in its corresponding dataset

CNNDM	Num	arXiv	Num
(cnn) –	21k	in this paper	11k
according to the	3.5k	as a function	6.4k
the first time	2.4k	in the case	4.8k
the end of	1.3k	we find that	3.7k

Table 7.1: Examples about the high-frequency n-grams in oracle sentences from CNN/DailyMail and arXiv.

or domain and can not be generalized like the context information. Although both context information and pattern information are crucial for the task, it is hard to tell whether the improvement of the current extractive summarization models stems from a better understanding of the context information or overfitting the pattern information on specific data. Hence, the existing models may fail to achieve good performance when transferring to other domains or datasets with limited data due to the intermingling of domain-specific pattern information.

In this paper, we aim to apply disentangled representation learning to extractive summarization, and separate the two key factors for the task, context information and pattern information, for a better generalization ability in low-resource settings (zero-shot and few-shot). Our model is built on a pretraining-based extractive summarization model [92] that uses a BERT to encode each sentence with its context to the latent representation. We would like the latent representation to be disentangled with respect to the context and pattern information. Following the previous works [61, 15], we combine the multitask objectives and adversarial objectives/mutual information (MI) minimizing objectives to accomplish this. The multitask objectives aim to encourage the two latent spaces to learn its corresponding information. For the context information, we propose to approximate it by predicting the high-frequency non-stop word appearing in a sentence and its context. For the pattern information, we divide it into two parts: the position pattern feature and the n-gram pattern

feature. The former one can be transferred into a sentence position predicting problem, while the latter one is approximated by predicting whether the target sentence contains any high-frequency n-gram patterns. Then we try two commonly used disentangled representation learning approaches, adversarial objectives/MI minimizing objectives, to further ensure the independence between the two latent spaces.

After the model is trained on a source dataset, it can be transferred to a target dataset for low-resource extractive summarization. In the zero-shot setting, we only utilize the context representation to do the extractive summarization. In the few-shot setting, we choose to fine-tune the pattern-related parameters with a few training instances to automatically select useful patterns for the target dataset.

To evaluate our proposed model, we conduct the experiments on three datasets from different domains: CNN/DaliyMail from the news summarization domain, arXiv from the science article summarization domain, and QMSum from the dialogue summarization domain. These experiments suggest the effectiveness of our model by disentangling context and pattern information.

7.2 Related Work

Extractive Summarization Extractive summarization is an important sub-topic for text summarization. Early works [104, 107, 192, 178] formulated it as a sentence binary classification problem and further extend it with different techniques. With the development of the pretrained model, using a transformer-based pretrained model as encoder [92, 2, 179] leads to a huge improvement in the task. Recently, MATCHSUM [186] has achieved a state-of-the-art performance by combining contrastive learning with extractive summarization. These models mainly focus on improving the performance on a certain dataset or domain. Research on low-resource text summarization is also increasing. AdaptSum [167] propose a pre-train and then fine-tune strategy

for low-resource domain adaptation for abstractive summarization. Other researchers [33] present a similar idea but further enhance it with a data augmentation method using the large corpus from Wikipedia. [181] combines domain words and a prompt-based language model to achieve zero-shot domain adaption in dialogue abstractive summarization. In this work, we aim to explore the low-resource extractive summarization by disentangling context and pattern information.

Disentanglement Representation Learning Disentanglement representation has first been explored in computer vision to disentangle features such as color or rotation. Recently, a growing amount of work has been proposed to investigate learning disentangled representations in NLP tasks. Early works [53, 132, 61] follow a similar idea, and applied disentanglement representation learning on style/sentiment transferring. Later, researchers further extend its application to different topics such as cross-lingual transfer [157], negation and uncertainty learning [144], and fair classification[111]. Generally, there are mainly three types of approaches for disentanglement representation learning. A common approach [61] is to add an adversary that competes against the encoder trying to avoid learning certain types of attribute. Another approach [15, 20] is to adopt the mutual information theory, and attempt to minimize the mutual information upper bound between two disentangle representations. Recently, some researchers [21] propose a simpler approach by adding a set of regulizers to achieve disentanglement representation learning. Similar to cross-lingual transfer, in this work, we also aim to adopt disentanglement representation learning to domain transferring, but in the context of extractive summarization.

7.3 Model

In section 3.1, we would like to introduce the problem we face and three essential requirements that we need to address the problem. Our overall model is described

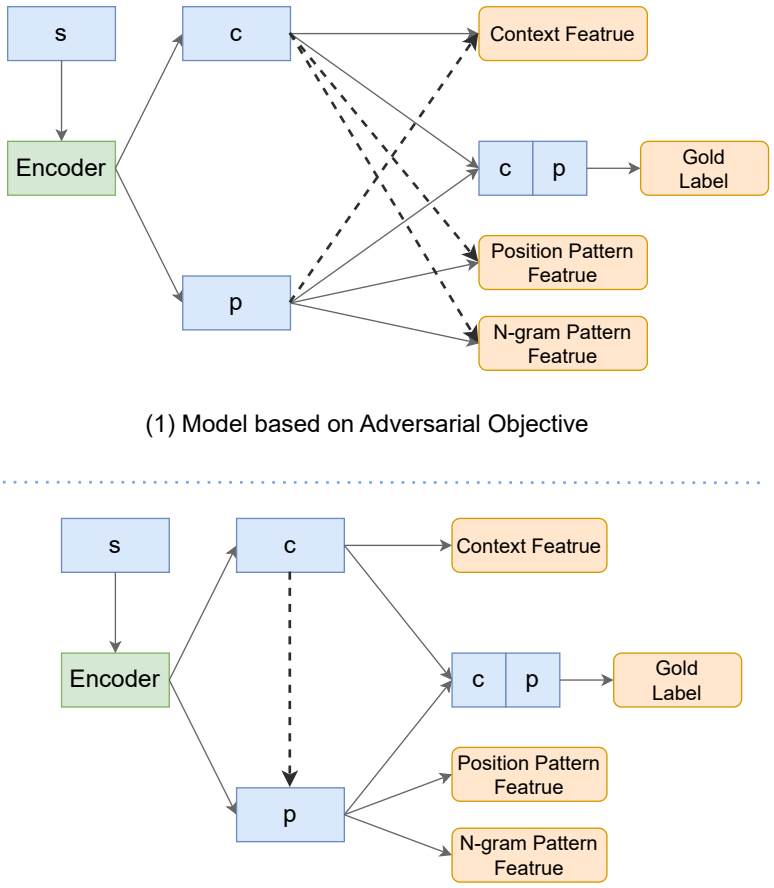

Figure 7.2: The framework of our proposed model. Part (1) shows the model based on the adversarial objective, while part (2) displays the one based on the MI minimization objective. The blue blocks refer to different sentence representations, the green blocks stand for the model components and the yellow blocks represent the target features. The solid lines represent normal classification loss, and the dashed lines stand for the discriminator loss plus the adversary loss.

in section 3.2 to 3.5. Then, we introduce the details of the training process and the application for low-resource extractive summarization in section 3.6 and 3.7.

7.3.1 Problem Statement

In this work, we disentangle the sentence representation for extractive summarization into two parts: context representation and pattern representation. To achieve this, we need to satisfy the following requirements for an effective disentanglement.

- The context and pattern representation need to have the ability to predict sentence importance and contribute to the extractive summarization.
- The context and pattern representation should be predictive of the corresponding ground-truth information. For example, the pattern representation of a sentence can predict its pattern feature such as its position.
- The context and pattern representation should lie in independent vector space, and one representation can not predict the corresponding ground-truth information of the other one.

