Event-driven data collection and summarization

Zheng, Xin

2018

Zheng, X. (2018). Event-driven data collection and summarization. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/89715>

<https://doi.org/10.32657/10220/47146>

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

SINGAPORE

EVENT-DRIVEN DATA COLLECTION AND SUMMARIZATION

Xin Zheng

School of Computer Science and Engineering

A thesis submitted to the Nanyang Technological University
in fulfilment of the requirement for the degree of
Doctor of Philosophy

2018

Abstract

The online social networks have experienced an unprecedented proliferation. Various platforms change the way people learning about information, particularly on ongoing events, which can only be known from mainstream media in the past. The social media platforms have many pleasing properties compared with traditional media: *convenient, detailed, fast* and *interactive*. This provides us an opportunity to learn immediate and detailed information about event of interest, which is highly preferred by decision makers and the public especially when emergencies and significant events happen.

Keyword search function is provided by social media platforms like Twitter for users to search messages containing the query keyword(s). However, the returned results are piecemeal due to length limitation, could be mixed with irrelevant tweets or are incomplete due to inappropriate query. This calls for research on collecting clean and complete event-related messages from social media platform. In this dissertation, the researches are conducted on Twitter platform. The collected event relevant tweets could be a large set. Presenting the data in a concise and representative form could help end users to have a general idea of the collected information. Therefore, after collecting event-related tweets, we aim to construct a summary for the large set of data.

Collecting clean and complete event-related tweets from Twitter Stream is not a trivial problem. The challenges are as follows: (i) the great volume of tweets make the filtering process a heavy workload; (ii) tweets are short, noisy and with many abbreviations, dialects and misspellings which make it difficult to identify event-related messages, especially when we do not have enough training data for distinction; (iii) events are evolving and the collection should be adaptive to the development of events. The proposed methods in this dissertation deal with these challenges. Basically, the methods consist of two components. Keyword search is applied to collect candidate tweets. Next, tweets are

further identified to be event-related or not. To make the collected set of tweets be adaptive to event development, we propose dynamic keyword extraction methods. Properties of keywords which benefit candidate tweets collection are well explored. To relieve the requirement of labeled data for event-related tweets identification, we incorporate active learning with uncertainty sampling. In addition, the property information of keywords is also leveraged to further reduce the requirement of human labels.

As stated before, tweets are noisy and casually written. It is not suitable to extract tweets as summary directly. Therefore, we turn to well-written news articles linked by URLs in tweets, which should be of the same topic with event-related tweets. We call tweets with URLs linking to news as linking tweets. The news report the main information about the event of interest, while tweets could be diverse, including people's focuses, comments, and other complementary information to news. We aim to construct a summary based on both the linked news articles and event-related tweets. The summary should not only highlight the key points in news articles but also address people's focuses about the event. Peoples' focuses are usually ignored when summarizing only news articles, but they are important when presenting the core information to people who care about the event. We start from the summarization of single news document and linking tweets to multiple news documents and their linking tweets. Basically, we apply neural network models to tackle the semantic information in news and linking tweets. However, the proposed models in this dissertation do not depend on reference summaries which are of few number and not enough for training deep learning models with good performances.

To sum up, in this dissertation, we propose approaches of collecting clean and complete event-related tweets from Twitter Stream, and unsupervised models on summarizing single and multiple news documents with linking tweets.

Acknowledgments

First of all, I would like to express my sincere gratitude towards my supervisor Prof. Aixin Sun. Without his patient guidance, encouragement, immense knowledge and advice, I would not be able to finish my Ph.D career. Under his elaborative supervision, I have not only learned the state-of-the-art information retrieval knowledge, but also acquired a conscientious research attitude through critical thinking. The knowledge and methodology I learnt are invaluable to my research, and even to my life. I feel very grateful to Prof. Gao Cong for his introducing me to my supervisor. Without him, I may miss the chance to study in NTU and do research with Prof. Sun.

I would also like to express my great thankfulness to SAP for providing me Ph.D scholarship, and the chance to get access to real industry working environment and work with talented colleagues. I appreciate all the supports, guidances and suggestions given by my mentors: Dr. Karthik Muthuswamy, Zhenzhou Wu, Dr. Satish Verma, Naveen Nandan, and Dr. Dean Wang.

I would like to express my appreciation to all my seniors for their patience on our discussions and providing valuable suggestions: Dr. Chenliang Li, Dr. Zongyang Ma, Dr. Jialong Han, Dr. Surendra Sedhai and Dr. Jing Li. I also want to thank all my peers for their company and supports: Jianglei Han, Minh C. Phan, Keke Huang, Yequan Wang, Grace E. Lee, Ting Lin and Zhe Chen, Peng Han, Xiaoshi Zhong. I enjoy the active and pleasant environment of sharing knowledge and experiences. Many thanks for all my friends for their constant companion and precious friendship. They include Dr. Wenya Wang, Jin Wang, Haodan Yang, Peng Ni, Shuhao Zhang, Bingshui Da, Hangwei Qian, Qing Guo, Jianshu Li, Ruidan He, Kun Ouyang, Wei Cui, Haiyan Yin. I would also like to appreciate all my friends at the DMAL Lab for their friendly support and help: Dr. Jun Zhang, Dr. Sibao Wang, Dr. Lu Chen, Dr. Le Wu, Dr. Bolong Zheng, Youze Tang,

Yanchen Shi, Renchi Yang, Xi Lin, Dr. Kaiyu Feng, Dr. Kaiqi Zhao, Dr. Tao Guo, Yiding Liu. I am truly thankful for the convenient space and excellent facilities provided by the Data Management and Analysis Lab. Many thanks for the technique support from our administrator Thomas Loo Kian Hock. Thanks Irene Goh for providing us the well managed GPU server platform.

Last but not least, I would like to sincerely thank my family, my mother and father, for their constant supports, encouragement and love all the time. I really appreciate for all the selfless sacrifices that they have made for me, especially for my parents' respects on all my decisions. I love you as always.

Contents

Abstract	i
Acknowledgements	iii
List of Figures	ix
List of Tables	x
1 Introduction	3
1.1 Motivation	3
1.2 Challenges	7
1.3 Contributions	12
1.4 Thesis Organization	14
2 Literature Review	17
2.1 Methods for Event-related Data Collection	17
2.1.1 Methods for Keyword Extraction	19
2.1.2 Methods for Event-related Document Identification	23
2.2 Methods for Summarization	25
2.2.1 Single Document Summarization	25
2.2.2 Multi-Document Summarization	28
2.2.3 Summarization with Tweets	31
3 Collecting Event-related Tweets from Twitter Stream with Active Learning based on Multiple Instance Learning	35
3.1 Introduction	35
3.2 The Proposed Method: ALMIK	36
3.2.1 Event-related Tweets Identification	37

3.2.2	Keyword Extraction	40
3.3	Experiments	42
3.3.1	Dataset	42
3.3.2	Initial Keyword Generation	44
3.3.3	Baseline Methods	45
3.3.4	Evaluation on Event-related Tweets	50
3.3.5	Evaluation on Event-related Keywords	52
3.4	Summary	56
4	Semi-Supervised Event-related Tweet Identification with Dynamic Key- word Generation	59
4.1	Introduction	59
4.2	The Proposed Method: SSEM	60
4.2.1	Expectation Maximization Identifier	61
4.2.2	Word Importance Score	63
4.2.3	Determination of Changeable Tweets	66
4.2.4	Dynamic Keyword Graph	68
4.3	Experiments	71
4.3.1	Dataset	71
4.3.2	Baseline Methods	72
4.3.3	Evaluation on Event-related Tweets	76
4.3.4	Evaluation on Keywords Generation	77
4.4	Summary	80
5	Tweet-aware News Summarization by Attention and Reconstruction	83
5.1	Introduction	83
5.2	Data Analysis	85
5.3	The Proposed Method: ARS	89
5.3.1	Attentive Information Extraction	90
5.3.2	Reconstruction Regularization	92
5.3.3	Sentence Selection	96
5.4	Experiments	96

5.4.1	Dataset and Model Implementations	96
5.4.2	Baseline Methods	99
5.4.3	Performance Comparison	100
5.4.4	Sentence Selection Strategies	102
5.4.5	Model Sensitivity to Parameter λ	105
5.4.6	Case Studies	106
5.5	Summary	108
6	Aspect-driven Multi-Document Summarization	111
6.1	Introduction	111
6.2	Preliminary	112
6.3	The Proposed Method: ADS	113
6.3.1	Document Representation	114
6.3.2	Aspect Representation	115
6.3.3	Saliency Estimation for Aspect	116
6.3.4	Saliency Estimation for Sentence	117
6.4	Experiments	119
6.4.1	Dataset and Experimental Settings	119
6.4.2	Baseline Methods	120
6.4.3	Performance Comparison	121
6.4.4	Comparison on Different Sentence Selection Strategies	123
6.5	Summary	125
7	Conclusion and Future Work	127
7.1	Conclusion	127
7.2	Future Work	130
	List of Publications	133
	References	135

List of Figures

1.1	Overview of researches in the thesis.	5
3.1	Overview of the proposed ALMIK. The main techniques for event-related tweets identification and event-related keywords extraction are further illustrated on its right. Wikipedia is only utilized to derive the seed keywords.	37
3.2	Performance comparison.	46
3.3	F_1 comparison of AL _s , AL _p and ALMI over different number of manually labeled clusters on <i>EQ</i> data, from t_1 to t_6 . Since tweets in one cluster are nearly duplicate or similar with each other, the labeling effort for one cluster is similar to that of one tweet.	49
3.4	F_1 comparison of AL _s , AL _p and ALMI over different number of manually labeled clusters on <i>Flood</i> data from day_1 to day_3	51
4.1	The information flow for one iteration is from Step 1 to 5. After getting new training data from Steps 4 and 5, a new iteration would start to retrain the classifier.	61
4.2	An example of word graph changes over time.	69
4.3	Performances comparison of the proposed method SSEM and baseline methods DL, LDA, TTM, ADP, and AL, on EQ.	75
4.4	Performances comparison of the proposed method SSEM and baseline methods DL, LDA, TTM, ADP, and AL, on Flood.	75
4.5	Performances comparison of the proposed method SSEM and baseline methods DL, LDA, TTM, ADP, and AL, on NBA.	76
4.6	Changing of F_1 measures of AL and SSEM, with different number of human labeled clusters on the three events EQ, Flood and NBA.	78

5.1	Distribution of documents with respect to the corresponding ratio of overlapping words, against tweets vocabulary and news vocabulary respectively.	87
5.2	Framework for summarization by attention and reconstruction.	89
5.3	Heat map of two examples of position-dependent word salience scores. . .	105
5.4	Model sensitivity to parameter λ on different sentence selection strategies.	106
6.1	Hierarchical structure for sentence and document representations construction.	114
6.2	Salience estimation for news sentences and aspects.	115

List of Tables

3.1	The total number of tweets $\mathcal{T}$ and the number of positive tweets $\mathcal{T}_P$ within each time window in the simulated Twitter Stream.	43
3.2	Example keywords extracted on <i>EQ</i> during t_3 . Keywords in bold are irrelevant to the event. Same as follows.	53
3.3	Example keywords on <i>Flood</i> during day_2	54
3.4	Number of tweets $ TC $ collected with keywords generated by each method on <i>EQ</i> and <i>Flood</i> . $ TC_P $ is the number of event-related tweets included in <i>TC</i> . $ CN $ is the number of tweet clusters. We do not include <i>ADP</i> , because the relevant tweets in testing set are not determined by keywords. Information of the first time window is not presented because it is the same for all the methods.	55
4.1	Example of word's configuration similarity. $sim(w, v)$ and $sim(w, k)$ are calculated according to Equation 4.6. If there is no edge between word w and k , the edge weight $sim(w, k) = 0$. m is the number of words in keyword set.	70
4.2	Number of tweets $\mathcal{T}$ fall within the time window in the simulated Twitter stream (<i>i.e.</i> , 1% sample of the entire Twitter), and number of positive tweets $\mathcal{T}_P$ in each time window on the three datasets.	72
4.3	Example keywords extracted on EQ in time t_3 . Keywords in bold are NOT related to the event of interest.	79
4.4	Example keywords extracted on Flood in time day_2 . Keywords in bold are NOT related to the event of interest.	80
4.5	Example keywords extracted on NBA in time t_4 . Keywords in bold are NOT related to the event of interest.	81

