

Neural Text Summarization and Generation

LI, Piji

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Doctor of Philosophy
in
Systems Engineering and Engineering Management

The Chinese University of Hong Kong

July 2018

Thesis Assessment Committee

Professor CHENG Hong (Chair)

Professor LAM Wai (Thesis Supervisor)

Professor YU Jeffrey Xu (Committee Member)

Professor WONG Man Leung (External Examiner)

Abstract

Automatic summarization is the process of reducing a text document or a document cluster with a computer program in order to create a summary that retains the most important information. A good summary should cover the most important points, while being coherent, non-redundant and grammatically readable.

The problem of automatic summarization has been studied for a long time and applied widely in various domains. Nevertheless, existing automatic systems still face a variety of major challenges. Most of the research works employ extractive methods to construct summaries. However, some previous research works show that human-written summaries are more abstractive. Abstractive summarization is full of challenges, and the performance depends on the techniques drawn from natural language understanding and abstractive text generation. Moreover, we observe that some works mainly use Bag-of-Words (BoWs) vectors to represent sentences. The BoWs representations are sparse and in high-dimensional size, which lead to poor performance on semantic modeling and relation detection. To address these problems, for abstractive text summarization, we propose a new framework based on a sequence-to-sequence oriented encoder-decoder model equipped with a deep recurrent generative decoder for latent summary structure modeling. For multi-document summarization, we propose a cascaded attention modeling based unsupervised framework to estimate the salience information from the text. We also introduce an unsupervised data reconstruction framework for sentence salience esti-

mation based on variational auto-encoders which jointly considers the reconstruction for latent semantic space and observed term vector space. Nowadays, with the development of social media and mobile equipments, more and more user generated content is available. One natural extension of the problem setting is to incorporate such content regarding the event so as to directly or indirectly improve the generated summaries with greater user satisfaction. However, no previous work has investigated how to incorporate the user generated content into text summarization models. To tackle this issue, a new multi-document summarization paradigm called reader-aware multi-document summarization (RA-MDS) is introduced. We propose a new framework to generate summaries jointly considering news reports and user comments. We also introduce a new dataset and describe the details of data collection and annotation. Finally, text summarization, especially abstractive text summarization can be regarded as a branch of automatic text generation research. Recently, text generation for recommendation systems attracts much attention. But no previous works consider employing the user persona information to improve the quality of the generated text. Therefore, we propose a new task called persona-aware abstractive tips generation for recommendation systems. A neural network based model is introduced to conduct the tips generation and rating prediction. Persona information of users and items are incorporated with the model to generate better text.

摘要

自动摘要使用计算机程序来简化文档或文档集，生成一个能保留重要信息的摘要的过程。好的摘要应该覆盖最重要的信息，而且是连贯的、非冗余的、语法正确可读的。

自动摘要已经研究了很长时间，并且在各个领域得到了广泛的应用。尽管如此，传统的自动系统仍面临着各种挑战。大多数研究工作采用抽取的方法来构建摘要，然而，一些前人的研究工作表明，人类书写摘要更像是生成式的。生成式摘要挑战很大，性能取决于自然语言理解的性能和文本生成的技术。此外，我们观察到许多工作主要用词袋向量来表示句子。词袋向量表示具有稀疏和高维度的特点，导致语义建模和关系检测性能较差。为了解决这些问题，对于生成式文本摘要，我们提出了一种基于序列到序列的编码-解码模型的新框架，该模型配备有用于潜在摘要结构建模的深度递归生成式解码器。对于多文档摘要，我们提出了一种基于无监督的级联注意力模型来估计文本的重要性。我们还引入了一种基于变分自编码器的无监督数据重建框架，它同时考虑了潜在语义空间和观测样本向量空间的重构。如今，随着社交媒体和移动设备的发展，越来越多的内容由用户产生。一个自然的任务扩展就是融合这些关于事件的用户数据直接或间接的来提升摘要的质量。然而，以前的工作没有研究如何将用户生成的内容合并到文本摘要系统中。为解决这个问题，我们介绍了一种称为读者感知型多文档摘要 (RA-MDS) 的新型多文档摘要范式。我们提出了一个新的框架来同时考虑新闻报道和用户评论来产生摘要。我们还引入了一个新的数据集并描述了数据收集和标注的过程细节。实际上，文本摘要，特别是生成式文本摘要任务可以看作是文本生成任务的一个分

支。最近，推荐系统的自动文本生成引起了很多关注。但是，以前的作品没有考虑用户个性化信息来提高生成文本的质量。所以，我们提出了一个推荐系统结合文本生成的新任务。我们引入了基于神经网络的模型来进行文本生成和打分预测。用户和产品的个性化信息被整合到模型中用来生成更好的文本。

Acknowledgement

First and for most, I would like to express my sincere gratitude to my supervisor, Prof. LAM Wai for his supervision of my doctoral research during the past four years. I have been amazingly fortunate to have an advisor who gave me the freedom to explore on my own. He taught me how to tackle research questions and express ideas. He gave me enormous advice, patience, and support. His guidance helped me in all the time of research and writing of this thesis. I benefit a lot from his enthusiasm for research, critical thinking, profound knowledge, scholarly expertise, etc.

I would also like to thank my outstanding committee members: Prof. CHENG Hong, Prof. YU Jeffrey Xu, and Prof. WONG Man Leung. Thanks for your helpful comments and suggestions on my research and thesis.

I would like to especially thank all my co-authors: Dr. Hang Li, Prof. Rebecca J. Passonneau, Zhaochun Ren, Zhongyu Wei, Shangsong Liang, and Weiwei Guo for enthusiastic support and help during our research collaborations.

I want to thank Dr. Hang Li and Dr. Zhengdong Lu for the frequent communications and helpful suggestions on the collaborative projects between Noah's Ark Lab, Huawei and The Chinese University of Hong Kong. Your excellent research works in the fields of natural language processing and deep learning also inspired me deeply.

I also want to thank all the people in the Text Mining Group. I feel honoured

and fortunate to work with these brilliant colleagues: Lidong Bing, Shoaib Jameel, Chunliang Lu, Yinqing Xu, Yi Liao, Bei Shi, Xinshi Lin, Xin Li, Qian Yu, Tao Wang, Zihao Fu, Xin Shen, Kwun Ping Lai, and Zihao Wang.

I have spent four wonderful years in the Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong. I' m thankful to Zhongyu Wei, Junwen Chen, Jing Li, Baolin Peng, Ming Liao, Shichao Dong, Xingshan Zeng, Jing Ma, Yu Rong, Wentao Tian, Siyuan Zhang, Can Lu, Hao Wei, Kangfei Zhao, Yuli Jiang, and Hao Zhang for their friendship, encouragement and sharing in the past four years.

I would like to thank my Master Degree supervisor in Shandong University, Prof. Jun Ma, who introduces me into the fields of information retrieval and machine learning. Thanks for your continuous attention, communication, and stimulation in my research life.

Last, but not least, I would like to thank my parents, Xiufen Li and Weizong Li, for always supporting me spiritually throughout my studies. Special thanks to my wife, Dr. Jiaxing Xu, for her understanding, encouragement, and love.

Contents

| | |
|--|-----------|
| Abstract | i |
| Acknowledgement | v |
| 1 Introduction | 1 |
| 1.1 Contributions | 7 |
| 1.2 Publication List | 10 |
| 1.3 Thesis Outline | 11 |
| 2 Literature Survey | 13 |
| 2.1 Text Summarization | 13 |
| 2.2 Abstractive Text Generation | 15 |
| 2.3 Neural Sequence Modeling | 17 |
| 2.4 Variational Auto-Encoders | 18 |
| 3 Latent Structure Modeling for Single-Document Summarization | 20 |
| 3.1 Background | 20 |
| 3.2 Framework Description | 22 |
| 3.2.1 Overview | 22 |
| 3.2.2 Recurrent Generative Decoder | 24 |
| 3.2.3 Abstractive Summary Generation | 27 |

| | | |
|----------|---|-----------|
| 3.2.4 | Learning | 31 |
| 3.3 | Experimental Setup | 32 |
| 3.3.1 | Datesets | 32 |
| 3.3.2 | Evaluation Metrics | 32 |
| 3.3.3 | Comparative Methods | 33 |
| 3.3.4 | Experimental Settings | 35 |
| 3.4 | Results and Discussions | 35 |
| 3.4.1 | ROUGE Evaluation | 35 |
| 3.4.2 | Summary Case Analysis | 37 |
| 3.5 | Summary | 39 |
| 4 | Cascaded Attention Modeling for Multi-Document Summarization | 40 |
| 4.1 | Background | 40 |
| 4.2 | Framework Description | 42 |
| 4.2.1 | Overview | 42 |
| 4.2.2 | Attention Modeling for Distillation | 43 |
| 4.2.3 | Compressive Summary Generation Phase | 48 |
| 4.3 | Experimental Setup | 53 |
| 4.3.1 | Datasets | 53 |
| 4.3.2 | Settings | 54 |
| 4.4 | Results and Discussions | 54 |
| 4.4.1 | Effect of Different Architectures | 54 |
| 4.4.2 | Main Results of Compressive MDS | 56 |
| 4.4.3 | Case Study: Distilled Word Saliency | 59 |
| 4.4.4 | Case Study: Attention-based Sentence Saliency | 59 |
| 4.4.5 | Case Analysis | 60 |
| 4.5 | Summary | 61 |

| | | |
|----------|---|-----------|
| 5 | Variational Auto-Encoders for Multi-Document Summarization | 62 |
| 5.1 | Background | 62 |
| 5.2 | Overview of Our Proposed Framework | 65 |
| 5.3 | Sentence Saliency Framework | 66 |
| 5.3.1 | Latent Semantic Modeling | 66 |
| 5.3.2 | Saliency Estimation | 69 |
| 5.3.3 | Multi-Task Learning | 72 |
| 5.4 | Summary Generation | 73 |
| 5.5 | Experiments and Results | 75 |
| 5.5.1 | Datasets | 75 |
| 5.5.2 | Evaluation Metric | 75 |
| 5.5.3 | Settings | 75 |
| 5.5.4 | Results and Discussions | 76 |
| 5.6 | Summary | 79 |
| 6 | Reader-Aware Multi-Document Summarization | 80 |
| 6.1 | Background | 80 |
| 6.2 | Framework | 82 |
| 6.2.1 | Overview | 82 |
| 6.2.2 | Reader-Aware Saliency Estimation | 84 |
| 6.2.3 | Preparation of Entity Mentions for Rewriting | 87 |
| 6.2.4 | Summary Construction | 89 |
| 6.3 | Data Description | 90 |
| 6.3.1 | Background | 90 |
| 6.3.2 | Data Collection | 91 |
| 6.3.3 | Data Properties | 92 |
| 6.4 | Experimental Setup | 92 |
| 6.4.1 | Dataset and Metrics | 92 |

| | | |
|----------|--|------------|
| 6.4.2 | Comparative Methods | 93 |
| 6.4.3 | Experimental Settings | 94 |
| 6.5 | Results and Discussions | 94 |
| 6.5.1 | Results on Our Dataset | 94 |
| 6.5.2 | Further Investigation of Our Framework | 95 |
| 6.5.3 | Case Study | 96 |
| 6.5.4 | Topics | 97 |
| 6.6 | Summary | 99 |
| 7 | Persona-Aware Abstractive Tips Generation | 100 |
| 7.1 | Background | 100 |
| 7.2 | Framework Description | 104 |
| 7.2.1 | Overview | 104 |
| 7.2.2 | Persona Modeling | 105 |
| 7.2.3 | Abstractive Tips Generation | 112 |
| 7.3 | Experimental Setup | 117 |
| 7.3.1 | Datasets | 117 |
| 7.3.2 | Evaluation Metrics | 117 |
| 7.3.3 | Comparative Methods | 118 |
| 7.3.4 | Experimental Settings | 120 |
| 7.4 | Results and Discussions | 121 |
| 7.4.1 | Abstractive Tips Generation | 121 |
| 7.4.2 | Ablation Experimental Results | 124 |
| 7.4.3 | Rating Prediction | 125 |
| 7.4.4 | Further Investigations | 126 |
| 7.5 | Summary | 127 |
| 8 | Conclusion | 129 |

List of Figures

| | | |
|-----|--|-----|
| 1.1 | Single-document summarization. | 2 |
| 1.2 | Multi-document summarization. | 3 |
| 1.3 | Reader comments. | 5 |
| 1.4 | Examples of reviews and tips. | 6 |
| 2.1 | Sequence-to-sequence framework. | 17 |
| 2.2 | Variational Auto-Encoders. | 18 |
| 3.1 | Headlines of the top stories from the channel “Technology” of CNN. | 21 |
| 3.2 | Deep recurrent generative decoder | 23 |
| 4.1 | Our cascaded attention modeling framework | 42 |
| 4.2 | The constituency tree of a sentence. | 49 |
| 4.3 | Visualization for sentence attention. | 59 |
| 5.1 | Our proposed sentence salience framework | 64 |
| 6.1 | Example of reader comments | 81 |
| 6.2 | Our proposed RAVAESum framework | 83 |
| 7.1 | Example of tips. | 101 |
| 7.2 | Framework for persona-aware abstractive tips generation | 103 |

List of Tables

| | | |
|-----|---|----|
| 3.1 | ROUGE-F1 on validation sets | 35 |
| 3.2 | ROUGE-F1 on Gigawords | 36 |
| 3.3 | ROUGE-Recall on DUC2004 | 36 |
| 3.4 | ROUGE-F1 on LCSTS | 37 |
| 3.5 | Examples of the generated summaries. | 38 |
| 4.1 | Comparisons on TAC 2010 | 55 |
| 4.2 | Results on DUC 2006. | 57 |
| 4.3 | Results on DUC 2007. | 57 |
| 4.4 | Results on TAC 2011. | 58 |
| 4.5 | Top-10 terms extracted from each topic according to the word salience | 58 |
| 4.6 | The summary of the topic “ <i>Hawkins Robert Van Maur</i> ”. | 61 |
| 5.1 | Results on DUC 2006. | 76 |
| 5.2 | Results on DUC 2007. | 76 |
| 5.3 | Results on TAC 2011. | 77 |
| 5.4 | Top-10 terms extracted from each topic according to the output of VAEs-A | 78 |
| 6.1 | Summarization performance. | 95 |
| 6.2 | Further investigation of RAVAESum. | 95 |
| 6.3 | Top-10 terms | 96 |

| | | |
|-----|---|-----|
| 6.4 | Generated summaries for the topic “Sony Virtual Reality PS4”. . . . | 97 |
| 6.5 | All the topics and the corresponding categories. | 98 |
| 7.1 | Overview of the datasets. | 117 |
| 7.2 | Baselines and methods used for comparison. | 120 |
| 7.3 | ROUGE (R-1 and R-2) evaluation on the five datasets from different domains. | 122 |
| 7.4 | ROUGE (R-L and R-SU4) evaluation on the five datasets from different domains. | 123 |
| 7.5 | Examples of the predicted ratings and the generated tips | 125 |
| 7.6 | Ablation experiments on the dataset Home | 126 |
| 7.7 | MAE and RMSE values for rating prediction on datasets Electronics and Movies. | 126 |
| 7.8 | MAE and RMSE values for rating prediction on datasets Yelp, Clothing, and Home. | 127 |
| 7.9 | Rating controlled tips generation. | 127 |

Chapter 1

Introduction

Due to the enormous amount of information in this era, we are facing an inevitable and challenging problem of information overload. Lots of information from different sources in different types rush to people through computers and mobile equipments. To address the problem of data disaster, there is an intensive need to refine and compress the information. Automatic summarization is the process of reducing a text document or a document cluster with a computer program in order to create a summary that retains the most important information. A good summary should cover the most important points, while being coherent, non-redundant and grammatically readable. The problem of automatic summarization has been studied for a long time and applied widely in various domains [25, 33, 84, 102, 141]. For example, many news websites such as Dailymail¹ provide highlights to help users to capture the main topics of the news report quickly. Meanwhile, most of the search engines generate snippets for each result document in order to convey more details information in the content of the document.

Considering the kind of input documents, summarization tasks can be divided into two categories: single-document summarization (SDS) [81] and multi-document

¹<http://www.dailymail.co.uk>

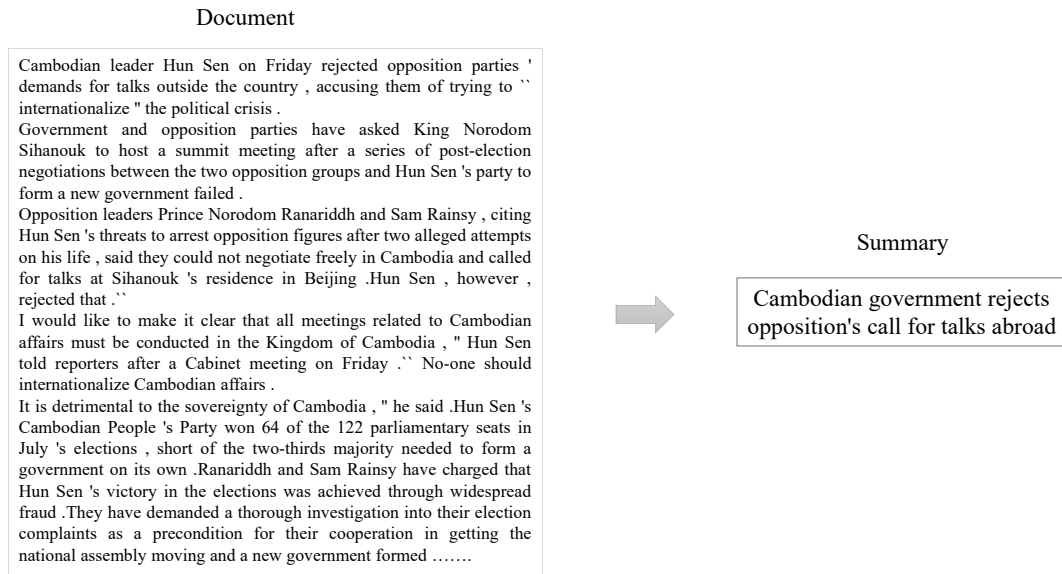


Figure 1.1: Single-document summarization.

summarization (MDS) [86]. The goal of single-document summarization is to generate a short summary for one document at a time. As shown in Figure 1.1, the text in the left part is a news document, and the generated short summary is shown in the right column. Actually, the task of news headline generation [114] can be regarded as a special task of single-document summarization.

The purpose of multi-document summarization is to generate a summary for a topic which describes an event discussed in a set of documents from different sources. For example, Figure 1.2 is an illustration of the summarization setting for the topic “Malaysia Airlines Disappearance”. This topic contains 10 documents coming from different news Web sites and reporting the news of “Malaysia Airlines Disappearance”. To save the time of readers from reading the whole set of documents, multi-document summarization aims at producing a short summary, e.g., 100-word length, that covers the essential information of all the news documents in this topic or event.

According to the different summary generation methods, the summarization

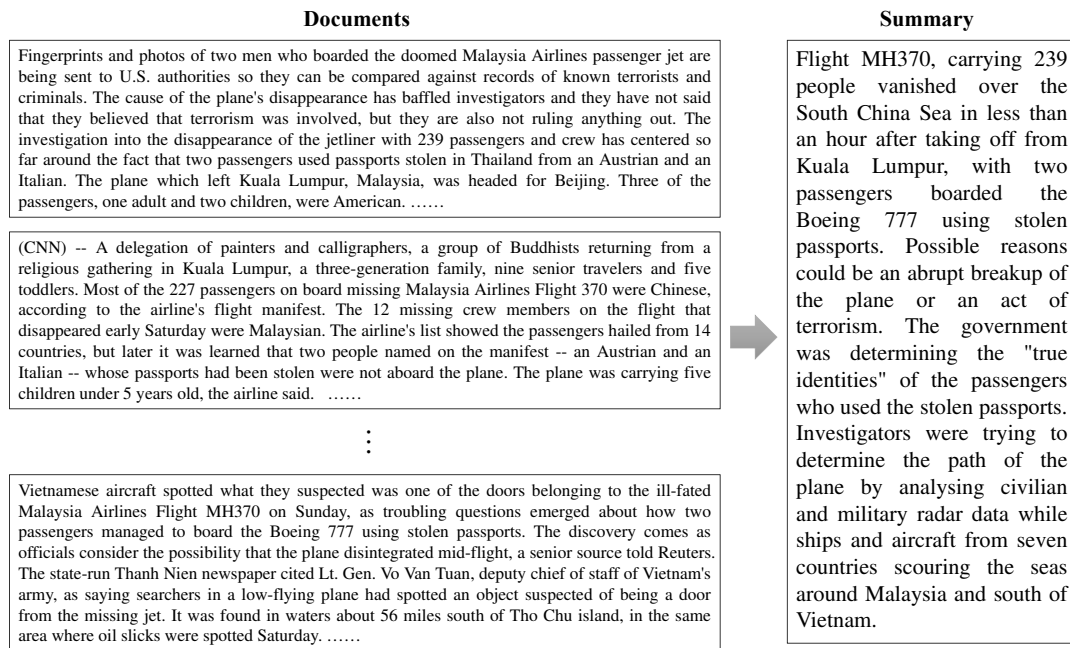


Figure 1.2: Multi-document summarization for the topic “Malaysia Airlines Disappearance”.

techniques can be classified into three categories: extraction-based approaches, compression-based approaches, and abstraction-based approaches. Extraction-based approaches are the most studied approach of the three. Early studies mainly followed a greedy strategy in sentence selection [13, 35, 128]. Each sentence in the documents is firstly assigned a salience score. Then, sentence selection is performed by greedily selecting the sentence with the largest salience score among the remaining ones. The redundancy is controlled during the selection. Compression-based approaches adopted a two steps [34, 70, 78, 145]. The first step selects the sentences, and the second step removes the unimportant or redundant units from the sentences. Different from the common extraction-based and compression-based methods, abstraction-based methods aim at constructing new sentences as summaries, thus they require a deeper understanding of the text and the capability of generating new sentences, which provide an obvious advantage in improving the focus of a summary, reducing

the redundancy, and keeping a good compression rate [3, 6].

Actually, some previous research works show that human-written summaries are more abstractive [48]. However, abstractive summarization is full of challenges, and the performance depends on the techniques drawn from natural language understanding and abstractive text generation. Therefore, some works employ indirect techniques to construct new sentences. For example, Barzilay and McKeown [3] followed by [30, 31] employ sentence fusion techniques to construct new sentences. Bing et al. [6] propose a fine-grained sentence construction method by merging non-phrases and verb-phrases from different sentences. Nevertheless, these methods are indirect strategies and sometimes will do harm to the linguistic quality of the constructed sentences. Therefore, better methods need to be proposed to address the abstractive summarization problem.

Considering the prohibitive resources for labeling multi-document summarization datasets, some methods adopt unsupervised data reconstruction methods to conduct salience estimation and achieve comparable results [42, 70, 82, 108, 116, 140]. After investigating these works, we observe that they mainly use Bag-of-Words (BoWs) vectors in sentence representation and reconstruction loss function. The BoWs representations are in high-dimensional size and very sparse, which lead to poor performance on semantic modeling and relation detection. On the other hand, some research works [44, 52, 59, 96] have demonstrated that distributed representations outperform BoWs in modeling sentence and document semantics, and obtain significant improvements in the tasks of sentence matching, sentiment analysis, and text classification. Intuitively, employing distributed representations to model sentences can improve the performance semantic modeling and similarity measurement, which can further improve the performance of summarization. However, no previous automatic summarization approaches consider this problem.

With the development of social media and mobile equipments, more and more

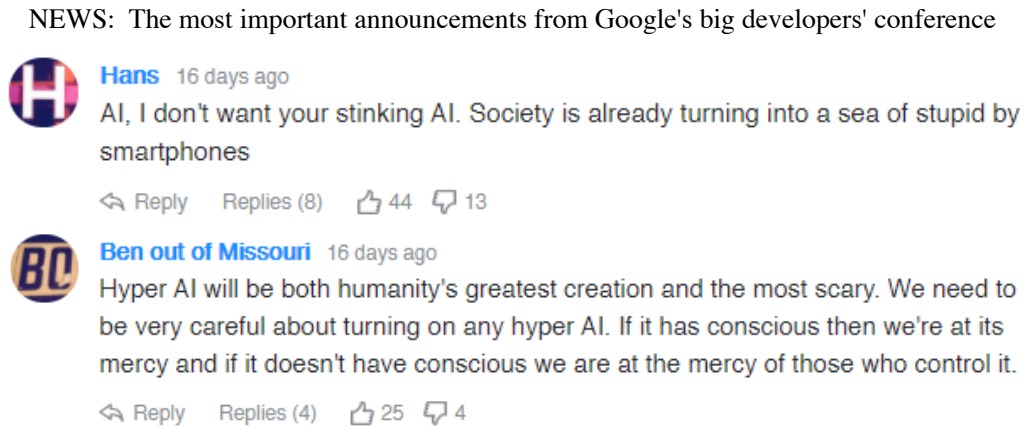



Figure 1.3: Reader comments of the news “The most important announcements from Google’s big developers’ conference (May, 2017)”.

user generated content is available. Figure 1.3 is a snapshot of reader comments under the news report “The most important announcements from Google’s big developers’ conference”². The content of the original news report talks about some new products based on AI techniques. The news report generally conveys an enthusiastic tone. However, while some readers share similar enthusiasms, some others express their worries about new products and technologies and these comments can also reflect their interests which may not be very salient in the original news reports. In order to improve the generated summaries with greater user satisfaction, the generated summaries from the reports for the event should be salient according to not only the reports but also the reader comments. However, no previous work has investigated how to incorporate comments into MDS problem. One challenge is how to conduct salience calculation by jointly considering the focus of news reports and the reader interests revealed by comments. Meanwhile, the model should not be sensitive to the availability of diverse aspects of reader comments. Another challenge is that reader comments are very noisy, grammatically and content-wise. Some previous works explore the effect of comments or social contexts in single document

²<https://goo.gl/DdU0vL>

Review



Monica H.


👍 60 🌟 10 📷 26

★★★★★
📷 7
7 days ago

This place is amazing! If I could give more than 5 stars I would! Not only is the food impeccable but the service and hospitality is top notch. The staff was so attentive and detail oriented, making it a truly one of a kind experience. It is an intimate restaurant, not overly crowded with tables. So if you want to have the Gary Danko experience you need to make reservations well in advance.


They have options of 3-5 course meals. And all food categories are absolutely delicious. We each did the four course meal and shared each course so that we could try as many things as possible. To start we had the butternut squash soup and the risotto. Both fabulous. The risotto was like none I have ever tasted; creamy without being overly heavy. And the soup was lite and fit the season well. Next we had the lobster and potato purée and the seafood curry. Both amazing. I'm usually

Tips




T D. 6/21/15

Pass on the bison. Lobster tail, risotto, beef, duck breast are good




Morgan G. 6/21/15

Everything was absolutely incredible. Service. Food. Atmosphere. All perfect!




Praveen K. 11/30/14

The risotto was excellent. Amazing service.




Amy L. 9/4/14

Great service and food. Definitely not a jeans and t-shirt place.



Michelle D. 8/31/14

Service and staff here is one of the best in all of SF! I was so impressed!



Madhulika G. 7/23/14

You have to make reservations much in advance

Figure 1.4: Examples of reviews and tips selected from the restaurant “Gary Danko” on Yelp. Tips are more concise than reviews and can reveal user experience, feelings, and suggestions with only a few words. Users will get conclusions about this restaurant immediately after scanning the tips with their mobile phones.

summarization (such as blog summarization) [46, 139]. However, our problem is more challenging because the considered comments are about an event with multiple reports spanning a time period, resulting in diverse and noisy comments.

Generally, text summarization, especially the abstractive text summarization can be regarded as a branch of text generation. Also, there are lots of other popular text generation tasks such as machine translation [2], dialogue systems [113], and caption generation for images and videos [123, 138]. Recently, automatic text generation for recommendation systems attracts much attention and some approaches have been proposed to address this challenging problem. According to different nature of texts, the text generation tasks for recommendation systems can be classified into two categories: review generation [23, 103, 119, 142] and tips generation [75]. As

shown in Figure 1.4, the left column is a review from the user “Monica H.”, and tips from several other users are shown on the right column. In the review text, Monica first generally introduced the restaurant, and then narrated her dining experience in detail. In the tips text, users expressed their experience and feelings plainly using short texts, such as “The risotto was excellent. Amazing service.”. They also provide some suggestions to other people directly in several words, such as “You have to make reservations much in advance.” In contrast to item specifications and user reviews, tips have several characteristics: (1) tips are typically single-topic nuggets of information, and shorter than reviews with a length of about 10 words on average; (2) tips can express user experience, feelings, and suggestions directly; (3) tips can give other people quick insights, saving the time of reading long reviews. However, existing works only consider text information such as item specifications and user reviews in their systems. No previous works incorporate the tips information to improve the performance of recommendation system.

In this thesis, we investigate the above mentioned problems and propose several frameworks to tackle the corresponding tasks.

1.1 Contributions

- **Latent Structure Modeling for Single-Document Summarization**

For single-document summarization, after analyzing the summaries carefully, we can find some common structures from them, such as “*What*”, “*What-Happened*”, “*Who Action What*”, etc. Intuitively, if we can incorporate the latent structure information of summaries into the abstractive summarization model, it will improve the quality of the generated summaries. To address the problem, we propose a new framework based on a sequence-to-sequence oriented encoder-decoder model equipped with a deep recurrent generative decoder (DRGN). Latent struc-

ture information implied in the target summaries is learned based on a recurrent latent random model for improving the summarization quality. Neural variational inference is employed to address the intractable posterior inference for the recurrent latent variables.

- **Cascaded Attention Modeling for Multi-Document Summarization**

In the context of multi-document summarization, to generate a summary sentence for a key aspect of the topic, we need to find its relevant parts in the original documents, which may attract more attention. The semantic parts with high attention weights plausibly represent and reconstruct the topic's main idea. Inspired by this observation, considering the helpfulness of the attention modeling mechanism used in the models for abstractive summarization, we propose a cascaded attention based unsupervised model to estimate the salience information from the text for compressive multi-document summarization. The attention weights are learned automatically by an unsupervised data reconstruction framework which can capture the sentence salience.

- **Variational Auto-Encoders for Multi-Document Summarization**

Recall that the distributed sentence representations perform much better than the BoWs vectors in many tasks such sentence matching and sentiment analysis. In order to employ the distributed sentence representations to improve the performance of summarization, we propose a new unsupervised sentence salience framework which can be divided into two components: latent semantic modeling and salience estimation. For latent semantic modeling, a neural generative model called Variational Auto-Encoders (VAEs) is employed to describe the observed sentences and the corresponding latent semantic representations. Neural variational inference is used for the posterior inference of the latent variables. For salience estimation, we propose an unsupervised data reconstruction framework,

which jointly considers the reconstruction for latent semantic space and observed term vector space. Therefore, we can capture the salience of sentences from these two different and complementary vector spaces. Thereafter, the VAEs-based latent semantic model is integrated into the sentence salience estimation component in a unified fashion.

- **Reader-Aware Multi-Document Summarization**

To generate the summaries by jointly considering the news reports and user comments, we propose a new multi-document summarization paradigm called reader-aware multi-document summarization (RA-MDS). Specifically, a set of reader comments associated with the news reports are also collected. The generated summaries from the reports for the event should be salient according to not only the reports but also the reader comments. To tackle this RA-MDS problem, we propose a neural network based method that is able to calculate the salience of the text units by jointly considering news reports and reader comments. Another reader-aware characteristic of our framework is to improve linguistic quality via entity rewriting. The rewriting consideration is jointly assessed together with other summarization requirements under a unified optimization model. To support the generation of compressive summaries via optimization, we explore a finer syntactic unit, namely, noun/verb phrase. In this work, we also generate a data set for conducting RA-MDS. We describe the methods for data collection, aspect annotation, and summary writing as well as scrutinizing by experts.

- **Persona-Aware Abstractive Tips Generation**

We investigate the task of abstractive tips generation for recommendation systems. Different from existing methods, our framework considers persona information when conducting tips text generation. In order to exploit the persona information, we propose a framework based on adversarial variational auto-encoders

(aVAE) for persona modeling from the historical tips and reviews for users and items. The latent variables from aVAE are regarded as persona embeddings. Besides representing persona using the latent embeddings, we design a persona memory for directly storing the persona related words for the current user and item. Pointer Networks is used to retrieve persona related information from the memory when generating tips. The distilled persona embeddings are used as latent factors for users and items and are fed into the rating prediction component for detecting sentiment. Finally, the persona embeddings and the sentiment information are incorporated into the recurrent neural networks (RNN) based tips generation component.

1.2 Publication List

The contributions and results have been published in the following venues:

- Piji Li, Lidong Bing, Wai Lam, Hang Li and Yi Liao. Reader-Aware Multi-Document Summarization via Sparse Coding. In Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI), pp. 1270-1276. 2015. [70]
- Piji Li, Zihao Wang, Wai Lam, Zhaochun Ren, and Lidong Bing. Saliency Estimation via Variational Auto-Encoders for Multi-Document Summarization. In Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI), pp. 3497-3503. 2017. [74].
- Piji Li, Zihao Wang, Zhaochun Ren, Lidong Bing, and Wai Lam. Neural rating regression with abstractive tips generation for recommendation. In Proceedings of the 40th International ACM SIGIR conference on Research

and Development in Information Retrieval (SIGIR), pp. 345-354. ACM, 2017. [75]

- Piji Li, Wai Lam, Lidong Bing, and Zihao Wang. Deep Recurrent Generative Decoder for Abstractive Text Summarization. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 2091-2100. 2017. [73]
- Piji Li, Wai Lam, Lidong Bing, Weiwei Guo, and Hang Li. Cascaded Attention based Unsupervised Information Distillation for Compressive Summarization. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 2081-2090. 2017. [72]
- Piji Li, Lidong Bing, and Wai Lam. Reader-Aware Multi-Document Summarization: An Enhanced Model and The First Dataset. In Proceedings of the Workshop on New Frontiers in Summarization (EMNLP-NewSum), pp. 91-99. 2017. [71]
- Piji Li, Lidong Bing, and Wai Lam. Actor-Critic based Training Framework for Abstractive Summarization. arXiv preprint arXiv:1803.11070 (2018). [76]

1.3 Thesis Outline

After the high level introduction of the major problems focused on in this thesis, the rest of the chapters in the thesis are organized as follows.