7.3.2 Extractive Summarization Model

Given an input document containing n sentences $x = \{s_1, s_2, .., s_n\}$, we adopt a BERT to generate contextualized representations for each sentence. Since the output of BERT is grounded to tokens, we use a similar strategy with [92] to modify the input sequence of BERT. We insert a [cls] token at the beginning of each sentence and use the embedding of the [cls] token to represent its corresponding sentence. Considering our glob is to disentangle it to context and pattern representation, we add two additional multilayer perceptrons (MLP) that map the sentence representations generated by BERT to context representations c and pattern representations p. Here,

we collectively called the BERT and two MLP mappers encoder E. Then a sigmoid classifier F_{ext} takes the concatenation of both representations as input to predict a score y_i^e for sentence s_i, and the loss of the whole model is the binary classification loss of y_i^e against gold label t_i^e. Note that the gold label refers to the one-hot distribution of the oracle sentences (the sentence set that has the highest similarity with the reference summary). The loss is shown in the following:

$$y_i^e = F_{ext}(c_i; p_i) \tag{7.1}$$

$$l_{ext} = -\frac{1}{n}\sum_{i}^{n} t_i^e log(y_i^e) + (1 - t_i^e) log(1 - y_i^e) \tag{7.2}$$

This classification loss serves as our primary training objective for extractive summarization. Meanwhile, to better utilize the context representation and pattern representation in the low-resource setting, we expect the two disentangled representations can do extractive summarization independently. Hence, we add two similar classifiers that directly take context representation or pattern representation as input, and their losses are denoted as $l_{ext(c)}$ and $l_{ext(p)}$. Note that the gradients of the two classifiers are detached from the main model.

7.3.3 Learning Context Representation

The context representation c is expected to do extractive summarization using the context information. In addition to the extractive summarization loss, we add a multitask objective to ensure the context information is contained in it. The question that lies ahead is to define what "context" actually refers to. A widely accepted idea is that the effective context information in extractive summarization is salient words/phrases that repeat multiple times in the context. Inspired by this, given a sentence s_i, we propose to approximate the context information by predicting the

non-stop words existing in both s_i and its adjacent sentences. The distribution of these words on the vocabulary is considered as the context feature t_i^c for s_i.

We build a two-layer MLP classifier $F_{mul(c)}$ on the context representation c to predict the context feature, and the classifier is trained with cross-entropy loss against the ground-truth distribution:

$$l_{mul(c)} = -\frac{1}{n}\sum_{i}^{n}\sum_{j\in voc} t_{ij}^{c} log(y_{ij}^{c}) \tag{7.3}$$

where the *voc* stands for the vocabulary and $y_i^c = F_{mul(c)}(c_i)$ is the predicted context feature.

7.3.4 Learning Pattern Representation

The pattern representation p needs to predict both sentence importance and pattern-related features. In this paper, we mainly focus on the two types of pattern, position pattern and n-gram pattern, that contribute the most to extractive summarization. Position pattern refers to the position of the sentence in the document, which plays an important role in the news article summarization. We add a multitask objective that predicts the position of a sentence. In this case, the position pattern feature t_i^o is a one-hot vector with a length that is the same as the sentence number. N-gram pattern is another crucial factor that influences sentence importance, which represents the expressions/phrases that are commonly used for summaries. Inspired by [126], We count the frequencies of all n-grams that appear in the oracle sentences and select the top 500 as the n-gram pattern set. The glob of pattern representation is to predict whether a sentence contains any pattern from the pattern set, which is a binary classification problem.

Similarly, we also use two MLP classifiers on the pattern representation p to predict the pattern related feature:

$$l_{mul(p)} = -\frac{1}{n}\sum_{i}^{n} t_i^p log(y_i^p) + (1 - t_i^p) log(1 - y_i^p) \tag{7.4}$$

$$l_{mul(o)} = -\frac{1}{n}\sum_{i}^{n}\sum_{j}^{n} t_{ij}^o log(y_{ij}^o) \tag{7.5}$$

where $y_i^p = F_{mul(p)}(p_i)$ is the predicted n-gram pattern feature and $y_i^o = F_{mul(o)}(p_i)$ is the predicted position pattern feature.

7.3.5 Learning Disentangled Representation

Although the multitask objectives assist the model to learn context and pattern information in different latent spaces, they are not effective enough to ensure the independence between c and p. As shown in the Figure 7.2, we adopt two commonly used objectives for learning disentangled representation in this paper.

Adversarial Objective Considering one representation should be predictive of their corresponding information only, following [61], we add adversarial classifiers that try to predict the information related to the other one on both latent spaces, and the model is forced to structure the latent spaces such that the outputs of these adversarial classifiers are non-predictive. The adversarial objective is composed of two parts. The first part is the adversarial classifiers on each latent space for each type of non-target information. The second part is the adversarial loss aiming to maximize the entropy of the predicted distribution of the adversarial classifiers.

Taking the adversarial objective on the pattern space for example, we train a two-layer MLP classifier, context discriminator $F_{dis(c)}$, to predict whether it contains any context information. One thing that is worth noticing is that the gradients of these classifiers are not back-propagated to the encoder. In this case, the training of the context discriminator will not influence the encoder. Similar to equation (3) and (5), a cross-entropy loss is shown as follow, but with different input and parameters:

$$l_{dis(c)} = -\frac{1}{n}\sum_{i}^{n}\sum_{j \in voc} t_{ij}^{c} log(y_{ij}^{c}) \tag{7.6}$$

where $y_i^c = F_{dis(c)}(p_i)$ refers to the predicted context feature using pattern representation.

Then an adversarial loss is used to maximize the entropy of the output of context discriminator. Here, we only train the encoder with such adversarial loss and the parameters of the context discriminator are excluded.

$$l_{adv(c)} = -\frac{1}{n}\sum_{i}^{n}\sum_{j \in voc} y_{ij}^{c} log(y_{ij}^{c}) \tag{7.7}$$

We also impose the n-gram pattern discriminator and position pattern discriminator to disentangle the pattern information from the context space. These two adversarial objectives follow nearly the same way as the mentioned one and their corresponding loss are denoted as $l_{dis(p)}$, $l_{dis(o)}$, $l_{adv(p)}$ and $l_{adv(o)}$.

MI Minimization Objective Mutual information (MI) is a natural measure of the independence between two variables. Inspired by the previous works [15], minimizing the upper-bound estimate of the mutual information (MI) between two latent spaces is an effective way to disentangle them. Following the Contrastive Learning Upper-Bound (CLUB) estimate of the MI [15], we firstly train a neural network M that aims to estimate pattern representation by taking context representation as input:

$$l_{map} = \frac{1}{n}\sum_{i}^{n} kl(M(c_i), p_i) \tag{7.8}$$

where kl stands for the Kullback–Leibler divergence. Just like the discriminator in the adversarial objective, we fix the parameters of the encoder when we train the neural network M with this loss.

We minimize the Mutual information between the two latent spaces by minimizing

the following equation:

$$l_{mi} = \frac{1}{n}\sum_{i}^{n} kl(M(p_i), c_i) - kl(M(p_i), c_k) \tag{7.9}$$

where k is selected uniformly from indices $\{1, ..., n\}$. Here, the optimization is only performed with parameters of the encoder E.