5.1	Number of unique news linked by URLs in event-related tweets.	84
5.2	Quantitative evaluation: usefulness of linking tweets for news summarization.	86
5.3	Results of ARS and all baselines. All ROUGE scores reported in this chapter are with significance test and are in 95% confidence interval with at most ± 0.25 of reported results calculated by the official ROUGE script with default settings. Best F1 scores are in bold and second best are underlined.	98
5.4	Results of using different sentence selection strategies. Best F1 are in bold and second best underlined.	103
5.5	Summaries of a sampled news article.	107
5.6	Example summary of linking tweets.	108
6.1	Results of the proposed methods and baselines. “R” indicates ROUGE. All the ROUGE scores reported in this work are with significance test and the values fall in 95% confidence interval with at most ± 0.25 of reported results calculated by the official ROUGE script with default settings. . .	122
6.2	Results on different sentence selection strategies, which are also with significance test as above.	123
6.3	Example sentences.	124

Chapter 1

Introduction

1.1 Motivation

Recent decades have witnessed the great proliferation of various social media platforms with different functions and properties, like Twitter¹, Facebook², Instagram³, Pinterest⁴, just to name a few. Nowadays, people are changing their habits of learning about information from traditional TV programs and news articles. They are getting accustomed to posting, sharing, surfing and searching anything on various social platforms anytime and everywhere. For example, when a natural disaster happens, victims could post tweets reporting locations or conditions, and asking for help [128]. Families or friends of victims may post finding people when they cannot contact them [142]. When a significant sport event happens, the audiences on the spot would usually like to report the details of current situation on social media sites via text, photos or videos to record the moment.

In contrary to traditional information distributing channels, *e.g.*, TV programs, news articles, social network sites provide us an alternative. Meanwhile, they present many pleasing properties compared with traditional channels: *convenient, fast, detailed*, and

¹It is a social networking service on which users can post and interact with anyone through “tweets”. <https://twitter.com/>

²It is a social media platform which requires permission when adding friends. <https://www.facebook.com/>

³It is a photo and video-sharing social networking site. <https://www.instagram.com/>

⁴A software system aims to discover information based on catalogue mainly through images and videos. <https://www.pinterest.com/>

interactive. In reality, it is impossible for the journalists to learn about everything immediately, especially for emergencies like natural disasters, terrorist attacks and bursting events for organizations [25, 29, 85, 128, 129]. For such events, the involved parties or eyewitnesses usually could learn about first-hand information. On the other hand, some events could be prescheduled, *e.g.*, significant sport events and competitions. Even if the journalists could prepare beforehand and learn about the information immediately, they only report the primary or most attractive part of information from their view points. Audiences on the spot could probably be able to cover more details. Thanks to the social media sites, involved parties, eyewitnesses or the audiences are able to post relevant pieces of information on social platforms through various formats before the official news is reported by journalists.

Many people could benefit from the faster, detailed and interactive information on social media sites. For example, decision makers in natural disasters or government bodies responding to terrorist attacks could learn about the details much faster and come to quicker decisions, which could help save time, money or even more lives. Similarly, rescuers could know about more accurate information and avoid wasting time. Fans or people interested in significant sport events could enjoy or learn about more when they get access to the detailed information. Analogous situations are numerous and we just name a few here. In short, there is a lot of demand for acquiring event-related messages from social media sites. Due to limitations on data access, we make research on Twitter platform and only focus on text content in this dissertation. Specifically, we will make researches on collecting clean and complete event-related tweets from Twitter Stream. Note that the problem setting in this dissertation is different from event detection and event tracking. The event of interest is predefined in our problem setting, while it is needed to be detected in event detection [128, 129]. Event tracking means starting from a few sample stories and finding all subsequent stories that discuss the same event [2]. In our problem setting, only a few keywords defining the event are given as in Figure 1.1.

Figure 1.1: Overview of researches in the thesis.

Assuming we have obtained the clean and complete set of event-related tweets, presenting all the tweets to users is not a good idea. The reason is that the collection of tweets could be significantly large. It could take a rather long time to get through all the collected tweets. Further, retweets or near-duplicate posts are common on Twitter. People could get tired before they reach their concerned or interested information. Hence, a proper way to present the collection of tweets is needed. We choose summarization to consolidate the core information in collection.

As we know that, tweets are noisy and casually written. It is very common to find

@account, #hashtags, URLs, various symbols, abbreviations, slangs and misspellings in tweets. Hence, to construct a concise and representative summary, we should refer to some well-written documents. Note that the summarization is conducted after we obtained the collection of tweets, which means event-related news documents from every channel are starting to be reported as well. Such event-related news can be retrieved through URLs cited in the collected event-related tweets. Citing web pages, including news articles, in tweets through URLs is very common on Twitter as we discovered in Section 5.1 of chapter 5. And we take event-related tweets with URLs linking to a news article as its linking tweets. It is easy to obtain a set of news articles by selecting web pages of which the full URLs contains pre-defined major newspapers' names. Furthermore, there is great probability that the news document and its linking tweets describe the same event and share overlapping information. News usually contain the most significant information and are written in a formal way. This is enough to cover major information of the collected event-related tweets, so that news articles are good candidates to extract sentences from for summary generation. Moreover, news articles are generally to be objective. They tend to depict the truth without involving subjective comments or individual emotions, which could appear in tweets. Hence, the collected tweets and their linked news articles are kind of complementary to each other. By incorporating correlated tweets and news articles one could generate more reasonable and general summary. Therefore, we aim to summarize the news articles together with their linking tweets.

Overall, the researches in this thesis includes collecting clean and complete event-related tweets from Twitter Stream and presenting the summarization of both news documents and their linking tweets as shown in Figure 1.1.

1.2 Challenges

Neither collecting tweets nor summarization is an easy task. Many challenges are there and needed to be resolved to achieve the goals. The challenges are presented as follows.

Challenges for Collecting Event-related Tweets. Given few words defining an event of interest, which could be name of event (*e.g.*, Chile earthquake), the task is to collect clean and complete event-related tweets from Twitter Stream. Comparing with event detection, although the event is not needed to be detected, the main challenge is the completeness of the collected tweets. On the other hand, only few words are known about the event of interest, in contrast with event tracking in which a few stories related to the event are available. This makes our problem harder to resolve.

Existing methods for collecting event-related data from Twitter Stream are mostly location-based [21, 165], and keyword-based [29, 150]. Location-based methods define a geographical boundary and collect tweets based on geo-tags or location information in user profiles. This type of methods inherently would miss out tweets without geo-tags, or messages posted by users from other places and users with incomplete user profiles. Keyword-based methods, on the other hand, heavily depend on the comprehensiveness of query keywords. Usually, a set of high quality keywords is manually generated by human with good domain knowledge but is time-consuming [100, 128]. Thus, automatic keyword generating methods are proposed to solve the problem [8, 29]. According to our target of data entirety, we adopt keyword-based method and propose automatic keyword generation approaches without requiring domain knowledge and human judgment.

Twitter provides keyword search function (*i.e.*, both search box on user interface and application programming interface ⁵) for users to find their interested posts through keywords. However, it is often challenging for common users to formulate high quality

⁵<https://developer.twitter.com/en/docs/tweets/search/api-reference/get-search-tweets.html>

queries. It is usually seen that irrelevant messages would mix with important ones, or sometimes the desired messages are not retrieved at all [44, 164]. The reasons are as follows (i) query keywords are not appropriate and the relevant tweets do not contain keywords for searching; (ii) not all of the tweets containing query keywords are event-related. Besides the issues for keyword-based searching stated above, the development of events should also be taken into consideration. The dominant keywords for an event at different stages are different as well. For example, when an area is just struck by an earthquake, people are talking about rescuing, aftershock, death and injured people. Later, rebuilding and serving orphans could be the major issues.

Based on the analysis above, challenges for automatic keyword generation are concluded as follows. First, the keywords should be dynamically generated and updated together with the evolvement of an event. A preferable way is to predict the potential core keywords in the following stage beforehand, so that event-related tweets will not be missed out. Second, the selected keywords should be able to cover as many event-related tweets as possible so that we can collect desired tweets with fewer keywords. Note that, all the challenges for keyword generation can only be resolved based on the large volume of noisy tweets. No labeled data for training and learning but only a few words defining the event of interest. This makes keyword generation from tweets a difficult task.

Note that not all the tweets containing keywords are event-related. Some words could have many meanings and the tweet means differently could also been returned in keyword search. Besides, some tweets add several `#hashtags` just to attract more attentions, although the content could have nothing to do with the event of interest. For the goal of clean event-related tweet set, we need further identification for the candidate set of tweets as shown in Figure 1.1.

A straightforward method for identifying event-related tweets is classification [100, 158]. However, it is impractical to classify tweets with a pre-trained classifier. First,

such a classifier would require a large amount of annotated data for training on different events. Second, as events evolve, historical labeled data becomes less relevant to the current development of an event. The specificities of events are different from each other. Researchers also try to apply topic model [149, 164], but the models can hardly well handle all the aspects of an event. Moreover, post classification processing is often required. To construct a high quality identifier, we believe human labels are necessary as input. The challenge is how to minimize human effort on labeling data without compromising data quality.

Challenges for Summarization. For summarization task, many different settings have been proposed to fit different scenarios and been studied for decades, *e.g.*, single document summarization [106, 116], multiple documents summarization [88, 138], query-based summarization [45, 148] and so on. As stated in Section 1.1, we focus on summarization of news documents and their linking tweets. Recently, some researches start to explore auxiliary information beyond news articles, *e.g.*, comments [60, 61], linking tweets [80, 81, 152, 154], to help improve model performances. Hu *et al.*, Huang *et al.* and Li *et al.* [60, 61, 81] pay more attention to readers’ opinions or comments and construct comment-oriented summarization. However, in recent years many news websites including Reuters, NBCnews, and CNN, close their online reader comment section.⁶ One of the reasons is that many conversations between readers nowadays happen on social media and online forums, making comments on news sites unnecessary.⁷ Some researches like [80, 152, 154], extract features from linking tweets, but do not further explore relationships between news articles and linking tweets. In our study, we will explore such relationships and make better use of them to formulate reasonable summaries to general public.

⁶<http://thefederalist.com/2017/07/26/npr-cnn-news-sites-hit-mute-button-readers/>

⁷<https://edition.cnn.com/2014/11/21/tech/web/online-comment-sections/index.html>

We make data analysis in Section 5.2 of chapter 5 and Section 6.2 of chapter 6. Specifically, we observe that the information in a news document and linking tweets is not totally overlapped. Compared with content of news documents, their linking tweets are more diverse and detailed. This also coincides with the motivation for event-related tweets collection in Section 1.1. Except users' focuses that are also mentioned by news articles, personal comments containing complementary information do not appear in news. The supplemental information could be rather detailed, like individual cases in an emergency. In addition, news articles can report information not revealed in linking tweets. For example, from the linking tweets of a news article on Boston bombing in our investigated dataset, we know that the suspect deletes an important Instagram account. However, we can only learn this from the news article that digital traces remain for the deleted posts because no linking tweets mention this point.

Through data analysis, we have better understanding of the relationships between collected tweets and the linked news. Recall that the reason of incorporating linked news for tweets summarization is that news articles are usually written in concise and well-formatted way in contrast to the noisy and casually-written tweets. For information only appears in tweets, no news sentences could be referred to for the summarization. Additionally, it is not certain there exists well-written tweets to be considered as summary for such information. Therefore, for summarization works in this thesis, we do not specifically take information solely contained in tweets into consideration. This is left for our future works.