In Chapter 2, we review some related works for text summarization and generation, as well as some neural network models used in our frameworks. We compare our summarization and generation frameworks with the existing works, pointing out some shortcomings of these works and the superiority of our proposed frameworks.

In Chapter 3, we propose a new framework for abstractive text summarization based on a sequence-to-sequence oriented encoder-decoder model equipped with a deep recurrent generative decoder (DRGN).

In Chapter 4, we propose a cascaded attention based unsupervised model to estimate the salience information from the text for compressive multi-document summarization.

In Chapter 5, we introduce an unsupervised data reconstruction framework for salience estimation based on Variational Auto-Encoders (VAEs), which jointly considers the reconstruction for latent semantic space and observed term vector space.

In Chapter 6, a new multi-document summarization paradigm called reader-aware multi-document summarization (RA-MDS) is introduced. We propose a new framework to generate summaries jointly considering news reports and user comments. We also introduce a new dataset and describe the details of data collection and annotation.

In Chapter 7, we propose a new task called abstractive tips generation for recommendation system. A neural network based model is introduced to conduct the tips generation and rating prediction. Persona information of users and items are considered to improve the quality of the generated tips.

In Chapter 8, we review the main contributions of the thesis and summarize the significance and applicability of the proposed frameworks. We also discuss some possible extensions and future research directions of the research topics in this thesis.

Chapter 2

Literature Survey

2.1 Text Summarization

Automatic summarization is the process of automatically generating a summary that retains the most important content of the original text document. It has been studied by the researchers in the fields of text mining and natural language processing for nearly the last half century [1, 22, 33, 102, 141]. Back to the 1950s, Luhn [84] has already introduced an important research work to generate summaries for scientific documents, by extracting salient sentences from the text using features such as word and phrase frequency.

Traditionally, according to the kind of input documents, summarization tasks can be divided into single-document summarization (SDS) and multi-document summarization (MDS). The early works such as [4, 25, 84] began from single-document summarization. McKeown and Radev [89] seems are the pioneer of multi-document summarization and they developed a system called SUMMONS (SUMMArizing On-line NewS articles) to extract summary for a series of news articles on the same event.

According to different machine learning paradigms, summarization models can

be divided into supervised framework and unsupervised framework. Min et al. [97] and Wang et al. [131] extracted numeric features manually to represent sentences and designed a support vector regression machine [24] based framework to predict the sentence salience. For unsupervised frameworks, He et al. [42], Liu et al. [82], Li et al. [70] and Song et al. [116] employed sparse coding techniques for finding the salient sentences as summaries.

Considering the different summary constructing methods, summarization techniques can be classified into three categories: extractive summarization [10, 16, 27, 35, 69, 97, 100, 116, 128], compressive summarization [63, 70, 74, 131], and abstractive summarization [3, 6]. Most of the approaches are designed for extractive summarization. Sentence salience estimation is an important procedure which can provide the criteria for sentence selection. Erkan and Radev [27] and Mihalcea and Tarau [94] constructed a sentence graph and employed Pagerank algorithm [104] to calculate the importance value for each sentence. Wan et al. [128] proposed a manifold-ranking based approach to topic-focused multi-document summarization. The proposed approach employs the manifold-ranking process to make full use of the relationships among sentences and the relationships between the topic and the sentences. Radev et al. [106] obtain the centroids by clustering the sentences and conduct the salience estimation by considering the relationship between the sentences and the centroids. Compressive summarization approaches can be divided into two steps. The first step selects the sentences, and the second step removes the unimportant or redundant units from the sentences [34, 70, 78, 145]. Abstractive summarization can generate new sentences based on the facts from different source sentences. Barzilay and McKeown [3] employed sentence fusion to generate a new sentence. Bing et al. [6] proposed a more fine-grained fusion framework, where new sentences are generated by selecting and merging salient phrases using integer linear programming (ILP) based optimization strategy [88]. These methods

can be regarded as a kind of indirect abstractive summarization, and complicated constraints are used to guarantee the linguistic quality.

Recently, inspired by the attention based sequence-to-sequence (seq2seq) framework used in machine translation [2], some researchers employ neural network based framework to tackle the abstractive summarization problem. Rush et al. [111] proposed a neural network based model with local attention modeling, which is trained on the Gigaword corpus, but combined with an additional log-linear extractive summarization model with handcrafted features. Gu et al. [40] integrated a copying mechanism into a seq2seq framework to improve the quality of the generated summaries. Chen et al. [15] proposed a new attention mechanism that not only considers the important source segments, but also distracts them in the decoding step in order to better grasp the overall meaning of input documents. Nallapati et al. [99] also employed the typical attention modeling based seq2seq framework, but utilized a trick to control the vocabulary size to improve the training efficiency. Tan et al. [118] incorporated the graph-based sentence salience estimation component with the seq2seq framework by regarding the sentence salience as graph-based attention value. Zhou et al. [150] proposed a selective encoding framework to enhance the performance of seq2seq. See et al. [112] improved the seq2seq framework by jointly considering the copy mechanism [40, 125] and the coverage modeling strategy [121]. Paulus et al. [105] proposed a reinforcement learning framework to tackle the problem of abstractive summarization.

2.2 Abstractive Text Generation

Abstractive text generation is a challenging task. Recently, sequence modeling based on the gated recurrent neural networks such as Long Short-Term Memory (LSTM) [43] and Gated Recurrent Unit (GRU) [17] demonstrates high capability in text gen-

eration related tasks. Lebre et al. [60] introduced a neural model for concept-to-text generation, which can generate biographical sentences from fact tables on a dataset of biographies from Wikipedia. Wiseman et al. [134] investigated the problem of data-to-text generation and their methods can generate texts from data records, such as the news report generation from the data records of NBA games. Murakami et al. [98] presented a encoder-decoder model for automatically generating market comments from stock prices. Che et al. [14], Fedus et al. [29], Guo et al. [41], Liao et al. [77], Lin et al. [80], Yu et al. [143], Zhang et al. [148] employed the adversarial training strategy [36] and reinforcement learning techniques to enhance the performance of the original text generation frameworks. Moreover, neural text generation related techniques have improved the performance of tasks of different areas such as machine translation [2], abstractive summarization [99, 111], conversation system [113], question generation [28], and image caption generation [138].

In the area of recommendation systems, some researchers also apply LSTM or GRU based RNN models on abstractive text generation. Tang et al. [119] proposed a framework to generate context-aware reviews. Sentiments and products are encoded into a continuous semantic representation and use RNN to conduct the decoding and generation. Dong et al. [23] regarded users, products, and rating as attribute information and employ an attention modeling based sequence modeling framework to generate reviews. Ni et al. [103] proposed to combine collaborative filtering with generative networks to jointly perform the tasks of item recommendation and review generation. Low-dimensional user preferences and item properties are combined with a character-level LSTM model to conduct the review generation. Yao et al. [142] employed the adversarial strategy to make the generated review indistinguishable from human written ones so that can improve the performance of review generation.

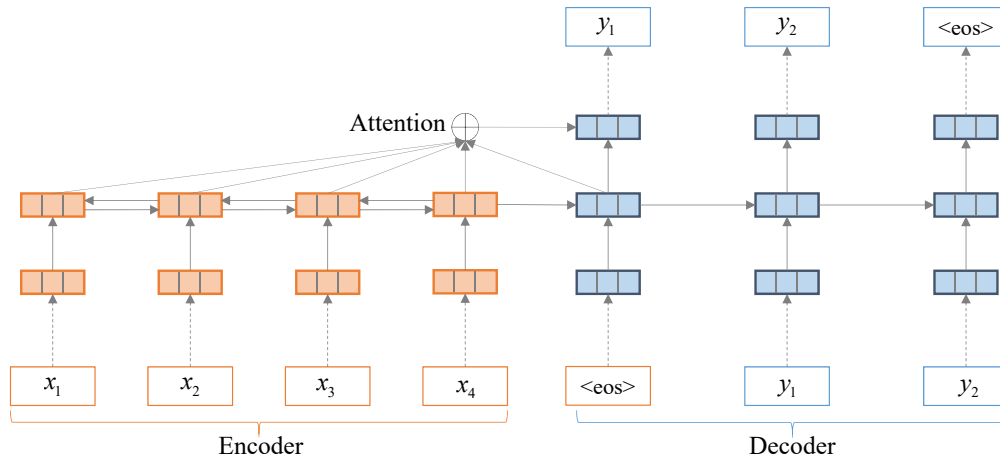


Figure 2.1: Sequence-to-sequence framework with attention modeling mechanism. It contains two components: encoder and decoder. Encoder is usually a bi-directional recurrent neural networks based on LSTM or GRU. It will conduct the sequence modeling for the input text sequence. Decoder will conduct the generation. Attention modeling mechanism can retrieve the relevant information from the input source text for better generation performance.

2.3 Neural Sequence Modeling

Recurrent Neural Networks (RNN) are specially designed for modeling sequential data. In the past, RNN was used to handle the time series data [26, 32]. At present, it has been successfully applied to the areas of natural language processing and text mining [95]. In RNN, the current hidden layer activation is generated based on the past hidden layer activation, which makes the RNN extremely deep and difficult to train due to the exploding and the vanishing gradient problems [5, 43]. To tackle this problem, Long Short-Term Memory (LSTM) neural network was proposed in [43] by introducing memory cells, linearly depending on their past values. LSTM also introduces three gating functions, namely input gate, forget gate and output gate [38]. Recently, Cho et al. [17] introduced the Gated Recurrent Unit (GRU), which is an architecture that can be comparable with LSTM on a suite of tasks with less parameters [19].

Sequence-to-sequence (seq2seq) framework [117] contains two components: encoder and decoder, as shown in Figure 2.1. Both components are designed based on RNNs with LSTM or GRU as the recurrent cells. The decoder component can be regarded as a language model, receiving context information provided by the encoder. Seq2seq framework with attention modeling mechanism as shown in Figure 2.1 was first proposed to handle the task of machine translation and yielded good performance [2, 85, 136]. Nowadays, it has been successfully extended to multiple natural language generation and text mining tasks such as abstractive text summarization [15, 40, 83, 99, 111, 112, 118, 150], text generation [60, 134], keyphrase extraction [90], dialogue systems [66, 67, 113, 124], caption generation for images and videos [50, 123, 126, 138], etc.

Besides using RNNs as the basic component for seq2seq models, CNN can also be employed to conduct the sequence modeling [122]. Well-designed attention modeling mechanism plays an import role in the framework proposed in [122], and they claim that “Attention is all you need”. Besides attention modeling, copy mechanism [40, 125] and coverage strategy [121] are also very useful and can be incorporated with the seq2seq framework to obtain better sequence prediction performance.

2.4 Variational Auto-Encoders

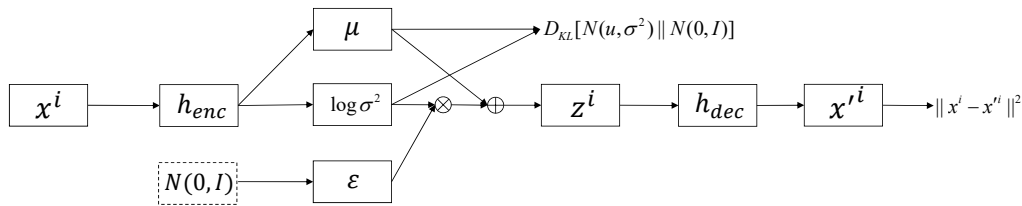


Figure 2.2: Variational Auto-Encoders.

Variational Auto-Encoders (VAEs) [54, 110] is a neural generative model which can be used to conduct the latent variable modeling and data generation. As shown

in Figure 2.2, VAEs contains two stages: inference (variational encoder) and generation (variational decoder). In the inference stage, the variational encoder can approximate the posterior distribution of the latent variables. During the generation state, the variational decoder can generate examples given the sampled random variables.

In fact, some works [20, 93] have demonstrated that VAEs outperform the general Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN) in generating high-level semantic representations. Compared with deterministic autoencoder for sequential data, the VAEs is able to capture more coherent latent space, which is attributed to the prior of the latent variable. Due to the power of the VAE, many models have adopted VAEs to solve various task. For example, Chung et al. [20] incorporated a high-level latent random variables into standard RNN to model highly structured sequential data such as natural speech. Zhang et al. [147] introduced a continuous latent variable to explicitly model underlying semantics of source sentences and to guide the generation of target translations. Hu et al. [47] proposed a deep generative model that learns interpretable latent representations and generates sentences with specified attributes such as the review ratings.

Chapter 3

Latent Structure Modeling for Single-Document Summarization

3.1 Background

Some previous research works show that human-written summaries are more abstractive [6, 48]. Moreover, our investigation reveals that people may naturally follow some inherent structures when they write the abstractive summaries. To illustrate this observation, we show some examples in Figure 3.1, which are some top story summaries or headlines from the channel “Technology” of CNN. After analyzing the summaries carefully, we can find some common structures from them, such as “**What**”, “**What-Happened**”, “**Who Action What**”, etc. For example, the summary “*Apple sues Qualcomm for nearly \$1 billion*” can be structuralized as “Who (Apple) Action (sues) What (Qualcomm)”. Similarly, the summaries “[*Twitter*] [*fixes*] [*botched @POTUS account transfer*]”, “[*Uber*] [*to pay*] [*\$20 million*] for [*misleading drivers*]”, and “[*Bipartisan bill*] [*aims to*] [*reform*] [*H-1B visa system*]” also follow the structure of “Who Action What”. The summary “*The emergence of the ‘cyber cold war’*” matches with the structure of “What”, and the summary “*St.*

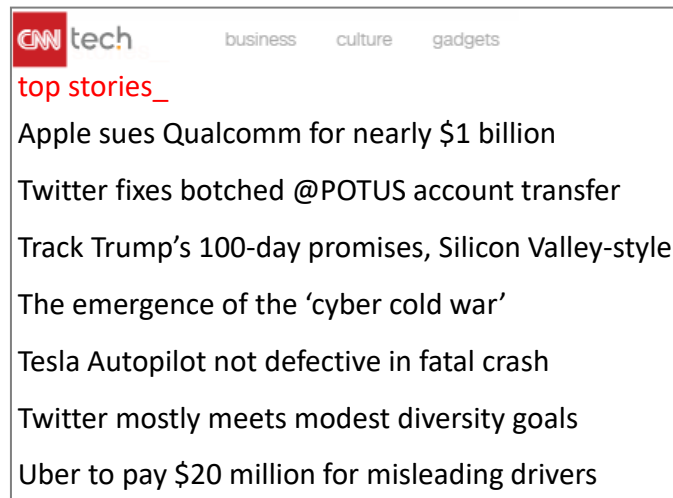


Figure 3.1: Headlines of the top stories from the channel “Technology” of CNN.

Louis’ public library computers hacked” follows the structure of “What-Happened”.

Intuitively, if we can incorporate the latent structure information of summaries into the abstractive summarization model, it will improve the quality of the generated summaries. However, very few existing works specifically consider the latent structure information of summaries in their summarization models. Several research works employ topic models to capture the latent information from source documents or sentences [12, 130]. However, they only use the latent information to conduct the estimation of sentence salience and improve the performance of extractive summarization. In contrast, our purpose is to model and learn the latent structure information from the target summaries and use it to enhance the performance of abstractive summarization. Although a very popular neural network based sequence-to-sequence (seq2seq) framework as shown in Figure 2.1 has been proposed to tackle the abstractive summarization problem [15, 40, 83, 99, 111, 112, 118, 150], the calculation of the internal decoding states is entirely deterministic. The deterministic transformations in these discriminative models lead to limitations on the representation ability of the latent structure information. Miao and Blunsom [92]

extended the seq2seq framework and proposed a generative model to capture the latent summary information, but they did not consider the recurrent dependencies in their generative model leading to limited representation ability.

To tackle the above mentioned problems, we design a new framework based on sequence-to-sequence oriented encoder-decoder model equipped with a latent structure modeling component. We employ Variational Auto-Encoders (VAEs) [54, 110] as the base model for our generative framework which can handle the inference problem associated with complex generative modeling. However, the standard framework of VAEs is not designed for sequence modeling related tasks. Inspired by [20], we add historical dependencies on the latent variables of VAEs and propose a deep recurrent generative decoder (DRGD) for latent structure modeling. Then the standard discriminative deterministic decoder and the recurrent generative decoder are integrated into a unified decoding framework. The target summaries will be decoded based on both the discriminative deterministic variables and the generative latent structural information. All the neural parameters are learned by back-propagation in an end-to-end training paradigm.

3.2 Framework Description

3.2.1 Overview

As shown in Figure 3.2, the basic framework of our approach is a neural network based encoder-decoder framework for sequence-to-sequence learning. The input is a variable-length sequence $X = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m\}$ representing the source text. The word embedding \mathbf{x}_t is initialized randomly and learned during the optimization process. The output is also a sequence $Y = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n\}$, which represents the generated abstractive summaries. Gated Recurrent Unit (GRU) [17] is employed as the basic sequence modeling component for the encoder and the decoder. For

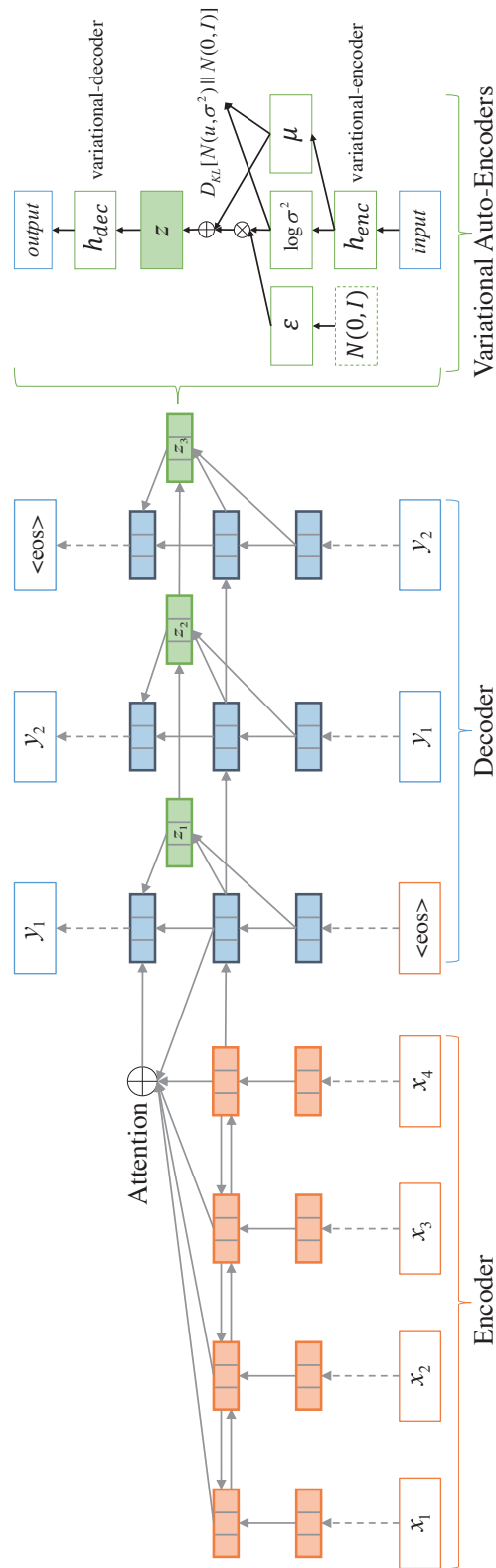


Figure 3.2: Our deep recurrent generative decoder (DRGD) for latent structure modeling.

latent structure modeling, we add historical dependencies on the latent variables of Variational Auto-Encoders (VAEs) and propose a deep recurrent generative decoder (DRGD) to distill the complex latent structures implied in the target summaries of the training data. Finally, the abstractive summaries will be decoded out based on both the discriminative deterministic variables H and the generative latent structural information Z .

3.2.2 Recurrent Generative Decoder

Assume that we have obtained the source text representation $\mathbf{h}^e \in \mathbb{R}^{k_h}$. The purpose of the decoder is to translate this source code \mathbf{h}^e into a series of hidden states $\{\mathbf{h}_1^d, \mathbf{h}_2^d, \dots, \mathbf{h}_n^d\}$, and then revert these hidden states to an actual word sequence and generate the summary.

For standard recurrent decoders, at each time step t , the hidden state $\mathbf{h}_t^d \in \mathbb{R}^{k_h}$ is calculated using the dependent input symbol $\mathbf{y}_{t-1} \in \mathbb{R}^{k_w}$ and the previous hidden state \mathbf{h}_{t-1}^d :

$$\mathbf{h}_t^d = f(\mathbf{y}_{t-1}, \mathbf{h}_{t-1}^d) \quad (3.1)$$

where $f(\cdot)$ is a recurrent neural network such as vanilla RNN, Long Short-Term Memory (LSTM) [43], and Gated Recurrent Unit (GRU) [17]. No matter which one we use for $f(\cdot)$, the common transformation operation is as follows:

$$\mathbf{h}_t^d = g(\mathbf{W}_{yh}^d \mathbf{y}_{t-1} + \mathbf{W}_{hh}^d \mathbf{h}_{t-1}^d + \mathbf{b}_h^d) \quad (3.2)$$

where $\mathbf{W}_{yh}^d \in \mathbb{R}^{k_h \times k_w}$ and $\mathbf{W}_{hh}^d \in \mathbb{R}^{k_h \times k_h}$ are the linear transformation matrices. \mathbf{b}_h^d is the bias. k_h is the dimension of the hidden layers, and k_w is the dimension of the word embeddings. $g(\cdot)$ is the non-linear activation function. From Equation 3.2, we can see that all the transformations are deterministic, which leads to a deterministic recurrent hidden state h_t^d . From our investigations, we find that the representational

power of such deterministic variables are limited. Some more complex latent structures in the target summaries, such as the high-level syntactic features and latent topics, cannot be modeled effectively by the deterministic operations and variables.

Recently, a generative model called Variational Auto-Encoders (VAEs) [54, 110] shows strong capability in modeling latent random variables and improves the performance of tasks in different fields such as sentence generation [9] and image generation [39]. However, the standard VAEs is not designed for modeling sequence directly. Inspired by [20], we extend the standard VAEs by introducing the historical latent variable dependencies to make it be capable of modeling sequence data. Our proposed latent structure modeling framework can be viewed as a sequence generative model which can be divided into two parts: inference (variational-encoder) and generation (variational-decoder). As shown in the decoder component of Figure 3.2, the input of the original VAEs only contains the observed variable \mathbf{y}_t , and the variational-encoder can map it to a latent variable $\mathbf{z} \in \mathbb{R}^{k_z}$, which can be used to reconstruct the original input. For the task of summarization, in the sequence decoder component, the previous latent structure information needs to be considered for constructing more effective representations for the generation of the next state.

For the inference stage, the variational-encoder can map the observed variable $\mathbf{y}_{<t}$ and the previous latent structure information $\mathbf{z}_{<t}$ to the posterior probability distribution of the latent structure variable $p_\theta(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})$. It is obvious that this is a recurrent inference process in which \mathbf{z}_t contains the historical dynamic latent structure information. Compared with the variational inference process $p_\theta(\mathbf{z}_t|\mathbf{y}_t)$ of the typical VAEs model, the recurrent framework can extract more complex and effective latent structure features implied in the sequence data.

For the generation process, based on the latent structure variable \mathbf{z}_t , the target word y_t at the time step t is drawn from a conditional probability distribution

$p_\theta(\mathbf{y}_t|\mathbf{z}_t)$. The target is to maximize the probability of each generated summary $y = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_T\}$ based on the generation process according to:

$$p_\theta(y) = \prod_{t=1}^T \int p_\theta(\mathbf{y}_t|\mathbf{z}_t)p_\theta(\mathbf{z}_t)d\mathbf{z}_t \quad (3.3)$$

For the purpose of solving the intractable integral of the marginal likelihood as shown in Equation 3.3, a recognition model $q_\phi(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})$ is introduced as an approximation to the intractable true posterior $p_\theta(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})$. The recognition model parameters ϕ and the generative model parameters θ can be learned jointly. The aim is to reduce the Kullback-Leibler divergence (KL) between $q_\phi(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})$ and $p_\theta(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})$:

$$\begin{aligned} & D_{KL}[q_\phi(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})||p_\theta(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})] \\ &= \int_{\mathbf{z}} q_\phi(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t}) \log \frac{q_\phi(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})}{p_\theta(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})} d\mathbf{z} \\ &= \mathbb{E}_{q_\phi(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})}[\log q_\phi(\mathbf{z}_t|\cdot) - \log p_\theta(\mathbf{z}_t|\cdot)] \end{aligned}$$

where \cdot denotes the conditional variables $\mathbf{y}_{<t}$ and $\mathbf{z}_{<t}$. Bayes rule is applied to $p_\theta(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})$,

$$\begin{aligned} & D_{KL}[q_\phi(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})||p_\theta(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})] \\ &= \log p_\theta(\mathbf{y}_{<t}) + \mathbb{E}_{q_\phi(\mathbf{z}_t|\cdot)}[\log q_\phi(\mathbf{z}_t|\cdot) \\ & \quad - \log p_\theta(\mathbf{y}_{<t}|\mathbf{z}_t) - \log p_\theta(\mathbf{z}_t)] \end{aligned} \quad (3.4)$$

and we can extract $\log p_\theta(\mathbf{z})$ from the expectation, transfer the expectation term $\mathbb{E}_{q_\phi(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})}$ back to KL-divergence, and rearrange all the terms. Consequently the

following holds:

$$\begin{aligned} \log p_{\theta}(\mathbf{y}_{<t}) &= D_{KL}[q_{\phi}(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})||p_{\theta}(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})] \\ &\quad + \mathbb{E}_{q_{\phi}(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})}[\log p_{\theta}(\mathbf{y}_{<t}|\mathbf{z}_t)] \\ &\quad - D_{KL}[q_{\phi}(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})||p_{\theta}(\mathbf{z}_t)] \end{aligned} \quad (3.5)$$

Let $\mathcal{L}(\theta, \phi; y)$ represent the last two terms from the right part of Equation 3.5:

$$\begin{aligned} \mathcal{L}(\theta, \phi; y) &= \mathbb{E}_{q_{\phi}(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})} \left\{ \sum_{t=1}^T \log p_{\theta}(\mathbf{y}_t|\mathbf{z}_t) \right. \\ &\quad \left. - D_{KL}[q_{\phi}(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})||p_{\theta}(\mathbf{z}_t)] \right\} \end{aligned} \quad (3.6)$$

Since the first KL-divergence term of Equation 3.5 is non-negative, we have $\log p_{\theta}(\mathbf{y}_{<t}) \geq \mathcal{L}(\theta, \phi; y)$ meaning that $\mathcal{L}(\theta, \phi; y)$ is a lower bound (the objective to be maximized) on the marginal likelihood. In order to differentiate and optimize the lower bound $\mathcal{L}(\theta, \phi; y)$, following the core idea of VAEs, we use a neural network framework for the probabilistic encoder $q_{\phi}(\mathbf{z}_t|\mathbf{y}_{<t}, \mathbf{z}_{<t})$ for better approximation.

3.2.3 Abstractive Summary Generation

We also design a neural network based framework to conduct the variational inference and generation for the recurrent generative decoder component similar to some design in previous works [39, 54, 110]. The encoder component and the decoder component are integrated into a unified abstractive summarization framework. Considering that GRU has comparable performance but with less parameters and more efficient computation, we employ GRU as the basic recurrent model which

updates the variables according to the following operations:

$$\begin{aligned}
\mathbf{r}_t &= \sigma(\mathbf{W}_{xr}\mathbf{x}_t + \mathbf{W}_{hr}\mathbf{h}_{t-1} + \mathbf{b}_r) \\
\mathbf{z}_t &= \sigma(\mathbf{W}_{xz}\mathbf{x}_t + \mathbf{W}_{hz}\mathbf{h}_{t-1} + \mathbf{b}_z) \\
\mathbf{g}_t &= \tanh(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{W}_{hh}(\mathbf{r}_t \odot \mathbf{h}_{t-1}) + \mathbf{b}_h) \\
\mathbf{h}_t &= \mathbf{z}_t \odot \mathbf{h}_{t-1} + (1 - \mathbf{z}_t) \odot \mathbf{g}_t
\end{aligned} \tag{3.7}$$

where \mathbf{r}_t is the reset gate, \mathbf{z}_t is the update gate. \odot denotes the element-wise multiplication. \tanh is the hyperbolic tangent activation function.

As shown in the left block of Figure 3.2, the encoder is designed based on bidirectional recurrent neural networks. Let \mathbf{x}_t be the word embedding vector of the t -th word in the source sequence. GRU maps \mathbf{x}_t and the previous hidden state \mathbf{h}_{t-1} to the current hidden state \mathbf{h}_t in feed-forward direction and back-forward direction respectively:

$$\begin{aligned}
\vec{\mathbf{h}}_t &= GRU(x_t, \vec{\mathbf{h}}_{t-1}) \\
\overleftarrow{\mathbf{h}}_t &= GRU(x_t, \overleftarrow{\mathbf{h}}_{t-1})
\end{aligned} \tag{3.8}$$

Then the final hidden state $\mathbf{h}_t^e \in \mathbb{R}^{2k_h}$ is concatenated using the hidden states from the two directions:

$$\mathbf{h}_t^e = \vec{\mathbf{h}}_t || \overleftarrow{\mathbf{h}}_t \tag{3.9}$$

As shown in the middle block of Figure 3.2, the decoder consists of two components: discriminative deterministic decoding and generative latent structure modeling.

The discriminative deterministic decoding is an improved attention modeling based recurrent sequence decoder. The first hidden state \mathbf{h}_1^d is initialized using the average of all the source input states:

$$\mathbf{h}_1^d = \frac{1}{T^e} \sum_{t=1}^{T^e} \mathbf{h}_t^e \tag{3.10}$$

where \mathbf{h}_t^e is the source input hidden state. T^e is the input sequence length. The deterministic decoder hidden state \mathbf{h}_t^d is calculated using two layers of GRUs. On the first layer, the hidden state is calculated only using the current input word embedding \mathbf{y}_{t-1} and the previous hidden state $\mathbf{h}_{t-1}^{d_1}$:

$$\mathbf{h}_t^{d_1} = GRU_1(\mathbf{y}_{t-1}, \mathbf{h}_{t-1}^{d_1}) \quad (3.11)$$

where the superscript d_1 denotes the first decoder GRU layer. Then the attention weights at the time step t are calculated based on the relationship of $\mathbf{h}_t^{d_1}$ and all the source hidden states $\{\mathbf{h}_i^e\}$. Let $a_{i,j}$ be the attention weight between $\mathbf{h}_i^{d_1}$ and \mathbf{h}_j^e , which can be calculated using the following formulation:

$$\begin{aligned} a_{i,j} &= \frac{\exp(e_{i,j})}{\sum_{j'=1}^{T^e} \exp(e_{i,j'})} \\ e_{i,j} &= \mathbf{v}^T \tanh(\mathbf{W}_{hh}^d \mathbf{h}_i^{d_1} + \mathbf{W}_{hh}^e \mathbf{h}_j^e + \mathbf{b}_a) \end{aligned} \quad (3.12)$$

where $\mathbf{W}_{hh}^d \in \mathbb{R}^{k_h \times k_h}$, $\mathbf{W}_{hh}^e \in \mathbb{R}^{k_h \times 2k_h}$, $\mathbf{b}_a \in \mathbb{R}^{k_h}$, and $\mathbf{v} \in \mathbb{R}^{k_h}$. The attention context is obtained by the weighted linear combination of all the source hidden states:

$$\mathbf{c}_t = \sum_{j'=1}^{T^e} a_{t,j'} \mathbf{h}_{j'}^e \quad (3.13)$$

The final deterministic hidden state $\mathbf{h}_t^{d_2}$ is the output of the second decoder GRU layer, jointly considering the word \mathbf{y}_{t-1} , the previous hidden state $\mathbf{h}_{t-1}^{d_2}$, and the attention context \mathbf{c}_t :

$$\mathbf{h}_t^{d_2} = GRU_2(\mathbf{y}_{t-1}, \mathbf{h}_{t-1}^{d_2}, \mathbf{c}_t) \quad (3.14)$$

For the component of recurrent generative model, inspired by some ideas in previous works [39, 54, 110], we assume that both the prior and posterior of the latent

variables are Gaussian, i.e., $p_\theta(\mathbf{z}_t) = \mathcal{N}(0, \mathbf{I})$ and $q_\phi(\mathbf{z}_t | \mathbf{y}_{<t}, \mathbf{z}_{<t}) = \mathcal{N}(\mathbf{z}_t; \boldsymbol{\mu}, \boldsymbol{\sigma}^2 \mathbf{I})$, where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ denote the variational mean and standard deviation respectively, which can be calculated via a multilayer perceptron. Precisely, given the word embedding \mathbf{y}_{t-1} , the previous latent structure variable \mathbf{z}_{t-1} , and the previous deterministic hidden state \mathbf{h}_{t-1}^d , we first project it to a new hidden space:

$$\mathbf{h}_t^{e_z} = g(\mathbf{W}_{yh}^{e_z} \mathbf{y}_{t-1} + \mathbf{W}_{zh}^{e_z} \mathbf{z}_{t-1} + \mathbf{W}_{hh}^{e_z} \mathbf{h}_{t-1}^d + \mathbf{b}_h^{e_z})$$

where $\mathbf{W}_{yh}^{e_z} \in \mathbb{R}^{k_h \times k_w}$, $\mathbf{W}_{zh}^{e_z} \in \mathbb{R}^{k_h \times k_z}$, $\mathbf{W}_{hh}^{e_z} \in \mathbb{R}^{k_h \times k_h}$, and $\mathbf{b}_h^{e_z} \in \mathbb{R}^{k_h}$. g is the sigmoid activation function: $\sigma(\mathbf{x}) = 1/(1 + e^{-\mathbf{x}})$. Then the Gaussian parameters $\boldsymbol{\mu}_t \in \mathbb{R}^{k_z}$ and $\boldsymbol{\sigma}_t \in \mathbb{R}^{k_z}$ can be obtained via a linear transformation based on $\mathbf{h}_t^{e_z}$:

$$\begin{aligned} \boldsymbol{\mu}_t &= \mathbf{W}_{h\mu}^{e_z} \mathbf{h}_t^{e_z} + \mathbf{b}_\mu^{e_z} \\ \log(\boldsymbol{\sigma}_t^2) &= \mathbf{W}_{h\sigma}^{e_z} \mathbf{h}_t^{e_z} + \mathbf{b}_\sigma^{e_z} \end{aligned} \quad (3.15)$$

The latent structure variable $\mathbf{z}_t \in \mathbb{R}^{k_z}$ can be calculated using the reparameterization trick:

$$\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad \mathbf{z}_t = \boldsymbol{\mu}_t + \boldsymbol{\sigma}_t \otimes \boldsymbol{\varepsilon} \quad (3.16)$$

where $\boldsymbol{\varepsilon} \in \mathbb{R}^{k_z}$ is an auxiliary noise variable. The process of inference for finding \mathbf{z}_t based on neural networks can be treated as a variational encoding process.