7.3.6 Training Strategy

The loss of our model mainly consists of two parts, the losses that update the discriminator (for MI Objective, it is M) and the main loss (all the other losses). In the training process, for each batch, we first optimize the discriminator by $l_{dis(c)}$, $l_{dis(p)}$ and $l_{dis(o)}$ with a weight λ_{dis} (for MI Objective, it is l_{map}), and then optimize the encoder and all other classifiers with the main loss. The main loss L_{all} for our model comprises three types of terms: the extractive summarization objectives, the context/pattern feature learning objectives and adversarial objectives (for MI Objective, it is l_{mi}), given by

$$\begin{aligned} l_{all} = {} & l_{ext} + l_{ext(c)} + l_{ext(p)} + \\ & \lambda_{mul} l_{mul(c)} - \lambda_{adv} l_{adv(c)} + \\ & \lambda_{mul} l_{mul(p)} - \lambda_{adv} l_{adv(p)} + \\ & \lambda_{mul} l_{mul(o)} - \lambda_{adv} l_{adv(o)} \end{aligned} \tag{7.10}$$

The checkpoint selection strategy and hyperparameter searching are also crucial for model training. Considering the glob of our model is to effectively utilize the context information in the target dataset rather than achieve the best performance on the source dataset, we follow two rules: (1) The disentanglement is successful (based on the training log); (2) We select the checkpoint with the best performance when using context representation on the validation set. In the experiment, the weights are

Dataset	Type	Domain	Size	Source Len	Target Len
QMsum	dialogue	meeting	1257/272/281	9070	70
arXiv	document	science	202914/6436/6440	6030	273
CNN/DM	document	news	287227/13368/11490	766	53

Table 7.2: The statistics and comparison of the datasets. The Source Len and Target Len stand for the token number of the source document and the summaries.

$\lambda_{mul} = 1, \lambda_{adv} = 1, \lambda_{dis} = 3$

7.3.7 Application in Low-Resource Setting

After we train the model on a source dataset, we can transfer it to a target dataset with limited data. Considering the pattern information in the source dataset may be misleading in a target dataset, we use the context representation to do the extractive summarization in the zero-shot setting. As for the few-shot setting, the data samples from the target dataset provide the model a chance to accomplish a quick adjustment on its pattern information. In this case, we choose to fine-tune the pattern-related parameters with the given samples to select useful patterns for the target dataset.

7.4 Experiment

7.4.1 Experiment Details

Dataset: We evaluate our proposed methods in three English datasets from different domains. The detailed information and comparison are shown in Table 7.2. **arXiv** [19] collects academic articles from arXiv.org as source documents and uses the abstracts of these articles as the target summaries. **QMSum** [189] is one of the benchmark datasets for dialogue summarization. Considering the QMSum dataset

contains both data samples for normal text summarization and query-focused summarization, we only use the data samples that contain no query. Meanwhile, the number of training data in QMSum is relatively small, so we only use it for testing. **CNN/DailyMail** [105] is the classic dataset for news summarization. It is also known for suffering from lead bias, where the summaries that consist of the lead three sentences can achieve a relatively good performance.

Model Details: In this work, we adopt BERT-base as the encoder of our model. Our implementation is based on Transformers from Hugging Face. In the training, the learning rate is set to 2e-5, and the batch size is set to 16. We conduct the validation for every 2000 steps and train the model for a maximum of 30000 steps. We truncate all the input documents to 500 tokens. For the long-input summarization dataset such as arXiv and QMSum, we split the original document into multiple chunks and generate extractive summarization scores for the sentences in each chunk independently. In all experiments, we select 3 sentences for CNN/DM and 6 sentences for arXiv and QMSum. Following previous works, we also adopt the trigram blocking trick during inference.

Evaluation Metric: We adopt Rouge as our evaluation metric [86] including Rouge-1 (R-1), Rouge-2 (R-2), and Rouge-L (R-L) as evaluation metrics. In practice, we use a python wrapper pyrouge to apply the classic Rouge 1.5.5.

7.4.2 Comparison

We compare our method with some commonly used baselines and previous state-of-the-art methods designed for low-resource text summarization. There are three types of methods: unsupervised baselines, comparable unsupervised models based on domain transferring or pretraining, and other reference models that are not directly comparable.

Unsupervised Baselines Lead-n aims to select the lead sentences in the docu-

ment as the summaries, and it always plays an important role in the news summarization dataset that heavily relies on the position pattern information such as CNN/DailyMail. We also show the result of two strong unsupervised baselines TextRank [101] and LexRank [32].

Comparable Models AdaptSum [167] focuses on one-to-one domain adaption in text summarization. It proposes a Source Domain Pre-Training (SDPT) strategy that first fine-tunes a pretrained model on the source domain and then applies it to the target domain. Another research [33] also proposes a similar method with it and further extends with a data augmentation method. However, this data augmentation method requires the pattern information from the target dataset and is not comparable with our model.

Other Reference Models We display the result of BERTSum [92] training on the full target dataset, which can be considered as the upper bound of our model.

7.4.3 Experiment Results

Zero-shot application We first evaluate the performance of our model in the zero-shot setting in Table 7.3 and Table 7.4, where the information of the target dataset is totally unknown. Here, we display the two variants of the model, Our_adv using the adversarial objective and Our_mi adopting the MI minimization objective. Based on the results, we have the following observation. Firstly, Our_adv achieves the best result in most cases. This indicates the effectiveness of context information in the zero-shot setting. Meanwhile, we also observe that Our_mi obtains a lower performance compared to Our_adv. Further investigation of the training process shows that using the MI minimization objective is more difficult to disentangle pattern and context information. We think the reason is that the two types of information are not naturally disentangled and are optimized by the same extractive summarization objectives. In this case, the model requires more clear guidance to achieve the disen-

To arXiv	R-1	R-2	R-L
Lead*	33.66	8.94	22.19
TextRank*	24.38	10.57	22.18
LexRank*	33.85	10.73	28.99
AdaptSum	36.28	9.17	32.26
Our_adv	**37.03**	**9.64**	**33.03**
Our_mi	36.89	9.44	32.75
BERT(full)	41.04	13.92	36.61
To QMSum	R-1	R-2	R-L
Lead-5*	12.84	1.69	9.17
TextRank*	16.27	2.69	15.41
AdaptSum	26.41	4.67	23.80
Our_adv	**27.27**	**5.11**	**24.91**
Our_mi	26.71	4.49	24.18

Table 7.3: The results of models trained on CNN/DM in zero-shot setting.

tanglement.

Analysis of context and pattern information To understand the influence of both context and pattern information on the target dataset, we compare the performance of using context representation, using pattern representation, and using both representations in Table 7.5. Considering the huge gap in the pattern between the two datasets, it is not surprising that using the pattern representation achieves the worst result. Meanwhile, its misleading information also pulls down the results of using both representations. We also display the position distribution of extracted sentences on arXiv using the model trained on CNN/DM in Figure 7.3. Since CNN/DM is known for its lead bias, the pattern latent space learned on it inevitably tend to select the lead sentences. This trend further dominates the situation when using both representations. As for using context representation alone, the lead bias is relatively

To CNN/DM	R-1	R-2	R-L
Lead*	40.49	17.66	36.75
TextRank*	33.85	13.61	30.14
LexRank*	34.68	12.82	31.12
AdaptSum	37.21	15.07	33.64
Our_adv	**38.37**	**15.81**	**34.64**
Our_mi	38.05	15.74	34.37
BERT(full)	42.83	19.82	39.13
To QMSum	R-1	R-2	R-L
Lead-5*	12.84	1.69	9.17
TextRank*	16.27	2.69	15.41
AdaptSum	**28.28**	**4.78**	**25.28**
Our_adv	28.01	4.74	24.94
Our_mi	27.63	4.66	25.13

Table 7.4: The results of models trained on arXiv in zero-shot setting.

weaker.

Few-shot application Directly using the pattern information in an unsuitable dataset leads to a decrease in the model performance. However, this does not mean the pattern representation is completely useless. In the few-shot setting, we can obtain some information from the target dataset and fine-tune the pattern latent space. To simulate this situation, for each target dataset, we build its few-shot version by randomly taking 50 data samples from its original training set and splitting it into 25 training data and 25 validation data. Here, despite our proposed model and AdaptSum, we also show the result of directly fine-tuning a BERTSum model on the limited data. In Table 7.6, the performance of all models is improved with the help of the limited data, while the gap between Our_adv and AdaptSum still exists. This shows our model is capable of selecting the effective pattern information for the target dataset

arXiv to CNN/DM	R-1	R-2	R-L
Both	37.71	15.28	33.98
Context	38.37	15.81	34.64
Pattern	36.65	14.39	32.95

Table 7.5: The results on CNN/DM when using context/pattern representation.