For the overlapping information between tweets and news, we can find representative news sentences for corresponding tweets. We agree with the discussion in previous works [80, 163] that the points cared about by some people could also be focuses to others. Hence, such information should be summarized which could be valuable and useful to others. We explore the attractive points to users from tweets which share overlapping

information with news. Although such information appears in both news documents and linking tweets, there exist differences. The matching information in linking tweets could highlight some content that is not obvious in news articles but is cared about by readers, which is also claimed in [80]. Only considering news may not capture such points. Besides, from the example above, we notice sometimes tweets may only cover part of the information, whereas that in news are usually complete. To consolidate the summary, we also summarize information only appears in news documents. Therefore, we define the problem as constructing summary from both focuses revealed in linking tweets and the information in news articles.

However, it is not easy to deal with our summarization problem. Firstly, the representation of news articles and their linking tweets has major impact on what kind of information could be conveyed. We claim that the traditional bag-of-words representation method [53] only tells statistical knowledge, but cannot reveal semantic, sequential and contextual information. However, such are important to better understand the content, especially when comparing information contained in news documents and their linking tweets. Thus, we need to construct an appropriate sentence or document representation method. Secondly, information comparison between news articles and linking tweets is challenging. Before comparing the information, we need to understand the content reported in news and linking tweets. However, natural language understanding is considered an AI-hard problem [157] which refers to the most difficult problems in the field of artificial intelligence. Although it is not required to comprehend news and linking tweets deeply, the semantic meaning probably expressed in totally different formats should be understood to some extent. Therefore, it requires to pay more efforts to understand and compare the reported content in news and linking tweets. Next, when dealing with multiple news documents, the relationships among them should be tackled carefully. The relationships are complicated, which could be similar, complementary, and evolutionary,

and cannot be concluded as one type. Proper approaches should be adopted to deal with the problem. Due to the difficulties, we start from single news document and its linking tweets, then generalize to multiple news articles with their linking tweets.

Overall, collecting event-related clean and complete tweets and summarization of collected tweets with news articles linked by them are not trivial. Many challenges are to be overcome for producing pleasant performances.

1.3 Contributions

This thesis focuses on developing and improving techniques in IR and NLP field, specifically on collecting and presenting event-driven social media data. The main contributions in this dissertation are summarized as follows:

- We define and investigate event-related tweets collection problem. To be concrete, the aim is to gather clean and complete set of event-related tweets from Twitter Stream given only a few words defining an event of interest. As stated in Section 1.1, the problem is different from event detection and tracking. Besides, the collected data could benefit many people and be leveraged for downstream applications.
- Ranking approach is adopted to select keywords for tweets collection. We propose preferred properties for keywords evaluation: *relevance*, *coverage* and *evolvment*. Existing methods do not formally investigate the coverage ability and development of query keywords. The coverage property help to avoid excluding event-related tweets. Besides, the selected keywords should also be adapted to event development, so that tweets related to the event yet with little relationship with previous information could also be collected. We propose specific measures for these properties. Besides, we notice some core keywords always ranked top but they have already been added into keyword list. Such keywords are not preferred to be ranked high

for keyword extraction. We propose dynamic keyword graph and configuration similarity to evaluate the above situation. Thus, keywords that are event-related, with high coverage, dynamically evolving, are extracted for tweets collection.

- To ensure high quality of collected tweets, we adopt active learning to incorporate human labeling process. Uncertainty sampling is used to reduce unnecessary label requirement (as in the model proposed in chapter 3, ALMIK). Further, we find keywords also contain rich information which could be utilized to identify event-related tweets. On the contrary, the identified event-related tweets also influence the estimation of keywords properties. We propose to bridge communications between event-related tweets identification and keywords extraction through expectation maximization process (*i.e.*, the model proposed in chapter 4, SSEM). Therefore, human labeling efforts could be further reduced and making the identification as semi-supervised.
- We propose a tweet-aware single news document summarization model which captures not only important information highlighted by readers in linking tweets but also information reported in news. Content of linking tweets are diverse. The overlapping information of tweets and news are identified through attention mechanism. On the other hand, the attention mechanism is also utilized to estimate word salience scores, which are position-dependent. The reason is that relative importance of a word should be different in different contexts. In addition, the other important information in news should also be added in final summary. Thus, we propose reconstruction regularization to enrich saliences of words. From word saliences, we can estimate sentence salience scores and extract summary sentences. With a large number of linking tweets and the reconstruction regularizers, our

model could be trained in an unsupervised fashion, *i.e.*, without using any reference summaries⁸.

- We further generalize the summarization of single news article to multiple news documents with their linking tweets. We propose to deal with both sequential context information in single document and the relationships among multiple documents in the given set. The sequential context information could be encoded by bidirectional Recurrent Neural Network (RNN) inherently. To avoid distinguishing the complex relationships among multiple documents, we distribute news sentences and tweets into latent aspects. Here, readers' focuses in tweets could help identify attractive information for general readers. Thus, the underlying topic could be represented from document perspective and aspect perspective. The representations of both views should be similar, which contributes to model unsupervised fashion. Instead of estimating sentence saliences directly, we adopt hierarchical style. Different aspects should be of different importance, which further influence salience scores of sentences which belong to each aspects.

1.4 Thesis Organization

The rest of the thesis is organized as follows. We make a detailed literature review of methods on data collection and summarization in chapter 2. We present the supervised keyword-based approach on event-related tweets collection in chapter 3. To further reduce human efforts on data labeling, we propose a semi-supervised method that incorporates keyword information when identifying event-related tweets and present it in chapter 4. In chapters 5 and 6, we introduce the works on summarizing news together with tweets linking to the news articles. We start from single document summarization in chapter 5

⁸Reference summaries are usually manually generated and are costly. We consider them as ground truth in this work.

and move to multi-document summarization in chapter 6. Finally, chapter 7 concludes the research works and propose promising directions for future works.

Chapter 2

Literature Review

In this chapter, we provide an overview of diverse methods on event-related social media collection and summarization. Specifically, we first introduce commonly used methods for event-related data collection. Because we adopt keyword-based data collection method, keyword generation approaches and event-related social media data identification are reviewed in details. Next, we review methods for summarization. We do not only consider approaches that incorporate social media data, but also those without such auxiliary data. Specifically, we review the methods from extractive and abstractive views on both single and multiple document summarization. Models combining social media data are reviewed separately.

2.1 Methods for Event-related Data Collection

Location-based Methods. There are methods for collecting data using location-based constraints [21, 165] for the purposes of detecting emerging local events or local recommendations. Location-based method defines the boundary of a geographical region. Tweets that are posted within the region or posted by the users with location profile at that region are collected. However, it is easy to miss tweets posting from other locations and tweets posted by users without location profiles or with locations indicated in location profiles out of predefined boundary.

Topic-Model-based Methods. Collecting event-related data could also leverage topic models. Yang *et al.* [158] collect messages related to adverse drug reaction from social media through Latent Dirichlet allocation (LDA) [17] and a partially supervised classification approach. Wang *et al.* [149] propose a targeted topic model which leverages a latent variable to identify documents belonging to some specific aspects. Although [63] is to detect novel topics, the authors claim their dictionary learning based method could collect topic-related data with good precision and recall on test data. However, it is not very easy to apply topic model on streaming data. Besides, events are evolving, it is not suitable to judge whether a set of tweets related to an event or not only based on the representative keywords which are produced statistically. Even the representative keywords for each topic set seems to be irrelevant to the event, tweets in the set are still possible to be event-related. In other words, it is not efficient to filter event-irrelevant tweets by topic-model-based models.

Keyword-based Methods. A straightforward approach to collect desired data is keyword search [29, 128]. Sakaki *et al.* [128] collect earthquake-related tweets using a pre-defined set of keywords. Similarly, Magdy and Elsayed [100] prepare a static set of well-defined queries about the topic of interest. They consider tweets containing words in the set of queries as topic-related. Constructing event-related keywords requires good domain knowledge. Thus an automatic way of generating event-related keywords is preferred. We mainly focus on automatic keywords generation methods. Many different kinds of methods are proposed to generate preferred keywords to collect event-related data [8, 29, 78, 127]. Again, event development is also an issue for keyword-based methods. Therefore, keywords generation should take this into consideration and produce dynamic query candidates. Since every kind of method is not perfect, we choose keyword-based method and aim to overcome the drawbacks with other techniques together.

2.1.1 Methods for Keyword Extraction

Keyword extraction has been studied for a long time because of the great usefulness on downstream applications including summarization, topic extraction and query, just to name a few. At the early stage, many researches focus on keywords or keyphrases extraction from formal documents, *e.g.*, news and academic articles [125, 141, 155]. This kind of documents is well-structured and well-written, which makes keywords or keyphrases generation with few noises. With the development of social media, larger and larger volume of data is generated by users everyday. Unlike formal document, such kind of short texts are casually written. Many abbreviations, slang, misspellings and noises are within the short texts. Methods on formal data cannot perform well on social media data and methods specifically for social media data are proposed [86, 101, 159].

From another angle, methods for keywords or keyphrases extraction could be categorized into supervised and unsupervised methods. Supervised methods usually formulate the problem as a binary classification task, that a word is a keyword or not [86, 92, 125, 141, 146, 155]. However, this type of methods require large amount of training data which is not practical for event-related keywords extraction due to costly labeling effort and lack of adaptive ability. Therefore, we mainly focus on unsupervised methods on keywords extraction and review relevant works in the following.

Many approaches are proposed to extract representative keywords, which could be used to query event-related tweet candidates. Three main approaches are (i) ranking keywords through frequency-based methods, (ii) ranking keywords through graph-based approaches, and (iii) ranking keywords through topic models. Later, some dynamic keyword generation methods are proposed to fit data changing.

Frequency-based Methods. Actually, word frequencies [96], TFIDF [131, 132] and pairwise word co-occurrence frequencies [102] remain as simple yet strong baselines for unsupervised keyword extraction. But these frequency-based methods do not take lexical

and semantic information into consideration. To resolve this issue, some researchers turn to WordNet ¹ [161] or Wikipedia [37, 95, 140] to obtain the thesaurus. Some others even build their own corpus. Liu *et al.* [95] measure word relatedness through Wikipedia and they take each Wikipedia page as a concept. For each word, a concept vector could be calculated where the value is TFIDF score of each term in each Wikipedia page. A similarity measure, like cosine similarity, Pointwise Mutual Information, could be used to calculate the relatedness score. Then they adopt different clustering methods to group similar words together which considers terms' semantic meaning. Similarly, Liu *et al.* [90] also cluster words, but they calculate cluster term frequencies instead of selecting exemplar terms from each cluster as in [95]. Besides, Liu *et al.* [90] also incorporate part-of-speech (POS) information and sentence salience score for keywords extraction. They assume that only noun, verb and adjective words are likely to be keywords and other type of words identified by POS are not considered. The word importance score is further adjusted by sentence importance. In keyphrase extraction, frequency-based method can be easily affected by the frequencies of substrings. Thus, Yang [160] propose to incorporate local significant features together with global statistical information. They extract the max-duplicated strings by building PAT-tree [33] for the document, so that the keywords can be picked out from the max-duplicated strings by their significance values. For more detailed keyphrase extraction methods review, we refer readers to a recent survey [54].

Graph-based Methods. Usually the graph refers to word graph, where words are vertices and an edge exists when two words appear in the same document. Either edge weight or node weight could be added in to the graph. As a classic keyword generation method, Mihalcea and Tarau [104] apply PageRank [113] to word graph constructed from documents. Top ranked words calculated by their own defined function are representative

¹<https://wordnet.princeton.edu/>

keywords. Li *et al.* [79] build a phrase semantic network from Wikipedia and compute phrase importance through which the influence of title phrases is propagated to the other phrases iteratively. They assume that the phrases in title could reflect the document content and their importance could influence the phrases in the document consequently. Liu *et al.* [94] also build a variation of PageRank algorithm. Each document is assigned to different topics by LDA. Then they build a Topical PageRank to decompose traditional random walks specific to various topics. As the result, the words are measured by their importance to each topic. Wang *et al.* [150] propose a double ranking approach to select a set of keywords which could be used to collect tweets satisfying a given topic or a research problem. Graph-based keyword extraction method usually rank words according to centrality measure [1]. Actually, properties of nodes (*i.e.*, words) also influence the importance of a word. Bellaachia and Al-Dhelaan [11] propose a Node and Edge Rank (NE-Rank) algorithm which incorporates node weights into PageRank. TFIDF values are taken as node weight. In addition to statistical TFIDF feature of node, other information of words also influence their importances. Biswas *et al.* [16] integrate features like frequency, centrality, position and strength of neighbors as node (*i.e.*, word) weight and select keywords based on NE-Rank. We refer readers to a detailed survey [10] on graph-based keyword extraction approaches.