To generate summaries precisely, we first integrate the recurrent generative decoding component with the discriminative deterministic decoding component, and map the latent structure variable \mathbf{z}_t and the deterministic decoding hidden state $\mathbf{h}_t^{d_2}$ to a new hidden variable:

$$\mathbf{h}_t^{d_y} = \tanh(\mathbf{W}_{zh}^{d_y} \mathbf{z}_t + \mathbf{W}_{hh}^{d_y} \mathbf{h}_t^{d_2} + \mathbf{b}_h^{d_y}) \quad (3.17)$$

Given the combined decoding state $\mathbf{h}_t^{d_y}$ at the time t , the probability of gener-

ating any target word y_t is given as follows:

$$\mathbf{y}_t = \varsigma(\mathbf{W}_{hy}^d \mathbf{h}_t^{d_y} + \mathbf{b}_{hy}^d) \quad (3.18)$$

where $\mathbf{W}_{hy}^d \in \mathbb{R}^{k_y \times k_h}$ and $\mathbf{b}_{hy}^d \in \mathbb{R}^{k_y}$. $\varsigma(\cdot)$ is the softmax function. Finally, we use a beam search algorithm [56] for decoding and generating the best summary.

3.2.4 Learning

Although the proposed model contains a recurrent generative decoder, the whole framework is fully differentiable. As shown in Section 3.2.3, both the recurrent deterministic decoder and the recurrent generative decoder are designed based on neural networks. Therefore, all the parameters in our model can be optimized in an end-to-end paradigm using back-propagation. We use $\{X\}_N$ and $\{Y\}_N$ to denote the training source and target sequence. Generally, the objective of our framework consists of two terms. One term is the negative log-likelihood of the generated summaries, and the other one is the variational lower bound $\mathcal{L}(\theta, \phi; Y)$ mentioned in Equation 3.6. Since the variational lower bound $\mathcal{L}(\theta, \phi; Y)$ also contains a likelihood term, we can merge it with the likelihood term of summaries. The final objective function, which needs to be minimized, is formulated as follows:

$$\begin{aligned} \mathcal{J} = \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^T \left\{ -\log \left[p(y_t^{(n)} | y_{<t}^{(n)}, X^{(n)}) \right] \right. \\ \left. + D_{KL} \left[q_\phi(\mathbf{z}_t^{(n)} | \mathbf{y}_{<t}^{(n)}, \mathbf{z}_{<t}^{(n)}) \| p_\theta(\mathbf{z}_t^{(n)}) \right] \right\} \end{aligned} \quad (3.19)$$

3.3 Experimental Setup

3.3.1 Datasets

We train and evaluate our framework on three popular datasets.

Gigawords is an English sentence summarization dataset prepared based on Annotated Gigawords¹ by extracting the first sentence from articles with the headline to form a source-summary pair. We directly download the prepared dataset used in Rush et al. [111]. It roughly contains 3.8M training pairs, 190K validation pairs, and 2,000 test pairs. The test set is identical to the one used in all the comparative baseline methods.

DUC-2004² is another English dataset only used for testing in our experiments. It contains 500 news documents from the New York Times and Associated Press Wire services. Each document contains 4 model summaries written by experts. The length of the summary is limited to 75 bytes.

LCSTS [45] is a large-scale Chinese short text summarization dataset, consisting of pairs of (short text, summary) collected from Sina Weibo³. We take Part-I as the training set, Part-II as the development set, and Part-III as the test set. There is a score in range 1 ~ 5 labeled by human to indicate how relevant an article and its summary is. We only reserve those pairs with scores no less than 3. The size of the three sets are 2.4M, 8.7k, and 725 respectively. In our experiments, we only take Chinese character sequence as input, without performing word segmentation.

3.3.2 Evaluation Metrics

We use ROUGE (Recall-Oriented Understudy for Gisty Evaluation) [79] as our evaluation metric. The basic idea of ROUGE is to count the number of overlap-

¹<https://catalog.ldc.upenn.edu/ldc2012t21>

²<http://duc.nist.gov/duc2004>

³<http://www.weibo.com>

ping units between generated summaries and the reference summaries. There are several variants of ROUGE according to the different semantic units used for evaluation, including ROUGE-N (n-grams), ROUGE-L (the longest common sequence), ROUGE-SU (skip-bigrams and uni-grams). For example, the most commonly used ROUGE-N is computed respectively as follows:

$$ROUGE-N_{recall} = \frac{\sum_{S \in \{\text{ReferenceSummaries}\}} \sum_{\text{gram}_n \in S} \text{Count}_{\text{match}}(\text{gram}_n)}{\sum_{S \in \{\text{ReferenceSummaries}\}} \sum_{\text{gram}_n \in S} \text{Count}(\text{gram}_n)} \quad (3.20)$$

$$ROUGE-N_{precision} = \frac{\sum_{S \in \{\text{ReferenceSummaries}\}} \sum_{\text{gram}_n \in S} \text{Count}_{\text{match}}(\text{gram}_n)}{\sum_{S \in \{\text{SystemSummaries}\}} \sum_{\text{gram}_n \in S} \text{Count}(\text{gram}_n)} \quad (3.21)$$

$$ROUGE-N_{F\text{-measure}} = \frac{2 \times ROUGE-N_{precision} \times ROUGE-N_{recall}}{ROUGE-N_{precision} + ROUGE-N_{recall}} \quad (3.22)$$

Considering that many previous works employ F-measures of ROUGE-1 (R-1), ROUGE-2 (R-2) and ROUGE-L (R-L) as the metrics for evaluation, we also report the results under these three metrics in our work.

3.3.3 Comparative Methods

We compare our model with some baselines and state-of-the-art methods. Because the datasets are quite standard, so we just extract the results from their papers. Therefore the baseline methods on different datasets may be slightly different.

- **TOPIARY** [144] is the best on DUC2004 Task-1 for compressive text summarization. It combines a system using linguistic based transformations and an unsupervised topic detection algorithm for compressive text summarization.

- **MOSES+** [111] uses a phrase-based statistical machine translation system trained on Gigaword to produce summaries. It also augments the phrase table with “deletion” rules to improve the baseline performance, and MERT is also used to improve the quality of generated summaries.
- **ABS** and **ABS+** [111] are both the neural network based models with local attention modeling for abstractive sentence summarization. ABS+ is trained on the Gigaword corpus, but combined with an additional log-linear extractive summarization model with handcrafted features.
- **RNN** and **RNN-context** [45] are two seq2seq architectures. RNN-context integrates attention mechanism to model the context.
- **CopyNet** [40] integrates a copying mechanism into the sequence-to-sequence framework.
- **RNN-distract** [15] uses a new attention mechanism by distracting the historical attention in the decoding steps.
- **RAS-LSTM** and **RAS-Elman** [18] both consider words and word positions as input and use convolutional encoders to handle the source information. For the attention based sequence decoding process, RAS-Elman selects Elman RNN [26] as decoder, and RAS-LSTM selects Long Short-Term Memory architecture [43].
- **LenEmb** [51] uses a mechanism to control the summary length by considering the length embedding vector as the input.
- **ASC+FSC₁** [92] uses a generative model with attention mechanism to conduct the sentence compression problem. The model first draws a latent summary sentence from a background language model, and then subsequently draws the observed sentence conditioned on this latent summary.

- **lvt2k-1sent** and **lvt5k-1sent** [99] utilize a trick to control the vocabulary size to improve the training efficiency.

3.3.4 Experimental Settings

For the experiments on the English dataset Gigawords, we set the dimension of word embeddings to 300, and the dimension of hidden states and latent variables to 500. The maximum length of documents and summaries is 100 and 50 respectively. The batch size of mini-batch training is 256. For DUC-2004, the maximum length of summaries is 75 bytes. For the dataset of LCSTS, the dimension of word embeddings is 350. We also set the dimension of hidden states and latent variables to 500. The maximum length of documents and summaries is 120 and 25 respectively, and the batch size is also 256. The beam size of the decoder was set to be 10.

We used mini-batch stochastic gradient descent (SGD) to optimize the log-likelihood. Adadelta [146] with hyperparameter $\rho = 0.95$ and $\epsilon = 1e - 6$ is used for gradient based optimization. Gradient clipping is adopted by scaling gradients when the norm exceeds a threshold of 10. Our neural network based framework is implemented using Theano [120] on a single Tesla K80 GPU.

3.4 Results and Discussions

3.4.1 ROUGE Evaluation

Table 3.1: ROUGE-F1 on validation sets

| Dataset | System | ROUGE-1 | ROUGE-2 | ROUGE-L |
|---------|--------|--------------|--------------|--------------|
| GIGA | StanD | 32.69 | 15.29 | 30.60 |
| | DRGD | 36.25 | 17.61 | 33.55 |
| LCSTS | StanD | 33.88 | 21.49 | 31.05 |
| | DRGD | 36.71 | 24.00 | 34.10 |

Table 3.2: ROUGE-F1 on Gigawords

| System | ROUGE-1 | ROUGE-2 | ROUGE-L |
|------------------------|--------------|--------------|--------------|
| ABS | 29.55 | 11.32 | 26.42 |
| ABS+ | 29.78 | 11.89 | 26.97 |
| RAS-LSTM | 32.55 | 14.70 | 30.03 |
| RAS-Elman | 33.78 | 15.97 | 31.15 |
| ASC + FSC ₁ | 34.17 | 15.94 | 31.92 |
| lvt2k-1sent | 32.67 | 15.59 | 30.64 |
| lvt5k-1sent | 35.30 | 16.64 | 32.62 |
| DRGD | 36.27 | 17.57 | 33.62 |

Table 3.3: ROUGE-Recall on DUC2004

| System | ROUGE-1 | ROUGE-2 | ROUGE-L |
|-------------|--------------|-------------|--------------|
| TOPIARY | 25.12 | 6.46 | 20.12 |
| MOSES+ | 26.50 | 8.13 | 22.85 |
| ABS | 26.55 | 7.06 | 22.05 |
| ABS+ | 28.18 | 8.49 | 23.81 |
| RAS-Elman | 28.97 | 8.26 | 24.06 |
| RAS-LSTM | 27.41 | 7.69 | 23.06 |
| LenEmb | 26.73 | 8.39 | 23.88 |
| lvt2k-1sen | 28.35 | 9.46 | 24.59 |
| lvt5k-1sen | 28.61 | 9.42 | 25.24 |
| DRGD | 28.99 | 9.72 | 25.28 |

We first depict the performance of our model DRGD by comparing to the standard decoders (StanD) of our own implementation. The comparison results on the validation datasets of Gigawords and LCSTS are shown in Table 3.1. From the results we can see that our proposed generative decoders DRGD can obtain obvious improvements on abstractive summarization than the standard decoders. Actually, the performance of the standard decoders is similar with those mentioned popular baseline methods.

The results on the English datasets of Gigawords and DUC-2004 are shown in Table 3.2 and Table 3.3 respectively. Our model DRGD achieves the best summarization performance on all the ROUGE metrics. Although ASC+FSC₁ also uses a

Table 3.4: ROUGE-F1 on LCSTS

| System | ROUGE-1 | ROUGE-2 | ROUGE-L |
|---------------|----------------|----------------|----------------|
| RNN | 21.50 | 8.90 | 18.60 |
| RNN-context | 29.90 | 17.40 | 27.20 |
| CopyNet | 34.40 | 21.60 | 31.30 |
| RNN-distract | 35.20 | 22.60 | 32.50 |
| DRGD | 36.99 | 24.15 | 34.21 |

generative method to model the latent summary variables, the representation ability is limited and it cannot bring in noticeable improvements. It is worth noting that the methods lvt2k-1sent and lvt5k-1sent [99] utilize linguistic features such as parts-of-speech tags, named-entity tags, and TF and IDF statistics of the words as part of the document representation. In fact, extracting all such features is a time consuming work, especially on large-scale datasets such as Gigawords. lvt2k and lvt5k are not end-to-end style models and are more complicated than our model in practical applications.

The results on the Chinese dataset LCSTS are shown in Table 3.4. Our model DRGD also achieves the best performance. Although CopyNet employs a copying mechanism to improve the summary quality and RNN-distract considers attention information diversity in their decoders, our model is still better than those two methods demonstrating that the latent structure information learned from target summaries indeed plays a role in abstractive summarization. We also believe that integrating the copying mechanism and coverage diversity in our framework will further improve the summarization performance.

3.4.2 Summary Case Analysis

In order to analyze the reasons of improving the performance, we compare the generated summaries by DRGD and the standard decoders StanD used in some other works such as [18]. The source texts, golden summaries, and the generated sum-

Table 3.5: Examples of the generated summaries.

| |
|--|
| <p>S(1): hosts wuhan won the men ’s soccer title by beating beijing shunyi #-# here at the #th chinese city games on friday.</p> <p>Golden: hosts wuhan wins men ’s soccer title at chinese city games.</p> <p>Stand: results of men ’s volleyball at chinese city games.</p> <p>DRGD: wuhan wins men ’s soccer title at chinese city games.</p> |
| <p>S(2): UNK and the china meteorological administration tuesday signed an agreement here on long - and short-term cooperation in projects involving meteorological satellites and satellite meteorology.</p> <p>Golden: UNK china to cooperate in meteorology.</p> <p>Stand: weather forecast for major chinese cities.</p> <p>DRGD: china to cooperate in meteorological satellites.</p> |
| <p>S(3): the rand gained ground against the dollar at the opening here wednesday , to #.# to the greenback from #.# at the close tuesday.</p> <p>Golden: rand gains ground.</p> <p>Stand: rand slightly higher against dollar.</p> <p>DRGD: rand gains ground against dollar.</p> |
| <p>S(4): new zealand women are having more children and the country ’s birth rate reached its highest level in ## years , statistics new zealand said on wednesday.</p> <p>Golden: new zealand birth rate reaches ##-year high.</p> <p>Stand: new zealand women are having more children birth rate hits highest level in ## years.</p> <p>DRGD: new zealand ’s birth rate hits ##-year high.</p> |

maries are shown in Table 3.5. From the cases we can observe that DRGD can indeed capture some latent structures which are consistent with the golden summaries. For example, our result for S(1) “*Wuhan wins men’s soccer title at Chinese city games*” matches the “Who Action What” structure. However, the standard de-

coder StanD ignores the latent structures and generates some loose sentences, such as the results for S(1) “*Results of men’s volleyball at Chinese city games*” does not catch the main points. The reason is that the recurrent variational auto-encoders used in our framework have better representation ability and can capture more effective and complicated latent structures from the sequence data. Therefore, the summaries generated by DRGD have consistent latent structures with the ground truth, leading to a better ROUGE evaluation.

3.5 Summary

In this Chapter, we propose a deep recurrent generative decoder (DRGD) to improve the abstractive summarization performance. The model is a sequence-to-sequence oriented encoder-decoder framework equipped with a latent structure modeling component. Abstractive summaries are generated based on both the latent variables and the deterministic states. Extensive experiments on benchmark datasets show that DRGD achieves improvements over the state-of-the-art methods.

Chapter 4

Cascaded Attention Modeling for Multi-Document Summarization

4.1 Background

Considering the procedure of summary writing by humans, when people read, they will **remember** and **forget** part of the content. Information which is more important may make a deep impression easily. When people recall and digest what they have read to write summaries, the important information usually attracts more **attention** (*the behavioral and cognitive process of selectively concentrating on a discrete aspect of information, whether deemed subjective or objective, while ignoring other perceivable information*¹) since it may repeatedly appears in some documents, or be positioned in the beginning paragraphs.

In the context of multi-document summarization, to generate a summary sentence for a key aspect of the topic, we need to find its relevant parts in the original documents, which may attract more attention. The semantic parts with high attention weights plausibly represent and reconstruct the topic's main idea. To this end,

¹<https://en.wikipedia.org/wiki/Attention> (Apr., 2018)

we propose a cascaded neural attention model to distill salient information from the input documents in an unsupervised data reconstruction manner, which includes two components: reader and recaller. The reader is a gated recurrent neural network (LSTM or GRU) based sentence sequence encoder which can map all the sentences of the topic into a global representation, with the mechanism of remembering and forgetting. The recaller decodes the global representation into significantly fewer diversified vectors for distillation and concentration. A cascaded attention mechanism is designed by incorporating attentions on both the hidden layer (dense distributed representation of a sentence) and the output layer (sparse bag-of-words representation of summary information). It is worth noting that the output vectors of the recaller can be viewed as word salience, and the attention matrix can be used as sentence salience. Both of them are automatically learned by data reconstruction in an **unsupervised** manner. Thereafter, the word salience is fed into a coarse-grained sentence compression component. Finally, the attention weights are integrated into a phrase-based optimization framework for compressive summary generation.

In fact, the notion of “attention” has gained popularity recently in neural network modeling, which has improved the performance of many tasks such as machine translation [2, 85]. However, very few previous works employ attention mechanism to tackle MDS. Rush et al. [111] and Nallapati et al. [99] employed attention-based sequence-to-sequence (seq2seq) framework only for sentence summarization. Gu et al. [40], Cheng and Lapata [16], and Nallapati et al. [99] also utilized seq2seq based framework with attention modeling for short text or single document summarization. Different from their works, our framework aims at conducting multi-document summarization in an unsupervised manner.

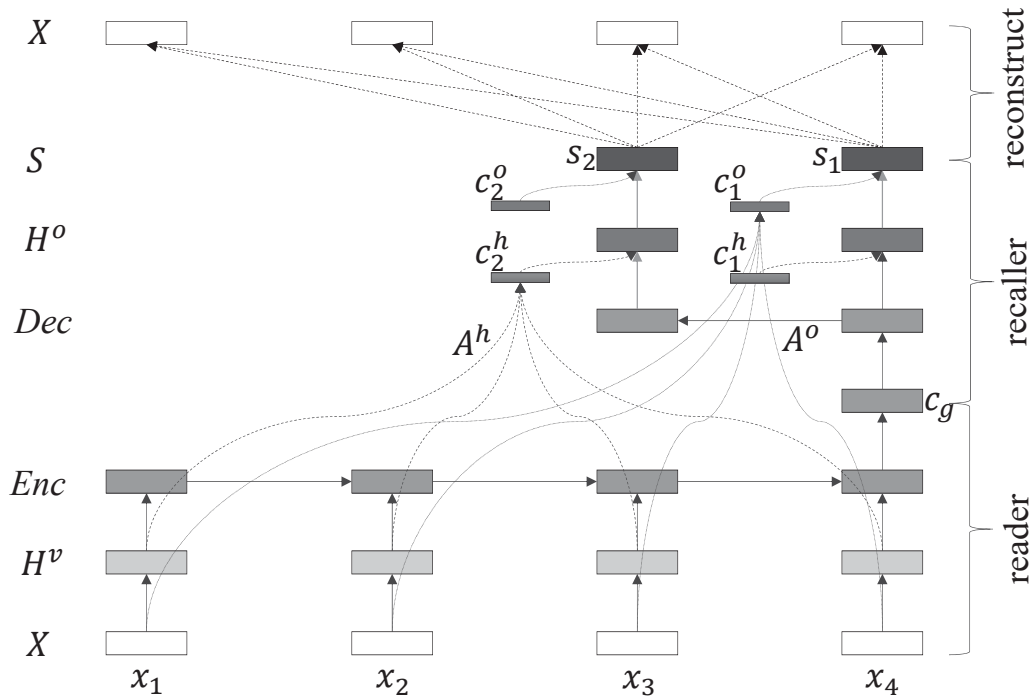


Figure 4.1: Our cascaded attention based unsupervised information distillation framework. X is the original input sentence sequence of a topic. H^i is the hidden vectors of sentences. “*Enc*” and “*Dec*” represent the RNN-based encoding and decoding layer respectively. c_g is the global representation for the whole topic. A^h and A^o are the distilled attention matrices for the hidden layer and the output layer respectively, representing the salience of sentences. H^o is the output hidden layer. s_1 and s_2 are the distilled condensed vectors representing the salience of words. Note that they are neither origin inputs nor golden summaries.

4.2 Framework Description

4.2.1 Overview

Our framework has two phases, namely, information distillation for finding salient words/sentences, and compressive summary generation. For the first phase, our cascaded neural attention model consists of two components: reader and recaller as shown in Figure 4.1. The reader component reads in all the sentences in the document set corresponding to the topic/event. The information distillation happens

in the recaller component where only the most important information is preserved. Precisely, the recaller outputs fewer vectors s than that of the input sentences x for the reader.

After the learning of the neural attention model finishes, the obtained salience information will be used in the second phase for compressive summary generation. This phase consists of two components: (i) the coarse-grained sentence compression component which can filter the trivial information based on the output vectors S from the neural attention model; (ii) the unified phrase-based optimization method for summary generation in which the attention matrix A^o is used to conduct fine-grained compression and summary construction.

4.2.2 Attention Modeling for Distillation

Reader

In the reader stage, for each topic, we extract all the sentences $X = \{x_1, x_2, \dots, x_m\}$ from the set of input documents corresponding to a topic and generate a sentence sequence with length m . The sentence order is the same as the original order of the documents. Then the reader reads the whole sequence sentence by sentence. We employ the bag-of-words (BOW) representation as the initial semantic representation for sentences. Assume that the dictionary size is k , then $x_i \in \mathbb{R}^k$.

Sparsity is one common problem for the BOW representation, especially when each vector is generated from a single sentence. Moreover, downstream algorithms might suffer from the curse of dimensionality. To solve these problems, we add a hidden layer H^v (v for input layer) which is a densely distributed representation above the input layer as shown in Figure 4.1. Such distributed representation can provide better generalization than BOW representation in many different tasks [44, 52, 59, 96]. Specifically, the input hidden layer will project the input sentence vector

x_j to a new space \mathbb{R}^h according to Equation 4.1. Then we obtain a new sentence sequence $H^v = [h_1^v, h_2^v, \dots, h_m^v]$.

$$h_j^v = \tanh(W_{xh}^v x_j + b_h^v) \quad (4.1)$$

where W_{xh}^v and b_h^v are the weight and bias respectively. The superscript v means that the variables are from the input layer.

While reading the sentence sequence, the reader should have the ability of remembering and forgetting. Therefore, we employ the RNN models with various gates (input gate, forget gate, etc.) to imitate the remembering and forgetting mechanism. Then the RNN based neural encoder (the third layer in Figure 4.1) will map the whole embedding sequence to a single vector c_g which can be regarded as a global representation for the whole topic. Let t be the index of the sequence state for the sentence x_t , the hidden unit h_t^e (e for encoder RNN) of the RNN encoder can be computed as:

$$h_t^e = f(h_{t-1}^e, h_t^v) \quad (4.2)$$

where the RNN $f(\cdot)$ computes the current hidden state given the previous hidden state h_{t-1}^e and the sentence embedding h_t^v . The encoder generates hidden states $\{h_t^e\}$ over all time steps. The last state $\{h_m^e\}$ is extracted as the global representation c_g for the whole topic. The structure for $f(\cdot)$ can be either an LSTM [43] or GRU [17].

Recaller

The recaller stage is a reverse of the reader stage, but it outputs less number of vectors in S as shown in Figure 4.1. Given the global representation c_g , the past hidden state h_{t-1}^d (d for decoder RNN) from the decoder layer, an RNN based

decoder generates several hidden states according to:

$$h_t^d = f(h_{t-1}^d, c_g) \quad (4.3)$$

We use c_g to initialize the first decoder hidden state. The decoder will generate several hidden states $\{h_t^d\}$ over pre-defined time steps. Then, similar to the reader stage, we add an output hidden layer after the decoder layer:

$$h_t^o = \tanh(W_{hh}^o h_t^d + b_h^o) \quad (4.4)$$

where W_{hh}^o and b_h^o are the weight and bias respectively for the projection from h_t^d to h_t^o . Finally, the output layer maps these hidden vectors to the condensed vectors $S = [s_1, s_2, \dots, s_n]$, Each output vector s_t has the same dimension k as the input BOW vectors and is obtained as follows:

$$s_t = \sigma(W_{hs} h_t^o + b_s) \quad (4.5)$$

For the purpose of distillation and concentration, we restrict n to be very small.

Cascaded Attention Modeling

Saliency estimation for words and sentences is a crucial component in MDS, especially in the unsupervised summarization setting. We propose a cascaded attention model for information distillation to tackle the saliency estimation task for MDS. We add attention mechanism not only in the hidden layer, but also in the output layer. By this cascaded attention model, we can capture the saliency of sentences from two different and complementary vector spaces. One is the embedding space that provides better generalization, and the other one is the BOW vector space that captures more nuanced and subtle difference.

For each output hidden state h_t^o , we align it with each input hidden state h_i^v by an attention vector $a_{t,i}^h \in \mathbb{R}^m$ (recall that m is the number of input sentences). $a_{t,i}^h$ is derived by comparing h_t^o with each input sentence hidden state h_i^v :

$$a_{t,i}^h = \frac{\exp(\text{score}(h_t^o, h_i^v))}{\sum_{i'} \exp(\text{score}(h_t^o, h_{i'}^v))} \quad (4.6)$$

where $\text{score}(\cdot)$ is a content-based function to capture the relation between two vectors. Several different formulations can be used as the function $\text{score}(\cdot)$ which will be elaborated later.

Based on the alignment vectors $\{a_{t,i}^h\}$, we can create a context vector c_t^h by linearly blending the sentence hidden states $\{h_{i'}^v\}$:

$$c_t^h = \sum_{i'} a_{t,i'}^h h_{i'}^v \quad (4.7)$$

Then the output hidden state can be updated based on the context vector. Let $\tilde{h}_t^o = h_t^o$, then update the original state according to the following operation:

$$h_t^o = \tanh(W_{ch}^a c_t^h + W_{hh}^a \tilde{h}_t^o) \quad (4.8)$$

The alignment vector $a_{t,i}^h$ captures which sentence should be attended more in the hidden space when generating the condensed representation for the whole topic.

Besides the attention mechanism on the hidden layer, we also directly add attention on the output BOW layer which can capture more nuanced and subtle difference information from the BOW vector space. The hidden attention vector $a_{t,i}^h$ is integrated with the output attention by a weight $\lambda_a \in [0, 1]$:

$$\bar{a}_{t,i}^o = \frac{\exp(\text{score}(s_t, x_i))}{\sum_{i'} \exp(\text{score}(s_t, x_{i'}))} \quad (4.9)$$

$$a_{t,i}^o = \lambda_a \bar{a}_{t,i}^o + (1 - \lambda_a) a_{t,i}^h \quad (4.10)$$

The output context vector is computed as:

$$c_t^o = \sum_{i'} a_{t,i'}^o x_{i'} \quad (4.11)$$

To update the output vector s_t in Equation 4.5, we develop a different method from that of the hidden attentions. Specifically we use a weighted combination of the context vectors and the original outputs with $\lambda_c \in [0, 1]$. Let $\tilde{s}_t = s_t$, then the updated s_t is:

$$s_t = \lambda_c c_t^o + (1 - \lambda_c) \tilde{s}_t \quad (4.12)$$

The parameters λ_a and λ_c can also be learned during training.

There are several different alternatives for the function $score(\cdot)$:

$$score(h_t, h_s) = \begin{cases} h_t^T h_s & \textit{dot} \\ h_t^T W h_s & \textit{tensor} \\ v^T \tanh(W[h_t; h_s]) & \textit{concat} \end{cases} \quad (4.13)$$

Considering their behaviors as studied in [85], we adopt “*concat*” for the hidden attention layer, and “*dot*” for the output attention layer.

Unsupervised Learning

By minimizing the loss owing to using the condensed output vectors to reconstruct the original input sentence vectors, we are able to learn the solutions for all the parameters as follows.

$$\min_{\Theta} \frac{1}{2m} \sum_{i=1}^m \|x_i - \sum_{j=1}^n s_j a_{j,i}^o\|_2^2 + \lambda_s \|S\|_1 \quad (4.14)$$

where Θ denotes all the parameters in our model. In order to penalize the unimportant terms in the output vectors, we put a sparsity constraint on the rows of S using l_1 -regularization, with the weight λ_s as a scaling constant for determining its relative importance.

Let \bar{s} be the magnitude vector computed from the columns in S ($S \in \mathbb{R}^{n \times k}$). Once the training is finished, each dimension of the vector \bar{s} can be regarded as the **word salience** score. According to Equation 4.14, $s_i \in S$ is used to reconstruct the original sentence space X , and $n \ll m$ (the number of sentences in X is much more than the number of vectors in S) Therefore a large value in \bar{s} means that the corresponding word contains important information about this topic and it can serve as the word salience.

Moreover, the output layer attention matrix A^o can be regarded as containing the **sentence salience** information. Note that each output vector s_i is generated based on the cascaded attention mechanism. Assume that $a_i^o = A_{i,:}^o \in \mathbb{R}^m$ is the attention weight vector for s_i . According to Equation 4.9, a large value in a_i^o conveys a meaning that the corresponding sentence should contribute more when generating s_i . We also use the magnitude of the columns in A^o to represent the salience of sentences.

4.2.3 Compressive Summary Generation Phase

Coarse-grained Sentence Compression

Using the information distillation result from the cascaded neural attention model, we conduct coarse-grained compression for each individual sentence. Such strategy has been adopted in some multi-document summarization methods [63, 131, 140]. Our coarse-grained sentence compression jointly considers word salience obtained from the neural attention model and grammaticality constraints. First, we assign

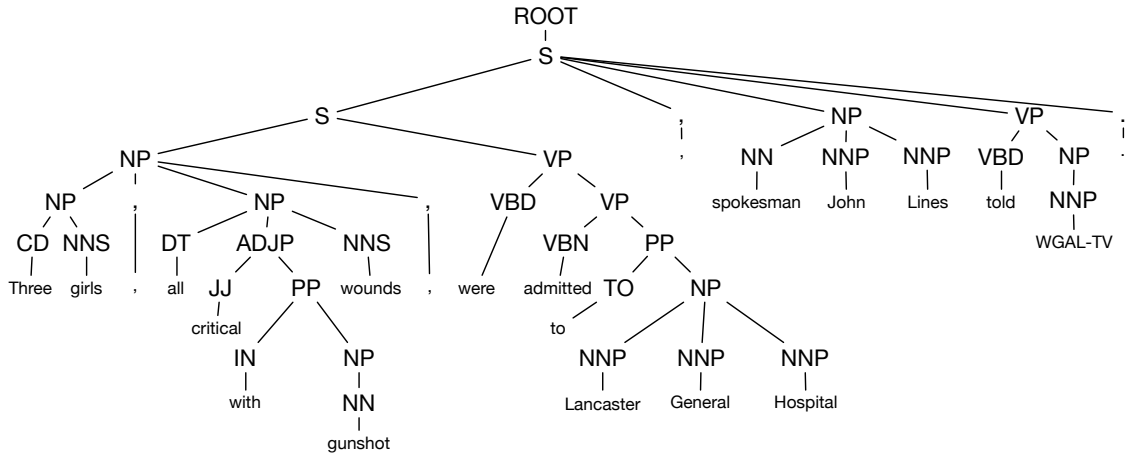


Figure 4.2: The constituency tree of a sentence.

the word salience score to each word in the sentence. For the words not in the dictionary, e.g., stop words, the score is 0, implying that these words are not important. Although many words will get a low salience score, we cannot simply delete them from the sentence since they might be critical to maintain the grammaticality. To solve this problem, we add some grammaticality constraints, such as both the subject and verb should be kept. In order to extract the grammaticality dependency from the sentence, we employ the Stanford parser [55] to generate a constituency tree for each sentence. , as shown in Figure 4.2. Inspired by Wang et al. [131], besides the word salience score and grammaticality constraints, we also design a set of rules to guide the compression. The linguistically-motivated rules are designed based on the observed obvious evidence for uncritical information from the word level to the clause level, which include news headers such as “BEIJING, Nov. 24 (Xinhua) –”, intra-sentential attribution such as “, police said Thursday”, “, he said”, etc. The information filtered by the rules will be processed according to the word salience score. Information with smaller salience score ($< \epsilon$) will be removed. We also design a function *trivial-block-labeling* to label the blocks of a sentence if it triggers the rules. Before deleting a block from the sentence, we add

Algorithm 1 Word salience guided sentence compression

Input: Sentence s , word salience Y , salience threshold ε .**Output:** Compressed sentence s' .

```

1: trivial-block-labeling according to rules in  $\Lambda$ 
2: for block  $b_i \in s$  do
3:   if  $b_i$  trigger rules in  $\Lambda$  then
4:     block salience  $a_i = 0$ 
5:     for  $w_j \in b_i$  do
6:        $a_i \leftarrow a_i + Y[w_j]$ 
7:     end for
8:     if  $a_i / \#w_i < \varepsilon$  and grammatical-checking is true then
9:       discard  $s_i$ 
10:    end if
11:  end if
12: end for
13: return  $s'$ .