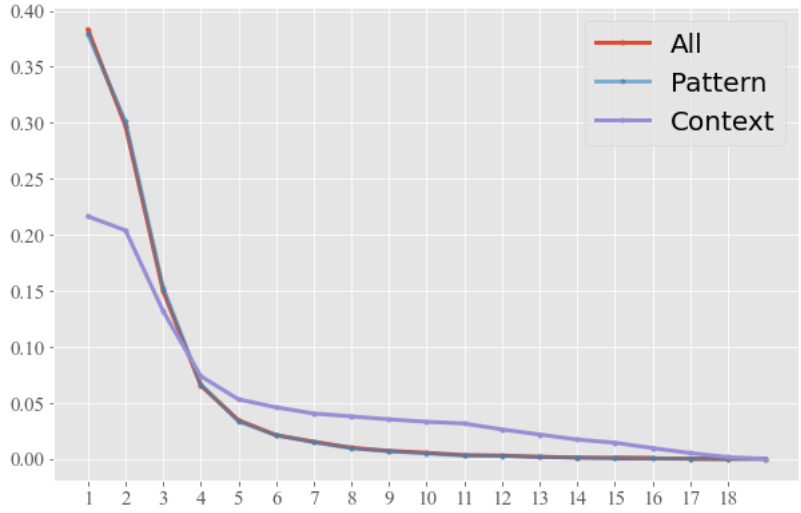

Figure 7.3: The predicted sentences position distribution on arXiv when using context/pattern representation.

and preserving its advantages on context information.

Ablation study We further conduct an ablation study. Firstly, we remove the adversary objectives from our model (–adv loss), which means the model can only learn the disentangled representation by approximating context/pattern features. Then we further remove the multitask objectives (–aux loss). In this case, the main difference between this model and the AdaptSum is that our classifier contains more parameters. Here we compare the result of only using context representations in the zero-shot setting. As shown in Table 7.7, we find that removing the adversary objectives leads to a clear performance drop. This suggests that using the adversary objectives alone is far enough to disentangle the context and pattern information. We also find that the

	arXiv to CNN/DM			**CNN/DM to arXiv**		
	Rouge-1	Rouge-2	Rouge-L	Rouge-1	Rouge-2	Rouge-L
BERT	37.36	15.21	33.86	32.55	7.68	28.92
AdaptSum	38.21	15.91	34.60	39.12	11.25	34.78
Our_adv	39.27	16.56	35.47	39.39	11.35	34.97

Table 7.6: The results on arXiv and CNN/DM in few-shot setting.

arXiv to CNNDM	R-1	R-2	R-L
Our_adv	38.37	15.81	34.64
–adv loss	37.72	15.44	34.05
–aux loss	37.11	14.75	33.44

Table 7.7: The ablation study in the zero-shot setting.

result of model "–aux loss" is similar to the result of AdaptSum in Table 7.4, which shows the improvement of our model is not brought by the additional parameters.

7.4.4 Visualization

To have a more direct observation, we visualize the context and pattern representations by using the t-SNE algorithm [143] to reduce them to two dimensions in Figure 7.4. These representations are taken from 1000 randomly sampled examples from CNN/DM using the model trained on arXiv. Each point refers to a context/pattern representation of a sentence from the source document. The figure shows that the context latent space and pattern latent space are well separated into two parts, which supports the effectiveness of our model in disentangling context and pattern information.

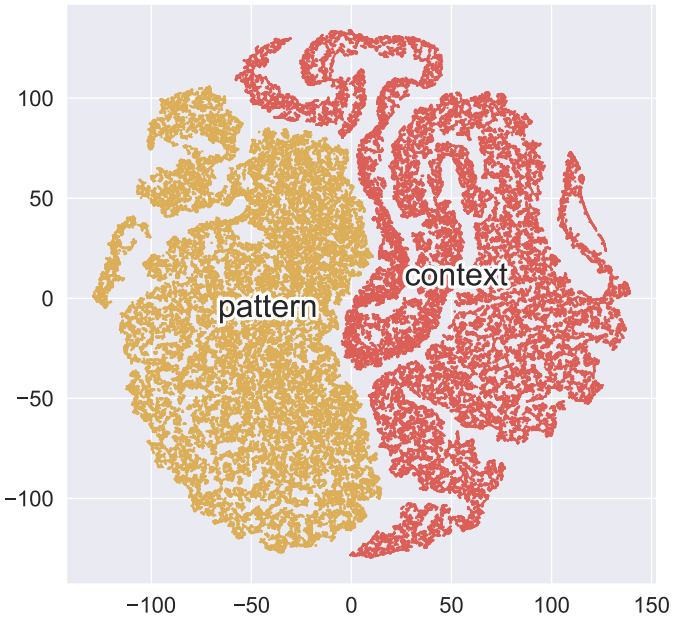

Figure 7.4: Visualization of context and pattern representations.

7.5 Discussion and Conclusion

We adopt two types of representative pattern information, position pattern, and n-gram pattern, but it does not mean they cover all effective pattern information. In this case, the way to efficiently include all types of pattern information is still an important problem. Secondly, we do not put too much effort into investigating the influence of different feature forms (pattern feature and context feature) for the multitask objectives. Thirdly, due to the limitation of time and paper length, we only evaluate our method in three representative domains. Other domains such as review summarization (Reddit [146]) and legislation document summarization (BillSum [66]) are also worth exploring.

In this paper, we propose a novel extractive summarization model that aims to improve the generalization ability in low-resource setting. It disentangles the sentence representation to context and pattern representation and utilize the context infor-

mation to reduce the influence of domain-specific pattern information during model transferring. The experiment suggests the ability of our model in the disentanglement, and it also supports the claim that the context information tends to have better generalization ability facing the dataset from a different domain. With in the era of large language model, when LLM can successfully handle general summarization, it is more important to explore abstractive summarization problems in vertical area such as dialogue or medical. Multi-model summarization and multi-document summarization are also potential directions. How to further extend this idea to these problems becomes a challenge.

Chapter 8

Conclusions and Future Work

With the growing numbers of online news and documents, people are overwhelmed by the great amount of information that exists in today's society. How to organize and summarize these information in an efficient way becomes a challenge. Therefore, automatic summarization models have been proposed to cope with this situation and free human labor from such tedious work.

In this thesis, we investigate the problems in text summarization. To fill the gap left by the previous mainstream research paradigm "news data + general model", we propose three research problems, i.e. (1) How to use the nature features of news articles to improve the general summarization model on news summarization? (2) How to extend the current general summarization models to summarization tasks/domains where these models cannot be directly applied? (3) How to utilize the large-scale data in news summarization to assist summarization tasks/domains with insufficient data? Based on the transformer model, we propose a series of models targeting at specific summarization domains or tasks to address these problems. Our proposed models obtain significant improvements compared with the state-of-the-art studies and achieve our research goal.

8.1 Summary of Contributions

Work 1 & 2: News Summarization Via Event Information

- Different from previous sentence-level extractive summarization approaches, we introduce a novel approach to extractive summarization by extracting event-level semantic units.
- We are the first to utilize an event graph to solve the sentence fusion problem in text summarization, introducing the use of an event graph with graph flow attention to capture both node representations and structural information.
- The two works consist of a complete framework for event-aware news summarization framework by first extracting salient events and then fusing them into abstractive summary sentences.
- The experiments and further ablation studies on news datasets demonstrate the effectiveness of event-level information in news summarization.