Topic-Model-based Methods. Topic models, *e.g.*, Probabilistic Latent Semantic Analysis (PLSA) [59], Latent Dirichlet Allocation (LDA) [17], could generate keywords automatically which is based on statistical analysis. Chen *et al.* [30] propose a topic model which leverages general knowledge, *i.e.*, lexical semantic relations of words such as synonyms, antonyms, and adjective attributes, to help produce more coherent topics and representative keywords. Yan *et al.* [32] build a topic model by combining term correlation matrix with non-negative matrix factorization. The term correlation matrix could further be decomposed to term-topic matrix, and topic representative keywords

could be obtained. After calculating topic distributions with LDA in a given conversation transcript, Habibi and Popescu-Belis [50] explicitly set topic diversity constraint for limited number of keywords which is resolved using a submodule method as in [88]. Except extracting keywords for all the topics, Wang *et al.* [149] propose a targeted topic model which could identify keywords of specific aspects. For example, there are many subtopics under camera domain, and users may only concern about the weight or lens. Given a few words which define the aspect, their model, which is a variation of LDA, could return keywords related to the specified aspect. Thus, they provide a way to focus on keywords of a targeted subtopic.

Dynamic Keyword Generation Methods. The aforementioned methods all generate keywords from a static collection and do not take temporal development of event into consideration. This is also one of the main focus of our research. Before we have the relevant data, one may leverage the predefined keyword set to collect desired data from data stream [128]. Then an automatic way of generating keywords that are related to an ongoing event is much desired. Chen *et al.* [29] filter organization-related tweets through a set of fixed keywords and also keywords generated dynamically. The fixed keyword set is generated based on domain knowledge. The dynamic keywords are identified by using chi-square test [72] on word frequencies of the predefined foreground and background tweets. Becker *et al.* [8] leverage term frequency analysis and event-related concept extraction from external sources to generate dynamic keyphrases. Li *et al.* [78] adopt Rocchio query expansion [20] to dynamically change query keyphrases. Wang *et al.* [150] propose a double ranking approach to select a set of keywords which could be used to collect tweets satisfying a given topic or a research problem. This method relies on human input to judge the quality of keywords and assumes the dataset is static. Sadri *et al.* [127] map query phrases generation process to a Multi-armed Bandit problem [162] to maximize query coverage. The existing methods do not consider what kind of words could

introduce more event-related tweets. We analyze the problem by considering the changes of keyword graph over time. In this study, we do not consider keyphrase extraction and we refer readers to a recent survey on keyphrase extraction methods [54].

Word variations on social media is a key issue to hinder keyword extraction techniques to be applied on. Marujo *et al.* [101] propose to leverage brown clustering [112] and continuous word vectors [89] to normalize variants of words in tweets.

2.1.2 Methods for Event-related Document Identification

A set of tweets could be collected by selected keywords generated by various methods. However, we cannot consider them as event-related directly due to following reasons:

- A keyword could have multiple meanings and tweets containing the words but with other meanings can be collected as well;
- Several irrelevant hashtags, which could happen to be one of the selected keywords, are usually added in tweets to attract more attentions.
- Misspellings always happen in tweets and such tweets could contain keywords.

Therefore, to obtain a satisfied set of documents, further identification of event-related tweets is necessary. We review methods on event-related text identification methods in this section.

Classification-based Methods. Classification-based methods are widely adopted for this task. Verma *et al.* [142] build a classifier to automatically detect event-related messages, utilizing a combination of hand annotated and automatically-extracted linguistic features. Becker *et al.* [9] leverage a supervised method to identify tweets as event-related or not. However, tweets related to any event are identified, not necessarily to a specific event. Magdy and Elsayed [100] adopt a classification method. They prepare a static set

of well-defined queries about the topic of interest and assume tweets containing words in the query set as topic related. TF-IDF score is used to rank topic related keywords and filter potential topic relevant tweets from irrelevant ones. The trained classifier is used to infer labels for newly incoming data. A drawback of this solution is that the labels of training data is not confidently of good quality compared with human labels. To reduce human annotation efforts, Tong and Koller [139] apply active learning to text classification and analyze the effectiveness of different query strategies. An example of leveraging active learning to query needed data from search engine is [78]. The framework heavily relies on the ranked documents returned by the search engine. Besides classification, Sadri *et al.* [127] identify topic related tweets by measuring phrase-based relevance, textual relevance and user historical tweet relevance.

Topic-Model-based Methods. Topic model has also been applied to identify event-related data. Yang *et al.* [158] collect messages from social media related to adverse drug reaction by leveraging Latent Dirichlet allocation (LDA) and a partially supervised classification method. Wang *et al.* [149] propose a targeted topic model which leverages a latent variable to identify documents that belong to specific aspects. Although [63] studies novel topics detection, they claim that their dictionary learning based method could also identify topic relevant documents with high precision. Yan *et al.* [32] also adopt dictionary learning method to identify topic relevant documents. They build a topic model by combining term correlation matrix with non-negative matrix factorization. Non-negative matrix factorization is a variant of Dictionary Learning [63]. Bommanavar *et al.* [18] estimate the volume of tweets about a specific event or topic at a given precision level. This is achieved by Sequential Probability Ratio Test [144].

2.2 Methods for Summarization

Given a tweet set that is relevant to an event, it is not practical for the public to learn about the detailed information by getting through the huge number of messages. A reasonable way is to summarize the main content described by the set of tweets. However, tweets are noisy, redundant and casually written. It is not very easy to select high quality and concise tweets as the summary. We further incorporate the news documents linked by the tweets for the summarization task. Our research works are related to single-document and multi-document summarization. Thus, we review the related literatures from the two perspectives. Then, we review the studies exploring tweets for document summarization. For systematic and comprehensive review of summarization techniques, we recommend survey papers [38, 110].

Summarization has been studied for a long time in both IR and NLP [40, 97]. The basic requirements for summarization, whatever the specific scenario is, are as follows:

- **Representative:** The summary should cover the core information in the document or set of documents;
- **Nonredundant:** A good summary should be concise and does not contain redundant information;
- **Length:** The length of summary should be within a limit.

The reviewed researches on summarization mainly aim to solve the above requirements and we do not emphasize them one by one.

2.2.1 Single Document Summarization

Summarization in general could be categorized into extractive-based and abstractive-based. This classification depends on whether sentences in resultant summary are ex-

tracted from the source document or generated based on information in the document. We will review works of both types.

Extractive Methods. Graph-based model for summarization is widely adopted to explore the statistical information within document. Usually sentences are vertices of the graph, and edge exists when similarity of two sentences exceeds a threshold or some other preferred relationships exists between the two sentences. Weights could be added on either edges or nodes. LexRank [43] is a well-known unsupervised approach which ranks sentences with a degree-based graph model. Instead of applying sentence graph, Parveen and Strube [115, 116] construct a bipartite graph consisting of sentence and entity vertices, which is an extension of [49]. HITS [68] is conducted on the graph to generate sentence ranking. In addition, they take text local coherence into consideration which is estimated as in [49]. Besides, sentence salience and non-redundancy are also important for a good summary. They formulate coherence, importance and redundancy into the general optimization framework and generate the final summary. Later, they improve this work and apply the solution on a weighted and dense topical graph [114].

Another type of unsupervised approach is based on reconstruction mechanism. The summary should be able to recover most of the important information of the document through some reconstruction methods. He *et al.* [58] propose to reconstruct each news sentence by a non-negative linear combination of the selected summary sentences and to minimize the reconstruction error.

Recently, deep learning methods are also applied to the summarization task. Cheng and Lapata [31] employ RNN encoder-decoder process to extract sentences and words. Ren *et al.* [122] rank news sentences by sentence salience scores which are estimated through both hand-crafted features and contextual information explored by an attentive neural model. Nallapati *et al.* [106] design a classifier to select summary sentences based on the features generated from a neural network framework. An obstacle for supervised

methods is the lack of reference summaries for training. Especially for deep learning based methods, large amount of training data is necessary. Contrary to supervised models, unsupervised models are not restricted by the shortage of training data.

Many approaches try to deal with redundancy issue after they obtain the text unit’s relevance. Maximum Marginal Relevance (MMR) [24] provides a greedy re-ranking strategy which considers the trade-off between redundancy and salience. However, this approach cannot produce the global optimal solution. Thus, McDonald [103] proposes global inference algorithms to resolve the solution. Both dynamic programming algorithm could produce near optimal solution as generated by Integer Linear Programming (ILP) algorithm. One drawback for ILP-based methods is the poor scalability. Gillick and Favre [48] and Li *et al.* [77] explore a finer granularity than sentence, like concept level (*i.e.*, n-gram) or sub-sentence level and inference by ILP (*i.e.*, budgeted maximum coverage problem) as well. To solve the ILP problem efficiently, they prune the concepts or sub-sentences to reduce solution space. Boudin *et al.* [19] find the pruning process would lead to poorer performance and propose an approximation approach which extends the objective function with the corresponding frequencies and can produce near-optimal result. Note that, these re-ranking methods could also be applied on multi-document summarization.

Abstractive Methods. Recently, with the development in language generation, abstractive summarization gained much research interests. One of the first successful applications is [126]. Rush *et al.* apply a neural language model to generate headline sentences. Chopra *et al.* [35] extend this work by replacing the neural language model with an attentive recurrent neural network. Instead of adopting traditional maximum likelihood estimation, Shen *et al.* [136] claim that such method only focuses on word-level errors to maximize the probability of the next correct word, which might hardly correlate well with corpus-level and sentence-level evaluation metrics. Thus, they apply minimum

risk training [135] which introduces evaluation metrics as loss functions and assume that the optimal set of model parameters should minimize the expected loss on the training data. Some recent works take the advantage of attention mechanism for this task [5, 98]. Tan *et al.* [137] combine topic sensitive PageRank [55] with attention model to assign weights for sentences with concern of their relevance to the topic. Li *et al.* [83] apply variational inference to attentive sequence-to-sequence model [34] in order to address the intractable posterior inference for the recurrent latent variables and form decoder with better performance. Both Nallapati *et al.* [107] and See *et al.* [134] incorporate pointer network with attentive sequence-to-sequence network together to deal with rare and unseen words by copying source text for summary generation. In addition, they propose a coverage mechanism to relieve repetition issue of sequence-to-sequence model. Similarly, Paulus *et al.* [118] deal with the repetitive and incoherent issue generated by RNN through convolutional sequence-to-sequence [47] together with a novel intra-temporal attention mechanism [118] which takes into account words generated by decoder. Besides, they incorporate rewards from policy gradient reinforcement learning [123] to reduce exposure bias [121] in supervised training. Wang *et al.* [147] propose to incorporate topic information into the convolutional sequence-to-sequence model. More importantly, they apply self-critical sequence training [123] for optimization which can directly optimize the proposed model with respect to the non-differentiable metric ROUGE [87], which also avoids the exposure bias during inference.

2.2.2 Multi-Document Summarization

Multi-document summarization (MDS) is a challenging task due to information diversity, redundancy, and noise. The problem attracts many research efforts. We review them according to extractive and abstractive methods.

Extractive Methods. Extractive methods select sentences from documents to form the summary. A typical framework is based on graph where sentences are vertices and similarities between sentences are edge weights. For example, TextRank [104] and LexRank [43], both leverage random walk to explore relationships among sentences and then produce sentence ranking. MEAD [120] is a centroid-based method, which ranks sentences by a set of parameters including centroid value, positional value, and first sentence overlap. Besides, redundancy is considered for each sentence with those that have higher ranking scores.