```

another function *grammatical-checking* to check the grammaticality based on the constituency tree. For example, subject and object relations should be kept since deleting any word in a subject-verb-object path will result in an ungrammatical sentence. The details of the coarse-grained compression are shown in Algorithm 1, where $\#w_i$ represents the number of words in block b_i .

Phrase-based Optimization for Summary Construction

After coarse-grained compression on each single sentence as described above, we design a unified optimization method for summary generation. We consider the salience information obtained by our neural attention model and the compressed sentences in the coarse-grained compression component.

Based on the parsed constituency tree for each input sentence as shown in Figure 4.2, we extract the noun-phrases (NPs) and verb-phrases (VPs) from the tree as follows: (1) The NPs and VPs that are the direct children of the sentence node (represented by the S node) are extracted. (2) VPs (NPs) in a path on which all the

nodes are VPs (NPs) are also recursively extracted and regarded as having the same parent node S. Recursive operation in the second step will only be carried out in two levels since the phrases in the lower levels may not be able to convey a complete fact.

The salience S_i of a phrase P_i is defined as:

$$S_i = \left\{ \frac{\sum_{t \in P_i} tf(t)}{\sum_{t \in Topic} tf(t)} \right\} \times a_i \quad (4.15)$$

where a_i is the salience of the sentence containing P_i . $tf(t)$ is the frequency of the concept t (unigram/bigram) in the whole topic. Thus, S_i inherits the salience of its sentence, and also considers the importance of its concepts.

The overall objective function of our optimization formulation for selecting salient NPs and VPs is formulated as an integer linear programming (ILP) problem:

$$\max \left\{ \sum_i \alpha_i S_i - \sum_{i < j} \alpha_{ij} (S_i + S_j) R_{ij} \right\} \quad (4.16)$$

where α_i is the selection indicator for the phrase P_i , S_i is the salience scores of P_i , α_{ij} and R_{ij} is the co-occurrence indicator and the similarity of a pair of phrases (P_i , P_j) respectively. The similarity is calculated by the Jaccard Index based method. Specifically, this objective maximizes the salience score of the selected phrases as indicated by the first term, and penalizes the selection of similar phrase pairs.

In order to obtain coherent summaries with good readability, we add some constraints into the ILP framework such as sentence generation constraint:

Constraint 1. Let β_k denote the selection indicator of the sentence x_k . If any phrase from x_k is selected, $\beta_k = 1$. Otherwise, $\beta_k = 0$. For generating a compressed summary sentence, it is required that if $\beta_k = 1$, at least one NP and at least one

VP of the sentence should be selected. It is expressed as:

$$\forall P_i \in x_k, \alpha_i \leq \beta_k \wedge \sum_i \alpha_i \geq \beta_k, \quad (4.17)$$

Constraint 2. Two phrases in the same path of the constituency tree cannot be selected at the same time:

$$\text{if } \exists P_k \rightsquigarrow P_j, \text{ then } \alpha_k + \alpha_j \leq 1, \quad (4.18)$$

For example, “Three girls, all critical with gunshot wounds” and “Three girls” cannot be both selected.

Constraint 3. These constraints control the co-occurrence relation of two phrases:

$$\alpha_{ij} - \alpha_i \leq 0, \quad \alpha_{ij} - \alpha_j \leq 0, \quad \alpha_i + \alpha_j - \alpha_{ij} \leq 1; \quad (4.19)$$

The first two constraints state that if the summary includes both the units P_i and P_j , then we have to include them individually. The third constraint is the inverse of the first two.

Constraint 4. The overall length of the selected NPs and VPs is no larger than a limit L .

$$\sum_i \{l(P_i) * \alpha_i\} \leq L, \quad (4.20)$$

where $l()$ is the word-based length of a phrase.

Other constraints include sentence number, summary length, phrase co-occurrence, etc. For details, please refer to McDonald [88], Woodsend and Lapata [135], and Bing et al. [6].

The objective function and constraints are linear. Therefore the optimization can be solved by existing ILP solvers such as the simplex algorithm [21]. In the

implementation, we use a package called `lp_solve`².

Postprocessing

In the post-processing, if the total length is smaller than L , we add conjunctions such as “and” and “then” to concatenate the VPs for improving the readability of the newly generated sentences. The pseudo-timestamp of a sentence is defined as the earliest timestamp of its VPs and the sentences are ordered based on their pseudo-timestamps. After postprocessing, the final compressive summaries have been generated.

4.3 Experimental Setup

4.3.1 Datasets

DUC: Both DUC 2006 and DUC 2007 are used in our evaluation. DUC 2006 and DUC 2007 contain 50 and 45 topics respectively. Each topic has 25 news documents and 4 model summaries. The length of the model summary is limited to 250 words.

TAC: We also use TAC 2010 and TAC 2011 in our experiments. TAC 2011 is the latest standard summarization benchmark data set and it contains 44 topics. Each topic falls into one of 5 predefined event categories and contains 10 related news documents and 4 model summaries. TAC 2010 contains 46 topics from the same predefined categories. Each topic also has 10 documents and 4 model summaries. TAC 2010 is used as the parameter tuning data set of our TAC evaluation.

²<http://lpsolve.sourceforge.net/5.5/>

4.3.2 Settings

For text processing, the input sentences are represented as BOW vectors with dimension k . The dictionary is created using unigrams and named entity terms. The word salience threshold ϵ used in sentence compression is 0.005. For the neural network framework, we set the hidden size as 500. All the neural matrix parameters \mathcal{W} in hidden layers and RNN layers are initialized from a uniform distribution between $[-0.1, 0.1]$. Adadelta [146] is used for gradient based optimization. Gradient clipping is adopted by scaling gradients then the norm exceeded a threshold of 10. The maximum epoch number in the optimization procedure is 200. We limit the number of distilled vectors $n = 5$. The attention cascaded parameter λ_a and λ_c can be learned by our model. The sparsity penalty λ_s in Equation 4.14 is 0.001. Our neural network based framework is implemented using Theano [120] on a single GPU of Tesla K80. For training the attention-based distillation component, each topic can be finished in less than 1 minute.

We use ROUGE score as our evaluation metric [79] with standard options³. F-measures of ROUGE-1 (R-1), ROUGE-2 (R-2) and ROUGE-SU4 (R-SU4) are reported. For the definition of ROUGE, please refer to Section 3.3.2 of Chapter 3.

4.4 Results and Discussions

4.4.1 Effect of Existing Salience Models and Different Attention Architectures

We quantitatively evaluate the performance of different variants on the dataset of TAC 2010. The experimental results are shown in Table 4.1. Note that the summary generation phase for different methods are the same, and only the salience

³ROUGE-1.5.5.pl -n 4 -w 1.2 -m -2 4 -u -c 95 -r 1000 -f A -p 0.5 -t 0

Table 4.1: Comparisons on TAC 2010

| System | ROUGE-1 | ROUGE-2 | ROUGE-SU4 |
|-------------------|--------------|--------------|--------------|
| CW | 0.353 | 0.092 | 0.123 |
| SC | 0.346 | 0.083 | 0.116 |
| AttenC-tensor-gru | 0.339 | 0.078 | 0.115 |
| AttenC-concat-gru | 0.353 | 0.089 | 0.121 |
| AttenC-dot-lstm | 0.352 | 0.089 | 0.121 |
| AttenH-dot-gru | 0.348 | 0.086 | 0.119 |
| AttenO-dot-gru | 0.348 | 0.085 | 0.118 |
| AttenC-dot-gru | 0.359 | 0.092 | 0.124 |
| (w\o coarse-comp) | 0.351 | 0.089 | 0.122 |

estimation methods are different. Commonly used existing methods for salience estimation include: concept weight (**CW**) [6] and sparse coding (**SC**) [70]. As mentioned in Section 4.2.2, there are several alternatives for the attention scoring function $score(\cdot)$: **dot**, **tensor**, and **concat**. The comparisons using different RNN models (LSTM and GRU) with different attention scoring functions are reported. Moreover, we also design experiments to show the benefit of our cascaded attention mechanism versus the single attention method. **AttenC** denotes the cascaded attention mechanism. **AttenH** and **AttenO** represent the attention only on the hidden layer or the output layer respectively without cascaded combination.

Among all the methods, the cascaded attention model with *dot* structure achieves the best performance. The effect of different RNN models, such as LSTM and GRU, is similar. However, there are less parameters in GRU resulting in improvements for the efficiency of training. Therefore, we choose **AttenC-dot-gru** as the attention structure of our framework in the subsequent experiments. Moreover, the results without coarse-grained sentence compression (Section 4.2.3) show that the compression can indeed improve the summarization performance.

4.4.2 Main Results of Compressive MDS

To compare the performance of our framework with existing approaches, our first priority is to get the summaries produced by their systems (or get their code to produce summaries by ourselves), then we run ROUGE evaluation on them with the same option. If the summaries of the comparative systems are not available, we implement their methods and communicate with the authors to clarify some details.

We compare our system **C-Attention** with several unsupervised summarization baselines and state-of-the-art models.

- **Random** baseline selects sentences randomly for each topic.
- **Lead** baseline [132] ranks the news chronologically and extracts the leading sentences one by one.
- **TextRank** [94] and **LexRank** [27] estimate sentence salience by applying the PageRank algorithm to the sentence graph.
- **PKUTM** [65] employs manifold-ranking for sentence scoring and selection.
- **ABS-Phrase** [6] generates abstractive summaries using phrase-based optimization framework.
- **DSDR** [42] employs sparse coding method to conduct the summary sentence selection.
- **MDS-Sparse** [82] proposes a two-level sparse representation model for summarization.
- **RA-MDS** [70] employs sparse coding method to estimate the salience of each original sentences.

Table 4.2: Results on DUC 2006.

| System | ROUGE-1 | ROUGE-2 | ROUGE-SU4 |
|-------------|---------------|---------------|---------------|
| Random | 0.280 | 0.046 | 0.088 |
| Lead | 0.308 | 0.048 | 0.087 |
| LexRank | 0.360 | 0.062 | 0.118 |
| TextRank | 0.373 | 0.066 | 0.125 |
| MDS-Sparse | 0.340 | 0.052 | 0.107 |
| DSDR | 0.377 | 0.073 | 0.117 |
| RA-MDS | 0.391 | 0.081 | 0.136 |
| ABS-Phrase | 0.392 | 0.082 | 0.137 |
| C-Attention | 0.393* | 0.087* | 0.141* |

Table 4.3: Results on DUC 2007.

| System | ROUGE-1 | ROUGE-2 | ROUGE-SU4 |
|-------------|---------------|---------------|---------------|
| Random | 0.302 | 0.046 | 0.088 |
| Lead | 0.312 | 0.058 | 0.102 |
| LexRank | 0.378 | 0.075 | 0.130 |
| TextRank | 0.403 | 0.083 | 0.144 |
| MDS-Sparse | 0.353 | 0.055 | 0.112 |
| DSDR | 0.398 | 0.087 | 0.137 |
| RA-MDS | 0.408 | 0.097 | 0.150 |
| ABS-Phrase | 0.419 | 0.103 | 0.156 |
| C-Attention | 0.423* | 0.107* | 0.161* |

We would like to mention that **SpOpt** [140] also presents some good results in their paper, however, it is difficult to rebuild their system to faithfully reproduce their results.

As shown in Table 4.2, Table 4.3, and Table 4.4, our system achieves the best results on all the ROUGE metrics. The reasons are as follows: (1) The attention model can directly capture the salient sentences, which are obtained by minimizing the global data reconstruction error; (2) The cascaded structure of attentions can jointly consider the embedding vector space and bag-of-words vector space when conducting the estimation of sentence salience; (3) The coarse-grained sentence compression based on distilled word salience, and the fine-grained compression via

Table 4.4: Results on TAC 2011.

| System | ROUGE-1 | ROUGE-2 | ROUGE-SU4 |
|-------------|---------------|---------------|---------------|
| Random | 0.303 | 0.045 | 0.090 |
| Lead | 0.315 | 0.071 | 0.103 |
| LexRank | 0.313 | 0.060 | 0.102 |
| TextRank | 0.332 | 0.064 | 0.107 |
| PKUTM | 0.396 | 0.113 | 0.148 |
| ABS-Phrase | 0.393 | 0.117 | 0.148 |
| RA-MDS | 0.400 | 0.117 | 0.151 |
| C-Attention | 0.400* | 0.121* | 0.153* |

* : Statistical significance tests show that our method is better than the best baselines.

Table 4.5: Top-10 terms extracted from each topic according to the word salience

| Topic 1 | Topic 2 | Topic 3 |
|-----------|----------|--------------|
| school | heart | HIV |
| shooting | disease | Africa |
| Auvinen | study | circumcision |
| Finland | risk | study |
| police | test | infection |
| video | blood | trial |
| Wednesday | red | woman |
| gunman | telomere | drug |
| post | level | health |

phrase-based unified optimization framework can generate more concise and salient summaries. It is worth noting that PKUTM used a Wikipedia corpus for providing domain knowledge. The system **SWING** [97] is the best system for TAC 2011. Our results are not as good as SWING. The reason is that SWING employs category-specific features and requires supervised training. These features help them select better category-specific content for the summary. In contrast, our model is basically **unsupervised**.

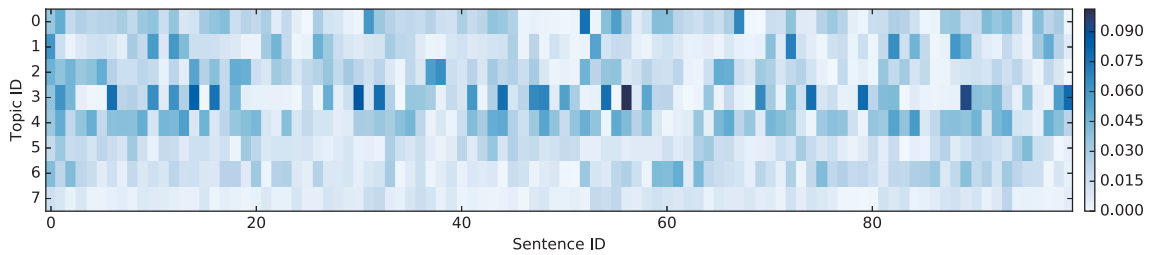


Figure 4.3: Visualization for sentence attention.

4.4.3 Case Study: Distilled Word Saliency

As mentioned above, the output vectors S in our neural model contain the distilled word saliency information. In order to show the performance of word saliency estimation, we select 3 topics (events) from different categories of TAC 2011: “Finland Shooting”, “Heart Disease”, and “Hiv Infection Africa”. For each topic, we sort the dictionary terms according to their saliency scores, and extract the top-10 terms as the saliency estimation results as shown in Table 4.5. We can see that the top-10 terms reveal the most important information of each topic. For the topic “Finland Shooting”, there is a sentence from the golden summary “A teenager at a school in Finland went on a shooting rampage Wednesday, November 11, 2007, killing 8 people, then himself.” It is obvious that the top-10 terms from Table 4.5 can capture this main point.

4.4.4 Case Study: Attention-based Sentence Saliency

In our model, the distilled attention matrix A^o can be treated as sentence saliency estimation. Let \hat{a} be the magnitude of the columns in A^o and $\hat{a} \in \mathbb{R}^m$. \hat{a}_i represents the saliency of the sentence x_i . We collect all the attention vectors for 8 topics of TAC 2011, and display them as an image as shown in Figure 4.3. The x-axis represents the sentence id (we show at most 100 sentences), and the y-axis represents the topic id. The gray level of pixels in the image indicates different saliency scores, where

dark represents a high salience score and *light* represents a small score. Note that different topics seem to hold different ranges of salience scores because they have different number of sentences, i.e. m . According to Equation 4.9, topics containing more sentences will distribute the attention to more units, therefore, each sentence will get a relatively smaller attention weight. But this issue does not affect the performance of MDS since different topics are independently processed.

In Figure 4.3, there are some chunks in each topic (see Topic 3 as an example) having higher attention weights, which indeed automatically captures one characteristic of MDS: *sentence position is an important feature for news summarization*. As observed by several previous studies [70, 97], the sentences in the beginning of a news document are usually more important and tend to be used for writing model summaries. Manual checking verified that those high-attention chunks correspond to the beginning sentences. Our model is able to automatically capture this information by assigning the latter sentences in each topic lower attention weights.

4.4.5 Case Analysis

Table 4.6 shows the summary of the topic “*Hawkins Robert Van Maur*” in TAC 2011. The summary contains four sentences, which are all compressed with different compression ratio. Some uncritical information is excluded from the summary sentences, such as “*police said Thursday*” in S2, “*But*” in S3, and “*he said*” in S4. In addition, the VP “*killing eight people*” in S2 is also excluded since it is duplicate with the phrase “*killed eight people*” in S3. Moreover, from the case we can find that the compression operation did not harm the linguistic quality.

Table 4.6: The summary of the topic “*Hawkins Robert Van Maur*”.

S1: The young gunman who opened fire at a mall busy with holiday shoppers appeared to choose his victims at random, according to police[~~-, but a note he left behind hinted at a troubled life~~].

S2: The teenage gunman who went on a shooting rampage in a department store, [~~killling eight people,~~] may have smuggled an assault rifle into the mall underneath clothing[~~-, police said Thursday~~].

S3: [~~But~~] police said it was Hawkins who went into an Omaha shopping mall on Wednesday and began a shooting rampage that killed eight people.

S4: Mall security officers noticed Hawkins briefly enter the Von Maur department store at Omaha’s Westroads Mall earlier Wednesday[~~-, he said~~].

4.5 Summary

In this Chapter, we propose a cascaded neural attention based unsupervised salience estimation method for compressive multi-document summarization. The attention weights for sentences and salience values for words are both learned by data reconstruction in an unsupervised manner. We thoroughly investigate the performance of combining different attention architectures and cascaded structures. Experimental results on some benchmark data sets show that our framework achieves good performance compared with the state-of-the-art methods.

Chapter 5

Variational Auto-Encoders for Multi-Document Summarization

5.1 Background

Considering the scalability restriction of labeling multi-document summarization datasets, some works adopt unsupervised data reconstruction methods to conduct salience estimation and achieve comparable results [42, 70, 82, 108, 116, 140]. After investigating these works, we observe that they mainly use Bag-of-Words (BoWs) vectors in sentence representation and reconstruction loss function. On the other hand, some research works [44, 52, 59, 96] have demonstrated that distributed representations outperform BoWs in modeling sentence and document semantics. In this paper, instead of using BoWs vectors, we explore a distributed representation for modeling the latent semantics of sentences for the MDS task. We propose a framework based on probabilistic generative models to describe the observed sentences and latent semantic vectors.

Given a topic (event) composed of a set of documents, we build a distributed latent semantic vector to model each sentence with a generative framework, where

each sentence is generated from an unobserved latent semantic space. Another characteristic is that the generative process employs a neural network conditioned on the input text approximating the distributions over the latent semantic vector. Markov Chain Monte Carlo (MCMC) sampling and Variational Inference (VI) are the most common methods used in generative models [8, 49, 127]. Nevertheless, some integrals of the marginal likelihood are intractable due to the continuous latent variables and neural network based generative modeling. Standard variational inference methods such as mean-field algorithms [137] cannot be used. Moreover, MCMC based sampling methods are too slow to extend to large-scale machine learning tasks. Recently, Variational Autoencoders (VAEs) [54, 110] and Generative Adversarial Networks (GANs) [36, 107] have been proposed that can handle the inference problem associated with complex generative modeling frameworks. In our work, we employ VAEs as the basic framework for the generative model. In fact, some works [20, 93] have demonstrated that VAEs outperform the general Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN) in generating high-level semantic representations.

To address the sentence salience estimation problem for MDS, we propose an unsupervised data reconstruction framework which jointly reconstructs the latent semantic space and the observed term vector space. The basic idea behind the data reconstruction is that each original sentence can be reconstructed using a linear combination of several other representative sentences. These representative sentences are able to capture different aspects implied in the event, such as “what happened”, “damages”, “countermeasures”, etc. We name the vectors which are used to represent the aspect sentences as aspect vectors. Then, salience estimation can be conducted during the reconstruction process using aspect vectors. Based on the spirit of generative model and data reconstruction process, we design several latent aspect vectors and use them to reconstruct the whole original latent semantic

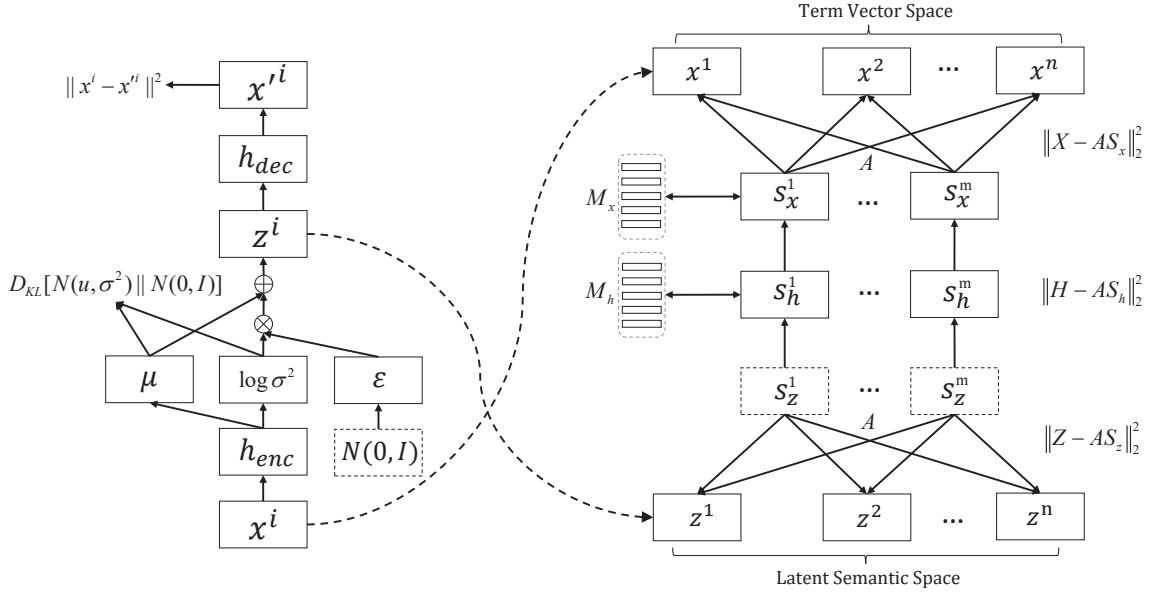


Figure 5.1: Our proposed sentence salience framework. **Left:** Latent semantic modeling via variational auto-encoders for sentence \mathbf{x}^i . **Right:** Saliency estimation by a data reconstruction method during the variational-decoding process. \mathbf{x} is the sentence term vector, and \mathbf{z} is the corresponding latent semantic vector. \mathbf{S}_z are the latent aspect vectors. \mathbf{S}_h and \mathbf{S}_x are hidden vectors and the output aspect term vectors. \mathbf{M}_h and \mathbf{M}_x are two memories used to refine \mathbf{S}_h and \mathbf{S}_x based on the neural alignment mechanism. \mathbf{A} is a reconstruction coefficient matrix which contains the sentence salience information.

space. In parallel with such idea, we also design some aspect term vectors which are used to reconstruct the original observed term vector space. Thereafter, the VAEs-based latent semantic model is integrated into the sentence salience estimation component in a unified fashion, and the whole framework can be trained jointly by back-propagation via multi-task learning. After estimating the sentence salience, we employ a phrase merging based unified optimization framework to generate a final summary.

5.2 Overview of Our Proposed Framework

As shown in Figure 5.1, our sentence salience framework has two main components: (1) latent semantic modeling; (2) salience estimation. To tackle the latent semantic modeling problem, a VAEs-based generative model is designed to project sentences from the term vector space to the latent semantic space. Consider a dataset $\mathbf{X} = \{\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^n\}$ consisting of n sentences from all the documents in a topic (event), represented by BoWs term vectors. The left part of Figure 5.1 illustrates a VAEs-based component implemented as a feed-forward neural network for associating a latent semantic vector $\mathbf{z}^i \in \mathbb{R}^K$ with each sentence $\mathbf{x}^i \in \mathbb{R}^{|V|}$, where V is the term dictionary. Based on generative modeling, a latent semantic vector $\mathbf{z}^i \in \mathbb{R}^K$ is generated from some prior distribution $p_\theta(\mathbf{z}^i)$. Then the sentence term vector \mathbf{x}^i is generated from a conditional distribution $p_\theta(\mathbf{x}^i|\mathbf{z}^i)$. To find the parameter θ , the reparameterization trick is applied to obtain a differentiable estimator of the variational lower bound. Then back-propagation can be employed to train the neural network. For sentence salience estimation, we propose **VAEs-A**, an unsupervised data reconstruction framework with the alignment mechanism for aspect vector discovery. The general idea is shown in the right part of Figure 5.1. Note that $\{\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^n\}$ and $\{\mathbf{z}^1, \mathbf{z}^2, \dots, \mathbf{z}^n\}$ are exactly the same vectors as those depicted in the left part of Figure 5.1. We design some latent aspect vectors \mathbf{S}_z for capturing the latent aspect information of a topic. The corresponding aspect term vectors \mathbf{S}_x are generated according to the conditional distribution $p_\theta(\mathbf{x}|\mathbf{z})$. By reconstructing the original sentence term vectors \mathbf{X} and the corresponding latent semantic vectors \mathbf{Z} using \mathbf{S}_x and \mathbf{S}_z jointly, the sentence salience can be estimated from the optimized coefficient matrix. Finally, inspired by [6], a phrase-based unified numerical optimization framework is employed to conduct the summary generation.

5.3 Sentence Saliency Framework

5.3.1 Latent Semantic Modeling

VAEs-based latent semantic modeling can be viewed as an instance of unsupervised learning, which can be divided into two parts: inference (variational-encoder) and generation (variational-decoder). Recall that the dictionary is V . As shown in the left part of Figure 5.1, for each sentence term vector $\mathbf{x} \in \mathbb{R}^{|V|}$, the variational-encoder can map it to a latent semantic vector $\mathbf{z} \in \mathbb{R}^K$, which can be used to generate the original sentence term vector via the variational-decoder component. The target is to maximize the probability of each \mathbf{x} in the dataset based on the generation process according to:

$$p_{\theta}(\mathbf{x}) = \int p_{\theta}(\mathbf{x}|\mathbf{z})p_{\theta}(\mathbf{z})d\mathbf{z} \quad (5.1)$$

For the purpose of solving the intractable integral of the marginal likelihood as shown in Equation 5.1, a recognition model $q_{\phi}(\mathbf{z}|\mathbf{x})$ is introduced as the approximation to the intractable of true posterior $p_{\theta}(\mathbf{z}|\mathbf{x})$. It is obvious that $q_{\phi}(\mathbf{z}|\mathbf{x})$ and $p_{\theta}(\mathbf{x}|\mathbf{z})$ can be regarded as a probabilistic encoder and decoder respectively. The recognition model parameters ϕ and the generative model parameters θ can be learnt jointly. The aim is to reduce the Kullback-Leibler divergence (KL) between $q_{\phi}(\mathbf{z}|\mathbf{x})$ and $p_{\theta}(\mathbf{z}|\mathbf{x})$:

$$\begin{aligned} D_{KL}[q_{\phi}(z|x)||p_{\theta}(z|x)] &= \int_z q_{\phi}(z|x) \log \frac{q_{\phi}(z|x)}{p_{\theta}(z|x)} dz \\ &= \mathbb{E}_{q_{\phi}(z|x)}[\log q_{\phi}(z|x) - \log p_{\theta}(z|x)] \end{aligned} \quad (5.2)$$

By applying Bayes rule to $p_{\theta}(\mathbf{z}|\mathbf{x})$:

$$\begin{aligned} D_{KL}[q_{\phi}(z|x)||p_{\theta}(z|x)] &= \log p_{\theta}(x) + \\ &\mathbb{E}_{q_{\phi}(z|x)}[\log q_{\phi}(z|x) - \log p_{\theta}(x|z) - \log p_{\theta}(z)] \end{aligned} \quad (5.3)$$

We can extract $\log p_\theta(\mathbf{x})$ from the expectation, transfer the expectation term $\mathbb{E}_{q_\varphi(z|x)}$ back to KL-divergence, and rearrange all the terms. Then we yield:

$$\begin{aligned} \log p_\theta(x) &= D_{KL}[q_\varphi(z|x)||p_\theta(z|x)] \\ &\quad + \mathbb{E}_{q_\varphi(z|x)}[\log p_\theta(x|z)] \\ &\quad - D_{KL}[q_\varphi(z|x)||p_\theta(z)] \end{aligned} \quad (5.4)$$

Let $\mathcal{L}(\theta, \varphi; \mathbf{x})$ represent the last two terms from the right part of Equation 5.4:

$$\mathcal{L}(\theta, \varphi; x) = \mathbb{E}_{q_\varphi(z|x)}[\log p_\theta(x|z)] - D_{KL}[q_\varphi(z|x)||p_\theta(z)] \quad (5.5)$$

Because the first KL-divergence term of Equation 5.4 is non-negative, so we have $\log p_\theta(x) \geq \mathcal{L}(\theta, \varphi; \mathbf{x})$, which means that $\mathcal{L}(\theta, \varphi; \mathbf{x})$ is a lower bound (the objective to be maximized) on the marginal likelihood. In order to differentiate and optimize the lower bound $\mathcal{L}(\theta, \varphi; \mathbf{x})$, following the core idea of VAEs, we use a neural network framework for the probabilistic encoder $q_\varphi(\mathbf{z}|\mathbf{x})$ for better approximation.

Similar to previous works [39, 54, 110], we assume that both the prior and posterior of the latent variables are Gaussian, i.e., $p_\theta(\mathbf{z}) = \mathcal{N}(0, \mathbf{I})$ and $q_\varphi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2\mathbf{I})$, where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ denote the variational mean and standard deviation respectively, which can be calculated with a multilayer perceptron (MLP). Precisely, given the term vector representation of an input sentence \mathbf{x} , we first project it to a hidden space:

$$h_{enc} = \text{relu}(W_{xh}x + b_{xh}) \quad (5.6)$$

where $h_{enc} \in \mathbb{R}^{d_h}$, W_{xh} and b_{xh} are the neural parameters. $\text{relu}(x) = \max(0, x)$ is the activation function.

Then the Gaussian parameters $\boldsymbol{\mu} \in \mathbb{R}^K$ and $\boldsymbol{\sigma} \in \mathbb{R}^K$ can be obtained via a linear

transformation based on h_{enc} :

$$\begin{aligned}\mu &= W_{h\mu}h_{enc} + b_{h\mu} \\ \log(\sigma^2) &= W_{h\sigma}h_{enc} + b_{h\sigma}\end{aligned}\tag{5.7}$$

The latent semantic vector $\mathbf{z} \in \mathbb{R}^K$ can be calculated using the reparameterization trick:

$$\varepsilon \sim \mathcal{N}(0, \mathbf{I}), \quad z = \mu + \sigma \otimes \varepsilon\tag{5.8}$$

where $\varepsilon \in \mathbb{R}^K$ is an auxiliary noise variable. It is obvious that the mapping from \mathbf{x} to \mathbf{z} is similar with the process of general auto-encoder. Therefore this process can be named variational-encoding process.

Given the latent semantic vector \mathbf{z} , a new term vector \mathbf{x}' is generated via the conditional distribution $p_\theta(\mathbf{x}|\mathbf{z})$. Under the neural network framework, the generation process is similar with the decoding process of the typical auto-encoder model:

$$h_{dec} = \text{relu}(W_{zh}z + b_{zh})\tag{5.9}$$

$$x' = \text{sigmoid}(W_{hx}h_{dec} + b_{hx})\tag{5.10}$$

Finally, based on the reparameterization trick in Equation 5.8, we can get the analytical representation of the variational lower bound $\mathcal{L}(\theta, \varphi; \mathbf{x})$:

$$\begin{aligned}\log p(x|z) &= \sum_{i=1}^{|V|} x_i \log x'_i + (1 - x_i) \cdot \log(1 - x'_i) \\ -D_{KL}[q_\varphi(z|x)||p_\theta(z)] &= \frac{1}{2} \sum_{i=1}^K (1 + \log(\sigma_i^2) - \mu_i^2 - \sigma_i^2)\end{aligned}\tag{5.11}$$

In this work we let $p_\theta(\mathbf{x}|\mathbf{z})$ be a multivariate Bernoulli. All the parameters $\{\mathbf{W}, \mathbf{b}\}$ can be learnt using the back-propagation method.