Work 3: Long-Input Summarization Via Context-Aware Extract-Generate Framework

- We first investigate the information loss in the extract-generate framework for long-input summarization and propose to focus on context information preservation.
- We develop a novel approach to generate the context prompts based on the interpretation of the extractor. These context-related text spans are capable of transferring the context information from the extractor to the generator without decreasing the efficiency of the model.
- The experiments further show its effectiveness and efficiency in capturing and

preserving the context information for the long-input summarization. And it is flexible enough to apply on most extract-generate frameworks.

Work 4 & 5: Low-Resource Summarization via Knowledge Transferring

- We propose prefix-merging, an effective approach for transferring and integrating task knowledge from multiple auxiliary tasks to a target task with limited data. In the context of query-focused summarization, integrating text summarization and QA, our approach outperforms the traditional approach of fine-tuning.

- We further explore prompt-based approaches in multi-task situations, providing a potential solution for the few-shot learning in complex tasks that can be integrated by the basic tasks.

- We develop a novel extractive summarization model that aims to improve the generalization ability in low-resource settings.

- We first investigate the context and pattern information in the view of knowledge transferring in extractive summarization, and develop a disentangled representation learning model to detach them.

- The experiments suggest the great potential of the knowledge contained in the large-scale news summarization data in improving the summarization system in other tasks or domains.

8.2 Future Work

We discuss the following potential future directions of our work in this thesis.

- In Chapter 3 and Chapter 4, we discuss the usage of event information in improving the news summarization model. The very essence here is to utilize the domain-specific structure or features to do the summarization. Except the news summarization, there also exist different domains such as dialogue or medical reports with unique characteristics. For example, the dialogue records contain the role or action information that might be effective for text summarization [184, 12]. In this case, providing customized models for these domains is a meaningful direction. Moreover, this also brings a question of whether there is a unified framework that can quickly adapt to these various domains together with their uniqueness. A similar framework has been proposed in question answering [62], and we believe it is worth further exploration.

- In Chapter 5, we bypass the information bottleneck of the pretrained language model through an efficient divide-and-conquer strategy. Another potential solution for this problem is reducing the memory cost of the transformer itself. The main idea for this type of model [6, 173, 64] is to replace full-connected attention with sparse attention, which leads to much lower memory consumption. However, when summarizing text like books with hundreds of thousands of tokens, the current two approaches are far from enough in both feasibility and efficiency. A potential solution is to combine both types of approaches. The challenge is how to maintain the dependency across the input with both sparse attention and truncated text. A hierarchical or recurrence structure that combines the two approaches might be a solution.

- Recently, large language models (LLM) such as GPT-3 [7] or LLaMA [142] have dominated the field of natural language process. In this case, combining text summarization with LLM is also a crucial direction for future research. In Chapter 6 and Chapter 7, we introduce knowledge transferring for text summarization in tasks/domains with limited data. LLMs are native few-shot learners that have strong abilities in generalization and in-context learning. Hence,

LLM based summarization system can be a solution for summarization problems in these tasks/domains. Moreover, the evaluation for text summarization has been a challenging problem, since a summary can often be expressed in multiple ways. Considering human evaluation is always expensive, the evaluation based on LLM is a good substitute for it. LLM evaluator is capable of providing evaluation in different views including coherence, diversity, etc, far more than the traditional overlapping-based metrics. In turn, the various metrics can promote the development of the summarization system. Overall, the usage of LLM in text summarization is an inevitable and exciting problem to be solved.

References

[1] Samira Abnar and Willem Zuidema. Quantifying attention flow in transformers. *arXiv preprint arXiv:2005.00928*, 2020.

[2] Sanghwan Bae, Taeuk Kim, Jihoon Kim, and Sang-goo Lee. Summary level training of sentence rewriting for abstractive summarization. *arXiv preprint arXiv:1909.08752*, 2019.

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

[4] Ahsaas Bajaj, Pavitra Dangati, Kalpesh Krishna, Pradhiksha Ashok Kumar, Rheeya Uppaal, Bradford Windsor, Eliot Brenner, Dominic Dotterrer, Rajarshi Das, and Andrew McCallum. Long document summarization in a low resource setting using pretrained language models. *arXiv preprint arXiv:2103.00751*, 2021.

[5] Regina Barzilay, Kathleen McKeown, and Michael Elhadad. Information fusion in the context of multi-document summarization. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics*, pages 550–557, 1999.

[6] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[7] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. 2020.

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

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

[10] Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. Faithful to the original: Fact aware neural abstractive summarization. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[11] Ying-Lang Chang and Jen-Tzung Chien. Latent dirichlet learning for document summarization. In *2009 IEEE international conference on acoustics, speech and signal processing*, pages 1689–1692. IEEE, 2009.

[12] Jiaao Chen and Diyi Yang. Multi-view sequence-to-sequence models with conversational structure for abstractive dialogue summarization. *arXiv preprint arXiv:2010.01672*, 2020.

[13] Mingda Chen, Zewei Chu, Sam Wiseman, and Kevin Gimpel. Summscreen: A dataset for abstractive screenplay summarization. *arXiv preprint arXiv:2104.07091*, 2021.

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

[15] Pengyu Cheng, Martin Renqiang Min, Dinghan Shen, Christopher Malon, Yizhe Zhang, Yitong Li, and Lawrence Carin. Improving disentangled text representation learning with information-theoretic guidance. *arXiv preprint arXiv:2006.00693*, 2020.

[16] Jackie Chi Kit Cheung and Gerald Penn. Unsupervised sentence enhancement for automatic summarization. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 775–786, 2014.

[17] 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.

[18] Sumit Chopra, Michael Auli, and Alexander M Rush. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 93–98, 2016.

[19] Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. A discourse-aware attention model for abstractive summarization of long documents. *arXiv preprint arXiv:1804.05685*, 2018.

[20] Pierre Colombo, Chloe Clavel, and Pablo Piantanida. A novel estimator of mutual information for learning to disentangle textual representations. *arXiv preprint arXiv:2105.02685*, 2021.

[21] Pierre Colombo, Guillaume Staerman, Nathan Noiry, and Pablo Piantanida. Learning disentangled textual representations via statistical measures of similarity. *arXiv preprint arXiv:2205.03589*, 2022.

[22] Peng Cui and Le Hu. Sliding selector network with dynamic memory for extractive summarization of long documents. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5881–5891, 2021.

[23] Hoa Trang Dang. Duc 2005: Evaluation of question-focused summarization systems. In *Proceedings of the Workshop on Task-Focused Summarization and Question Answering*, pages 48–55, 2006.

[24] Naomi Daniel, Dragomir Radev, and Timothy Allison. Sub-event based multi-document summarization. In *Proceedings of the HLT-NAACL 03 Text Summarization Workshop*, pages 9–16, 2003.

[25] Yang Deng, Wenxuan Zhang, and Wai Lam. Multi-hop inference for question-driven summarization. *arXiv preprint arXiv:2010.03738*, 2020.

[26] Daniel Deutsch and Dan Roth. Summary cloze: A new task for content selection in topic-focused summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3720–3729, 2019.

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

[28] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, pages 13042–13054, 2019.

[29] Yue Dong, Yikang Shen, Eric Crawford, Herke van Hoof, and Jackie Chi Kit Cheung. Banditsum: Extractive summarization as a contextual bandit. *arXiv preprint arXiv:1809.09672*, 2018.

[30] Zi-Yi Dou, Pengfei Liu, Hiroaki Hayashi, Zhengbao Jiang, and Graham Neubig. Gsum: A general framework for guided neural abstractive summarization. *arXiv preprint arXiv:2010.08014*, 2020.

[31] Micha Elsner and Deepak Santhanam. Learning to fuse disparate sentences. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 54–63, 2011.

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

[33] Alexander R Fabbri, Simeng Han, Haoyuan Li, Haoran Li, Marjan Ghazvininejad, Shafiq Joty, Dragomir Radev, and Yashar Mehdad. Improving zero and few-shot abstractive summarization with intermediate fine-tuning and data augmentation. *arXiv preprint arXiv:2010.12836*, 2020.