One drawback of the graph-based model is that it cannot perform well when the sentence graph are not well connected. Wan *et al.* [145] claim that the document set is usually composed of a few themes which are represented by sets of sentences and some themes are more important than the others. They apply conditional Markov Random Walk and HITS on each cluster separately. However, their solution only considers statistical information. Semantic or contextual information among sentences is neglected. Following the similar idea, Banerjee *et al.* [6] propose that sentences in the most important document of the set are relevant to sentences in the other documents. Hence they cluster sentences based on those in the most important document. However, the most important document in the set may not always exist. Further, the relationships among documents is not necessarily to be conclusive. On the contrary, there are many kinds of relationships among documents (*e.g.*, similar, complementary or evolutionary). In our solution, we group similar sentences by exploiting correlations among input data.

Some works formulate the summarization task as submodularity problem [88, 130] and resolve it by optimization solution with constraints. Note that, the objective functions should be monotone and submodular. The advantages for this kind of methods are: (i) the applicability to many useful submodular objective functions, (ii) the efficiency of greedy algorithm, (iii) the performance could be guaranteed by mathematical proof.

The constraints could be various, *e.g.*, size constraint [108] for summary length, knapsack constraint [64, 75] for summary coverage, and tree constraint [62, 71, 130] for finer-grained sentence structure. The constraints could also be combined together. Tixier *et al.* [138] adopt the same near-optimal objective function, which is a linear combination of coverage and positively reward diversity, proposed by Lin and Bilmes [88]. But the coverage is a weighted sum of keywords saliences, which is generated by k-core decomposition [7] of word graph. And the diversity reward becomes the keywords redundancy evaluation instead of sentences. Elhamifar and Kaluza [42] propose a unconstrained submodular optimization method to select representative subset for streaming data, which means the data could come incrementally. Additionally, a convex relaxation-based algorithm is also proposed in the same paper. Such kind of methods could be applied on streaming data summarization, like summarizing the ongoing events.

Ye *et al.* [161] formulate summarization as selecting sentences which contain as many salient concepts (*i.e.*, terms) as possible. This is intrinsic a knapsack problem which is NP-hard. To relieve the computational cost, they propose document concept lattice (DCL) to represent sentences which aims to balance the redundancy and diversity of concepts. DCL indexes all existing sentences in the concept space so that the sentences for a good summary can be optimized from promising sub-spaces defined by local topics.

Liu *et al.* [91] adopt the idea of reconstruction. They apply a two-level sparse representation model and reconstruct the document by extracted sentences with different constraints. Similarly, Ma *et al.* [99] try to minimize the reconstruction error of selected sentences with the document set through a neural network. Cao *et al.* [23] apply multi-task learning to train a better sentence representation, which incorporates classification task with summarization. The sentences are ranked according to similarities between representations for sentences and summaries which are learned through both tasks.

Abstractive Methods. Summarizing multiple documents in an abstractive way is even more challenging. The generation should take many inputs into consideration. In contrast

to abstractive methods for single document summarization, the abstractive approaches on multi-document summarization are more like compression and fusion.

Bing *et al.* [15] propose an abstractive multi-document summarization solution. They extract salient noun and verb phrases from constituency tree, then reconstruct sentences with representative phrases via integer linear programming (*i.e.*, ILP). Similarly, Li *et al.* [82, 84] adopt a two-stage fashion. They first estimate sentence and phrase saliences via auto-encoder framework, then apply ILP to find the optimal combination of phrases for abstractive summary generation.

Chali *et al.* [26] apply a two-stage method which includes sentence compression and merging, and summary generation by a submodular maximization. They remove unimportant content from sentences as in [14] and sub-clauses related to the reporting verbs as in [27]. Following [15], they merge sentences with coreferent subject, which is identified by Stanford Coreference Resolution engine [74], but different verbs. They evaluate importance, coverage and non-redundancy properties to generate the summary and formulate them as a budget constrained submodular problem as [88].

2.2.3 Summarization with Tweets

Due to the lack of reference summaries in many summarization problem settings, researchers start to explore tweets linking to the news articles, or other form of social data. Hu *et al.* [60] propose comments-oriented document summarization which incorporates both web document and online comments to construct summary. Similarly, Huang *et al.* [61] also take readers' comments into consideration. A difference is that they group sentences in document and its corresponding comments into clusters based on their claim that the sentences describe different aspects of the document. The summary sentences are selected by considering the importances of aspects. Cao *et al.* [22] leverage tweets to guide the summary construction. They select summary sentences from both news and

high quality tweets to make them cover the key points presented in tweets. In addition, Yulianti *et al.* [163] make use of tweets to guide summarization in a query-biased fashion. They take document related tweets as queries, and select the most novel sentences from document to be included in summary.

Besides using tweets to guide the summarization process, many works adopt tweets as auxiliary information. Wei *et al.* [152] explore hand-crafted features from both news and relevant tweets to rank news sentences in a supervised manner. In their following work, they [154] apply an unsupervised dependency-tree based sentence compression approach for summarization. Tweet information is incorporated as edge weight in the tree. Li *et al.* [81] propose a sparse-coding-based method to generate summaries that are not only salient to the set of documents but also readers' comments. However, the produced summaries mainly rely on readers' comments. Later, they propose a deep learning method [80] that not only joint models the focus of document set and readers' comments, but also makes it insensitive to the diverse comments. This part is similar to our work. The fundamental difference is that we also deal with sequential contextual information within single documents and cluster sentences into aspects to tackle the relationships among documents.

Some other works summarize sets of tweets directly. Liu *et al.* [93] select representative tweets from the tweet set with the same topic by considering the readability, social influence (*i.e.*, the number of times a tweet is retweeted) and user diversity. However, it is difficult to quantitatively evaluate the readability of a tweet. He *et al.* [57] explore deeper on social-temporal information in Twitter. A tweet is likely to be important if it is posted or retweeted by authoritative users, and they formulate this as social information. The temporal heat signals of topic-related tweets is considered as temporal information. Both features above and tweet content information together decide the importance of a tweet. However, this method only considers statistical information of a tweet. Later, He

and Duan [56] propose a sparse reconstruction framework as [91, 99] by jointly considering coverage and sparsity.

Chapter 3

Collecting Event-related Tweets from Twitter Stream with Active Learning based on Multiple Instance Learning

3.1 Introduction

As stated in chapter 1, many users search or report event-related information on social media in nearly real-time when an event happens [85, 129]. To better integrate and leverage the piecemeal information on social media, we propose to collect clean and complete event-related tweets along with event development in this chapter. In other words, we will collect tweets from Twitter Stream.

Comparing with location-based methods [21, 165], we adopt keyword-based approach as in [29, 150] to collect candidate event-related tweets. Next, we process further classification on identifying event relevant tweets and remove noise. There are two challenges in our problem. One is to generate keywords that are adaptive to event evolvement and can cover as many event-related tweets as possible. The other is to identify event relevant tweets with as little human input as possible. We will resolve the challenges with an automatic keyword extraction module and event related tweets identification module as detailed in Section 3.2. Then, we introduce the collected dataset of two real-life events

and conduct model evaluation, comparison and analysis with them in Section 3.3. In the end, Section 3.4 concludes this work.

3.2 The Proposed Method: ALMIK

Our proposed solution, named *ALMIK* (for Active Learning based on Multiple-Instance learning with Keyword extraction), is illustrated in Figure 3.1. The method works in a batch mode with a predefined time-window size. A time window can be a day or several hours depending on the number of tweets collected. The batch mode processing enables us to derive keywords from tweets collected in the past time window, and to expand keywords for tweet collection in subsequent time windows.

The process starts with an initial set of keywords. The seed keywords could be generated from event-related documents or provided by users. In our implementation, we derive event-related words from relevant Wikipedia entries (detailed in Experiment Section). With seed keywords supplying to Twitter API¹, we start to receive the first batch of tweets.

With the first batch of tweets, we now need to address two problems. The first is to identify event-related tweets from the collection because not all tweets containing keywords are event-related. The second is to update keywords making them to be adaptive to event evolvement, because initial keywords extracted from Wikipedia are relatively general and cannot describe specific characteristics of the event of interest. To tackle the first problem, we apply active learning based on multiple-instance learning which could maintain high precision with less human input. For the second problem, we rank keywords by investigating three properties of candidate words and select the most representative ones to expand the keyword set. This two-stage process repeats in the subsequent time windows, and stops when only few relevant tweets can be collected in a time window,

¹Either Streaming API <https://dev.twitter.com/streaming/overview> or Rest API <https://dev.twitter.com/rest/public> can be used. In our experiments, we used a simulated Twitter Stream.

Figure 3.1: Overview of the proposed ALMIK. The main techniques for event-related tweets identification and event-related keywords extraction are further illustrated on its right. Wikipedia is only utilized to derive the seed keywords.

or the collected tweets satisfy user’s need. It is natural that the number of tweets on a specific topic reduces significantly after a short time period.

3.2.1 Event-related Tweets Identification

Because of reporting on the same event and “retweet” action, many event-related tweets are similar to each other or near-duplicate. This phenomenon can help reduce manual effort in identifying event-related tweets by grouping similar tweets into clusters and applying multiple-instance learning to label these clusters.

Tweet clustering. We combine MinHash clustering and incremental clustering to group tweets collected by keywords TC within a time window into clusters as shown in Algorithm 1. It has been reported that MinHash algorithm is effective in grouping near-duplicate tweets [133]. However, tweets with different minimum hash value could also be similar. To avoid similar tweets being clustered into different groups, we combine MinHash clustering and incremental clustering to get reasonable tweet clusters. Incremental clustering is chosen here because it is efficient and suitable for unknown data distribution.

Algorithm 1: MinHash Incremental Clustering

input : Tweets TC collected in a time window

output: Clusters of tweets M

```
1  $M \leftarrow \emptyset$  ;
2 Initialize the map  $TM \langle hashValue, clusterID \rangle$  ;
3  $hashValue \leftarrow 0$  ;
4 foreach  $t \in TC$  do
5    $hashValue \leftarrow \text{MinHash}(t)$  ;
6   if  $TM$  contains  $hashValue$  then
7     | Update  $M$ : assign  $t$  to  $clusterID = TM.get(hashvalue)$  ;
8   else
9     | Update  $M$ :  $clusterID = \text{IncrementalClustering}(t)$  ;
10    | Put  $hashValue$  and  $clusterID$  of  $t$  into  $TM$  ;
11 return  $M$ 
```

Given a coming tweet t , we first calculate the minimum hash values for its uni-gram, bi-gram and tri-gram representations. The hash value for tweet t is the concatenation of the three minimum hash values. If there exists a tweet which has the same hash value, then t is assigned to the same cluster as the existing tweet (*i.e.*, MinHash clustering). Otherwise, t is clustered based on its cosine similarity with existing clusters, as in typical incremental clustering. The similarity is computed by the bag-of-words features with TFIDF weighting. If the similarity between t and any of the clusters is smaller than a threshold², t is assigned to a new cluster; otherwise, t is assigned to the most similar cluster.

Active learning with multiple-instance learning. Now we identify event-related tweets through active learning. Our situation fits pool-based active learning, where a pool of unlabeled data U is available. An active learner $\mathcal{L}$ consists of three components (f, q, X) , where f is a decision function to label instances (*i.e.*, tweets in our setting), q is a query strategy for selecting instances (*i.e.*, tweet clusters in our setting) for human annotations, and X denotes the annotated instances.

²We empirically set the threshold as 0.3 which gives relative good results after several tries.