5.3.2 Saliency Estimation

The right part of Figure 5.1 depicts the general framework for saliency estimation. Note that \mathbf{x}^i and \mathbf{z}^i are the same vectors as those in the left part of Figure 5.1. Considering the spirit of summarization, we design a set of latent aspect vectors \mathbf{S}_z from the latent space which can be regarded as the representatives of the whole semantic space. Inspired by previous works [42, 70, 108, 140], we propose an unsupervised data reconstruction framework, named **VAEs-A**, for sentence saliency estimation. The main idea is to jointly consider the reconstruction for latent semantic space and observed term vector space. This framework can capture the saliency of sentences from these two different and complementary vector spaces.

VAEs-A

Assume that $\mathbf{S}_z = \{\mathbf{s}_z^1, \mathbf{s}_z^2, \dots, \mathbf{s}_z^m\}$ are m latent aspect vectors used for reconstructing all the latent semantic vectors $\mathbf{Z} = \{\mathbf{z}^1, \mathbf{z}^2, \dots, \mathbf{z}^n\}$, and $m \ll n$. Recall that n is the number of original sentences. Here, we do not use the standard probabilistic sampling methods, instead we propose a more efficient and straightforward estimation method based on a neural network, which can be trained using back-propagation. More specifically, \mathbf{S}_z is initialized using values from $[-0.1, 0.1]$ randomly. Thereafter, the variational-decoding progress of VAEs can map the latent aspect vector \mathbf{S}_z to \mathbf{S}_h , and then produce m new aspect term vectors \mathbf{S}_x :

$$s_h = \text{relu}(W_{zh}s_z + b_{zh}) \quad (5.12)$$

$$s_x = \text{sigmoid}(W_{hx}s_h + b_{hx}) \quad (5.13)$$

where the neural parameters \mathbf{W} and \mathbf{b} are shared from the decoder of VAEs.

Although VAEs are able to generate high-level abstract latent semantic representations for sentences, they may not be sufficient for generating high-quality sentence

term vectors. The top-down generating process may lose detailed information [64]. In order to address this problem and to estimate the sentence salience more precisely, we add an alignment mechanism [2, 85] to the decoding hidden layer and output layer respectively. The purpose of the alignment mechanism is to recall the lost detailed information from the sentence term vector memory $\mathbf{M}_x = \{\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^n\}$ and the encoder hidden state memory $\mathbf{M}_h = \{\mathbf{h}_{enc}^1, \mathbf{h}_{enc}^2, \dots, \mathbf{h}_{enc}^n\}$.

For each decoder hidden state s_h^i , we align it with each encoder hidden state $h_{enc}^j \in M_h$ by an alignment vector $a^h \in \mathbb{R}^n$. $a_{i,j}^h$ is derived by comparing s_h^i with each input sentence hidden state h_{enc}^j :

$$a_{i,j}^h = \frac{\exp(e_{i,j}^h)}{\sum_{j'} \exp(e_{i,j'}^h)} \quad (5.14)$$

$$e_{i,j}^h = v_{ha}^T \tanh(W_{ha} h_{enc}^j + U_{ha} s_h^i)$$

The alignment vector $a_{i,j}^h$ captures much more detailed information from the source hidden space when generating the new representations. Based on the alignment vectors $\{a_{i,j}^h\}$, we can create a context vector c_h^i by linearly blending the sentence hidden states $h_{enc}^{j'}$:

$$c_h^i = \sum_{j'} a_{i,j'}^h h_{enc}^{j'} \quad (5.15)$$

Then the output hidden state can be updated based on the context vector:

$$\tilde{s}_h^i = \tanh(W_{ch}^h c_h^i + W_{hh}^a s_h^i) \quad (5.16)$$

And a temporal output vector is generated according to:

$$\tilde{s}_x^i = \text{sigmoid}(W_{hx} \tilde{s}_h^i + b_{hx}) \quad (5.17)$$

Besides the alignment mechanism on the hidden layer, we also directly add alignment

on the output layer, which can capture more nuanced and subtle difference information from the BoWs term vector space. The alignment is conducted by comparing \tilde{s}_x^i with each observed term vector $x^j \in M_x$:

$$\begin{aligned} a_{i,j}^x &= \frac{\exp(e_{i,j}^x)}{\sum_{j'} \exp(e_{i,j'}^x)} \\ e_{i,j}^x &= \tilde{s}_x^i \cdot x^j \end{aligned} \quad (5.18)$$

where \cdot in the inner product operation. Then the output context vector is computed as:

$$c_x^i = \sum_j a_{i,j}^x x^j \quad (5.19)$$

To update the output vector, we develop a different method from that of the hidden alignments. Specifically we use a weighted combination of the context vectors and the original outputs with $\omega_a \in [0, 1]$:

$$s_x^i = \omega_a c_x^i + (1 - \omega_a) \tilde{s}_x^i \quad (5.20)$$

Intuitively, \mathbf{S}_z , \mathbf{S}_h , and \mathbf{S}_x can be used to reconstruct the space to which they belong respectively. Let $\mathbf{A} \in \mathbb{R}^{n \times m}$ be the reconstruction coefficient matrix. Specifically, we do not create the new variable \mathbf{A} here. Instead, we represent it using the decoder output layer alignment matrix $\mathbf{A} = \{a_{i,j}^x\}$, then refine it during optimization. We define the magnitude of each row of \mathbf{A} as the salience scores for the corresponding sentences.

The optimization objective contains three reconstruction terms, jointly considering the latent semantic reconstruction and the term vector space reconstruction:

$$\mathcal{L}_A = \lambda_z \|Z - AS_z\|_2^2 + \lambda_h \|H - AS_h\|_2^2 + \lambda_x \|X - AS_x\|_2^2 \quad (5.21)$$

This objective is integrated with the variational lower bound of VAEs and optimized in a multi-task learning fashion.

VAEs-Zero

We also investigate a simpler VAEs-based model named VAEs-Zero which can also conduct salience estimation. Recall the reparameterization trick, the prior and posterior of the latent semantic vector \mathbf{z} are both from Gaussian, and the vectors drawn from the zero mean will hold larger probability density. Based on this idea, we can generate a term vector $\mathbf{s}_x \in \mathbb{R}^{|V|}$ from a special latent semantic vector $\mathbf{s}_z = \mathbf{0}$ via the variational-decoding process. Intuitively, \mathbf{s}_x contains richer information than the other vectors, which should be distilled as the summary information. Therefore, we assume that sentences which are more similar with \mathbf{s}_x hold larger salience values. For each sentence $\mathbf{x}^i \in \mathbf{X}$, we use the cosine similarity as the salience estimation:

$$a^i = \frac{\mathbf{x}^i \cdot \mathbf{s}_x}{\|\mathbf{x}^i\| \|\mathbf{s}_x\|} \quad (5.22)$$

Interestingly, \mathbf{s}_x can also be treated as the word salience information, so it can be employed to conduct the keyword extraction task.

5.3.3 Multi-Task Learning

As mentioned before, we integrate VAEs-based latent semantic modeling and salience estimation into a unified framework. Then the new optimization objective is:

$$\mathcal{J} = \min_{\Theta} (-\mathcal{L}(\theta, \varphi; x) + \lambda \mathcal{L}_{\text{salience}}) \quad (5.23)$$

where Θ is a set of all the parameters related to this task. $\mathcal{L}_{\text{salience}}$ is the reconstruction loss function for VAEs-A or VAEs-Zero. The whole framework can be trained

using back-propagation efficiently. After the training, we calculate the magnitude of each row of \mathbf{A} as the salience score for each corresponding sentence, which will be fed into a phrase-based optimization framework to generate a summary.

5.4 Summary Generation

Inspired by the phrase-based model in Bing et al. [6] and Li et al. [70], we refine this model to consider the salience information obtained by our VAEs-based salience estimation framework. Based on the parsed constituency tree for each input sentence, we extract the noun-phrases (NPs) and verb-phrases (VPs). The salience S_i of a phrase P_i is defined as:

$$S_i = \left\{ \sum_{t \in P_i} tf(t) / \sum_{t \in Topic} tf(t) \right\} \times a_i, \quad (5.24)$$

where a_i is the salience of the sentence containing P_i ; $tf(t)$ be the frequency of the concept t (unigram/bigram) in the whole topic. Thus, S_i inherits the salience of its sentence, and also considers the importance of its concepts.

The overall objective function of this optimization formulation for selecting salient NPs and VPs is formulated as an integer linear programming (ILP) problem:

$$\begin{aligned} \max \{ & \sum_i \alpha_i S_i^N - \sum_{i < j} \alpha_{ij} (S_i^N + S_j^N) R_{ij}^N \\ & + \sum_i \beta_i S_i^V - \sum_{i < j} \beta_{ij} (S_i^V + S_j^V) R_{ij}^V \} \end{aligned} \quad (5.25)$$

where α_i and β_i are selection indicators for the NP N_i and the VP V_i , respectively. S_i^N and S_i^V are the salience scores of N_i and V_i . α_{ij} and β_{ij} are co-occurrence indicators of pairs (N_i, N_j) and (V_i, V_j) . R_{ij}^N and R_{ij}^V are the similarity of pairs (N_i, N_j) and (V_i, V_j) . The similarity is calculated by the Jaccard Index based method.

Specifically, this objective maximizes the salience score of the selected phrases, and penalizes the selection of similar phrase pairs.

In order to obtain coherent summaries with good readability, we add some constraints into the ILP framework, such as phrase co-occurrence constraint, which control the co-occurrence relation of NPs or VPs: For NPs, we introduce three constraints:

$$\alpha_{ij} - \alpha_i \leq 0, \quad (5.26)$$

$$\alpha_{ij} - \alpha_j \leq 0, \quad (5.27)$$

$$\alpha_i + \alpha_j - \alpha_{ij} \leq 1. \quad (5.28)$$

Constraints 5.26 to 5.28 ensure a valid solution of NP selection. The first two constraints state that if the units N_i and N_j co-occur in the summary (i.e., $\alpha_{ij} = 1$), then we have to include them individually (i.e., $\alpha_i = 1$ and $\alpha_j = 1$). The third constraint is the inverse of the first two. Similarly, the constraints for VPs are as follows:

$$\beta_{ij} - \beta_i \leq 0, \quad (5.29)$$

$$\beta_{ij} - \beta_j \leq 0, \quad (5.30)$$

$$\beta_i + \beta_j - \beta_{ij} \leq 1. \quad (5.31)$$

Other constraints include sentence number, summary length, phrase co-occurrence, etc. For details, please refer to Woodsend and Lapata [135], Bing et al. [6], and Li et al. [70]. The objective function and constraints are linear. Therefore the optimization can be solved by existing ILP solvers such as simplex algorithms [21]. In the implementation, we use a package called `lp_solve`¹.

¹<http://lpsolve.sourceforge.net/5.5/>

5.5 Experiments and Results

5.5.1 Datasets

The standard MDS datasets from DUC and TAC are used in our experiments. DUC 2006 and DUC 2007 contain 50 and 45 topics respectively. Each topic has 25 news documents and 4 model summaries. The length of the model summary is limited to 250 words. TAC 2011 is the latest standard summarization benchmark data set and it contains 44 topics. Each topic contains 10 related news documents and 4 model summaries. TAC 2010 is used as the parameter tuning data set of our TAC evaluation. The length of the model summary is limited to 100 words.

5.5.2 Evaluation Metric

We use ROUGE score as our evaluation metric [79]. F-measures of ROUGE-1, ROUGE-2 and ROUGE-SU4 are reported. For the definition of ROUGE, please refer to Section 3.3.2 of Chapter 3.

5.5.3 Settings

For text processing, the input sentences are represented as BoWs vectors with dimension $|V|$. The dictionary V is created using unigrams, bigrams and named entity terms. n is the number of sentences in all the documents of a topic (event). For the number of aspects, we let $m = 5$. For the neural network framework, we set the hidden size $d_h = 500$ and the latent size $K = 100$. For the optimization objective, we let $\lambda_z = 1$, $\lambda_h = 400$, $\lambda_x = 800$, and $\lambda = 1$. Adam [53] is used for gradient based optimization with a learning rate 0.001. Our neural network based framework is implemented using Theano [120] on a single GPU². For training the attention-based

²Tesla K80, 1 Kepler GK210 is used, 2496 Cuda cores, 12G GDDR5 memory.

Table 5.1: Results on DUC 2006.

| System | Rouge-1 | Rouge-2 | Rouge-SU4 |
|-------------|---------------|---------------|---------------|
| Random | 0.280 | 0.046 | 0.088 |
| Lead | 0.308 | 0.048 | 0.087 |
| MDS-Sparse | 0.340 | 0.052 | 0.107 |
| DSDR | 0.377 | 0.073 | 0.117 |
| RA-MDS | 0.391 | 0.081 | 0.136 |
| ABS-Phrase | 0.392 | 0.082 | 0.137 |
| C-Attention | 0.393 | 0.087 | 0.141 |
| VAEs-Zero | 0.382 | 0.080 | 0.135 |
| VAEs-A | 0.396* | 0.089* | 0.143* |

Table 5.2: Results on DUC 2007.

| System | Rouge-1 | Rouge-2 | Rouge-SU4 |
|-------------|---------------|---------------|---------------|
| Random | 0.302 | 0.046 | 0.088 |
| Lead | 0.312 | 0.058 | 0.102 |
| MDS-Sparse | 0.353 | 0.055 | 0.112 |
| DSDR | 0.398 | 0.087 | 0.137 |
| RA-MDS | 0.408 | 0.097 | 0.150 |
| ABS-Phrase | 0.419 | 0.103 | 0.156 |
| C-Attention | 0.423 | 0.107 | 0.161 |
| VAEs-Zero | 0.416 | 0.106 | 0.158 |
| VAEs-A | 0.423* | 0.110* | 0.164* |

distillation component, each topic can be finished in less than 1 minute.

5.5.4 Results and Discussions

To compare the performance of our framework with previous methods, our first priority is to get the summaries produced by their systems (or get their code to produce summaries by ourselves). Then we run ROUGE evaluation on them with the same option.

We compare our system with several summarization baselines and existing unsupervised methods. **Random** baseline selects sentences randomly for each topic. **Lead** baseline [132] ranks the news chronologically and extracts the leading sen-

Table 5.3: Results on TAC 2011.

| System | Rouge-1 | Rouge-2 | Rouge-SU4 |
|-------------|---------------|---------------|---------------|
| Random | 0.303 | 0.045 | 0.090 |
| Lead | 0.315 | 0.071 | 0.103 |
| PKUTM | 0.396 | 0.113 | 0.148 |
| RA-MDS | 0.400 | 0.117 | 0.151 |
| ABS-Phrase | 0.393 | 0.117 | 0.148 |
| C-Attention | 0.400 | 0.121 | 0.153 |
| VAEs-Zero | 0.388 | 0.113 | 0.145 |
| VAEs-A | 0.405* | 0.122* | 0.155* |

tences one by one. Three other unsupervised existing methods based on sparse coding are also compared, namely, **DSDR** [42], **MDS-Sparse** [82], and **RA-MDS** [70]. **ABS-Phrase** [6] generates abstractive summaries using phrase-based optimization framework with weighted term frequency as salience estimation. We also conduct comparisons with our method **C-Attention** proposed in Chapter 4. “*” means that statistical significance tests show that our method is better than the best baselines.

As shown in Table 5.1 and Table 5.2, our system achieves the best results on all the ROUGE metrics. It demonstrates that VAEs based latent semantic modeling and jointly semantic space reconstruction can improve the MDS performance considerably. It is worth to note that VAEs-Zero also achieves comparable performance. Although it is not as good as VAEs-A, it is better than most of the existing methods. Therefore, VAEs based latent semantic modeling can benefit the MDS performance. Besides those **unsupervised** models, to our knowledge, the method presented in Wang et al. [131] achieved the best performance on DUC 2007. The reason is that it uses **supervised** learning framework to train the sentence compression and document summarization models. In the evaluation, it provides two supervised learning based sentence selection methods: Support Vector Regression (SVR) and LambdaMART. SVR obtains 0.095 and 0.147 on Rouge-2 and Rouge-SU4 respectively.

Table 5.4: Top-10 terms extracted from each topic according to the output of VAEs-A

| Topic 1 | Topic 2 | Topic 3 |
|---------|------------|---------|
| Roberts | China | food |
| amish | earthquake | recall |
| girl | Sichuan | pet |
| school | province | cat |
| Miller | tuesday | dog |
| family | million | company |
| child | relief | menu |
| police | people | sell |
| kill | government | product |

LambdaMART obtains 0.123 and 0.156. Our framework, which is unsupervised, outperforms SVR and achieves similar results compared with LambdaMART.

For the data set TAC 2011, besides the above mentioned baselines, we compare our framework with several more top systems: **PKUTM** [65] employs manifold-ranking for sentence scoring and selection; Table 5.3 shows that our performance is better than both PKUTM. It is worth noting that PKUTM used a Wikipedia corpus for providing domain knowledge. The method **SWING** [97] is the best TAC 2011 system. However, our results are not as good as SWING. The reason is that SWING uses category-specific features and trains the feature weights with the category information of TAC 2010 data in a supervised manner. These features help them select better category-specific content for the summary. In contrast, our model is **unsupervised**, and we only use TAC 2010 for general parameter tuning purpose.

We mention that \mathbf{S}_z and \mathbf{S}_x represent different aspects of an event. To validate this idea, we take the topic “Pet Food Recall” in TAC 2011 and extract some keywords from each aspect. **Aspect-1** contains words “*Nutro, purchase, dozen, drop, 60, timing, protein, research*”, **Aspect-2** is “*Sarah, Tuite, source, protein, Food, and, Drug Administration*”, and **Aspect-3** is “*food, company, recall, pet, menu,*

cat, product, foods, dog”. It demonstrates that our framework is able to capture the main aspects of a topic. Moreover, we find that the magnitude of \mathbf{S}_x can represent the word salience information. We select 3 topics from TAC 2011: “Amish Shooting”, “Earthquake Sichuan”, and “Pet Food Recall”. For each topic, we sort the dictionary terms according to their salience scores, and extract the top-10 terms, as shown in Table 5.4. We can see that the top-10 terms reveal the most important information of each topic. For the topic “Amish Shooting”, we notice a sentence from the golden summary: “On October 2, 2006, a gunman, Charles Roberts, entered an Amish school near Lancaster, PA, took the children hostage, killed five girls and wounded seven other children before killing himself.” It is obvious that the top-10 terms can capture the main semantics.

5.6 Summary

In this Chapter, we propose an new unsupervised Multi-Document Summarization (MDS) framework. First, a VAEs based generative model is employed to map the sentence from term vector space to latent semantic space. Then an unsupervised data reconstruction model is proposed to conduct salience estimation, by jointly reconstructing latent semantic space and observed term vector space using aspect related vectors. Experimental results on the benchmark data sets DUC and TAC show that our framework achieves better performance than the state-of-the-art models.

Chapter 6

Reader-Aware Multi-Document Summarization

6.1 Background

With the development of social media and mobile equipments, more and more user generated content is available. Figure 6.1 is a snapshot of reader comments under the news report “The most important announcements from Google’s big developers’ conference”¹. The content of the original news report talks about some new products based on AI techniques. The news report generally conveys an enthusiastic tone. However, while some readers share similar enthusiasms, some others express their worries about new products and technologies and these comments can also reflect their interests which may not be very salient in the original news reports. one natural extension of the setting is to incorporate such content regarding the event so as to directly or indirectly improve the generated summaries with greater user satisfaction. Unfortunately, existing multi-document summarization approaches cannot handle this issue. In this work, we investigate a new setting in this direction. Specifically,

¹<https://goo.gl/DdU0vL>

NEWS: The most important announcements from Google's big developers' conference

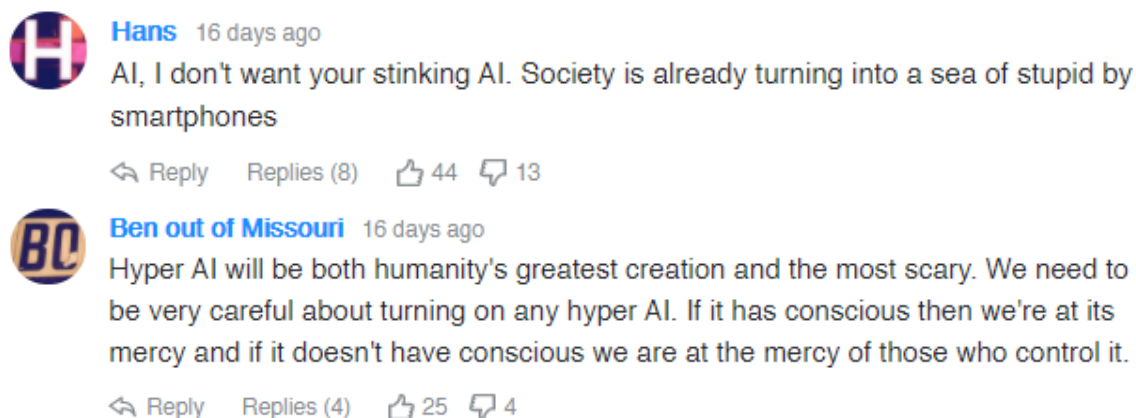


Figure 6.1: Reader comments of the news “The most important announcements from Google’s big developers’ conference (May, 2017)”.

a set of reader comments associated with the news reports are also collected. The generated summaries from the reports for the event should be salient according to not only the reports but also the reader comments. We name such a paradigm of extension as reader-aware multi-document summarization (RA-MDS).

One challenge of the RA-MDS problem is how to conduct salience estimation by jointly considering the focus of news reports and the reader interests revealed by comments. Meanwhile, the model should be insensitive to the availability of diverse aspects of reader comments. Another challenge is that reader comments are very noisy, not fully grammatical and often expressed in informal expressions. Some previous works explore the effect of comments or social contexts in single document summarization such as blog summarization [46, 139]. However, the problem setting of RA-MDS is more challenging because the considered comments are about an event which is described by multiple documents spanning a time period. Another challenge is that reader comments are very diverse and noisy.

Recall that in Chapter 5 we proposed a sentence salience estimation framework known as *VAESum* based on a neural generative model called Variational Auto-

Encoders (VAEs) [54, 110]. We find that the Gaussian based VAEs have a strong ability to capture the salience information and filter the noise from texts. Intuitively, if we feed both the news sentences and the comment sentences into the VAEs, commonly existed latent aspect information from both of them will be enhanced and become salient. Inspired by this consideration, to address the sentence salience estimation problem for RA-MDS by jointly considering news documents and reader comments, we extend the VAESum framework by training the news sentence latent model and the comment sentence latent model simultaneously by sharing the neural parameters. After estimating the sentence salience, we employ a phrase based compressive unified optimization framework to generate a final summary.

There is a lack of high-quality dataset suitable for RA-MDS. Existing datasets from DUC² and TAC³ are not appropriate. Therefore, we introduce a new dataset for RA-MDS. We employed some experts to conduct the tasks of data collection, aspect annotation, and summary writing as well as scrutinizing. To our best knowledge, this is the first dataset for RA-MDS.

6.2 Framework

6.2.1 Overview

As shown in Figure 6.2, our reader-aware news sentence salience framework has three main components: (1) latent semantic modeling; (2) comment weight estimation; (3) joint reconstruction. Consider a dataset X_d and X_c consisting of n_d news sentences and n_c comment sentences respectively from all the documents in a topic (event), represented by bag-of-words vectors. Our proposed news sentence salience estimation framework is extended from VAESum [74], which can jointly consider

²<http://duc.nist.gov/>

³<http://tac.nist.gov/>

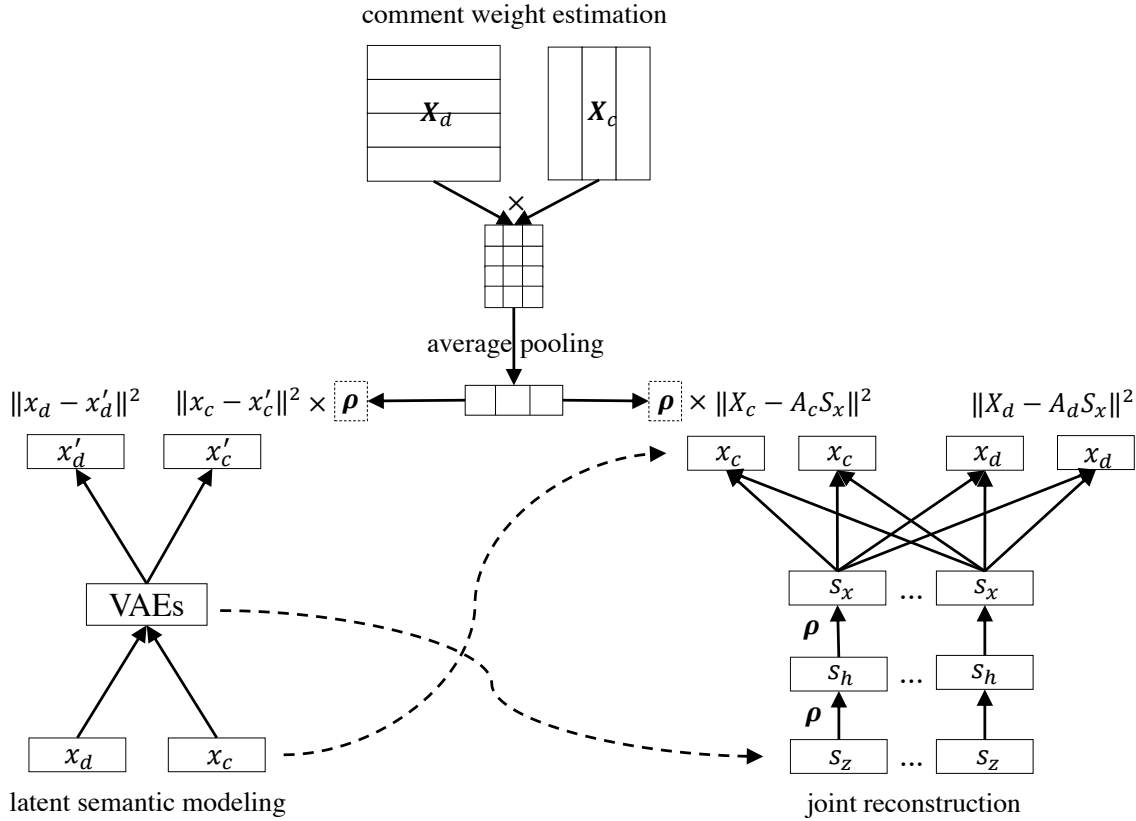


Figure 6.2: Our proposed framework. **Left:** Latent semantic modeling via variation auto-encoders for news sentence \mathbf{x}_d and comment sentence \mathbf{x}_c . **Middle:** Comment sentence weight estimation. **Right:** Saliency estimation by a joint data reconstruction method. \mathbf{A}_d is a news reconstruction coefficient matrix which contains the news sentence saliency information.

news documents and reader comments. One extension is that, in order to absorb more useful information and filter the noisy data from comments, we design a weight estimation mechanism which can assign a real value ρ_i for a comment sentence \mathbf{x}_c^i . The comment weight $\rho \in \mathbb{R}^{n_c}$ is integrated into the VAEs based sentence modeling and data reconstruction component to handle comments.

6.2.2 Reader-Aware Saliency Estimation

Variational Autoencoders (VAEs) [54, 110] is a generative model based on neural networks which can be used to conduct latent semantic modeling. In Chapter 5, we employ VAEs to map the news sentences into a latent semantic space, which is helpful in improving the MDS performance. Similarly, we also employ VAEs to conduct the semantic modeling for news sentences and comment sentences. Assume that both the prior and posterior of the latent variables are Gaussian, i.e., $p_\theta(\mathbf{z}) = \mathcal{N}(0, \mathbf{I})$ and $q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2\mathbf{I})$, where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ denote the variational mean and standard deviation respectively, which can be calculated with a multi-layer perceptron (MLP). VAEs can be divided into two phases, namely, encoding (inference), and decoding (generation). All the operations are depicted as follows:

$$\begin{aligned}
 h_{enc} &= \text{relu}(W_{xh}x + b_{xh}) \\
 \boldsymbol{\mu} &= W_{h\mu}h_{enc} + b_{h\mu} \\
 \log(\boldsymbol{\sigma}^2) &= W_{h\sigma}h_{enc} + b_{h\sigma} \\
 \boldsymbol{\varepsilon} &\sim \mathcal{N}(0, \mathbf{I}), \quad z = \boldsymbol{\mu} + \boldsymbol{\sigma} \otimes \boldsymbol{\varepsilon} \\
 h_{dec} &= \text{relu}(W_{zh}z + b_{zh}) \\
 x' &= \text{sigmoid}(W_{hx}h_{dec} + b_{hx})
 \end{aligned} \tag{6.1}$$

By feeding both the news documents and the reader comments into VAEs, we equip the model a ability of capturing the information from them jointly. However, there is a large amount of noisy information hidden in the comments. Hence we design a weighted combination mechanism for fusing news and comments in the VAEs. Precisely, we split the variational lower bound $\mathcal{L}(\theta, \varphi; \mathbf{x})$ into two parts and fuse them using the comment weight $\boldsymbol{\rho}$:

$$\mathcal{L}(\theta, \varphi; \mathbf{x}) = \mathcal{L}(\theta, \varphi; \mathbf{x}_d) + \boldsymbol{\rho} \times \mathcal{L}(\theta, \varphi; \mathbf{x}_c) \tag{6.2}$$

The calculation of $\boldsymbol{\rho}$ will be discussed later.

The news sentence salience estimation is conducted by an unsupervised data reconstruction framework. Assume that $\mathbf{S}_z = \{\mathbf{s}_z^1, \mathbf{s}_z^2, \dots, \mathbf{s}_z^m\}$ are m latent aspect vectors used for reconstructing all the latent semantic vectors $\mathbf{Z} = \{\mathbf{z}^1, \mathbf{z}^2, \dots, \mathbf{z}^n\}$. Thereafter, the variational-decoding progress of VAEs can map the latent aspect vector \mathbf{S}_z to \mathbf{S}_h , and then produce m new aspect term vectors \mathbf{S}_x :

$$\begin{aligned} s_h &= \text{relu}(W_{zh}s_z + b_{zh}) \\ s_x &= \text{sigmoid}(W_{hx}s_h + b_{hx}) \end{aligned} \quad (6.3)$$

VAESum employs an alignment mechanism [2, 85] to recall the lost detailed information from the input sentence. Inspired this idea, we design a jointly weighted alignment mechanism by considering the news sentence and the comment sentence simultaneously. For each decoder hidden state s_h^i , we align it with each news encoder hidden state h_d^j by an alignment vector $a^d \in \mathbb{R}^{n_d}$. We also align it with each comments encoder hidden state h_c^j by an alignment vector $a^c \in \mathbb{R}^{n_c}$. In order to filter the noisy information from the comments, we again employ the comment weight $\boldsymbol{\rho}$ to adjust the alignment vector of comments:

$$\tilde{a}^c = a^c \times \boldsymbol{\rho} \quad (6.4)$$

The news-based context vector c_d^i and the comment-based context vector c_c^i can be obtained by linearly blending the input hidden states respectively. Then the output hidden state can be updated based on the context vectors:

$$\tilde{s}_h^i = \tanh(W_{dh}^h c_d^i + W_{ch}^h c_c^i + W_{hh}^a s_h^i) \quad (6.5)$$

Then we can generate the updated output aspect vectors based on \tilde{s}_h^i . We add a similar alignment mechanism into the output layer.

\mathbf{S}_z , \mathbf{S}_h , and \mathbf{S}_x can be used to reconstruct the space to which they belong respectively. In order to capture the information from comments, we design a joint reconstruction approach here. Let $\mathbf{A}_d \in \mathbb{R}^{n_d \times m}$ be the reconstruction coefficient matrix for news sentences, and $\mathbf{A}_c \in \mathbb{R}^{n_c \times m}$ be the reconstruction coefficient matrix for comment sentences. The optimization objective contains three reconstruction terms, jointly considering the latent semantic reconstruction and the term vector space reconstruction for news and comments respectively:

$$\begin{aligned} \mathcal{L}_A = & (\|Z_d - A_d S_z\|_2^2 + \|H_d - A_d S_h\|_2^2 + \|X_d - A_d S_x\|_2^2) \\ & + \boldsymbol{\rho} \times (\|Z_c - A_c S_z\|_2^2 + \|H_c - A_c S_h\|_2^2 + \|X_c - A_c S_x\|_2^2) \end{aligned} \quad (6.6)$$

This objective is integrated with the variational lower bound of VAEs $\mathcal{L}(\theta, \varphi; \mathbf{x})$ and optimized in a multi-task learning fashion. Then the new optimization objective is:

$$\mathcal{J} = \min_{\Theta} (-\mathcal{L}(\theta, \varphi; x) + \mathcal{L}_A) \quad (6.7)$$

where Θ is a set of all the parameters related to this task. We define the magnitude of each row of \mathbf{A}_d as the salience scores for the corresponding news sentences.