[34] Alexander R Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir R Radev. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. *arXiv preprint arXiv:1906.01749*, 2019.

[35] Hanyin Fang, Weiming Lu, Fei Wu, Yin Zhang, Xindi Shang, Jian Shao, and Yueting Zhuang. Topic aspect-oriented summarization via group selection. *Neurocomputing*, 149:1613–1619, 2015.

[36] Yuwei Fang, Siqi Sun, Zhe Gan, Rohit Pillai, Shuohang Wang, and Jingjing Liu. Hierarchical graph network for multi-hop question answering. *arXiv preprint arXiv:1911.03631*, 2019.

[37] Xiachong Feng, Xiaocheng Feng, Bing Qin, Xinwei Geng, and Ting Liu. Dialogue discourse-aware graph convolutional networks for abstractive meeting summarization. *arXiv preprint arXiv:2012.03502*, 17, 2020.

[38] Patrick Fernandes, Miltiadis Allamanis, and Marc Brockschmidt. Structured neural summarization. *arXiv preprint arXiv:1811.01824*, 2018.

[39] Katja Filippova and Michael Strube. Sentence fusion via dependency graph compression. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 177–185, 2008.

[40] Joseph L Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378, 1971.

[41] Lea Frermann and Alexandre Klementiev. Inducing document structure for aspect-based summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6263–6273, 2019.

[42] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. In *International conference on machine learning*, pages 1243–1252. PMLR, 2017.

[43] Sebastian Gehrmann, Yuntian Deng, and Alexander M Rush. Bottom-up abstractive summarization. *arXiv preprint arXiv:1808.10792*, 2018.

[44] Mor Geva, Eric Malmi, Idan Szpektor, and Jonathan Berant. Discofuse: A large-scale dataset for discourse-based sentence fusion. *arXiv preprint arXiv:1902.10526*, 2019.

[45] Alexios Gidiotis and Grigorios Tsoumakas. A divide-and-conquer approach to the summarization of long documents. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:3029–3040, 2020.

[46] Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. Samsum corpus: A human-annotated dialogue dataset for abstractive summarization. *arXiv preprint arXiv:1911.12237*, 2019.

[47] Max Grusky, Mor Naaman, and Yoav Artzi. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. *arXiv preprint arXiv:1804.11283*, 2018.

[48] Tahmid Hasan, Abhik Bhattacharjee, Md Saiful Islam, Kazi Samin, Yuan-Fang Li, Yong-Bin Kang, M Sohel Rahman, and Rifat Shahriyar. Xl-sum: Large-scale multilingual abstractive summarization for 44 languages. *arXiv preprint arXiv:2106.13822*, 2021.

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

[50] Johan Hasselqvist, Niklas Helmertz, and Mikael Kågebäck. Query-based abstractive summarization using neural networks. *arXiv preprint arXiv:1712.06100*, 2017.

[51] Karl Moritz Hermann, Tomáš Kočiskỳ, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. *arXiv preprint arXiv:1506.03340*, 2015.

[52] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.

[53] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. Toward controlled generation of text. In *International conference on machine learning*, pages 1587–1596. PMLR, 2017.

[54] Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. Efficient attentions for long document summarization. *arXiv preprint arXiv:2104.02112*, 2021.

[55] Luyang Huang, Lingfei Wu, and Lu Wang. Knowledge graph-augmented abstractive summarization with semantic-driven cloze reward. *arXiv preprint arXiv:2005.01159*, 2020.

[56] Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. On using very large target vocabulary for neural machine translation. *arXiv preprint arXiv:1412.2007*, 2014.

[57] Heng Ji, Ralph Grishman, Hoa Trang Dang, Kira Griffitt, and Joe Ellis. Overview of the tac 2010 knowledge base population track. In *Third text analysis conference (TAC 2010)*, volume 3, pages 3–3, 2010.

[58] Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438, 2020.

[59] Hanqi Jin, Tianming Wang, and Xiaojun Wan. Multi-granularity interaction network for extractive and abstractive multi-document summarization. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 6244–6254, 2020.

[60] Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William W Cohen, and Xinghua Lu. Pubmedqa: A dataset for biomedical research question answering. *arXiv preprint arXiv:1909.06146*, 2019.

[61] Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. Disentangled representation learning for non-parallel text style transfer. *arXiv preprint arXiv:1808.04339*, 2018.

[62] Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. Unifiedqa: Crossing format boundaries with a single qa system. *arXiv preprint arXiv:2005.00700*, 2020.

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

[64] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*, 2020.

[65] Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. Text generation from knowledge graphs with graph transformers. *arXiv preprint arXiv:1904.02342*, 2019.

[66] Anastassia Kornilova and Vlad Eidelman. Billsum: A corpus for automatic summarization of us legislation. *arXiv preprint arXiv:1910.00523*, 2019.

[67] Mahnaz Koupaee and William Yang Wang. Wikihow: A large scale text summarization dataset. *arXiv preprint arXiv:1810.09305*, 2018.

[68] Wessel Kraaij, Thomas Hain, Mike Lincoln, and Wilfried Post. The ami meeting corpus. In *Proc. International Conference on Methods and Techniques in Behavioral Research*, 2005.

[69] Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, 2020.

[70] Md Tahmid Rahman Laskar, Enamul Hoque, and Jimmy Xiangji Huang. Wsl-ds: Weakly supervised learning with distant supervision for query focused multi-document abstractive summarization. *arXiv preprint arXiv:2011.01421*, 2020.

[71] Logan Lebanoff, Franck Dernoncourt, Doo Soon Kim, Lidan Wang, Walter Chang, and Fei Liu. Learning to fuse sentences with transformers for summarization. *arXiv preprint arXiv:2010.03726*, 2020.

[72] Logan Lebanoff, John Muchovej, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. Analyzing sentence fusion in abstractive summarization. *arXiv preprint arXiv:1910.00203*, 2019.

[73] Logan Lebanoff, John Muchovej, Franck Dernoncourt, Doo Soon Kim, Lidan Wang, Walter Chang, and Fei Liu. Understanding points of correspondence between sentences for abstractive summarization. *arXiv preprint arXiv:2006.05621*, 2020.

[74] Logan Lebanoff, Kaiqiang Song, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. Scoring sentence singletons and pairs for abstractive summarization. *arXiv preprint arXiv:1906.00077*, 2019.

[75] Logan Lebanoff, Kaiqiang Song, and Fei Liu. Adapting the neural encoder-decoder framework from single to multi-document summarization. *arXiv preprint arXiv:1808.06218*, 2018.

[76] Katherine Lee, Orhan Firat, Ashish Agarwal, Clara Fannjiang, and David Sussillo. Hallucinations in neural machine translation. 2018.

[77] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[78] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising

sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[79] 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.

[80] Peng Li, Yinglin Wang, Wei Gao, and Jing Jiang. Generating aspect-oriented multi-document summarization with event-aspect model. Association for Computational Linguistics, 2011.

[81] Wei Li, Xinyan Xiao, Jiachen Liu, Hua Wu, Haifeng Wang, and Junping Du. Leveraging graph to improve abstractive multi-document summarization. *arXiv preprint arXiv:2005.10043*, 2020.

[82] Wenjie Li, Mingli Wu, Qin Lu, Wei Xu, and Chunfa Yuan. Extractive summarization using inter-and intra-event relevance. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 369–376, 2006.

[83] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[84] Yanran Li and Sujian Li. Query-focused multi-document summarization: Combining a topic model with graph-based semi-supervised learning. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1197–1207, 2014.

[85] Zhenwen Li, Wenhao Wu, and Sujian Li. Composing elementary discourse units in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6191–6196, 2020.