Algorithm 2: Support Vector Machine Active Learning

input : Tweet clusters M , iteration number L
output: Earthquake-related clusters M_P and non-related M_N

- 1 Select at least two clusters from M . One is earthquake-related, the other is not.
Add into training data set Tr the rest of tweet clusters in M as testing data Te ;
- 2 **for** $i < L$ **do**
- 3 Model $SVM(Tr) \leftarrow$ Train SVM using Tr ;
- 4 Classify Te on $SVM(Tr)$;
- 5 Select cluster q with minimum distance to decision boundary of $SVM(Tr)$;
- 6 Request label of q and add q into Tr ;
- 7 $M_P, M_N \leftarrow$ Classify Te on $SVM(Tr)$;
- 8 Add positive/negative labeled data in Tr into M_P and M_N ;
- 9 **return** M_P, M_N

We adopt Support Vector Machine (SVM) as the decision function as in [139], $f : \mathcal{X} \rightarrow \{-1, 1\}$.³ SVM has shown to be effective in many text classification tasks. It predicts label for a test instance based on a hyperplane, defined by support vectors. The difficulty of classifying a test instance can be estimated by the distance between an instance and hyperplane. We adopt the uncertainty sampling method [76] as query strategy q . That is, the most difficult instance(s) to classify from U based on current classifier will be selected for human annotation. Note that, to minimize human effort, we annotate tweet clusters instead of individual tweets as in the setting of multiple-instance learning [4] as shown in Algorithm 2. Therefore, it is to find the cluster m_q whose instances have the smallest average distance to SVM’s hyperplane.

$$q = \arg \min_{m_q \subseteq U} \frac{1}{|m_q|} \sum_{j \in m_q} d(j) \quad (3.1)$$

where $d(j)$ is the distance of instance j to SVM’s decision boundary.

The multiple-instance learning labeling strategy is as follows. When there is at least one positive instance in the cluster, it is labeled as positive. Negative label is assigned when all instances in a cluster are negative. Therefore, a positive cluster may contain

³We adopt default parameter settings for SVM which give good results in our experiments.

negative instances. Instances in a positive cluster would be reclassified by SVM after the model is trained in each iteration, similar to [3]. If all instances in a positive cluster labeled by a human annotator are predicted as negative by SVM, then the instance which has the smallest d will be relabeled as positive, because there should be at least one positive instance in an annotated positive cluster.

To start the process, we manually label at least two clusters (one positive cluster and one negative cluster) and use them as initial training examples. The two initial clusters can be selected in many ways. Here we select the tweet clusters that contain the most, and the least, relevant keywords respectively.⁴ There is a great probability that the two selected clusters to be labeled as positive and negative. After classifier is trained, it would reclassify labeled training data because of the labeling strategy described above. SVM would also classify all unlabeled instances. The most uncertain cluster, *i.e.*, the cluster that is closest to the hyperplane, will be selected for human annotation, and be added into training examples. The process repeats a few rounds and event-related tweets are those predicted as positive by the last round of classifier.

3.2.2 Keyword Extraction

We use P and N to denote event-related tweet set and irrelevant tweet set predicted by SVM in a time window. With P and N , we could generate event representative keywords which should have the following properties:

- *Relevance*: The word should be relevant to the event of interest;
- *Coverage*: The word should have high coverage which leads to more event-related tweets to be collected;
- *Evolution*: The word frequency should increase along the event development.

⁴The most and the least relevant keywords could be chosen from initial and the following keywords ranking list.

Relevance is the most important property for event representative keywords. For relevance, we adopt relative entropy to measure the difference between the distribution of each word w in P and that in N as follows:

$$RelEnt(w) = p(w, P) \times \log \left(\frac{p(w, P)}{p(w, N)} \right) \quad (3.2)$$

A word w with high relative entropy means the word has larger probability in P and smaller probability in N . Such words should be more related to the target event and could be potential representative keywords. Hence, we prefer words with high $RelEnt(w)$ values.

When an event happens, users would talk about different aspects of the event at the same time. Assume current keywords have not covered all the aspects of target event. The potential new keywords should be able to introduce tweets of aspect that are not covered by existing keywords. We therefore estimate the coverage property of word w as the percentage of tweets containing w in P over that in the whole Twitter Stream (through API) $\mathcal{T}$ during current time window:

$$Coverage(w) = \frac{\# \text{tweets in } P \text{ containing } w}{\# \text{tweets in } \mathcal{T} \text{ containing } w} \quad (3.3)$$

The coverage reflects the percentage of tweets containing w collected by current keyword set and we prefer words with lower coverage value. However, $Coverage(w)$ is not the smaller the better. If a word rarely appears in event-related tweets, but often appears in irrelevant tweets, the word has smaller coverage but is not related to the event. For this reason, we combine both relevance and coverage together, and construct a rank score for each word w in P , shown below. The relative entropy value and exponential function will penalize large coverage value. We then select top ranked words and add them into keyword set K .

$$Rank(w) = RelEnt(w) * \exp\left(-\frac{1}{Coverage(w)}\right) \quad (3.4)$$

Event evolves over time. Take earthquake as an example, in the first few days, people may talk more on rescuing victims and warning of aftershocks, while donation and rebuilding would be the main topics in later days. Thus representative keywords would change as well. To model word trend, we adopt a time series model, Autoregressive (AR) model [52]. It models a type of time-varying random process. AR specifies that the output variable x depends on its own value in the past. Specifically, AR(1) predicts the variable value in the next time window x_{t+1} based on its value x_t in current time window:

$$x_{t+1} = \alpha x_t + \varepsilon_{t+1} \quad (3.5)$$

where α is a parameter which could be inferred from historical data, and ε_t is white noise following Gaussian distribution. The details of learning the parameters can be found in [52]. In our case, we use a word’s frequency in P obtained in current time window to predict its frequency in the next time window. We select words with uptrend, and rank them according to the increasing percentage of frequency. Top N words will be added into keyword set K as well. The updated keyword set is used to collect tweets from Twitter Stream in the next time window. Since parameter estimation needs to acquire historical data, the AR component will start from the second time window.

3.3 Experiments

3.3.1 Dataset

In our experiments, we simulate Twitter Stream by using a large static dataset collected by Internet Archive⁵. The tweets are collected by Spritzer⁶, which is a sampled Twitter Streaming API and returns around 1% of the full Twitter Stream. Since the labeling effort is of great amount, we conduct experiments on two real-world events.

⁵<https://archive.org/details/twitterstream>

⁶<https://dev.twitter.com/streaming/overview>

Table 3.1: The total number of tweets $\mathcal{T}$ and the number of positive tweets $\mathcal{T}_{\mathcal{P}}$ within each time window in the simulated Twitter Stream.

EQ	$\mathcal{T}$	$\mathcal{T}_{\mathcal{P}}$	Flood	$\mathcal{T}$	$\mathcal{T}_{\mathcal{P}}$
t_1	79,726	280	d_1	1,613,579	193
t_2	244,529	513			
t_3	364,964	373			
t_4	499,713	1,153	d_2	1,618,825	236
t_5	246,540	280			
t_6	342,098	168	d_3	1,783,940	116

- *ChileEarthquake*: A magnitude 8.2 earthquake struck offshore of Iquique, Chile on 1 April 2014;
- *IndianFlood*: The 2015 South Indian floods caused by heavy rainfall from November to December.

We use *EQ* and *Flood* to denote the two events. Tweets are collected from the first day of *EQ*. For *Flood*, we take the data of the most severe three days because the rainfall takes some time to become flood.

We collect tweets in Twitter Stream within time windows as in Table 3.1. Specifically, we set 8-hour as time window size for *EQ* and collect data in the first two days. We take 1 day as time window size for *Flood* due to relatively small number of tweets for this event.

During each time window, we collect tweets with keyword set K . As long as a tweet contains one of the keywords in K either as a word or as a substring, it would be collected. For example, with a keyword “Chile”, tweets containing hashtag #chileearthquake will be collected as well.

For evaluation purpose, we construct ground truth data. We define event-related tweets according to [111], which provides several most popular aspects of a disaster

queried by users. Since the whole dataset is too big for manual annotation, we choose to label two subsets of tweets within each time window for evaluation⁷ as follows:

- Tweets that are found to be event-related by the proposed method and any of the baseline methods in each time window;
- Randomly collected 1,000 tweets without matching any keywords in K from Twitter Stream, which are a sample set of stream data that can be accessed at that time.

The remaining tweets are considered irrelevant. In this sense, the Recall value reported in our experiments is an estimation. Nevertheless, it is infeasible to obtain the true recall value due to the large number of tweets.

The tweets TC collected through keyword search in each time window are cleaned as follows. First, non-English tweets are removed by a language detection tool.⁸ Afterwards, URLs, punctuation marks (except @ and # which denote accounts and hashtags), special symbols, and stop words are removed. The proposed method and baseline methods are applied on the cleaned tweets.

3.3.2 Initial Keyword Generation

A comprehensive set of seed keywords is essential for the high-recall requirement in tweets collection. Here, we leverage knowledge from Wikipedia to construct a set of seed keywords. Note that the seed keywords could be generated from any domain-specific documents or even given by domain experts. We adopt Wikipedia to avoid the bias of personal knowledge.

Consider Chile earthquake as the event of interest. We consider all Wikipedia entries with the word *earthquake* appearing in their titles as earthquake-related.⁹ After collecting

⁷The data is annotated by the authors and another two volunteers.

⁸<https://github.com/optimaize/language-detector>

⁹We do not include Wikipedia entries that describe the events used in our evaluation.

all such entries, we perform Part-of-Speech (POS) tagging on the content of these documents and select nouns as candidate event-related keywords, assuming that keywords are nouns. To find out which words are most related to the target event *earthquake*, we adopt Mutual Information (MI) as a measure of the association strength between the word *earthquake* and each of the nouns in the candidate set, shown in Equation 3.6.

$$I(e; w) = p(e, w) \log \frac{p(e, w)}{p(e)p(w)} = \frac{n(e, w)}{n(W)} \log \frac{n(e, w)n(W)}{n(e)n(w)} \quad (3.6)$$

In this equation, e denotes the word *earthquake*, $w \in C$ denotes a word w (or a noun) from the candidate set C . $n(e)$ and $n(w)$ is the number of Wikipedia entries containing e and w , and $n(e, w)$ is the number of Wikipedia entries containing both e and w . $n(W)$ is the event-related Wikipedia entries number. The top-ranked M words are selected and added into the K (in our experiments, $M = 10$). Besides, we also add the event name and location into K , *e.g.*, “earthquake” and “Chile”. With these initial seed keywords, we use Twitter Stream API to collect all tweets that contain at least one word in K . We denote the set of collected tweets as candidate tweet set TC , $TC = \bigcup_{w \in K} T(w)$, where $T(w)$ is the set of tweets containing keyword w . The same process is used to obtain the initial seed keywords for event *Flood*. With the seed keywords, we collect the first batch of candidate tweet TC .

3.3.3 Baseline Methods

We compare the proposed method ALMIK with state-of-the-art methods: ADP [100], LDA [17], Dictionary Learning (DL) [63] and Targeted Topic Modeling (TTM) [149]. All of these methods could generate both event-related keywords and identify event-related tweets. We make all methods iterate over the time windows similar as ours. In other words, the two components (“Event-related Tweets Identification” and “Event-related Keywords Extraction”) shown in Figure 3.1 are replaced by the baseline methods.

(a) Precision of 7 methods on *EQ* and *Flood* data

(b) Recall of 7 methods on *EQ* and *Flood* data

(c) F1 of 7 methods on *EQ* and *Flood* data

Figure 3.2: Performance comparison.

The methods *LDA*, *DL*, and *TTM* need a predefined number of topics as input, and return clusters of tweets belonging to different topics and their corresponding keywords. We take the top ranked keywords of the most relevant topic(s) from *LDA* and *DL*, and

top ranked keywords from every cluster of *TTM* as event-related keywords and add them into keyword set K to collect tweets in the next time window. If more than one cluster is selected as event-related in these baseline methods, then the same number (*i.e.*, required keyword number over selected cluster number) of keywords are selected from each event-related cluster. The total number of selected keywords is the same for all the methods in each time window.

LDA Each tweet in TC is an input document for *LDA*, and the same for *DL* and *TTM*. We empirically try various topic numbers and select the one with the best performance on F_1 . For topic cluster, we manually check the returned keywords and tweets. If there is at least one keyword related to the target event, we would check all the tweets in the cluster, then we select the cluster(s) combination with highest F_1 . The manually checking process is the same for *DL*.

DL The objective function for *DL* is [63]:

$$l(X, D_T) = \min_{\Psi \in \mathbb{R}^{k \times m}} \frac{1}{2} \|X - D_T \Psi\|_2^2 + \lambda \|\Psi\|_1$$

where X is word $\times$ document matrix, D_T is word $\times$ topic matrix, and Ψ is topic $\times$ document matrix. We assign each word to the topic with largest value in row of D_T . The assignment of documents to topics is the same. When selecting top ranked words, we rank all the words belonging to the same topic according to their values in the corresponding column in D_T .