We should note that the most important variable in our framework is the comment weight vector $\boldsymbol{\rho}$, which appears in all the three components of our framework. The basic idea for calculating $\boldsymbol{\rho}$ is that if the comment sentence is more similar to the news content, then it contains less noisy information. For all the news sentences X_d and all the comment sentences X_c , calculate the relation matrix $R \in \mathbb{R}^{n_d \times n_c}$ by:

$$R = X_d \times X_c^T \quad (6.8)$$

Then we add an average pooling layer to get the coefficient value for each comment

sentence:

$$\mathbf{r} = \frac{1}{n_c} \sum_{i=1}^{n_c} R[:, i] \quad (6.9)$$

Finally, we add a sigmoid function to adjust the coefficient value to (0, 1):

$$\boldsymbol{\rho} = \text{sigmoid}(\mathbf{r}) \quad (6.10)$$

Because we have different representations from different vector space for the sentences, therefore we can calculate the comment weight in different semantic vector space. Here we use two spaces, namely, latent semantic space obtained by VAEs, and the original bag-of-words vector space. Then we can merge the weights by a parameter λ_p :

$$\boldsymbol{\rho} = \lambda_p \times \boldsymbol{\rho}_z + (1 - \lambda_p) \times \boldsymbol{\rho}_x \quad (6.11)$$

where $\boldsymbol{\rho}_z$ and $\boldsymbol{\rho}_x$ are the comment weight calculated from latent semantic space and term vector space. Actually, we can regard $\boldsymbol{\rho}$ as some gates to control the proportion of each comment sentence absorbed by the framework.

6.2.3 Preparation of Entity Mentions for Rewriting

Summaries may contain phrases that are not understandable out of context since the sentences compiled from different documents might contain too little, too much, or repeated information about the referent. A human summary writer only uses the full-form mention (e.g. President Barack Obama) of an entity one time and uses the short-form mention (e.g. Obama) in the other places. Analogously, for a particular entity, our framework requires that the full-form mention of the entity should only appear one time in the summary and its other appearances should use the most concise form. Some early works perform rewriting along with the greedy selection of individual sentence [101]. Some other works perform summary rewriting as a post-

processing step [115]. In contrast with such works, the rewriting consideration in our framework is jointly assessed together with other summarization requirements under a unified optimization model. This brings in two advantages. First, the assessment of rewriting operation is jointly considered with the generation of the compressive summary so that it has a global view to generate better rewriting results. Second, we can make full use of the length limit because the effect of rewriting operation on summary length is simultaneously considered with other constraints in the model. To support the generation of compressive summaries via optimization, we explore a finer syntactic unit, namely, noun/verb phrase. Precisely, we first decompose the sentences into noun/verb phrases and the salience of each phrase is calculated by jointly considering its importance in reports and comments.

We first conduct co-reference resolution for each document using Stanford co-reference resolution package [62]. We adopt those resolution rules that are able to achieve high quality and address our need for summarization. In particular, Sieve 1, 2, 3, 4, 5, 9, and 10 in the package are employed. A set of clusters are obtained and each cluster contains the mentions corresponding to the same entity in a document. The clusters from different documents in the same topic are merged by matching the named entities. Three types of entities are considered, namely, person, location, and organization.

Let M denote the mention cluster of an entity. The full-form mention m^f is determined as:

$$m^f = \arg \max_{m \in M} \sum_{t \in m} tf'(t) \quad (6.12)$$

where $tf'(t)$ is calculated in M . We do not simply select the longest one since it could be too verbose. The short-form mention m^s is determined as:

$$m^s = \arg \max_{m \in M'} \sum_{t \in m} tf'(t) \quad (6.13)$$

where M' contains the mentions that are the shortest and meanwhile are not pronouns.

6.2.4 Summary Construction

In order to produce reader-aware summaries, we refine the models in Chapter 4 and Chapter 5 to consider the news sentences salience information obtained by our framework. Based on the parsed constituency tree for each input sentence, we extract the noun-phrases (NPs) and verb-phrases (VPs). The overall objective function of this optimization formulation for selecting salient NPs and VPs is formulated as an integer linear programming (ILP) problem:

$$\max\left\{\sum_i \alpha_i S_i - \sum_{i < j} \alpha_{ij} (S_i + S_j) R_{ij}\right\}, \quad (6.14)$$

where α_i is the selection indicator for the phrase P_i , S_i is the salience scores of P_i , α_{ij} and R_{ij} is co-occurrence indicator and the similarity a pair of phrases (P_i, P_j) respectively. The similarity is calculated with the Jaccard Index based method.

In order to obtain coherent summaries with good readability, we add some constraints into the ILP framework. For details, please refer to Chapter 4 and Chapter 5. We just introduce the constraints for conducting entity rewriting here. Let \mathbf{P}_M be the phrases that contain the entity corresponding to the cluster M . For each $P_i \in \mathbf{P}_M$, two indicators γ_i^f and γ_i^s are defined. γ_i^f indicates that the entity in P_i is rewritten by the full-form, while γ_i^s indicates that the entity in P_i is rewritten by the short-form. To adopt our rewriting strategy, we design the following constraints:

$$\text{if } \exists P_i \in \mathbf{P}_M \wedge \alpha_i = 1, \sum_{P_j \in \mathbf{P}_M} \gamma_j^f = 1, \quad (6.15)$$

$$\text{if } P_i \in \mathbf{P}_M \wedge \alpha_i = 1, \gamma_i^f + \gamma_i^s = 1. \quad (6.16)$$

Note that if a phrase contains several mentions of the same entity, we can safely rewrite the latter appearances with the short-form mention and we only need to decide the rewriting strategy for the first appearance.

6.3 Data Description

In this section, we describe the preparation process of the dataset. Then we provide some properties and statistics.

6.3.1 Background

The definition of the terminology related to the dataset is given as follows.⁴

Topic: A topic refers to an event and it is composed of a set of news documents from different sources.

Document: A news article describing some aspects of the topic. The set of documents in the same topic typically span a period, say a few days.

Category: Each topic belongs to a category. There are 6 predefined categories: (1) Accidents and Natural Disasters, (2) Attacks (Criminal/Terrorist), (3) New Technology, (4) Health and Safety, (5) Endangered Resources, and (6) Investigations and Trials (Criminal/Legal/Other).

Aspect: Each category has a set of predefined aspects. Each aspect describes one important element of an event. For example, for the category “Accidents and Natural Disasters”, the aspects are “WHAT”, “WHEN”, “WHERE”, “WHY”, “WHO_AFFECTED”, “DAMAGES”, and “COUNTERMEASURES”.

Aspect facet: An aspect facet refers to the actual content of a particular aspect for a particular topic. Take the topic “Malaysia Airlines Disappearance” as an example,

⁴In fact, for the core terminology, namely, topic, document, category, and aspect, we follow the MDS task in TAC (<https://tac.nist.gov//2011/Summarization/Guided-Summ.2011.guidelines.html>).

facets for the aspect “WHAT” include “missing Malaysia Airlines Flight 370”, “two passengers used passports stolen in Thailand from an Austrian and an Italian.” etc. Facets for the aspect “WHEN” are “ Saturday morning”, “about an hour into its flight from Kuala Lumpur”, etc.

Comment: A piece of text written by a reader conveying his or her altitude, emotion, or any thought on a particular news document.

6.3.2 Data Collection

The first step is to select topics. The selected topics should be in one of the above categories. We make use of several ways to find topics. The first way is to search the category name using Google News. The second way is to follow the related tags on Twitter. One more useful method is to scan the list of event archives on the Web, such as earthquakes happened in 2017 ⁵.

For some news websites, in addition to provide news articles, they offer a platform to allow readers to enter comments. Regarding the collection of news documents, for a particular topic, one consideration is that reader comments can be easily found. Another consideration is that all the news documents under a topic must be collected from different websites as far as possible. Similar to the methods used in DUC and TAC, we also capture and store the content using XML format.

Each topic is assigned to 4 experts, who are major in journalism, to conduct the summary writing. The task of summary writing is divided into two phases, namely, aspect facet identification, and summary generation. For the aspect facet identification, the experts read and digested all the news documents and reader comments under the topic. Then for each aspect, the experts extracted the related facets from the news document. The summaries were generated based on the annotated aspect facets. When selecting facets, one consideration is those facets that are popular in

⁵https://en.wikipedia.org/wiki/Category:2017_earthquakes

both news documents and reader comments have higher priority. Next, the facets that are popular in news documents have the next priority. The generated summary should cover as many aspects as possible, and should be well-organized using complete sentences with a length restriction of 100 words.

After finishing the summary writing procedure, we employed another expert for scrutinizing the summaries. Each summary is checked from five linguistic quality perspectives: grammaticality, non-redundancy, referential clarity, focus, and coherence. Finally, all the model summaries are stored in XML files.

6.3.3 Data Properties

The dataset contains 45 topics from those 6 predefined categories. Some examples of topics are “Malaysia Airlines Disappearance”, “Flappy Bird”, “Bitcoin Mt. Gox”, etc. All the topics and categories are listed in Section 6.5.4. Each topic contains 10 news documents and 4 model summaries. The length limit of the model summary is 100 words (slitted by space). On average, each topic contains 215 pieces of comments and 940 comment sentences. Each news document contains an average of 27 sentences, and each sentence contains an average of 25 words. 85% of non-stop model summary terms (entities, unigrams, bigrams) appeared in the news documents, and 51% of that appeared in the reader comments. The dataset contains 19k annotated aspect facets.

6.4 Experimental Setup

6.4.1 Dataset and Metrics

The properties of our own dataset are depicted in Section 6.3.3. We use ROUGE score as our evaluation metric [79]. F-measures of ROUGE-1, ROUGE-2 and ROUGE-

SU4 are reported.

6.4.2 Comparative Methods

To evaluate the performance of our dataset and the proposed framework **RAVAE-Sum** for RA-MDS, we compare our model with the following methods:

- **RA-Sparse** [70]: A sparse-coding-based method we designed for sentence salience estimation by jointly considering news documents and reader comments. The global loss function is defined as follows:

$$J = \min_A \frac{1}{2m} \sum_{i=1}^m \rho_i \|\mathbf{x}_i - \sum_{j=1}^m a_j \mathbf{x}_j\|_2^2 + \frac{1}{2n} \sum_{i=1}^n \tau_i \|\mathbf{z}_i - \sum_{j=1}^m a_j \mathbf{x}_j\|_2^2 + \lambda \|A\|_1$$

s.t. $a_j \geq 0$ for $j \in \{1, \dots, m\}, \lambda > 0$

(6.17)

where x and z denote the sentences from news reports and user comments respectively. ρ is the sentence position information. τ is the comment weight. Please refer to Li et al. [70] for more details.

- **Lead** [132] : It ranks the news sentences chronologically and extracts the leading sentences one by one until the length limit.
- **Centroid** [106]: It summarizes clusters of news articles automatically grouped by a topic detection system, and then it uses information from the centroids of the clusters to select sentences.
- **LexRank** [27] and **TextRank** [94]: Both methods are graph-based unsupervised framework for sentence salience estimation based on PageRank algorithm.
- **Concept** [6]: It generates abstractive summaries using phrase-based optimization framework with concept weight as salience estimation. The concept

set contains unigrams, bigrams, and entities. The weighted term-frequency is used as the concept weight.

We can see that only the method RA-Sparse can handle RA-MDS. All the other methods are only for traditional MDS without comments.

6.4.3 Experimental Settings

The input news sentences and comment sentences are represented as BoWs vectors with dimension $|V|$. The dictionary V is created using unigrams, bigrams and named entity terms. n_d and n_c are the number of news sentences and comment sentences respectively. For the number of latent aspects used in data reconstruction, we let $m = 5$. For the neural network framework, we set the hidden size $d_h = 500$ and the latent size $K = 100$. For the parameter λ_p used in comment weight, we let $\lambda_p = 0.2$. Adam [53] is used for gradient based optimization with a learning rate 0.001. Our neural network based framework is implemented using Theano [120] on a single GPU⁶.

6.5 Results and Discussions

6.5.1 Results on Our Dataset

The results of our framework as well as the baseline methods are depicted in Table 6.1. It is obvious that our framework RAVAESum is the best among all the comparison methods. Specifically, it is better than RA-Sparse significantly ($p < 0.05$), which demonstrates that VAEs based latent semantic modeling and joint semantic space reconstruction can improve the MDS performance considerably. Both

⁶Tesla K80, 1 Kepler GK210 is used, 2496 Cuda cores, 12G GDDR5 memory.

Table 6.1: Summarization performance.

| System | ROUGE-1 | ROUGE-2 | ROUGE-SU4 |
|-----------|---------------|---------------|---------------|
| Lead | 0.384 | 0.110 | 0.144 |
| TextRank | 0.402 | 0.122 | 0.159 |
| LexRank | 0.425 | 0.135 | 0.165 |
| Centroid | 0.402 | 0.141 | 0.171 |
| Concept | 0.422 | 0.149 | 0.177 |
| RA-Sparse | 0.442 | 0.157 | 0.188 |
| RAVAESum | 0.443* | 0.171* | 0.196* |

Table 6.2: Further investigation of RAVAESum.

| System | ROUGE-1 | ROUGE-2 | ROUGE-SU4 |
|--------------|---------------|---------------|---------------|
| RAVAESum-noC | 0.437 | 0.162 | 0.189 |
| RAVAESum | 0.443* | 0.171* | 0.196* |

RAVAESum and RA-Sparse are better than the methods without considering reader comments.

6.5.2 Further Investigation of Our Framework

To further investigate the effectiveness of our proposed RAVAESum framework, we adjust our framework by removing the comments related components. Then the model settings of RAVAESum-noC are similar to VAESum [74]. The evaluation results are shown in Table 6.2, which illustrate that our framework with reader comments RAVAESum is better than RAVAESum-noC significantly ($p < 0.05$).

Moreover, as mentioned in VAESum [74], the output aspect vectors contain the word salience information. Then we select the top-10 terms for event ‘‘Sony Virtual Reality PS4’’, and ‘‘Bitcoin Mt. Gox Offile’’ for model RAVAESum (+C) and RAVAESum-noC (-C) respectively, and the results are shown in Table 6.3. It is obvious that the rank of the top salience terms are different. We check from the news documents and reader comments and find that some terms are enhanced by the reader comments successfully. For example, for the topic ‘‘Sony Virtual Reality

Table 6.3: Top-10 terms extracted from each topic according to the word salience values.

| Topic | $\pm C$ | Top-10 Terms |
|----------------------------|---------|--|
| “Sony Virtual Reality PS4” | −C | Sony, headset, game, virtual, morpheus, reality, vr, project, playstation, Yoshida |
| | +C | Sony, game, vr, virtual, headset, reality, morpheus, <i>oculus</i> , project, playstation |
| “Bitcoin Mt. Gox Offile” | −C | bitcoin, gox, exchange, mt., currency, Gox, virtual, company, money, price |
| | +C | bitcoin, currency, money, exchange, gox, mt., virtual, company, price, world |

PS4”, many readers talked about the product of “Oculus”, hence the word “oculus” is assigned a high salience by our model.

6.5.3 Case Study

Based on the news and comments of the topic “Sony Virtual Reality PS4”, we generate two summaries with our model considering comments (RAVAESum) and ignoring comments (RAVAESum-noC) respectively. The summaries and ROUGE evaluation are given in Table 6.4. All the ROUGE values of our model considering comments are better than those ignoring comments with large gaps. The sentences in ***italic bold*** of the two summaries are different. By reviewing the comments of this topic, we find that many readers talked about “Oculus”, the other product with virtual reality techniques. This issue is well identified by our model and select the sentence “*Mr. Yoshida said that Sony was inspired and encouraged to do its own virtual reality project after the enthusiastic response to the efforts of Oculus VR and Valve, another game company working on the technology.*”.

We also present an entity rewriting case study. For person name “Dong Nguyen” in the topic “Flappy Bird”, the summary without entity rewriting contains different mention forms such as “Dong Nguyen”, “Dong” and “Nguyen”. After rewriting,

Table 6.4: Generated summaries for the topic “Sony Virtual Reality PS4”.

| System | ROUGE-1 | ROUGE-2 | ROUGE-SU4 |
|--|--------------|--------------|--------------|
| RAVAESum-noC | 0.482 | 0.184 | 0.209 |
| A virtual reality headset that’s coming to the PlayStation 4. <i>Today announced the development of “Project Morpheus” (Morpheus) “a virtual reality (VR) system that takes the PlayStation4 (PS4)”</i> . Shuhei Yoshida, president of Sony Computer Entertainment, revealed a prototype of Morpheus at the Game Developers Conference in San Francisco on Tuesday. Sony showed off a prototype device V called Project Morpheus V that can be worn to create a virtual reality experience when playing games on its new PlayStation 4 console. <i>The camera on the Playstation 4 using sensors that track the player’s head movements.</i> | | | |
| RAVAESum | 0.490 | 0.230 | 0.243 |
| Shuhei Yoshida, president of Sony Computer Entertainment, revealed a prototype of Morpheus at the Game Developers Conference in San Francisco on Tuesday. A virtual reality headset that’s coming to the PlayStation 4. Sony showed off a prototype device V called Project Morpheus V that can be worn to create a virtual reality experience when playing games on its new PlayStation 4 console. <i>Mr. Yoshida said that Sony was inspired and encouraged to do its own virtual reality project after the enthusiastic response to the efforts of Oculus VR and Valve, another game company working on the technology.</i> | | | |

“Dong” is replaced by “Nguyen”, which makes the co-reference mentions clearer. As expected, there is only one full-form mention, such as “Nguyen Ha Dong, a Hanoi-based game developer” “Shuhei Yoshida, president of Sony Computer Entertainment Worldwide Studios”, and “The Australian Maritime Safety Authority’s Rescue Coordination Centre, which is overseeing the rescue ”, in each summary.

6.5.4 Topics

All the topics and the corresponding categories are shown in Table 6.5. The six predefined categories are: (1) Accidents and Natural Disasters, (2) Attacks (Criminal/Terrorist), (3) New Technology, (4) Health and Safety, (5) Endangered Resources, and (6) Investigations and Trials (Criminal/Legal/Other).

Table 6.5: All the topics and the corresponding categories.

| Topic | Category |
|--|-----------------|
| Boston Marathon Bomber Sister Arrested | 6 |
| iWatch | 3 |
| Facebook Offers App With Free Access in Zambia | 3 |
| 441 Species Discovered in Amazon | 5 |
| Beirut attack | 2 |
| Great White Shark Choked by Sea Lion | 1 |
| Sony virtual reality PS4 | 3 |
| Akademik Shokalskiy Trapping | 1 |
| Missing Oregon Woman Jennifer Huston Committed Suicide | 6 |
| Bremerton Teen Arrested Murder 6-year-old Girl | 6 |
| Apple And IBM Team Up | 3 |
| California Father Accused Killing Family | 6 |
| Los Angeles Earthquake | 1 |
| New Species of Colorful Monkey | 5 |
| Japan Whaling | 5 |
| Top Doctor Becomes Latest Ebola Victim | 4 |
| New South Wales Bushfires | 1 |
| UK David Cameron Joins Battle Against Dementia | 4 |
| UK Cameron Calls for Global Action on Superbug Threat | 4 |
| Karachi Airport Attack | 2 |
| Air Algeria Plane Crash | 1 |
| Flappy Bird | 3 |
| Moscow Subway Crash | 1 |
| Rick Perry Lawyers Dismissal of Charges | 6 |
| New York Two Missing Amish Girls Found | 6 |
| UK Contaminated Drip Poisoned Babies | 4 |
| Taiwan Police Evict Student Protesters | 2 |
| US General Killed in Afghan | 5 |
| Monarch butterflies drop | 5 |
| UN Host Summit to End Child Brides | 4 |
| Two Tornadoes in Nebraska | 1 |
| Global Warming Threatens Emperor Penguins | 5 |
| Malaysia Airlines Disappearance | 1 |
| Google Conference | 3 |
| Africa Ebola Out of Control in West Africa | 4 |
| Shut Down of Malaysia Airlines mh17 | 1 |
| Sochi Terrorist Attack | 2 |
| Fire Phone | 3 |
| ISIS executes David Haines | 2 |
| UK Rotherham 1400 Child Abuse Cases | 6 |
| Rare Pangolins Asians eating Extinction | 5 |
| Kunming Station Massacre | 2 |
| Bitcoin Mt. Gox | 3 |
| UK Jimmy Savile Abused Victims in Hospital | 6 |
| ISIS in Iraq | 2 |

6.6 Summary

In this Chapter, we investigate the problem of reader-aware multi-document summarization (RA-MDS) and introduce a new dataset. To tackle the RA-MDS, we extend a variational auto-encodes (VAEs) based MDS framework by jointly considering news documents and reader comments. The methods for data collection, aspect annotation, and summary writing and scrutinizing by experts are described. Experimental results show that reader comments can improve the summarization performance, which demonstrate the usefulness of the proposed dataset.

Chapter 7

Persona-Aware Abstractive Tips Generation

7.1 Background

We study the content of tips and the associated users, items, and ratings collected from a commercial E-commerce site. It can be observed that different users have different tips writing characteristics which include different writing styles. We use the term “**persona**” for denoting users’ written text characteristics such as wording and style. Figure 7.1a depicts some tips written for the item “Sony ICF-S79V Weather Band Shower Radio”¹ from different users. It is obvious that different tips for this item written by different users follow different styles, even though all of them assign the same ratings, namely, 5 for this item. Some users write tips such as “Great fit and finish for shower.”, “Easy to set up stations.”, and “Excellent design and quality construction.” to describe the product quality and their experience directly. These users prefer short sentences and direct complement words such as “great”, “perfect”, “excellent”, etc. Some users express their experience indirectly

¹<https://www.amazon.com/sony-icf-s79v-weather-shower-radio/dp/b00000dm9w>

| Tips | Rating |
|--|---------------|
| (1) Great fit and finish for shower. | 5 |
| (2) I selected this radio for myself several years ago and i have found that all claims for it are true. | 5 |
| (3) If your looking for a radio for your shower then look no further. | 5 |
| (4) Easy to set up stations. | 5 |
| (5) Excellent design and quality construction. | 5 |
| (6) First one lasted years just bought another one. | 5 |

(a) Tips for the item “Sony Weather Band Shower Radio”.

| Tips | Rating |
|--|---------------|
| (1) Works perfectly in my msi wind. | 5 |
| (2) Perfect size for a home office. | 5 |
| (3) Excellent player for price. | 5 |
| (4) Wonderful docking speaker with full sound. | 4 |
| (5) I like it when it not dropping the signal. | 4 |
| (6) Works fine in a pinch. | 3 |
| (7) Piece of crap do bother. | 1 |
| (8) Revised star piece of crap. | 1 |

(b) Tips for different items written by a particular user.

Figure 7.1: Example of tips.

by describing some facts using longer sentences such as “If your looking for a radio for your shower then look no further.” and “First one lasted years just bought another one.”. Therefore, different users indeed have different persona style when writing tips for items. Figure 7.1b depicts some tips written by a particular user for different items. We can see that this user prefers to use short sentences to express the experience. Moreover, by analyzing the tips with different ratings, we can know that the writing style is also controlled by the sentiment associated with the user. This user likes to use “perfect” and “excellent” to describe the item when she/he gives a high rating score for the item. On the other hand, this user also writes tips containing “piece of crap” to the items she/he does not like. Therefore, it is obvious that users possess an underlying persona style conditioned on the sentiment for the item.

Intuitively, the quality of abstractive tips generation can be improved if the model considers the user persona information when conducting the text generation process. However, to our best knowledge, previous works on tips generation such as [75] do not consider persona information. To tackle this problem, we investigate the approach called persona-aware tips generation, which can generate tips considering the persona information. There are two main challenges: (1) How to capture and represent the persona information; (2) How to integrate the sentiment signal with the persona information to control the style and the sentiment of the generated tips jointly.

Though abstractive text generation is a difficult task, with benefits derived from the development of deep learning especially recurrent neural networks (RNN), the performance of abstractive text generation has been improved significantly. Meanwhile, some researchers also apply RNN on review generation [23, 103, 119, 142] and obtain some good results. Hu et al. [47] revised the models for controlling the sentiment and tense of the generated reviews. However, all these works do not consider persona information in their models. Persona information plays an important role in recommendation systems which should not be neglected. Li et al. [67] propose two methods to conduct the persona modeling for text generation, but they focus on dialog systems and do not consider the sentiment information in their framework. Different from these previous works, our proposed framework can jointly consider the persona information and the sentiment signal when conducting the abstractive tips generation.

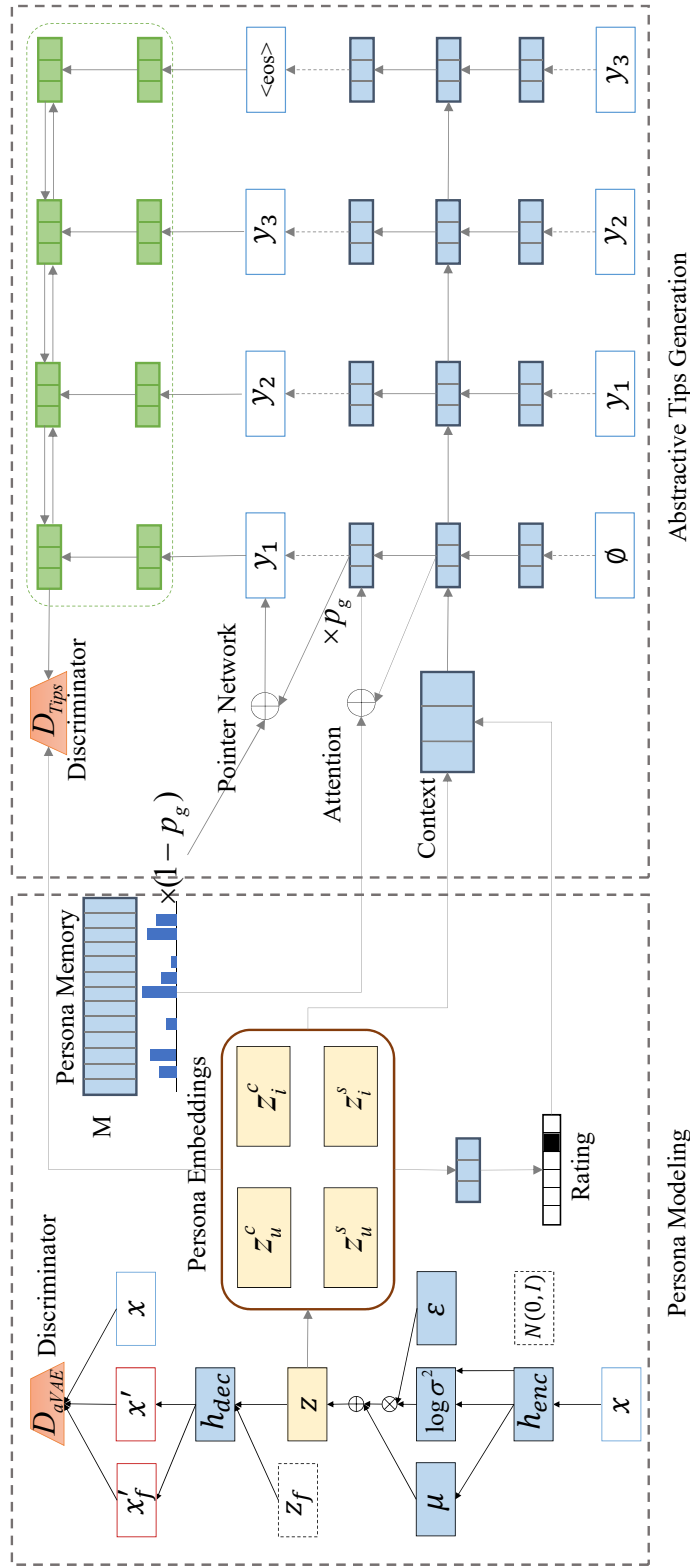


Figure 7.2: Our proposed framework for persona-aware abstractive tips generation.

7.2 Framework Description

7.2.1 Overview

The data consists of users, items, ratings, review content, and tips texts. We denote the whole training corpus by $\mathcal{X} = \{\mathcal{U}, \mathcal{I}, \mathcal{R}, \mathcal{C}, \mathcal{S}\}$, where \mathcal{U} and \mathcal{I} are the sets of users and items respectively, \mathcal{R} is the set of ratings, \mathcal{C} is the set of review documents, and \mathcal{S} is the set of tips texts. We use \mathcal{C}_u and \mathcal{S}_u to denote all the historical reviews and tips respectively of the user u .

As shown in Figure 7.2, our framework contains two major models: persona modeling on the left and abstractive tips generation on the right. For persona modeling, the process is conducted on the historical reviews \mathcal{C}_u and tips \mathcal{S}_u separately. Take the historical tips \mathcal{S}_u for example, we represent them using bag-of-words (BoWs) representations. We use \mathbf{x}_u^s to denote the BoWs vector. Then we feed \mathbf{x}_u^s into the adversarial variational auto-encoders (aVAE^s) and obtain the persona embedding \mathbf{z}_u^s for the user u . For the item i , there are some users writing tips for it. Then we can also conduct persona modeling for i based on the historical tips \mathcal{S}_i using the same aVAE model and get the persona embedding \mathbf{z}_i^s . The purpose of persona modeling for the item i is that when conducting tips generation for the user u , the model will also consider the tips from the other users having similar interests with u . We call this as *personalized collaborative influence*. Obviously, user reviews are different with tips. In order to distill persona information from reviews, we design another aVAE model (aVAE^c) to map the historical reviews \mathcal{C}_u and \mathcal{C}_i to persona embeddings \mathbf{z}_u^c and \mathbf{z}_i^c for the user u and the item i respectively. We also design an external persona memory \mathbf{M} for storing the persona related words for the current user and item which will be utilized in abstractive tips generation. In order to control the sentiment of the generated tips, the distilled persona embeddings are used as latent factors for users and items and are fed into a multilayer perceptron (MLP)

based neural network component to get the predicted rating r . Then we transform r to a one-hot vector \mathbf{r} which will be used as the sentiment controller when conducting the tips generation.

For abstractive tips generation, we design a sequence decoding model based on a gated recurrent neural network called Gated Recurrent Unit (GRU) [17]. Importantly, the persona embeddings and the rating vector are combined to construct a context vector which plays a significant role in abstractive tips generation. Pointer Networks is used to retrieve relevant words from the persona memory \mathbf{M} . During the training procedure, we add an adversarial training strategy to fine-tune the tips generation model.

7.2.2 Persona Modeling

Persona Embedding Learning

The target of persona modeling is to distill the persona information from the users' historical tips and reviews. Some previous works in recommendation systems [87, 109, 129] employ topic modeling methods such as Latent Dirichlet Allocation (LDA) [7] or its variants to analyze the text corpus and use the latent topic distribution to represent each document. Considering the fact that our tips generation component is based on neural networks, existing topic modeling paradigms cannot be incorporated into our framework in an elegant manner. Fortunately, Kingma and Welling [54] propose a method called variational auto-encoders (VAEs) which is able to detect latent topics using neural modeling paradigm [11]. VAEs consists of two parts: inference (variational-encoder) and generation (variational-decoder). Recall that the dictionary is V . For historical tips based persona modeling, the input are the BoWs vectors $\mathbf{x}_u^s \in \mathbb{R}^{|V|}$ and $\mathbf{x}_i^s \in \mathbb{R}^{|V|}$ for the user u and the item i respectively. For convenience, we will use \mathbf{x} to represent them in this section. As shown in the

left part of Figure 7.2, for each input BoWs vector \mathbf{x} , the variational-encoder can map it to a latent variable $\mathbf{z} \in \mathbb{R}^K$, which can be used to generate a new variable \mathbf{x}' via the variational-decoder component to reconstruct the original term vector. The target is to maximize the probability of each \mathbf{x} in the dataset based on the generation process according to:

$$p(\mathbf{x}) = \int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z} \quad (7.1)$$

For the purpose of solving the intractable integral of the marginal likelihood, a model $q(\mathbf{z}|\mathbf{x})$ is introduced as the approximation to the intractable of the true posterior $p(\mathbf{z}|\mathbf{x})$. The aim of optimization is to reduce the Kullback-Leibler divergence (KL) between $q(\mathbf{z}|\mathbf{x})$ and $p(\mathbf{z}|\mathbf{x})$ by maximizing the variational lower bound \mathcal{L}_{VAE} :

$$\mathcal{L}_{VAE} = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z})] - D_{KL}[q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})] \quad (7.2)$$

In order to differentiate and optimize the lower bound \mathcal{L}_{VAE} , following the core idea of VAEs, we use a neural network framework for the encoder $q(\mathbf{z}|\mathbf{x})$ for better approximation. Similar to previous works [39, 54], we assume that both the prior and posterior of the latent variables are Gaussian, i.e., $p(\mathbf{z}) = \mathcal{N}(0, \mathbf{I})$ and $q(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2\mathbf{I})$, where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ denote the variational mean and standard deviation respectively, which can be calculated with a multilayer perceptron (MLP). Precisely, given the BoWs vector \mathbf{x} of the historical tips, we first project it to a hidden space:

$$\mathbf{h}_{enc} = \text{relu}(\mathbf{W}_{xh}\mathbf{x} + \mathbf{b}_{xh}) \quad (7.3)$$

where $\mathbf{h}_{enc} \in \mathbb{R}^{d_h}$, \mathbf{W}_{xh} and \mathbf{b}_{xh} are the neural parameters. $\text{relu}(\mathbf{x}) = \max(0, \mathbf{x})$ is the activation function. Then the Gaussian parameters $\boldsymbol{\mu} \in \mathbb{R}^K$ and $\boldsymbol{\sigma} \in \mathbb{R}^K$ can

be obtained via a linear transformation based on \mathbf{h}_{enc} :

$$\begin{aligned}\boldsymbol{\mu} &= \mathbf{W}_{h\mu}\mathbf{h}_{enc} + \mathbf{b}_{h\mu} \\ \log(\boldsymbol{\sigma}^2) &= \mathbf{W}_{h\sigma}\mathbf{h}_{enc} + \mathbf{b}_{h\sigma}\end{aligned}\tag{7.4}$$

The latent variable $\mathbf{z} \in \mathbb{R}^K$ can be calculated using the reparameterization trick:

$$\boldsymbol{\varepsilon} \sim \mathcal{N}(0, \mathbf{I}), \quad \mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \otimes \boldsymbol{\varepsilon}\tag{7.5}$$

where $\boldsymbol{\varepsilon} \in \mathbb{R}^K$ is an auxiliary noise variable. This is the encoding process, and we denote all the parameters of this state as Θ_{Enc} .