[86] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[87] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[88] Jimmy Lin, Nitin Madnani, and Bonnie Dorr. Putting the user in the loop: interactive maximal marginal relevance for query-focused summarization. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 305–308, 2010.

[89] Zhaojiang Lin, Andrea Madotto, and Pascale Fung. Exploring versatile generative language model via parameter-efficient transfer learning. *arXiv preprint arXiv:2004.03829*, 2020.

[90] Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A Smith. Toward abstractive summarization using semantic representations. *arXiv preprint arXiv:1805.10399*, 2018.

[91] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too. *arXiv preprint arXiv:2103.10385*, 2021.

[92] Yang Liu and Mirella Lapata. Text summarization with pretrained encoders. *arXiv preprint arXiv:1908.08345*, 2019.

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

[94] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[95] Yixin Liu, Pengfei Liu, Dragomir Radev, and Graham Neubig. Brio: Bringing order to abstractive summarization. *arXiv preprint arXiv:2203.16804*, 2022.

[96] Elena Lloret, María Teresa Romá-Ferri, and Manuel Palomar. Compendium: A text summarization system for generating abstracts of research papers. *Data & Knowledge Engineering*, 88:164–175, 2013.

[97] Yuning Mao, Yanru Qu, Yiqing Xie, Xiang Ren, and Jiawei Han. Multi-document summarization with maximal marginal relevance-guided reinforcement learning. *arXiv preprint arXiv:2010.00117*, 2020.

[98] Ziming Mao, Chen Henry Wu, Ansong Ni, Yusen Zhang, Rui Zhang, Tao Yu, Budhaditya Deb, Chenguang Zhu, Ahmed H Awadallah, and Dragomir Radev. Dyle: Dynamic latent extraction for abstractive long-input summarization. *arXiv preprint arXiv:2110.08168*, 2021.

[99] Erwin Marsi and Emiel Krahmer. Explorations in sentence fusion. In *Proceedings of the Tenth European Workshop on Natural Language Generation (ENLG-05)*, 2005.

[100] Yashar Mehdad, Giuseppe Carenini, Frank Tompa, and Raymond Ng. Abstractive meeting summarization with entailment and fusion. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 136–146, 2013.

[101] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411, 2004.

[102] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Interspeech*, volume 2, pages 1045–1048. Makuhari, 2010.

[103] Amit Moryossef, Yoav Goldberg, and Ido Dagan. Step-by-step: Separating planning from realization in neural data-to-text generation. *arXiv preprint arXiv:1904.03396*, 2019.

[104] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[105] 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.

[106] Shashi Narayan, Shay B Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *arXiv preprint arXiv:1808.08745*, 2018.

[107] Shashi Narayan, Shay B Cohen, and Mirella Lapata. Ranking sentences for extractive summarization with reinforcement learning. *arXiv preprint arXiv:1802.08636*, 2018.

[108] Shashi Narayan, Joshua Maynez, Jakub Adamek, Daniele Pighin, Blaž Bratanič, and Ryan McDonald. Stepwise extractive summarization and planning with structured transformers. *arXiv preprint arXiv:2010.02744*, 2020.

[109] Preksha Nema, Mitesh Khapra, Anirban Laha, and Balaraman Ravindran. Diversity driven attention model for query-based abstractive summarization. *arXiv preprint arXiv:1704.08300*, 2017.

[110] 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 of the Association for Computational Linguistics*, pages 311–318, 2002.

[111] Sungho Park, Sunhee Hwang, Dohyung Kim, and Hyeran Byun. Learning disentangled representation for fair facial attribute classification via fairness-aware information alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 2403–2411, 2021.

[112] Ramakanth Pasunuru and Mohit Bansal. Multi-reward reinforced summarization with saliency and entailment. *arXiv preprint arXiv:1804.06451*, 2018.

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

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

[115] Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*, 2019.

[116] 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)*, pages 892–901, 2014.

[117] Jonathan Pilault, Raymond Li, Sandeep Subramanian, and Christopher Pal. On extractive and abstractive neural document summarization with transformer language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9308–9319, 2020.

[118] Guanghui Qin and Jason Eisner. Learning how to ask: Querying lms with mixtures of soft prompts. *arXiv preprint arXiv:2104.06599*, 2021.

[119] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[120] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

[121] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

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

[123] Tobias Rohde, Xiaoxia Wu, and Yinhan Liu. Hierarchical learning for generation with long source sequences. *arXiv preprint arXiv:2104.07545*, 2021.

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

[125] Itsumi Saito, Kyosuke Nishida, Kosuke Nishida, and Junji Tomita. Abstractive summarization with combination of pre-trained sequence-to-sequence and saliency models. *arXiv preprint arXiv:2003.13028*, 2020.

[126] Nikita Salkar, Thomas Trikalinos, Byron C Wallace, and Ani Nenkova. Self-repetition in abstractive neural summarizers. *arXiv preprint arXiv:2210.08145*, 2022.

[127] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[128] Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. Discriminative adversarial search for abstractive summarization. *arXiv preprint arXiv:2002.10375*, 2020.

[129] Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. Mlsum: The multilingual summarization corpus. *arXiv preprint arXiv:2004.14900*, 2020.

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

[131] Chao Shen and Tao Li. Learning to rank for query-focused multi-document summarization. In *2011 IEEE 11th International Conference on Data Mining*, pages 626–634. IEEE, 2011.

[132] Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. Style transfer from non-parallel text by cross-alignment. *Advances in neural information processing systems*, 30, 2017.

[133] Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*, 2020.

[134] Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. Supervised open information extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 885–895, 2018.

[135] Dan Su, Yan Xu, Tiezheng Yu, Farhad Bin Siddique, Elham J Barezi, and Pascale Fung. Caire-covid: a question answering and multi-document summarization system for covid-19 research. *arXiv e-prints*, pages arXiv–2005, 2020.

[136] Dan Su, Tiezheng Yu, and Pascale Fung. Improve query focused abstractive summarization by incorporating answer relevance. *arXiv preprint arXiv:2105.12969*, 2021.

[137] 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.

[138] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

[139] Bowen Tan, Lianhui Qin, Eric P Xing, and Zhiting Hu. Summarizing text on any aspects: A knowledge-informed weakly-supervised approach. *arXiv preprint arXiv:2010.06792*, 2020.

[140] 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)*, pages 1171–1181, 2017.

[141] Kapil Thadani and Kathleen McKeown. Supervised sentence fusion with single-stage inference. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1410–1418, 2013.

[142] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[143] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[144] Jake Vasilakes, Chrysoula Zerva, Makoto Miwa, and Sophia Ananiadou. Learning disentangled representations of negation and uncertainty. *arXiv preprint arXiv:2204.00511*, 2022.

[145] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[146] Michael Völske, Martin Potthast, Shahbaz Syed, and Benno Stein. Tl; dr: Mining reddit to learn automatic summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 59–63, 2017.

[147] Danqing Wang, Pengfei Liu, Yining Zheng, Xipeng Qiu, and Xuanjing Huang. Heterogeneous graph neural networks for extractive document summarization. *arXiv preprint arXiv:2004.12393*, 2020.

[148] Li Wang, Junlin Yao, Yunzhe Tao, Li Zhong, Wei Liu, and Qiang Du. A reinforced topic-aware convolutional sequence-to-sequence model for abstractive text summarization. *arXiv preprint arXiv:1805.03616*, 2018.

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

[150] Shuai Wang, Xiang Zhao, Bo Li, Bin Ge, and Daquan Tang. Integrating extractive and abstractive models for long text summarization. In *2017 IEEE international congress on big data (BigData congress)*, pages 305–312. IEEE, 2017.

[151] Zhengjue Wang, Zhibin Duan, Hao Zhang, Chaojie Wang, Long Tian, Bo Chen, and Mingyuan Zhou. Friendly topic assistant for transformer based abstractive summarization. In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*, pages 485–497, 2020.