TTM The method requires input documents from one domain but of different aspects [149]. Because we cannot assign all the tweets into a specific domain, we consider all tweets in TC from one domain. The method also needs some predefined aspect-specific keywords. Their model does not require the complete set of keywords to describe the target aspect, so we select the most representative keywords.¹⁰ Note that, experiments

¹⁰We set “earthquake”, “quake”, “tsunami”, “magnitude” and “Chile” for *EQ*, and “flood”, “flooding”, “floods” and “Indian” for *Flood*.

in the original paper only show the quality of keywords but not the results of the aspect-related documents. In their model, a status variable $r \in \{0, 1\}$ indicates a document’s relevance to the targeted aspect, where $r = 1$ means the document is relevant to the aspect and $r = 0$ otherwise. Thus we identify event-related tweets by variable r .

ADP This method requires a static well-defined set of queries Q on given topic [100]. Tweets containing words from query set are considered as positive training data P . Negative training tweets N are randomly selected from the rest of tweets. To avoid potential positive tweets mixed within N , a TFIDF score is adopted to rank words in P and top k are selected as event representative keywords E . Tweets in N containing E are removed. A classifier trained by P and N is used to classify tweets in the next time window, and the classifier is periodically retrained by new tweets.¹¹

We do not use the seed keywords derived from Wikipedia as query set for *TTM* and *ADP*, because of the poor experimental results caused by the general words like “people”, “area”. The presented query set for the two baselines are those achieving relatively good performance after many tries. Besides the above baselines, we also compare with two active learning methods, namely *AL-s* and *AL-p*, to measure the human labeling effort. For fair comparison, *AL-s* and *AL-p* also adopt SVM and uncertainty sampling, but the labeling strategy is different. The keyword extraction component in *AL-s* and *AL-p* are the same as *ALMIK*. Thus *AL-s* and *AL-p* could be considered as variants of our proposed method.

AL-s Active learning is applied to single tweet and the most uncertain tweet is selected to query label. By comparing with *AL-s*, we would evaluate the effectiveness of clustering similar tweets and the multiple-instance learning labeling strategy.

¹¹We set the same queries as in *TTM*. We consider tweets in the first 6 hours of each time window for *EQ* as training data candidates, and tweets in the rest 2 hours for test. Since event-related tweets for *Flood* are extremely unevenly distributed over time, we extract training data from time period covering the first 80% positive tweets in each time window, and the left as testing data.

Figure 3.3: F_1 comparison of AL_s, AL_p and ALMI over different number of manually labeled clusters on EQ data, from t_1 to t_6 . Since tweets in one cluster are nearly duplicate or similar with each other, the labeling effort for one cluster is similar to that of one tweet.

AL_p Active learning is applied to tweet clusters but each cluster is represented by a pseudo document which is the concatenation of all tweets in this cluster. The pseudo document could help relieve data representation sparsity issue. The labels for tweets in the most uncertain cluster are the same and will not be changed. When half of tweets in a cluster are positive, then the cluster is labeled as positive. Otherwise, the cluster is negative. Here the effectiveness of multiple-instance learning labeling strategy would be evaluated.

3.3.4 Evaluation on Event-related Tweets

First, we evaluate event-related tweets identification. Here, we call our model as *ALMI*, without keyword extraction. We compare the proposed method and baselines with Precision, Recall and F_1 on both datasets. Note that the metrics values for *ALMI*, *AL-s*, *AL-p* and *ADP* are calculated on test data.

Analysis of models performances. Figure 3.2 reports the results of our method and baseline methods. We have the following observations. First, the proposed *ALMI* outperforms all the baselines on both datasets, throughout all time windows, based on F_1 . Specifically, we could outperform the best F_1 results of *DL*, *LDA*, *TTM* and *ADP* by up to 13% on *EQ* and 16% on *Flood*, and those of *AL-p* and *AL-s* by up to 10% on *EQ* and 6% on *Flood*. A clearer comparison shows that the proposed *ALMI* outperforms all baselines on both Precision and Recall for most of time. The high values on three measures suggest that the proposed method could collect tweets with high precision and high recall. Second, *DL* outperforms *TTM* and *LDA* on both *EQ* and *Flood* based on F_1 . Third, the *ADP* performs rather unstable on both data. The reason is that not all the tweets containing query keywords are event-related. Training a classifier with such tweets could certainly introduce bias by not including other event-related tweets or containing too much irrelevant tweets. Fourth, *AL-p* and *AL-s* performance similarly. The reason is that the tweet cluster is to group same or near-duplicate tweets together. As a result, the representation of pseudo document and single tweet could also be similar.

Differences between datasets. According to experimental results in Figure 3.2, we observe that the average performance on *EQ* is better than *Flood*. In fact, the *Flood* dataset is much noisier than *EQ* dataset. A detailed look at the tweets in *Flood* and *EQ* reveals that the tweets relevant to flood are not very correlated with each other. Most of them state different aspects of the flood event and can hardly link them together. Another

Figure 3.4: F_1 comparison of AL_s , AL_p and $ALMI$ over different number of manually labeled clusters on *Flood* data from *day*₁ to *day*₃.

reason is that the Indian flood may not be as influential as Chile earthquake, and people who are talking about flood event are mostly local citizens. The tweet content revealing local information is different from one region to another. Thus, tweets from different affected areas are hardly to be linked and identified. Therefore, by adding location and semantic information might help on identifying event-related tweets. Even though, the proposed *ALMI* can still outperform baselines on *Flood* dataset.

Labeling effort. Recall that we group near-duplicate or similar tweets in clusters to reduce manual annotations. As shown in Table 3.4, the number of clusters are far fewer than the number of tweets. Since tweets in one cluster are almost the same, the annotation effort for one cluster is similar to that of one tweet. Thus, by clustering we reduce significant amount of unnecessary human annotations. By adopting multiple-instance learning based active learning, we further reduce human annotation and retain relative good results. Figures 3.3 and 3.4 report the changes of F_1 with different number of man-

ually labeled clusters in each time window on both datasets. All the three methods could achieve fairly high and stable F_1 with about 30 labeled clusters. Our proposed *ALMI* could get higher F_1 than AL_s and AL_p with the same number of human labels most of time. In other words, *ALMI* requires less human labeling effort to achieve similar F_1 score as AL_s and AL_p most of time. Thus, grouping similar tweets together and adopting multiple-instance learning based active learning do help improve performance even with less human effort than normal active learning.

Usually, for topic model-based methods, top k keywords are checked to determine event-related clusters. However, we find the results by doing so are poor, regardless the number of clusters we have tried. Sometimes event representative keywords are mixed together with irrelevant words. We cannot simply decide the cluster to be event-related or not. Under this condition, human annotations for tweets in clusters is needed. The number of tweets in candidate clusters are more than those required annotations by *ALMI*. In this sense, the proposed method requires much less human effort compared with baseline methods *LDA* and *DL*.

3.3.5 Evaluation on Event-related Keywords

The second stage of our proposed method is to extract event-related keywords. The quality of keywords determines the recall of event-related tweets. We show keyword examples generated by different methods in Tables 3.2 and 3.3. For *EQ*, we extract 20 keywords in each time window. The first 15 keywords selected by AL_p , AL_s and *ALMI* come from ranking method defined by Equation 3.4 and the last 5 keywords come from AR(1) model. For the other three baseline models, all the 20 keywords come from topics related to the event with equal chance. Similar for *Flood*, but we extract 10 keywords in total due to the small number of relevant tweets. The first 5 keywords come from ranking and the left from AR(1). Note that, the number of keywords in some baseline

Table 3.2: Example keywords extracted on EQ during t_3 . Keywords in bold are **irrelevant** to the event. Same as follows.

Methods	Keywords
Seed	damage, magnitude, people, area, epicenter, building, time, fault, region, utc, earthquake, chile
DL	magnitude(67), panama(41), earthquake(200), hits(28), 6.2(20), chile(258), 8.2(57), tsunami(38), coast(46), quake(56), download (11), prison(15), female(17), #prayforchile(62), @matthunter123(15), @kennyhamilton(9), hope (22), #chileearthquake(18)
LDA	chile(163), earthquake(142), tsunami(76), quake(66), magnitude(67), hits(28), panama(41), coast(44), death(24), massive(22), hit(19), japan(20), female(18), ice (11), damage(20), cream (12), powerful(15), prison(15), northern(17)
TTM	earthquake(214), chile(282), #prayforchile(37), quake(91), panama(37), hits(28), focus (16), tsunami(46), today (23), following (30), 6.2(20), coast(40), death(19), magnitude(64)
ADP	earthquake(125), chile(116), magnitude(60), 8.2(42), #prayforchile(19), tsunami(44), panama(32), coast(31), following (227), hits(116), 6.2(19), chilean(10), usgs(10), #earthquake(13), aftershocks(13), toll(10), californians(5), earthquakes(6), flee(28), damage(29)
AL-p	earthquake(161), magnitude(70), quake(81), chile(166), 8.2(52), tsunami(63), usgs(25), californians(16), panama(39), aftershocks(18), 6.2(20), #prayforchile(46), #earthquake(21), flee(15), inmates(16); earthquake(161), tsunami(63), chile(166), 8.2(52), #quake(4)
AL-s	earthquake(150), magnitude(71), quake(81), chile(143), #prayforchile(41), 8.2(51), usgs(26), panama(37), #earthquake(23), tsunami(53), 6.2(20), californians(16), aftershock(18), inmates(16), flee(15); tsunami(51), #prayforchile(57), strikes(12), #chile(15), president(1)
ALMIK	earthquake(168), chile(183), quake(87), magnitude(68), #prayforchile(55), 8.2(52), tsunami(66), panama(41), usgs(26), 6.2(20), californians(16), aftershocks(18), 5.8(15), #chileearthquake(18), inmates(16); tsunami(66), #prayforchile(55), strikes(12), president(1), declares(1)

methods listed in the tables are fewer than the predefined number. That is because the same keyword appears in different clusters and we avoid duplicates. The keywords

Table 3.3: Example keywords on *Flood* during *day₂*.

Methods	Keywords
Seed	flood, Indian, floods, rainfall, people, damage, water, displace, part, area, land, river
DL	#stopscientology (50), november(57), december(25), #weather (23), flooding(52), superb(42), craze(42), flood(3), great (1), partly (1)
LDA	river(13), blocked(14), flood(23), sam (3), shows (3), owblfn (13), ahmedabad(13), stopscientology (23), fatehbaad(29)
TTM	@karunainsan12 (18), flood(42), flooding(53), indian(30), flooded(26), haryana(13), explanation (14), superb(42), near (15), blocked(14)
ADP	indianapolis (24), #othercityofficials (33), area (408), ikea (83), indiana (26), flooded(24), plans (536), store (759), bloodflood(17), ii (301)
AL _p	#stopscientology (28), flooding(53), fatehbaad(29), #msgonthetopin-haryana (29), bahnhof(12), 12 (3), superb(23), craze(23), rain(6), tamil(1)
AL _s	#stopscientology (28), flooding(53), fatehbaad(29), #msgonthetopin-haryana (29), #banscientology (19); flooding(42), superb(23), craze(23), rain(6), tamil(1)
ALMIK	#stopscientology (28), flooding(53), #banscientology (28), #inflood-wetrust (11), #floodaware (5); flooding(53), superb(37), craze(37), rain(6), tamil(1)

are ordered based on their ranking scores. The number following each keyword is word frequency in event-related tweets identified by corresponding methods in current time window. For *ADP*, the word frequency is that in training data from where the list of words are generated. Note that, except *ADP* and *TTM*, all other methods are provided with the same set of seed keywords.