Given the latent variable \mathbf{z} , a new vector \mathbf{x}' is generated via the conditional distribution $p(\mathbf{x}|\mathbf{z})$ according to the variational-decoder:

$$\mathbf{h}_{dec} = \text{relu}(\mathbf{W}_{zh}\mathbf{z} + \mathbf{b}_{zh})\tag{7.6}$$

$$\mathbf{x}' = \sigma(\mathbf{W}_{hx}\mathbf{h}_{dec} + \mathbf{b}_{hx})\tag{7.7}$$

We denote all the parameters in the decoding stage using Θ_{Dec} . Finally, based on the reparameterization trick in Equation 7.5, we can get the analytical representation of \mathcal{L}_{VAE} :

$$\begin{aligned}\log p(\mathbf{x}|\mathbf{z}) &= \sum_{i=1}^{|\mathbf{V}|} x_i \log x'_i + (1 - x_i) \cdot \log(1 - x'_i) \\ -D_{KL}[q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})] &= \frac{1}{2} \sum_{i=1}^K (1 + \log(\sigma_i^2) - \mu_i^2 - \sigma_i^2)\end{aligned}\tag{7.8}$$

For presentation clarity, we let $\mathcal{L}_{Rec} = -\log p(\mathbf{x}|\mathbf{z})$ and $\mathcal{L}_{KL} = D_{KL}[q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})]$, both of them need to be minimized.

We wish to employ the latent variable \mathbf{z} as the distilled persona embeddings. So the quality of \mathbf{z} will affect the performance of tips generation. Some previous works [37, 91, 149] have also shown that the performance of \mathbf{z} is likely to be disturbed during

the training procedure, especially when combining VAEs with the RNN based text generation framework. In order to enhance the performance of the typical VAEs, inspired by the ideas in [36] and [58], we employ the adversarial strategy for the training of VAEs. Generally, we design a discriminator network D_{aVAE} with a vector $\tilde{\mathbf{x}}$ as input, and the target is to recognize if $\tilde{\mathbf{x}}$ is from the true data \mathbf{X} or from the generated samples \mathbf{X}' by VAEs. VAEs will “fool” the discriminator D_{aVAE} by trying the best to produce high quality latent variables \mathbf{z} as well as the generated sample \mathbf{x}' . Then the minimax game between the VAEs and the discriminator can be formulated as follows:

$$\begin{aligned} \min_{VAEs} \max_{D_{aVAE}} & \mathbb{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})} [\log D_{aVAE}(\mathbf{x})] \\ & + \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z}|\mathbf{x})} [\log(1 - D_{aVAE}(VAE_{Dec}(\mathbf{z})))] \\ & + \mathbb{E}_{\mathbf{z}_f \sim p(\mathbf{z})} [\log(1 - D_{aVAE}(VAE_{Dec}(\mathbf{z}_f)))] \end{aligned} \quad (7.9)$$

where VAE_{Dec} is the decoder component of the VAEs model. \mathbf{z} is the latent variable from VAEs, and \mathbf{z}_f is sampled from the prior distribution of \mathbf{z} .

For the design of the discriminator D_{aVAE} , we simply use a multilayer perceptron to process the data.

$$\begin{aligned} \mathbf{h}^{D_v} &= \tanh(\mathbf{W}_{xh}^{D_v} \tilde{\mathbf{x}} + \mathbf{b}_{xh}^{D_v}) \\ y^{D_v} &= \sigma(\mathbf{W}_{hy}^{D_v} \mathbf{h}^{D_v} + b_{hy}^{D_v}) \end{aligned} \quad (7.10)$$

where $\mathbf{W}_{xh}^{D_v} \in \mathbb{R}^{d_h \times |V|}$, $\mathbf{W}_{hy}^{D_v} \in \mathbb{R}^{1 \times d_h}$, $\mathbf{b}_{xh}^{D_v} \in \mathbb{R}^{d_h}$, and $b_{hy}^{D_v} \in \mathbb{R}$. The output y^{D_v} is a real value in the range of $[0, 1]$ and the value 1 means that the sample $\tilde{\mathbf{x}}$ is from the true data. We denote all the parameters in D_{aVAE} using Θ_{D_v} . The optimization

objective to be maximized for D_{aVAE} is formulated as:

$$\begin{aligned} \mathcal{L}_{D_{aVAE}} = & \log(D_{aVAE}(\mathbf{x})) \\ & + \log(1 - D_{aVAE}(VAE_{Dec}(VAE_{Enc}(\mathbf{x})))) \\ & + \log(1 - D_{aVAE}(VAE_{Dec}(\mathbf{z}_f))) \end{aligned} \quad (7.11)$$

Then the parameters Θ_{D_v} are updated using gradient methods:

$$\Theta_{D_v} \leftarrow \Theta_{D_v} - \nabla_{\Theta_{D_v}}(-\mathcal{L}_{D_{aVAE}}) \quad (7.12)$$

Conditioned on the aVAE framework, we will conduct the parameter learning for VAEs Encoder, VAEs Decoder, and discriminator D_{aVAE} using different loss functions respectively. Encoder transforms the input \mathbf{X} to the persona embeddings \mathbf{Z} . On one side, \mathbf{Z} are used to reconstruct the original input. On the other side, \mathbf{Z} are used to conduct the persona-aware tips generation. So the loss signals from both the aVAE and the tips generation framework are used to conduct the optimization for Θ_{Enc} :

$$\Theta_{Enc} \leftarrow \Theta_{Enc} - \nabla_{\Theta_{Enc}}(\mathcal{L}_{KL} + \mathcal{L}_{Rec} + \mathcal{L}_{D_{aVAE}}^z + \mathcal{L}_{Tips}) \quad (7.13)$$

where \mathcal{L}_{KL} and \mathcal{L}_{Rec} are the KL diversity and reconstruction loss from Equation 7.8. \mathcal{L}_{Tips} is the loss signal from the tips generation component. $\mathcal{L}_{D_{aVAE}}^z$ is the output of D_{aVAE} :

$$\mathcal{L}_{D_{aVAE}}^z = -\log(D_{aVAE}(VAE_{Dec}(VAE_{Enc}(\mathbf{x})))) \quad (7.14)$$

For the parameter optimization of VAEs Decoder, we use \mathcal{L}_{Rec} , $\mathcal{L}_{D_{aVAE}}$, \mathcal{L}_{Tips} as the loss signals:

$$\Theta_{Dec} \leftarrow \Theta_{Dec} - \nabla_{\Theta_{Dec}}(\mathcal{L}_{Rec} + \mathcal{L}_{D_{aVAE}} + \mathcal{L}_{Tips}) \quad (7.15)$$

Algorithm 2 Persona embedding learning.

Input: BoWs vectors of historical tips and reviews \mathbf{X} .

Output: The persona embeddings \mathbf{Z} .

- 1: Initialize $\Theta_{Enc}, \Theta_{Dec}, \Theta_{D_v}$;
 - 2: **while** not converged **do**
 - 3: Draw \mathbf{x} from p_{data} .
 - 4: Draw \mathbf{z}_f from prior $p(\mathbf{z})$.
 - 5: $\mathbf{z} = VAE_{Enc}(\mathbf{x})$
 - 6: $\mathbf{x}' = VAE_{Dec}(\mathbf{z})$
 - 7: $\mathbf{x}'_f = VAE_{Dec}(\mathbf{z}_f)$
 - 8: Get $\mathcal{L}_{Rec}, \mathcal{L}_{KL}, \mathcal{L}_{D_{aVAE}}$ according to Equation 7.8 and 7.11.
 - 9: Get \mathcal{L}_{Tips} from tips generation.
 - 10: Update parameters using gradient methods:

$$\Theta_{Enc} \leftarrow \Theta_{Enc} - \nabla_{\Theta_{Enc}} (\mathcal{L}_{KL} + \mathcal{L}_{Rec} + \mathcal{L}_{D_{aVAE}}^z + \mathcal{L}_{Tips})$$

$$\Theta_{Dec} \leftarrow \Theta_{Dec} - \nabla_{\Theta_{Dec}} (\mathcal{L}_{Rec} + \mathcal{L}_{D_{aVAE}} + \mathcal{L}_{Tips})$$

$$\Theta_{D_v} \leftarrow \Theta_{D_v} - \nabla_{\Theta_{D_v}} (-\mathcal{L}_{D_{aVAE}})$$
 - 11: **end while**
 - 12: **return** \mathbf{z} .
-

Finally, the training procedure of aVAE model is shown in Algorithm 2.

Feeding the historical reviews and tips representations ($\mathbf{x}_u^c, \mathbf{x}_i^c, \mathbf{x}_u^s$, and \mathbf{x}_i^s) into $aVAE^c$ (for reviews) and $aVAE^s$ (for tips) respectively, we can obtain four persona embeddings $\mathbf{z}_u^c, \mathbf{z}_i^c, \mathbf{z}_u^s$, and \mathbf{z}_i^s . These persona embeddings will be integrated into the rating prediction component and the tips generation component later.

Sentiment and Rating Modeling

We directly regard the persona embeddings as the latent factors of users and items, and feed them into a multilayer perceptron to conduct the rating prediction. The predicted ratings will be used to control the sentiment of the generated tips.

Specifically, we first map the persona embeddings to a hidden space:

$$\mathbf{h}^r = \tanh(\mathbf{W}_{uch}^r \mathbf{z}_u^c + \mathbf{W}_{ich}^r \mathbf{z}_i^c + \mathbf{W}_{ush}^r \mathbf{z}_u^s + \mathbf{W}_{ish}^r \mathbf{z}_i^s + \mathbf{b}_h^r) \quad (7.16)$$

where $\{\mathbf{W}_{uch}^r, \mathbf{W}_{ich}^r, \mathbf{W}_{ush}^r, \mathbf{W}_{ish}^r\} \in \mathbb{R}^{d_h \times k}$ are the mapping matrices. $\mathbf{b}_h^r \in \mathbb{R}^{d_h}$ is the bias term. \tanh is the hyperbolic tangent activation function. The superscript r refers to variables related to the rating prediction component. For better performance, we can add more layers of non-linear transformations into our model:

$$\mathbf{h}_l^r = \sigma(\mathbf{W}_{hh_l}^r \mathbf{h}_{l-1}^r + \mathbf{b}_{h_l}^r) \quad (7.17)$$

where $\mathbf{W}_{hh_l}^r \in \mathbb{R}^{d_h \times d_h}$ is the mapping matrix for the variables in the hidden layers. l is the index of a hidden layer. Assume that \mathbf{h}_L^r is the output of the last hidden layer. The output layer transforms \mathbf{h}_L^r into a real-valued rating \hat{r} :

$$\hat{r} = \mathbf{W}_{hr}^r \mathbf{h}_L^r + b^r \quad (7.18)$$

where $\mathbf{W}_{hr}^r \in \mathbb{R}^{1 \times d_h}$ and $b^r \in \mathbb{R}$. We formulate the optimization of the parameters Θ_r as a regression problem and the loss function is formulated as:

$$\mathcal{L}^r = \frac{1}{2|\mathcal{X}|} \sum_{u \in \mathcal{U}, i \in \mathcal{I}} (\hat{r}_{u,i} - r_{u,i})^2 \quad (7.19)$$

where \mathcal{X} represents the training set. $r_{u,i}$ is the ground truth rating assigned by the user u to the item i .

The predicted rating is a real value, not a vector, for example, $\hat{r}_{u,i} = 4.321$. In order to incorporate the rating information into the tips generation component, we cast it into an integer 4, and add a vectorization process to obtain the vector representation of rating $\hat{r}_{u,i}$. If the rating range is $[0, 5]$, we will get the rating vector $\hat{\mathbf{r}}_{u,i} = (0, 0, 0, 0, 1, 0)^T$.

External Persona Memory

In addition to represent persona information using the latent embeddings from aVAE, we design an external persona memory for directly storing the persona related words for both the current user u and the current item i . To build the memory, we first collect all the words for the current user u and the current item t from their historical tips. We add a filtering process to remove the stop-words and the low-frequency words. Then we get a local vocabulary storing the indices of the persona words. Recall that we have a global word embedding \mathbf{E} . Then we can get a sub-matrix from \mathbf{E} according to the word indices. We regard this sub-matrix as persona memory. We employ Pointer Networks to retrieve persona information from the memory when generating tips. The details are described in Section 7.2.3.

7.2.3 Abstractive Tips Generation

Overview of Tips Generation

The right part of Figure 7.2 depicts our tips generation model. The basic element is a RNN based sequence modeling component. Pointer Networks (attention modeling and copy mechanism) is introduced to conduct the memory reading. Context information plays an important role in the task of text generation. We combine the persona embeddings and the sentiment information as the context information and construct the context vector which can control the tips text generation. At the operational or testing stage, we use a beam search algorithm [56] for decoding and generating the best tips given a trained model.

Sequence Modeling

Assume that \mathbf{h}_t^s is the sequence hidden state at the time t . It depends on the input at the time t and the previous hidden state \mathbf{h}_{t-1}^s :

$$\mathbf{h}_t^s = f(\mathbf{h}_{t-1}^s, s_t) \quad (7.20)$$

$f(\cdot)$ can be the vanilla RNN, Long Short-Term Memory (LSTM) [43], or Gated Recurrent Unit (GRU) [17]. Considering that GRU has comparable performance but with less parameters and more efficient computation, we employ GRU as the basic model in our sequence modeling framework. In the case of GRU, the state updates are processed according to the following operations:

$$\begin{aligned} \mathbf{r}_t^s &= \sigma(\mathbf{W}_{sr}^s \mathbf{s}_t + \mathbf{W}_{hr}^s \mathbf{h}_{t-1}^s + \mathbf{b}_r^s) \\ \mathbf{z}_t^s &= \sigma(\mathbf{W}_{sz}^s \mathbf{s}_t + \mathbf{W}_{hz}^s \mathbf{h}_{t-1}^s + \mathbf{b}_z^s) \\ \mathbf{g}_t^s &= \tanh(\mathbf{W}_{sh}^s \mathbf{s}_t + \mathbf{W}_{hh}^s (\mathbf{r}_t^s \odot \mathbf{h}_{t-1}^s) + \mathbf{b}_h^s) \\ \mathbf{h}_t^s &= \mathbf{z}_t^s \odot \mathbf{h}_{t-1}^s + (1 - \mathbf{z}_t^s) \odot \mathbf{g}_t^s \end{aligned} \quad (7.21)$$

where $\mathbf{s}_t \in \mathbf{E}$ is the embedding vector for the word s_t of the tips and the vector is also learnt from our framework. \mathbf{r}_t^s is the reset gate, \mathbf{z}_t^s is the update gate. \odot denotes element-wise multiplication.

In order to conduct the persona-aware tips generation, we combine all the persona embeddings and the sentiment information as the context information and construct the context vector. Specifically, we initialize the hidden state \mathbf{h}_0 using the persona embeddings and the sentiment information:

$$\mathbf{h}_0^s = \tanh(\mathbf{W}_{uch}^s \mathbf{z}_u^c + \mathbf{W}_{ich}^s \mathbf{z}_i^c + \mathbf{W}_{ush}^s \mathbf{z}_u^s + \mathbf{W}_{ish}^s \mathbf{z}_i^s + \mathbf{W}_{rh}^s \hat{\mathbf{r}} + \mathbf{b}_h^s) \quad (7.22)$$

where $\{\mathbf{z}_*^*\}$ are the persona embeddings. $\hat{\mathbf{r}}$ is the vectorization for the predicted

rating \hat{r} . \mathbf{W} and \mathbf{b} are the neural parameters.

After getting all the sequence hidden states based on GRU, we feed them to the final output layer to predict the word sequence in tips.

$$\hat{\mathbf{s}}_{t+1} = \varsigma(\mathbf{W}_{hs}^s \mathbf{h}_t^s + \mathbf{b}^s) \quad (7.23)$$

where $\mathbf{W}_{hs}^s \in \mathbb{R}^{d \times |\mathcal{V}|}$ and $\mathbf{b}^s \in \mathbb{R}^{|\mathcal{V}|}$. $\varsigma(\cdot)$ is the softmax function. Then the word with the largest probability is the decoding result for the step $t + 1$:

$$w_{t+1}^* = \arg \max_{w_i \in \mathcal{V}} \hat{\mathbf{s}}_{t+1}^{(w_i)} \quad (7.24)$$

At the training stage, we use negative log-likelihood (NLL) as the loss function, where I_w is the vocabulary index of the word w :

$$\mathcal{L}_{Tips} = - \sum_{w \in Tips} \log \hat{\mathbf{s}}^{(I_w)} \quad (7.25)$$

Note that \mathcal{L}_{Tips} is also used in the persona modeling component to train the aVAE models.

At the testing stage, given a trained model, we employ the beam search algorithm [56] to find the best sequence s^* having the maximum log-likelihood.

$$S^* = \arg \max_{S \in \mathcal{S}} \sum_{w \in S} \log \hat{\mathbf{s}}^{(I_w)} \quad (7.26)$$

Exploiting Persona Memory

Recall that in Section 7.2.2, we build a local personal vocabulary V_{ui} for the user u and the item i . The persona memory \mathbf{M}_{ui} is extracted from the word embedding \mathbf{E} using the word indices in V_{ui} . Inspired by [2], we exploit the idea of attention modeling to conduct the addressing and reading operations on the memory \mathbf{M}_{ui} .

We can obtain the GRU hidden state \mathbf{h}_t^s according to Equation (7.20). Then the attention weights at the time step t are calculated based on the relationship between \mathbf{h}_t^s with all the word embeddings in \mathbf{M}_{ui} . Let $a_{i,j}$ be the attention weight between \mathbf{h}_t^s and \mathbf{m}_j , which can be calculated using:

$$a_{i,j} = \frac{\exp(e_{i,j})}{\sum_{j'=1}^{|V_{ui}|} \exp(e_{i,j'})} \quad (7.27)$$

$$e_{i,j} = \mathbf{v}_a^T \tanh(\mathbf{W}_{hh}^s \mathbf{h}_t^s + \mathbf{W}_{hh}^m \mathbf{m}_j + \mathbf{b}_a)$$

where $\mathbf{W}_{hh}^s \in \mathbb{R}^{d_h \times d_h}$, $\mathbf{W}_{hh}^m \in \mathbb{R}^{d_w \times d_h}$, $\mathbf{b}_a \in \mathbb{R}^{d_h}$, and $\mathbf{v}_a \in \mathbb{R}^{d_h}$. The attention context is obtained by the weighted linear combination of all the word embeddings in \mathbf{M}_{ui} :

$$\mathbf{c}_t = \sum_{j'=1}^{|V_{ui}|} a_{t,j'} \mathbf{m}_{j'} \quad (7.28)$$

The final hidden state \mathbf{h}_t^{s2} is the output of the second decoder GRU layer, jointly considering the word \mathbf{s}_t , the previous hidden state \mathbf{h}_{t-1}^{s2} , and the attention context \mathbf{c}_t :

$$\mathbf{h}_t^{s2} = GRU_2(\mathbf{h}_{t-1}^{s2}, \mathbf{s}_t, \mathbf{c}_t) \quad (7.29)$$

Then we can use \mathbf{h}_t^{s2} as the input to Equation 7.23 to conduct the decoding operation.

Besides using attention modeling to address and read the persona information from the the persona memory \mathbf{M} , we also employ the idea of Pointer Networks [125] to copy the target words from the memory to form the tips. At the state t , we can obtain the attention weights (distribution) $\mathbf{a}_{t,:}$ on the persona memory \mathbf{M}_{ui} . Then we project $\mathbf{a}_{t,:}$ to a $|V|$ -sized vector $\widehat{\mathbf{s}}_{t+1}^p$ according to the word indices in V_{ui} . Then we design a soft gate to decide that the word s_{t+1} should be generated or be copied from the memory:

$$p_g = \sigma(\mathbf{v}_p^T (\mathbf{W}_{hp}^s \mathbf{h}_t^{s2} + \mathbf{W}_{sp}^s \mathbf{s}_t + \mathbf{W}_{cp}^s \mathbf{c}_t + \mathbf{b}_p)) \quad (7.30)$$

where $\mathbf{v}_p \in \mathbb{R}^{d_h}$ and $p_g \in (0, 1)$. Then we merge the copy signal $\widehat{\mathbf{s}}_{t+1}^p$ and the original output $\widehat{\mathbf{s}}_{t+1}$ according to the gate p_g :

$$\widehat{\mathbf{s}}'_{t+1} = p_g \times \widehat{\mathbf{s}}_{t+1} + (1 - p_g) \times \widehat{\mathbf{s}}_{t+1}^p \quad (7.31)$$

Then the tips sampling process can be conducted on $\widehat{\mathbf{s}}'_{t+1}$.

Tips Quality Discriminator

Some previous works [68, 143] show that adversarial training strategy is beneficial to the text generation problem. To further improve the performance, we also employ this training strategy in our framework.

The tips discriminator D_{Tips} is a multilayer perceptron with the persona embeddings, the rating information, and the tips sequence as the input. The input tips sequence can be the ground truth S or the tips \widehat{S} generated by the system. We propose a Bidirectional-GRU model to conduct the representation learning for S and \widehat{S} :

$$\mathbf{h}^S = \overrightarrow{\mathbf{h}}^S || \overleftarrow{\mathbf{h}}^S \quad (7.32)$$

Then we combine all the information according to:

$$\mathbf{h}^q = \tanh(\mathbf{W}_{sh}^q \mathbf{h}^S + \mathbf{W}_{uch}^q \mathbf{z}_{\mathbf{u}}^c + \mathbf{W}_{ich}^q \mathbf{z}_{\mathbf{i}}^c + \mathbf{W}_{ush}^q \mathbf{z}_{\mathbf{u}}^s + \mathbf{W}_{ish}^q \mathbf{z}_{\mathbf{i}}^s + \mathbf{W}_{rh}^q \widehat{\mathbf{r}} + \mathbf{b}_h^q)$$

Finally, we add a softmax output layer to let the model output a binary category variable:

$$\mathbf{y}^q = \varsigma(\mathbf{W}_{hy}^q \mathbf{h}^q + \mathbf{b}_y^q) \quad (7.33)$$

We treat the ground truth S as the positive instance and the sampled sequence \widehat{S} as the negative instance. So we directly let the first dimension of \mathbf{y}^q represent the positive label. We define the value function as $V(S) = \mathbf{y}_{[0]}^q$. We utilize the

Table 7.1: Overview of the datasets.

| | Electr | Movies | Home | Clothing | Yelp |
|-----------------|---------------|---------------|-------------|-----------------|-------------|
| <i>users</i> | 191,522 | 123,340 | 66,212 | 39,085 | 115,781 |
| <i>items</i> | 62,333 | 49,823 | 27,991 | 22,794 | 60,224 |
| <i>reviews</i> | 1,684,779 | 1,693,441 | 550,461 | 277,521 | 1,393,257 |
| $ \mathcal{V} $ | 37,999 | 82,805 | 23,950 | 16,297 | 82,805 |

REINFORCE [133] method to integrate the tips quality signal $V(S)$ into the tips generation framework to conduct the parameter learning.

7.3 Experimental Setup

7.3.1 Datasets

In our experiments, we use five datasets from different domains to evaluate our framework. The ratings of these datasets are integers in the range of $[1, 5]$. There are four datasets from Amazon 5-core²: **Electronics**, **Movies & TV**, **Clothing, Shoes and Jewelry**, and **Home and Kitchen**. We regard the field “summary” as tips, and the number of tips texts is the same with the number of reviews. Another dataset is from **Yelp Challenge 2016**³. It is also a large-scale dataset consisting of restaurant reviews and tips. We filter out the words with low term frequency in the tips and review texts, and build a vocabulary \mathcal{V} for each dataset. We show the statistics of our datasets in Table 7.1.

7.3.2 Evaluation Metrics

For the evaluation of abstractive tips generation, the ground truth s_h is the tips written by the user. We use *ROUGE* [79] as our evaluation metric with standard

²<http://jmcauley.ucsd.edu/data/amazon>

³https://www.yelp.com/dataset_challenge

options⁴. It is a classical evaluation metric in the field of text summarization [79]. We use Recall, Precision, and F-measure of ROUGE-1 (R-1), ROUGE-2 (R-2), ROUGE-L (R-L), and ROUGE-SU4 (R-SU4) to evaluate the quality of the generated tips.

For the evaluation of rating prediction, we employ two metrics: Mean Absolute Error (*MAE*) and Root Mean Square Error (*RMSE*). Both of them are widely used for rating prediction in recommender systems. Given a predicted rating $\hat{r}_{u,i}$ and a ground-truth rating $r_{u,i}$ from the user u for the item i , the RMSE is calculated as:

$$RMSE = \sqrt{\frac{1}{N} \sum_{u,i} (r_{u,i} - \hat{r}_{u,i})^2} \quad (7.34)$$

where N indicates the number of ratings between users and items. Similarly, MAE is calculated as follows:

$$MAE = \frac{1}{N} \sum_{u,i} |r_{u,i} - \hat{r}_{u,i}| \quad (7.35)$$

7.3.3 Comparative Methods

To evaluate the performance of abstractive tips generation, we compare our framework PATG with the following baseline and state-of-the-art methods:

- **NRT** [75]: It is a previous framework we proposed for rating prediction and abstractive tips generation achieving state-of-the-art performance. Latent factors for users and items are learnt during the training procedure, and are used as the context information for tips generation. NRT does not consider the persona information.
- **LexRank** [27] is a classical method in the field of text summarization. We add a preprocessing procedure to prepare the input texts for LexRank, which consists of the following steps: (1) Retrieval: For the user u , we first retrieve

⁴ROUGE-1.5.5.pl -n 4 -w 1.2 -m -2 4 -u -c 95 -r 1000 -f A -p 0.5 -t 0

all her reviews \mathcal{C}_u from the training set. For the item i , we use the same method to get \mathcal{C}_i . (2) Filtering: Assuming that the ground truth rating for u and i is $r_{u,i}$, then we remove all the reviews from \mathcal{C}_u and \mathcal{C}_i whose ratings are not equal to $r_{u,i}$. The reviews whose words only appear in one set are also removed. (3) Tips extraction: We first merge \mathcal{C}_u and \mathcal{C}_i to get $\mathcal{C}_{u,i}$, then the problem can be regarded as a multi-document summarization problem. LexRank can extract a sentence from $\mathcal{C}_{u,i}$ as the final tips. Note that we give an advantage of this method since the ground truth ratings are used.

- **CTR_t**: Collaborative **T**opic **R**egression (CTR) [129] contains a topic model component and it can generate topics for items. So the topic related variables are employed to extract tips: (1) We first get the latent factor θ_i for item i , and draw the topic z with the largest probability from θ_i . Then from ϕ_z , which is a multinomial distribution of z on \mathcal{V} , we select the top-50 words with the largest probability. (2) The most similar sentence from $\mathcal{C}_{u,i}$ is extracted as the tips. This baseline is named **CTR_t**.
- **HFT_t**: **H**idden **F**actors and **H**idden **T**opics [87] utilizes a topic modeling technique to model the review texts for rating prediction. Then we can design a tips extraction method **HFT_t** using the similar technique in **CTR_t**.

To evaluate the performance of rating prediction, we compare our model with the following methods:

- **HFT**: **H**idden **F**actors and **T**opics [87]. It utilizes a topic modeling technique to model the review texts and achieves significant improvements compared with other strong topic modeling based methods.
- **CTR**: Collaborative **T**opic **R**egression [129]. It is a popular method for scientific articles recommendation by solving a one-class collaborative filtering problem. Note that CTR uses both ratings and item specifications.

Table 7.2: Baselines and methods used for comparison.

| Acronym | Gloss | Reference |
|--------------------------|--------------------------------------|-------------|
| PATG | Persona-aware tips generation | Section 7.2 |
| <i>Rating prediction</i> | | |
| HFT | Hidden factors and topics model | [87] |
| CTR | Collaborative topic regression model | [129] |
| NMF | Non-negative matrix factorization | [61] |
| SVD++ | Factorization meets the neighborhood | [57] |
| NRT | Neural Rating and Tips Generation | [75] |
| <i>Tips generation</i> | | |
| LexRank | Pagerank for summarization | [27] |
| CTR _t | CTR for tips topic extraction | [129] |
| HFT _t | HFT for tips topic extraction | [87] |
| NRT | Neural Rating and Tips Generation | [75] |

- **NMF**: **N**on-negative **M**atrix **F**actorization [61]. The non-negativity constraints are integrated in the typical matrix factorization and make the representation purely additive. NMF is a popular and strong baseline for CF-based recommendation. It only uses the rating matrix as the input.
- **SVD++**: It extends **S**ingular **V**alue **D**ecomposition by considering implicit feedback information for latent factor modeling [57].

Finally, we list all the methods and baselines in Table 7.2.

7.3.4 Experimental Settings

Each dataset is divided into three subsets: 80%, 10%, and 10%, for training, validation, and testing, respectively. All the parameters of our model are tuned with the validation set. After the tuning process, the number of latent factors k is set to 10 for NMF and SVD++. The number of topics K is set to 50 for the methods using topic models. The number of dimension for the persona embeddings is set

to 100. The dimension of the hidden size is 400. In our framework, the number of layers for the rating regression model is 2, and for the tips generation model is 1. We set the beam size $\beta = 5$, and the maximum length $\eta = 20$. All the neural matrix parameters in hidden layers and RNN layers are initialized from a uniform distribution between $[-0.1, 0.1]$. We also regard the word embedding \mathbf{E} used in the tips generation component as a neural parameter. Adadelta [146] is used for gradient based optimization.

7.4 Results and Discussions

7.4.1 Abstractive Tips Generation

The evaluation results of tips generation of our model and the comparative methods are given in Table 7.3 and Table 7.4. In order to capture more details, we report Recall, Precision, and F-measure (in percentage) of ROUGE-1, ROUGE-2, ROUGE-L, and ROUGE-SU4. Our model achieves the best performance in most of the metrics among all the five datasets. Moreover, from the results we can see that recall is not as good as precision. There are several reasons: (1) The ground truth tips used in the training set are very short, only about 10-word length on average. Naturally, the model trained using this dataset cannot generate long sentence. (2) The mechanism of typical beam search algorithm makes the model favor short sentences. (3) The comparison models are extraction-based approaches and these models favor to extract long sentence, although we add a length (i.e., 20 words) restriction on them.