[152] Furu Wei, Wenjie Li, Qin Lu, and Yanxiang He. Query-sensitive mutual reinforcement chain and its application in query-oriented multi-document summarization. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 283–290, 2008.

[153] Adina Williams, Andrew Drozdov*, and Samuel R Bowman. Do latent tree learning models identify meaningful structure in sentences? *Transactions of the Association for Computational Linguistics*, 6:253–267, 2018.

[154] Kam-Fai Wong, Mingli Wu, and Wenjie Li. Extractive summarization using supervised and semi-supervised learning. In *Proceedings of the 22nd international conference on computational linguistics (Coling 2008)*, pages 985–992, 2008.

[155] Kristian Woodsend and Mirella Lapata. Multiple aspect summarization using integer linear programming. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 233–243, 2012.

[156] Haibing Wu, Yiwei Gu, Shangdi Sun, and Xiaodong Gu. Aspect-based opinion summarization with convolutional neural networks. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 3157–3163. IEEE, 2016.

[157] Shaojuan Wu, Xiaowang Zhang, Deyi Xiong, Shizhan Chen, Zhiqiang Zhuang, Zhiyong Feng, et al. Learning disentangled semantic representations for zero-shot cross-lingual transfer in multilingual machine reading comprehension. *arXiv preprint arXiv:2204.00996*, 2022.

[158] Zongda Wu, Li Lei, Guiling Li, Hui Huang, Chengren Zheng, Enhong Chen, and Guandong Xu. A topic modeling based approach to novel document automatic summarization. *Expert Systems with Applications*, 84:12–23, 2017.

[159] Wen Xiao and Giuseppe Carenini. Extractive summarization of long documents by combining global and local context. *arXiv preprint arXiv:1909.08089*, 2019.

[160] Yujia Xie, Tianyi Zhou, Yi Mao, and Weizhu Chen. Conditional self-attention for query-based summarization. *arXiv preprint arXiv:2002.07338*, 2020.

[161] Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. Discourse-aware neural extractive text summarization. *arXiv preprint arXiv:1910.14142*, 2019.

[162] Runxin Xu, Fuli Luo, Zhiyuan Zhang, Chuanqi Tan, Baobao Chang, Songfang Huang, and Fei Huang. Raise a child in large language model: Towards effective and generalizable fine-tuning. *arXiv preprint arXiv:2109.05687*, 2021.

[163] Yumo Xu and Mirella Lapata. Coarse-to-fine query focused multi-document summarization. In *Proceedings of the 2020 Conference on empirical methods in natural language processing (EMNLP)*, pages 3632–3645, 2020.

[164] Yumo Xu and Mirella Lapata. Generating query focused summaries from query-free resources. *arXiv preprint arXiv:2012.14774*, 2020.

[165] Kaichun Yao, Libo Zhang, Tiejian Luo, and Yanjun Wu. Deep reinforcement learning for extractive document summarization. *Neurocomputing*, 284:52–62, 2018.

[166] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *arXiv preprint arXiv:1411.1792*, 2014.

[167] Tiezheng Yu, Zihan Liu, and Pascale Fung. Adaptsum: Towards low-resource domain adaptation for abstractive summarization. *arXiv preprint arXiv:2103.11332*, 2021.

[168] Ruifeng Yuan, Shichao Sun, Zili Wang, Ziqiang Cao, and Wenjie Li. Separating context and pattern: Learning disentangled sentence representations for low-resource extractive summarization. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7575–7586, 2023.

[169] Ruifeng Yuan, Zili Wang, Ziqiang Cao, and Wenjie Li. Few-shot query-focused summarization with prefix-merging. *arXiv preprint arXiv:2211.16164*, 2022.

[170] Ruifeng Yuan, Zili Wang, Ziqiang Cao, and Wenjie Li. Preserve context information for extract-generate long-input summarization framework. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 13932–13939, 2023.

[171] Ruifeng Yuan, Zili Wang, and Wenjie Li. Fact-level extractive summarization with hierarchical graph mask on bert. *arXiv preprint arXiv:2011.09739*, 2020.

[172] Ruifeng Yuan, Zili Wang, and Wenjie Li. Event graph based sentence fusion. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4075–4084, 2021.

[173] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33:17283–17297, 2020.

[174] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR, 2020.

[175] Justin Jian Zhang, Ricky Ho Yin Chan, and Pascale Fung. Extractive speech summarization using shallow rhetorical structure modeling. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1147–1157, 2009.

[176] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.

[177] Xiangwen Zhang, Jinsong Su, Yue Qin, Yang Liu, Rongrong Ji, and Hongji Wang. Asynchronous bidirectional decoding for neural machine translation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[178] Xingxing Zhang, Mirella Lapata, Furu Wei, and Ming Zhou. Neural latent extractive document summarization. *arXiv preprint arXiv:1808.07187*, 2018.

[179] Xingxing Zhang, Furu Wei, and Ming Zhou. Hibert: Document level pre-training of hierarchical bidirectional transformers for document summarization. *arXiv preprint arXiv:1905.06566*, 2019.

[180] Yusen Zhang, Ansong Ni, Tao Yu, Rui Zhang, Chenguang Zhu, Budhaditya Deb, Asli Celikyilmaz, Ahmed Hassan Awadallah, and Dragomir Radev. An exploratory study on long dialogue summarization: What works and what's next. *arXiv preprint arXiv:2109.04609*, 2021.

[181] Lulu Zhao, Fujia Zheng, Weihao Zeng, Keqing He, Weiran Xu, Huixing Jiang, Wei Wu, and Yanan Wu. Domain-oriented prefix-tuning: Towards efficient and generalizable fine-tuning for zero-shot dialogue summarization. *arXiv preprint arXiv:2204.04362*, 2022.

[182] Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M Meyer, and Steffen Eger. Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance. *arXiv preprint arXiv:1909.02622*, 2019.

[183] Yao Zhao, Mohammad Saleh, and Peter J Liu. Seal: Segment-wise extractive-abstractive long-form text summarization. *arXiv preprint arXiv:2006.10213*, 2020.

[184] Zhou Zhao, Haojie Pan, Changjie Fan, Yan Liu, Linlin Li, Min Yang, and Deng Cai. Abstractive meeting summarization via hierarchical adaptive segmental network learning. In *The world wide web conference*, pages 3455–3461, 2019.

[185] Chen Zheng and Parisa Kordjamshidi. Srlgrn: Semantic role labeling graph reasoning network. *arXiv preprint arXiv:2010.03604*, 2020.

[186] Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. Extractive summarization as text matching. *arXiv preprint arXiv:2004.08795*, 2020.

[187] Ming Zhong, Pengfei Liu, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. Searching for effective neural extractive summarization: What works and what's next. *arXiv preprint arXiv:1907.03491*, 2019.

[188] Ming Zhong, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. Dialoglm: Pre-trained model for long dialogue understanding and summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11765–11773, 2022.

[189] Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, et al. Qmsum: A new benchmark for query-based multi-domain meeting summarization. *arXiv preprint arXiv:2104.05938*, 2021.

[190] Sheng-hua Zhong, Yan Liu, Bin Li, and Jing Long. Query-oriented unsupervised multi-document summarization via deep learning model. *Expert systems with applications*, 42(21):8146–8155, 2015.

[191] 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.

[192] Qingyu Zhou, Nan Yang, Furu Wei, Shaohan Huang, Ming Zhou, and Tiejun Zhao. Neural document summarization by jointly learning to score and select sentences. *arXiv preprint arXiv:1807.02305*, 2018.

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

[194] Chenguang Zhu, Ruochen Xu, Michael Zeng, and Xuedong Huang. A hierarchical network for abstractive meeting summarization with cross-domain pretraining. *arXiv preprint arXiv:2004.02016*, 2020.