NRT does not consider persona information when generating tips. It only utilizes the learnt latent factors for users and items as the context information. Compared with NRT, our proposed framework PATG obtains better performance on all the metrics, which demonstrates that the consideration of persona information can indeed improve the tips generation performance. We also conduct statistical

Table 7.3: ROUGE (R-1 and R-2) evaluation on the five datasets from different domains.

| Dataset | Method | ROUGE-1 | | | ROUGE-2 | | |
|-------------|------------------|--------------|--------------|---------------|-------------|-------------|--------------|
| | | R | P | F1 | R | P | F1 |
| Electronics | LexRank | 10.97 | 12.93 | 11.58 | 0.95 | 1.05 | 0.97 |
| | HFT _t | 12.86 | 12.22 | 12.35 | 1.10 | 1.00 | 1.03 |
| | CTR _t | 12.69 | 11.72 | 12.02 | 1.13 | 1.05 | 1.07 |
| | NRT | 12.79 | 17.55 | 13.85 | 1.86 | 2.77 | 2.08 |
| | PATG | 13.00 | 19.26 | 14.52* | 2.29 | 3.12 | 2.44* |
| Movies&TV | LexRank | 11.10 | 13.50 | 11.89 | 1.06 | 1.29 | 1.12 |
| | HFT _t | 11.64 | 10.26 | 11.33 | 1.78 | 1.36 | 1.46 |
| | CTR _t | 11.37 | 10.33 | 10.68 | 1.43 | 1.31 | 1.34 |
| | NRT | 12.12 | 20.06 | 14.17 | 2.29 | 3.53 | 2.55 |
| | PATG | 12.46 | 21.22 | 14.63* | 2.38 | 3.88 | 2.67* |
| Home | LexRank | 12.91 | 15.47 | 13.77 | 1.73 | 2.06 | 1.82 |
| | HFT _t | 13.32 | 12.72 | 12.80 | 1.33 | 1.23 | 1.25 |
| | CTR _t | 14.30 | 13.21 | 13.55 | 1.73 | 1.50 | 1.58 |
| | NRT | 11.51 | 19.91 | 13.64 | 1.95 | 3.47 | 2.30 |
| | PATG | 12.21 | 21.46 | 14.61* | 2.32 | 4.32 | 2.78* |
| Clothing | LexRank | 13.31 | 12.73 | 12.85 | 1.06 | 1.02 | 1.02 |
| | HFT _t | 13.31 | 12.73 | 12.85 | 1.06 | 1.02 | 1.02 |
| | CTR _t | 13.79 | 13.82 | 13.37 | 1.26 | 1.23 | 1.22 |
| | NRT | 13.52 | 18.91 | 14.75 | 2.11 | 2.95 | 2.31 |
| | PATG | 14.45 | 21.49 | 16.14* | 2.49 | 3.77 | 2.79* |
| Yelp | LexRank | 9.19 | 12.09 | 10.28 | 1.07 | 1.33 | 1.15 |
| | HFT _t | 10.47 | 10.21 | 10.26 | 0.91 | 0.87 | 0.88 |
| | CTR _t | 10.68 | 10.51 | 10.51 | 0.98 | 0.94 | 0.96 |
| | NRT | 10.98 | 17.42 | 12.71 | 1.82 | 3.03 | 2.13 |
| | PATG | 12.05 | 19.15 | 14.02* | 2.15 | 3.44 | 2.47* |

Table 7.4: ROUGE (R-L and R-SU4) evaluation on the five datasets from different domains.

| Dataset | Method | ROUGE-L | | | ROUGE-SU4 | | |
|-------------|------------------|--------------|--------------|---------------|-------------|-------------|--------------|
| | | R | P | F1 | R | P | F1 |
| Electronics | LexRank | 9.96 | 11.70 | 10.50 | 3.08 | 3.91 | 3.22 |
| | HFT _t | 11.65 | 11.09 | 11.19 | 3.43 | 3.10 | 3.14 |
| | CTR _t | 11.65 | 10.74 | 11.02 | 3.45 | 3.06 | 3.14 |
| | NRT | 11.80 | 15.99 | 12.70 | 4.18 | 6.42 | 4.45 |
| | PATG | 11.91 | 17.42 | 13.24* | 4.50 | 7.44 | 4.89* |
| Movies&TV | LexRank | 10.02 | 12.12 | 10.70 | 3.25 | 4.33 | 3.46 |
| | HFT _t | 11.42 | 8.72 | 9.67 | 4.63 | 3.00 | 3.28 |
| | CTR _t | 10.40 | 9.44 | 9.76 | 3.17 | 2.73 | 2.84 |
| | NRT | 11.13 | 18.25 | 12.98 | 4.09 | 8.15 | 4.79 |
| | PATG | 11.51 | 19.25 | 14.73* | 6.04 | 8.76 | 6.33* |
| Home | LexRank | 11.72 | 13.97 | 12.46 | 3.93 | 5.02 | 4.15 |
| | HFT _t | 12.25 | 11.73 | 11.79 | 3.63 | 3.33 | 3.34 |
| | CTR _t | 13.14 | 12.11 | 12.43 | 4.18 | 3.66 | 3.78 |
| | NRT | 10.64 | 18.23 | 12.57 | 3.77 | 8.24 | 4.51 |
| | PATG | 11.32 | 19.65 | 13.48* | 4.03 | 8.71 | 4.82* |
| Clothing | LexRank | 11.97 | 11.43 | 11.54 | 3.47 | 3.24 | 3.26 |
| | HFT _t | 11.97 | 11.43 | 11.54 | 3.47 | 3.24 | 3.26 |
| | CTR _t | 12.54 | 12.14 | 12.16 | 3.70 | 3.52 | 3.49 |
| | NRT | 12.36 | 17.04 | 13.39 | 4.58 | 7.04 | 4.86 |
| | PATG | 13.09 | 19.24 | 14.55* | 4.93 | 8.39 | 5.39* |
| Yelp | LexRank | 8.45 | 11.13 | 9.45 | 2.65 | 3.90 | 3.01 |
| | HFT _t | 9.56 | 9.31 | 9.35 | 2.70 | 2.57 | 2.59 |
| | CTR _t | 9.70 | 9.53 | 9.54 | 2.77 | 2.68 | 2.68 |
| | NRT | 9.96 | 15.76 | 11.51 | 3.48 | 6.48 | 4.05 |
| | PATG | 10.94 | 17.21 | 12.66* | 3.96 | 7.15 | 4.57* |

* denotes that PATG achieves better performance than NRT with statistical significance test with $p < 0.05$.

significance test comparing PATG and NRT and the results indicate that the improvements are significant with $p < 0.05$.

In order to demonstrate the quality of the generated tips, we selected some real cases generated by our system PATG from different domains. The results are listed in Table 7.5. Although our model generates tips in an abstractive way, tips' linguistic quality is quite good. The persona properties of the generated tips match well with the ground truth. For example, in the first case, the generated tips is "This is a great hat for the price.", and the ground truth is "Thanks nice quality excellent price great deal.". Both of the sentences contain the terms "great" and "price". In the third case, the generated tips and the ground truth have a large overlapping with the terms "replace my old", and "processor". Interestingly, sometimes the framework can select some synonyms when conducting tips generation. For instance, the generated tips of the fourth case contains terms "bought" and "for my husband". The ground truth contains "purchased" and "for a male". Moreover, we also choose some generated tips with negative sentiment to conduct the sentiment correlation analysis. Take the generated tips "Please do not buy this coffee maker." as an example (the last case in Table 7.5), our model predicts a rating of 2.01, which clearly shows a consistent sentiment. The ground truth tips of this example is " They are still overpriced and all but worthless.", which also conveys a negative sentiment. The generated tips "The bottom line of the thin man." and the ground truth "Pretty dark story in book or movie form." are just describing some facts, with a neutral rating 3. Sometimes the overlapping between the generated tips and the ground truth is small, but they still convey similar information.

7.4.2 Ablation Experimental Results

In order to demonstrate the performance of each component of our framework, we conduct the ablation experiments on the dataset Home. The results are shown in

Table 7.5: Examples of the predicted ratings and the generated tips. The first line of each group shows the generated rating and tips. The second line shows the ground truth.

| Rating | Tips |
|-------------|---|
| 5.10 | <i>This is a great hat for the price.</i> |
| 5 | Thanks nice quality excellent price great deal. |
| 5.08 | <i>This is a great pitcher.</i> |
| 5 | Beautiful pitcher makes a great vase. |
| 5.17 | <i>I bought this food processor to replace my old one.</i> |
| 4 | I got this about a month ago to replace my old food processor. |
| 4.99 | <i>These shoes are so comfortable and I bought these for my husband.</i> |
| 5 | Comfortable good looking shoes purchased for a male that walks a lot. |
| 4.81 | <i>This is a great movie.</i> |
| 5 | Amazing love great movie and all teen should see it. |
| 2.57 | <i>The bottom line of the thin man.</i> |
| 3 | Pretty dark story in book or movie form. |
| 2.01 | <i>Please do not buy this coffee maker.</i> |
| 1 | They are still overpriced and all but worthless. |

Table 7.6, where “A” denotes the aVAE model, “M” represents the persona memory and the Pointer Networks, and “D” represents the tips quality discriminator D_{Tips} . For example, the method “PATG w/o A, M, D” means that A, M, and D are all removed and we only use the standard VAE for persona modeling. It is obvious that persona modeling based on aVAE can improve the tips generation performance. The persona memory and Pointer Networks are very helpful and contribute to the effectiveness of our framework.

7.4.3 Rating Prediction

The rating prediction results of our framework and comparative models on all datasets are given in Table 7.7 and Table 7.8. It shows that our model consis-

Table 7.6: Ablation experiments on the dataset Home. R-* represents the F1-Measure of ROUGE-*

| System | ROUGE-1 | ROUGE-2 | ROUGE-L | ROUGE-SU4 |
|------------------|---------|---------|---------|-----------|
| PATG w/o A, M, D | 13.76 | 2.27 | 12.64 | 4.45 |
| PATG w/o M, D | 13.99 | 2.61 | 12.95 | 4.71 |
| PATG w/o D | 14.32 | 2.72 | 13.30 | 4.81 |
| PATG | 14.51 | 2.72 | 13.48 | 4.81 |

Table 7.7: MAE and RMSE values for rating prediction on datasets Electronics and Movies.

| | Electronics | | Movies | |
|-------------|---------------|---------------|---------------|---------------|
| | MAE | RMSE | MAE | RMSE |
| NMF | 0.869 | 1.266 | 0.809 | 1.155 |
| SVD++ | 0.841 | 1.226 | 0.778 | 1.122 |
| CTR | 0.903 | 1.154 | 0.863 | 1.116 |
| HFT | 0.813 | 1.117 | 0.769 | 1.041 |
| NRT | 0.823 | 1.108 | 0.751 | 1.038 |
| PATG | 0.747* | 1.016* | 0.740* | 1.015* |

* denotes that PATG achieves better performance than NRT [75] with statistical significance test with $\alpha = 0.01$.

tently outperforms all comparative methods under both MAE and RMSE metrics on all datasets, which demonstrates that the persona embeddings can also improve the performance of rating prediction. Statistical significance of differences between the performance of PATG and the recent method NRT is tested using a two-tailed paired t-test. The result shows that PATG is significantly better than NRT.

7.4.4 Further Investigations

Recall that in addition to the persona embeddings as context information, rating information is also incorporated to control the sentiment of the generated tips. In order to show this additional ability of our framework, we design an experiment on

Table 7.8: MAE and RMSE values for rating prediction on datasets Yelp, Clothing, and Home.

| | Yelp | | Clothing | | Home | |
|-------------|---------------|---------------|---------------|---------------|---------------|---------------|
| | MAE | RMSE | MAE | RMSE | MAE | RMSE |
| NMF | 0.961 | 1.136 | 0.887 | 1.257 | 0.830 | 1.220 |
| SVD++ | 1.957 | 1.299 | 0.829 | 1.169 | 0.786 | 1.164 |
| CTR | 1.051 | 1.285 | 0.847 | 1.094 | 0.826 | 1.086 |
| HFT | 0.940 | 1.191 | 0.805 | 1.080 | 0.773 | 1.058 |
| NRT | 0.935 | 1.187 | 0.828 | 1.102 | 0.779 | 1.058 |
| PATG | 0.866* | 1.134* | 0.714* | 0.987* | 0.694* | 0.997* |

* denotes that PATG achieves better performance than NRT [75] with statistical significance test with $\alpha = 0.01$.

Table 7.9: Rating controlled tips generation.

| Rating | Tips | Evaluation |
|----------|----------------------------|------------|
| 5 | This is a great product. | ✓ |
| 4 | This is a good product. | ✓ |
| 3 | Not as good as my old one. | ✓ |
| 2 | Not as good as I expected. | ✓ |
| 1 | This is a good product. | × |

the domain “Home” for rating controlled tips generation. Specifically, during the prediction, we manually set the rating from 1 to 5 as sentiment context to control the generation. The results are shown in Table 7.9. It indicates that our framework can generate rating controlled tips for most rating levels (2 to 5). Due to the sparsity of rating-1 samples, the model just outputs the prior “This is a good product.” when $r = 1$.

7.5 Summary

In this Chapter, we propose a framework PATG to address the problem of persona-aware tips generation. A framework based on adversarial variational auto-encoders

(aVAE) is exploited for persona modeling from the historical tips and reviews. We also design an external persona memory for directly storing the persona related words for the current user and item. Pointer Networks is used to address and read the persona related information from the memory when generating tips. The distilled persona embeddings are used as latent factors and are fed into the rating prediction component for detecting sentiment. Then the persona embeddings and the sentiment information are incorporated into a recurrent neural networks (RNN) based tips generation component to control the tips generation. Experimental results show that our framework achieves better performance than the state-of-the-art models on abstractive tips generation.

Chapter 8

Conclusion

In this thesis, we introduce several methods for developing better text summarization and generation systems based on neural networks. We also discuss the problems of the existing methods, as well as the challenges, and the strategies to tackle them.

In Chapter 3, we propose a new framework for abstractive text summarization based on a sequence-to-sequence oriented encoder-decoder model equipped with a deep recurrent generative decoder (DRGN). Latent structure information implied in the target summaries is learned based on a recurrent latent random model for improving the summarization quality. Neural variational inference is employed to address the intractable posterior inference for the recurrent latent variables. Abstractive summaries are generated based on both the generative latent variables and the discriminative deterministic states. Extensive experiments on some benchmark datasets in different languages show that DRGN achieves improvements over the state-of-the-art methods.

In Chapter 4, we propose a cascaded attention based unsupervised model to estimate the salience information from the text for compressive multi-document summarization. The attention weights are learned automatically by an unsupervised data reconstruction framework which can capture the sentence salience. By adding

sparsity constraints on the number of output vectors, we can generate condensed information which can be treated as word salience. Fine-grained and coarse-grained sentence compression strategies are incorporated to produce compressive summaries. Experiments on some benchmark data sets demonstrate the effectiveness of our framework.

In Chapter 5, we propose a VAEs based unsupervised sentence salience framework for multi-document summarization. For latent semantic modeling, VAEs is employed to describe the observed sentences and the corresponding latent semantic representations. For salience estimation, we propose an unsupervised data reconstruction framework, which jointly considers the reconstruction for latent semantic space and observed term vector space. Thereafter, the VAEs-based latent semantic model is integrated into the sentence salience estimation component in a unified fashion, and the whole framework can be trained jointly by back-propagation via multi-task learning. Experimental results on the benchmark datasets DUC and TAC show that our framework achieves better performance.

In Chapter 6, we investigate the problem of reader-aware multi-document summarization (RA-MDS) and introduce a new dataset for this problem. To tackle RA-MDS, we extend the VAEs based MDS framework by jointly considering news documents and reader comments. To conduct evaluation for summarization performance, we prepare a new dataset. We describe the methods for data collection, aspect annotation, and summary writing as well as scrutinizing by experts. Experimental results show that reader comments can improve the summarization performance, which also demonstrates the usefulness of the proposed dataset. The annotated dataset for RA-MDS is available online¹.

In Chapter 7, we investigate the task of abstractive tips generation for recommendation systems. Different from existing methods, our framework considers persona

¹<http://www.se.cuhk.edu.hk/~textmine/dataset/ra-mds/>

information when conducting tips text generation. In order to exploit the persona information, we propose a framework based on adversarial VAEs for persona modeling from the historical tips and reviews for users and items. The latent variables from aVAE are regarded as persona embeddings. Besides representing persona using the latent embeddings, we design a persona memory for directly storing the persona related words for the current user and item. Pointer Networks is used to retrieve persona related information from the memory when generating tips. The distilled persona embeddings are used as latent factors for users and items and are fed into the rating prediction component for detecting sentiment. Finally, the persona embeddings and the sentiment information are incorporated into the recurrent neural networks (RNN) based tips generation component. Experimental results show that our framework achieves better performance than the state-of-the-art models on abstractive tips generation.

Bibliography

- [1] Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D Trippe, Juan B Gutierrez, and Krys Kochut. Text summarization techniques: A brief survey. *arXiv preprint arXiv:1707.02268*, 2017.
- [2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*, 2015.
- [3] Regina Barzilay and Kathleen R McKeown. Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–328, 2005.
- [4] Phyllis B Baxendale. Machine-made index for technical literature—an experiment. *IBM Journal of Research and Development*, 2(4):354–361, 1958.
- [5] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *Neural Networks, IEEE Transactions on*, 5(2):157–166, 1994.
- [6] Lidong Bing, Piji Li, Yi Liao, Wai Lam, Weiwei Guo, and Rebecca Passonneau. Abstractive multi-document summarization via phrase selection and merging. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural*

- Language Processing (Volume 1: Long Papers)*, volume 1, pages 1587–1597, 2015.
- [7] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.
- [8] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *arXiv preprint arXiv:1601.00670*, 2016.
- [9] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, 2016.
- [10] Ziqiang Cao, Wenjie Li, Sujian Li, Furu Wei, and Yanran Li. Attsum: Joint learning of focusing and summarization with neural attention. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 547–556, 2016.
- [11] Dallas Card, Chenhao Tan, and Noah A Smith. A neural framework for generalized topic models. *arXiv preprint arXiv:1705.09296*, 2017.
- [12] Asli Celikyilmaz and Dilek Hakkani-Tur. A hybrid hierarchical model for multi-document summarization. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 815–824. Association for Computational Linguistics, 2010.
- [13] Asli Celikyilmaz and Dilek Hakkani-Tür. Concept-based classification for multi-document summarization. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 5540–5543. IEEE, 2011.

-
- [14] Tong Che, Yanran Li, Ruixiang Zhang, R Devon Hjelm, Wenjie Li, Yangqiu Song, and Yoshua Bengio. Maximum-likelihood augmented discrete generative adversarial networks. *arXiv preprint arXiv:1702.07983*, 2017.
- [15] Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, and Hui Jiang. Distraction-based neural networks for modeling documents. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 2754–2760. AAAI Press, 2016.
- [16] Jianpeng Cheng and Mirella Lapata. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 484–494, 2016.
- [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. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, 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] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.
- [20] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. A recurrent latent variable model for sequential

- data. In *Advances in neural information processing systems*, pages 2980–2988, 2015.
- [21] George B Dantzig and Mukund N Thapa. *Linear programming 1: introduction*. Springer Science & Business Media, 2006.
- [22] Dipanjan Das and André FT Martins. A survey on automatic text summarization. *Literature Survey for the Language and Statistics II course at CMU*, 4:192–195, 2007.
- [23] Li Dong, Shaohan Huang, Furu Wei, Mirella Lapata, Ming Zhou, and Ke Xu. Learning to generate product reviews from attributes. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, volume 1, pages 623–632, 2017.
- [24] Harris Drucker, Christopher JC Burges, Linda Kaufman, Alex J Smola, and Vladimir Vapnik. Support vector regression machines. In *Advances in neural information processing systems*, pages 155–161, 1997.
- [25] Harold P Edmundson. New methods in automatic extracting. *Journal of the ACM (JACM)*, 16(2):264–285, 1969.
- [26] Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.
- [27] 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.
- [28] Zhihao Fan, Zhongyu Wei, Piji Li, Yanyan Lan³¹, and Xuanjing Huang. A question type driven framework to diversify visual question generation. In *The 27th International Joint Conference on Artificial Intelligence*, 2018.

-
- [29] William Fedus, Ian Goodfellow, and Andrew M Dai. Maskgan: Better text generation via filling in the __. *arXiv preprint arXiv:1801.07736*, 2018.
- [30] Katja Filippova. Multi-sentence compression: Finding shortest paths in word graphs. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 322–330. Association for Computational Linguistics, 2010.
- [31] Katja Filippova and Michael Strube. Sentence fusion via dependency graph compression. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 177–185. Association for Computational Linguistics, 2008.
- [32] Ken-ichi Funahashi and Yuichi Nakamura. Approximation of dynamical systems by continuous time recurrent neural networks. *Neural networks*, 6(6): 801–806, 1993.
- [33] Mahak Gambhir and Vishal Gupta. Recent automatic text summarization techniques: a survey. *Artificial Intelligence Review*, 47(1):1–66, 2017.
- [34] Dan Gillick and Benoit Favre. A scalable global model for summarization. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*, pages 10–18. Association for Computational Linguistics, 2009.
- [35] Jade Goldstein, Vibhu Mittal, Jaime Carbonell, and Mark Kantrowitz. Multi-document summarization by sentence extraction. In *Proceedings of the 2000 NAACL-ANLP Workshop on Automatic summarization*, pages 40–48. Association for Computational Linguistics, 2000.
- [36] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative ad-

- versarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.
- [37] Anirudh Goyal Alias Parth Goyal, Alessandro Sordoni, Marc-Alexandre Côté, Nan Ke, and Yoshua Bengio. Z-forcing: Training stochastic recurrent networks. In *Advances in neural information processing systems*, pages 6716–6726, 2017.
- [38] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*, pages 6645–6649. IEEE, 2013.
- [39] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. In *International Conference on Machine Learning*, pages 1462–1471, 2015.
- [40] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1631–1640, 2016.
- [41] Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. Long text generation via adversarial training with leaked information. *arXiv preprint arXiv:1709.08624*, 2017.
- [42] Zhanying He, Chun Chen, Jiajun Bu, Can Wang, Lijun Zhang, Deng Cai, and Xiaofei He. Document summarization based on data reconstruction. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, pages 620–626. AAAI Press, 2012.

-
- [43] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [44] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. Convolutional neural network architectures for matching natural language sentences. In *Advances in neural information processing systems*, pages 2042–2050, 2014.
- [45] Baotian Hu, Qingcai Chen, and Fangze Zhu. Lcsts: A large scale chinese short text summarization dataset. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1962–1972, 2015.
- [46] Meishan Hu, Aixin Sun, and Ee-Peng Lim. Comments-oriented document summarization: understanding documents with readers’ feedback. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 291–298. ACM, 2008.
- [47] 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, 2017.
- [48] Hongyan Jing and Kathleen R McKeown. Cut and paste based text summarization. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 178–185. Association for Computational Linguistics, 2000.
- [49] Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.
- [50] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015.

-
- [51] Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. Controlling output length in neural encoder-decoders. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1328–1338, 2016.
- [52] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1746–1751, 2014.
- [53] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [54] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- [55] Dan Klein and Christopher D Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st annual meeting of the association for computational linguistics*, pages 423–430, 2003.
- [56] Philipp Koehn. Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *Conference of the Association for Machine Translation in the Americas*, pages 115–124. Springer, 2004.
- [57] Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434. ACM, 2008.
- [58] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. In *International Conference on Machine Learning*, pages 1558–1566, 2016.

- [59] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196, 2014.
- [60] Rémi Lebret, David Grangier, and Michael Auli. Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213, 2016.
- [61] Daniel D Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.
- [62] Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational Linguistics*, 39(4):885–916, 2013.
- [63] Chen Li, Fei Liu, Fuliang Weng, and Yang Liu. Document summarization via guided sentence compression. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 490–500, 2013.
- [64] Chongxuan Li, Jun Zhu, and Bo Zhang. Learning to generate with memory. In *International Conference on Machine Learning*, pages 1177–1186, 2016.
- [65] Huiying Li, Yue Hu, Zeyuan Li, Xiaojun Wan, and Jianguo Xiao. Pkutm participation in tac2011. *Proceeding RTE*, 7, 2011.
- [66] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Asso-*

- ciation for Computational Linguistics: Human Language Technologies*, pages 110–119, 2016.
- [67] Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. A persona-based neural conversation model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 994–1003, 2016.
- [68] Jiwei Li, Will Monroe, Tianlin Shi, Sébastien Jean, Alan Ritter, and Dan Jurafsky. Adversarial learning for neural dialogue generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2157–2169, 2017.
- [69] Piji Li, Jun Ma, and Shuai Gao. Learning to summarize web image and text mutually. In *Proceedings of the 2nd ACM International Conference on Multimedia Retrieval*, pages 28–35. ACM, 2012.
- [70] Piji Li, Lidong Bing, Wai Lam, Hang Li, and Yi Liao. Reader-aware multi-document summarization via sparse coding. In *The 24th International Joint Conference on Artificial Intelligence*, pages 1270–1276, 2015.
- [71] Piji Li, Lidong Bing, and Wai Lam. Reader-aware multi-document summarization: An enhanced model and the first dataset. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 91–99, 2017.
- [72] Piji Li, Wai Lam, Lidong Bing, Weiwei Guo, and Hang Li. Cascaded attention based unsupervised information distillation for compressive summarization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2081–2090, 2017.
- [73] Piji Li, Wai Lam, Lidong Bing, and Zihao Wang. Deep recurrent generative

- decoder for abstractive text summarization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2091–2100, 2017.
- [74] Piji Li, Zihao Wang, Wai Lam, Zhaochun Ren, and Lidong Bing. Saliency estimation via variational auto-encoders for multi-document summarization. In *The Thirty-First AAAI Conference on Artificial Intelligence*, pages 3497–3503, 2017.
- [75] Piji Li, Zihao Wang, Zhaochun Ren, Lidong Bing, and Wai Lam. Neural rating regression with abstractive tips generation for recommendation. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 345–354. ACM, 2017.
- [76] Piji Li, Lidong Bing, and Wai Lam. Actor-critic based training framework for abstractive summarization. *arXiv preprint arXiv:1803.11070*, 2018.
- [77] Yi Liao, Lidong Bing, Piji Li, Shuming Shi, Wai Lam, and Tong Zhang. Incorporating pseudo-parallel data for quantifiable sequence editing. *arXiv preprint arXiv:1804.07007*, 2018.
- [78] Chin-Yew Lin. Improving summarization performance by sentence compression: a pilot study. In *Proceedings of the sixth international workshop on Information retrieval with Asian languages- Volume 11*, pages 1–8. Association for Computational Linguistics, 2003.
- [79] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*, volume 8, 2004.
- [80] Kevin Lin, Dianqi Li, Xiaodong He, Zhengyou Zhang, and Ming-Ting Sun.

- Adversarial ranking for language generation. In *Advances in Neural Information Processing Systems*, pages 3155–3165, 2017.
- [81] Marina Litvak and Mark Last. Graph-based keyword extraction for single-document summarization. In *Proceedings of the workshop on Multi-source Multilingual Information Extraction and Summarization*, pages 17–24. Association for Computational Linguistics, 2008.
- [82] He Liu, Hongliang Yu, and Zhi-Hong Deng. Multi-document summarization based on two-level sparse representation model. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 196–202, 2015.
- [83] Konstantin Lopyrev. Generating news headlines with recurrent neural networks. *arXiv preprint arXiv:1512.01712*, 2015.
- [84] Hans Peter Luhn. The automatic creation of literature abstracts. *IBM Journal of research and development*, 2(2):159–165, 1958.
- [85] Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, 2015.
- [86] Inderjeet Mani and Eric Bloedorn. Multi-document summarization by graph search and matching. In *Proceedings of the fourteenth national conference on artificial intelligence and ninth conference on Innovative applications of artificial intelligence*, pages 622–628. AAAI Press, 1997.
- [87] Julian McAuley and Jure Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 165–172. ACM, 2013.

-
- [88] Ryan McDonald. A study of global inference algorithms in multi-document summarization. In *European Conference on Information Retrieval*, pages 557–564. Springer, 2007.
- [89] Kathleen McKeown and Dragomir R Radev. Generating summaries of multiple news articles. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 74–82. ACM, 1995.
- [90] Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. Deep keyphrase generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 582–592, 2017.
- [91] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. In *International Conference on Machine Learning*, pages 2391–2400, 2017.
- [92] Yishu Miao and Phil Blunsom. Language as a latent variable: Discrete generative models for sentence compression. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 319–328, 2016.
- [93] Yishu Miao, Lei Yu, and Phil Blunsom. Neural variational inference for text processing. In *International Conference on Machine Learning*, pages 1727–1736, 2016.
- [94] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, 2004.

-
- [95] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*, 2010.
- [96] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.
- [97] Ziheng Lin Min, Yen Kan Chew, and Lim Tan. Exploiting category-specific information for multi-document summarization. *The 21th International Conference on Computational Linguistics (COLING)*, pages 2903–2108, 2012.
- [98] Soichiro Murakami, Akihiko Watanabe, Akira Miyazawa, Keiichi Goshima, Toshihiko Yanase, Hiroya Takamura, and Yusuke Miyao. Learning to generate market comments from stock prices. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1374–1384, 2017.
- [99] 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.
- [100] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *The Thirty-First AAAI Conference on Artificial Intelligence*, pages 3075–3081, 2017.
- [101] Ani Nenkova. Entity-driven rewrite for multi-document summarization. In *Third International Joint Conference on Natural Language Processing*, pages 118–125, 2008.

-
- [102] Ani Nenkova and Kathleen McKeown. A survey of text summarization techniques. In *Mining Text Data*, pages 43–76. Springer, 2012.
- [103] Jianmo Ni, Zachary C Lipton, Sharad Vikram, and Julian McAuley. Estimating reactions and recommending products with generative models of reviews. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 783–791, 2017.
- [104] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.
- [105] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*, 2017.
- [106] Dragomir R Radev, Hongyan Jing, Małgorzata Styś, and Daniel Tam. Centroid-based summarization of multiple documents. *Information Processing & Management*, 40(6):919–938, 2004.
- [107] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.
- [108] Zhaochun Ren, Hongya Song, Piji Li, Shangsong Liang, Jun Ma, and Maarten de Rijke. Using sparse coding for answer summarization in non-factoid community question-answering. In *SIGIR Workshop: Web Question Answering, Beyond Factoids*, 2016.
- [109] Zhaochun Ren, Shangsong Liang, Piji Li, Shuaiqiang Wang, and Maarten de Rijke. Social collaborative viewpoint regression with explainable recommendations. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 485–494. ACM, 2017.

-
- [110] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, pages 1278–1286, 2014.
- [111] Alexander M Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, 2015.
- [112] Abigail See, Peter J Liu, and Christopher D Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1073–1083, 2017.
- [113] Lifeng Shang, Zhengdong Lu, and Hang Li. Neural responding machine for short-text conversation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1577–1586, 2015.
- [114] Shi-Qi Shen, Yan-Kai Lin, Cun-Chao Tu, Yu Zhao, Zhi-Yuan Liu, Mao-Song Sun, et al. Recent advances on neural headline generation. *Journal of Computer Science and Technology*, 32(4):768–784, 2017.
- [115] Advait Siddharthan, Ani Nenkova, and Kathleen McKeown. Information status distinctions and referring expressions: An empirical study of references to people in news summaries. *Computational Linguistics*, 37(4):811–842, 2011.
- [116] Hongya Song, Zhaochun Ren, Shangsong Liang, Piji Li, Jun Ma, and Maarten de Rijke. Summarizing answers in non-factoid community question-answering.

- In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 405–414. ACM, 2017.
- [117] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.
- [118] Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1171–1181, 2017.
- [119] Jian Tang, Yifan Yang, Sam Carton, Ming Zhang, and Qiaozhu Mei. Context-aware natural language generation with recurrent neural networks. *arXiv preprint arXiv:1611.09900*, 2016.
- [120] Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May 2016. URL <http://arxiv.org/abs/1605.02688>.
- [121] Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 76–85, 2016.
- [122] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010, 2017.
- [123] Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko. Sequence to sequence-video to

- text. In *Proceedings of the IEEE international conference on computer vision*, pages 4534–4542, 2015.
- [124] Oriol Vinyals and Quoc Le. A neural conversational model. *arXiv preprint arXiv:1506.05869*, 2015.
- [125] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *Advances in neural information processing systems*, pages 2692–2700, 2015.
- [126] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 3156–3164. IEEE, 2015.
- [127] Martin J Wainwright and Michael I Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1-2):1–305, 2008.
- [128] Xiaojun Wan, Jianwu Yang, and Jianguo Xiao. Manifold-ranking based topic-focused multi-document summarization. In *Proceedings of the 20th international joint conference on Artificial intelligence*, pages 2903–2908. Morgan Kaufmann Publishers Inc., 2007.
- [129] Chong Wang and David M Blei. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 448–456. ACM, 2011.
- [130] Dingding Wang, Shenghuo Zhu, Tao Li, and Yihong Gong. Multi-document summarization using sentence-based topic models. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 297–300. Association for Computational Linguistics, 2009.

-
- [131] Lu Wang, Hema Raghavan, Vittorio Castelli, Radu Florian, and Claire Cardie. A sentence compression based framework to query-focused multi-document summarization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1384–1394, 2013.
- [132] Mark Wasson. Using leading text for news summaries: Evaluation results and implications for commercial summarization applications. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 2*, pages 1364–1368. Association for Computational Linguistics, 1998.
- [133] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, 1992.
- [134] Sam Wiseman, Stuart Shieber, and Alexander Rush. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263, 2017.
- [135] 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. Association for Computational Linguistics, 2012.
- [136] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google’s neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

- [137] Eric P Xing, Michael I Jordan, and Stuart Russell. A generalized mean field algorithm for variational inference in exponential families. In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence*, pages 583–591. Morgan Kaufmann Publishers Inc., 2002.
- [138] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057, 2015.
- [139] Zi Yang, Keke Cai, Jie Tang, Li Zhang, Zhong Su, and Juanzi Li. Social context summarization. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 255–264. ACM, 2011.
- [140] Jin-ge Yao, Xiaojun Wan, and Jianguo Xiao. Compressive document summarization via sparse optimization. In *The 24th International Joint Conference on Artificial Intelligence*, pages 1376–1382, 2015.
- [141] Jin-ge Yao, Xiaojun Wan, and Jianguo Xiao. Recent advances in document summarization. *Knowledge and Information Systems*, 53(2):297–336, 2017.
- [142] Yuanshun Yao, Bimal Viswanath, Jenna Cryan, Haitao Zheng, and Ben Y Zhao. Automated crowdturfing attacks and defenses in online review systems. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1143–1158. ACM, 2017.
- [143] L Yu, W Zhang, J Wang, and Y Yu. Seqgan: sequence generative adversarial nets with policy gradient. In *Thirty-First AAAI Conference on Artificial Intelligence*, volume 31, pages 2852–2858. Association for the Advancement of Artificial Intelligence (AAAI), 2017.

-
- [144] David Zajic, Bonnie Dorr, and Richard Schwartz. Bbn/umd at duc-2004: Topiary. In *Proceedings of the HLT-NAACL 2004 Document Understanding Workshop, Boston*, pages 112–119, 2004.
- [145] David M Zajic, Bonnie Dorr, Jimmy Lin, and Richard Schwartz. Sentence compression as a component of a multi-document summarization system. In *Proceedings of the 2006 document understanding workshop, New York*, 2006.
- [146] Matthew D Zeiler. Adadelata: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.
- [147] Biao Zhang, Deyi Xiong, Hong Duan, Min Zhang, et al. Variational neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 521–530, 2016.
- [148] Yizhe Zhang, Zhe Gan, Kai Fan, Zhi Chen, Ricardo Henao, Dinghan Shen, and Lawrence Carin. Adversarial feature matching for text generation. In *International Conference on Machine Learning*, pages 4006–4015, 2017.
- [149] Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 654–664, 2017.
- [150] Qingyu Zhou, Nan Yang, Furu Wei, and Ming Zhou. Selective encoding for abstractive sentence summarization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1095–1104, 2017.