Tables 3.2 and 3.3 show that all the methods could produce general descriptive words, such as *earthquake*, *Chile*, *magnitude* and *flooding*. The time series method AR(1) could generate more specific information, such as *president*, *declares* for comforting the victims and *Tamil* which indicates the flood affecting location. Word frequency for some keywords generated by AR(1) is not very high. That is because AR(1) predicts the word to have

Table 3.4: Number of tweets $|TC|$ collected with keywords generated by each method on *EQ* and *Flood*. $|TC_P|$ is the number of event-related tweets included in TC . $|CN|$ is the number of tweet clusters. We do not include *ADP*, because the relevant tweets in testing set are not determined by keywords. Information of the first time window is not presented because it is the same for all the methods.

T	DL		LDA		TTM		AL-p		AL-s		ALMI	
	$ TC $	$ TC_P $	$ TC $	$ TC_P $	$ TC $	$ TC_P $	$ TC $	$ TC_P $	$ TC $	$ TC_P $	$ TC $	$ CN $
t_2	18,131	512	21,379	512	15,968	512	15,438	512	15,554	512	15,391	4,676
t_3	34,175	370	39,383	371	36,055	370	24,043	370	26,054	370	23,897	5,023
t_4	55,799	1,147	82,821	1,150	63,048	1,146	35,226	1,146	38,140	1,146	24,900	5,514
t_5	59,836	273	133,677	277	115,046	276	29,637	275	38,820	273	29,597	6,182
t_6	149,324	168	193,406	168	174,058	168	46,798	168	57,013	168	45,192	7,459
d_2	102,679	208	182,649	208	119,985	208	82,617	222	82,204	222	97,539	10,067
d_3	148,021	114	217,988	114	135,699	97	105,116	114	103,722	114	107,316	11,870

increasing frequency in the next time window. From the results, AR(1)’s output does complement the ranking counterpart.

The baseline methods *DL*, *LDA*, *TTM* and *ADP* may generate keywords that are not so closely related to the events, *e.g.*, *download*, *ice*, *cream*, *following* and *great*, *sam*, *near*, *ikea*. Especially for *ADP*, the generated keywords on *Flood* are mostly irrelevant. The proposed ranking method could extract keywords that are not discovered by the other models, such as *californians* who are affected by the aftershock of Chile earthquake and *inmates* who escape from the prison during the earthquake, and *#infloodwetrust*, *#floodaware* which alert the flood.

Effectiveness of keywords in tweet collection. It’s fairly hard to judge which word is better when different models suggest different keywords and all the keywords are related to the event. Thus, we make a comparison of the number of tweets collected by the keywords versus those which are truly related to the event, as shown in Table 3.4. Heuristically, with equal number of event-related tweets, the fewer number of tweets that are collected the better. This means the keywords generated by the method are more correlated and more concentrated on the event. As we can see that the proposed method could extract smaller candidate dataset with similar positive tweets number than baselines. On the other hand, not all the positive tweets collected by the methods are predicted as event-related. That is, $|TC_p|$ does not reflect the final recall of a method.

3.4 Summary

In this chapter, we propose a method for collecting event-related tweets from Twitter Stream with high-precision and high-recall. Our proposed method consists of keyword extraction and event-related tweet identification. Due to the large volume of tweets, we collect candidate tweets from Twitter Stream by keyword search. We evaluate keywords to be qualified queries by three properties: *relevance*, *coverage* and *evolvement*. We

rank keywords by a ranking function which considers both *relevance* and *coverage*. The *evolvment* property is measured by Autoregressive time series model. Meanwhile, a tweet containing a keyword does not mean the tweet is event-related. Identifying event-related tweets from the collected tweets by keywords becomes another crucial component. To reduce human annotation effort, we group near-duplicate and similar tweets into clusters, and fit them in active learning process. Besides, uncertainty sampling is used as query strategy and multiple-instance learning is adopted as labeling strategy. We conduct experiments on two real-world events and annotate selective parts of data as ground truth for evaluation. Experiment results on both datasets suggest that the proposed method outperforms state-of-the-art methods. The method achieves the goal of high recall and high precision with little human annotation effort.

Chapter 4

Semi-Supervised Event-related Tweet Identification with Dynamic Keyword Generation

4.1 Introduction

The method reported in chapter 3 is a two-stage method which includes keyword extraction and event-related tweets identification. No information interacts between the two modules except the collected tweets containing selected keywords. In addition, we notice that some core keywords of the event may always rank at top over the others. Such keywords do not contribute to collect more event-related tweets. Therefore, we propose an improved model in this chapter to deal with the issues in previous work.

In the last work, the event-related tweets are only identified at sentence level by human annotators and classifier. Thus, we further explore word-level information in this method. We propose a unified framework in which both keyword extraction and tweet identification can communicate with and enhance each other. Specifically, the properties measured by keyword extraction module could be utilized to identify event-related tweets. On the contrary, event-related tweets could be used to update property values of words. Thus, the proposed method becomes semi-supervised and human labeling efforts are further reduced. Besides the single word properties considered in prior study, the word

co-occurrence property is also explored in this chapter. This is further used to evaluate the dynamic changes of keyword graphs over time which is not considered in existing works. Hence, the core keywords could be ranked lower and other trending words could have more chances to rank higher.

Next, we introduce the details of the unified model in Section 4.2. Then, extensive experiments are conducted on three real-life event datasets to evaluate the efficiency of proposed method in Section 4.3. Next, we conclude this method in Section 4.4.

4.2 The Proposed Method: SSEM

Similar as prior work, we define a time window and apply the proposed method on the tweets collected during each time window. Our solution starts with a set of seed keywords which are generated the same as in previous chapter. Using these seed keywords, we collect a set of candidate tweets, from the tweet stream. Due to the retweet action on Twitter, many tweets are nearly duplicate or rather similar with each other. We cluster similar tweets by incremental clustering [28]. Because our method requires human to label some data for the identifier, clustering similar tweets could reduces human effort. The clusters of tweets are then delivered to the proposed semi-supervised expectation maximization identifier, detailed in Section 4.2.1. The identifier leverages both human label and keyword information to infer whether a tweet is event-related. A keyword importance score is defined to quantify the keyword information, reported in Section 4.2.2. To make the expectation maximization identifier more efficient, constrains are proposed to reduce the solution space, discussed in Section 4.2.3. Next, we take changes of words and their neighbors into consideration and construct a keyword ranking function to select the preferred keywords for more event-related tweets collection. The details are presented in Section 4.2.4.

Figure 4.1: The information flow for one iteration is from Step 1 to 5. After getting new training data from Steps 4 and 5, a new iteration would start to retrain the classifier.

4.2.1 Expectation Maximization Identifier

The Expectation Maximization Identifier is illustrated in Figure 4.1. The process leverages both the tweet label information and the word importance information to identify event-related tweets.

Recall that we cluster similar tweets by incremental clustering. Afterwards, we randomly select two clusters to label: one is event-related, *i.e.*, positive, and the other is not, *i.e.*, negative. Many methods could ensure only 2 clusters are needed to check. One easy way is based on the keyword set. We select from tweet clusters that contain the most representative keywords in keyword set, and there is a great possibility for them to be positive. For the negative cluster, we select from the clusters that contain the least representative words in keyword set. The rest tweets are the testing data.

Expectation The two clusters of tweets are used as initial training data for SVM classifier. We could obtain current labels for testing data classified by the temporary classifier,

and the inference function is:

$$\begin{aligned} \min_{\mathbf{w}, b, \xi} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_i \xi_i \\ \text{s.t. } \forall i : y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i, \xi_i \geq 0, y_i \in \{-1, 1\} \end{aligned} \quad (4.1)$$

where $\mathbf{w}$ is the weight vector and b is a scalar value which would be learned. y_i is tweet label and x_i is tweet feature vector. Here we adopt simple bag-of-words weighted by TF-IDF as tweet representation vector. C and ξ_i are parameters learned during training. All the tweets in positive cluster are considered as event-related tweets *Pos*, and analogous to not event-related tweets *Neg*. Note that the labels are temporary and might change after each iteration. Tweets and their labels would be delivered to maximization stage.

Maximization We define a *word importance score* $fs(w)$ and a *sentence importance score* $I(t)$. The word importance score describes the representativeness of the word to the event of interest. The importance score is calculated based on single word properties and property of its co-occurrence words according to candidate tweet labels, to be detailed later. The sentence importance score describes the relatedness of the tweet to the event of interest. It is calculated by the multiplication of importance score for each word in the tweet. The rationale will be detailed in Section 4.2.2. Intuitively, tweets with high sentence importance score are more related to the event. Thus, we maximize the average sentence importance score for event-related tweets by adjusting the tweets labels which determines the word importance score:

$$\begin{aligned} \arg_{k_i} \max \frac{1}{|Pos|} \sum_{t_i \in Pos} I(t_i) - \beta \sum_{t_i \in S} (k_{t_i} - 1)^2 \\ k_{t_i} = \begin{cases} -1, & \text{if } y_i \text{ changes;} \\ 1, & \text{otherwise.} \end{cases} \end{aligned} \quad (4.2)$$

where t_i is a tweet and β is a hyperparameter. k indicates whether the tweet label changes or not, and $\beta(k_i - 1)^2$ is a regularizer which penalizes too many tweets adjusting labels

in case not event-related tweets dominate the sentence importance score. The details of the objective function will be presented next in Section 4.2.2. Note that only tweets belonging to a specific set S could adjust their labels, to be detailed in Section 4.2.3. The expectation and maximization process would iterate for a predefined number of times.

4.2.2 Word Importance Score

The sentence importance score is computed by the multiplication of the word importance scores. Importance score of a word is determined by single word properties and property by co-occurring words.

4.2.2.1 Single Word Properties

Relevance An event representative word should at least be relevant to the event. We adopt relative entropy to depict the relevance of each word to the event of interest:

$$r(w) = p(w, Pos) \cdot \log \frac{p(w, Pos)}{p(w, Neg)} = \frac{|Pos(w)|}{|Pos|} \cdot \log \frac{|Pos(w)||Neg|}{|Pos||Neg(w)|} \quad (4.3)$$

where $|Pos(w)|$ means the number of positive tweets that contain word w , and similarly $|Neg(w)|$. Here, the tweet label determines the word relevance score. The relevance value is scaled to $[0, 1]$.

Coverage Besides, we also prefer words with a relatively high coverage ratio, to avoid excluding event-related tweets. Thus the keyword set could maintain high recall for the event of interest. The coverage ratio is denoted by $\frac{|\mathcal{T}(w)|}{|Pos(w)|}$, where $|\mathcal{T}(w)|$ is the number of tweets that contain word w in the whole tweet stream within the current time window. However, it is not the higher coverage ratio the better. The irrelevant word could have a rather high coverage ratio as well. We prefer words with coverage ratio smaller than a threshold α to make sure most of the tweets containing the word w are positive. We assume this kind of words have higher probability to be event-related. Hence, we define

the coverage score as follows which could scale the coverage ratio and penalize high coverage ratio:

$$c(w) = \begin{cases} e^{(x-\alpha)}, x = \frac{|\mathcal{T}(w)|}{|\text{Pos}(w)|} \leq \alpha \\ e^{(\alpha-x)}, x = \frac{|\mathcal{T}(w)|}{|\text{Pos}(w)|} > \alpha \end{cases} \quad (4.4)$$

The hyper-parameter α is empirically set.

Quality Score The quality of a word is determined by both relevance and coverage. We call it quality score and model it as the multiplication of relevance and coverage:

$$rs(w) = r(w) \cdot c(w) \quad (4.5)$$

4.2.2.2 Word Co-Occurrence Property

Words do not exist independently in text. Their appearances rely on the contextual information. Thus, the word importance level to the event of interest is also related to the relevance extent of the co-occurring words. We assume that words which always co-occur with high quality words are also of great possibility to be of high quality. On the opposite, words often co-occur with poor quality words would be relatively less important to the event. This is like that nodes in a graph are influenced by its neighbor nodes. Therefore, we model words interplay on a word co-occurrence graph.

Each word in a collected tweet set is a node in the graph and is associated with the word quality score as its node weight. An edge exists between two nodes when they co-occur in a tweet. The edge weight is measured by Pointwise Mutual Information (PMI), which depicts the association between the two words:

$$sim(w, v) = \log \frac{p(w, v)}{p(w)p(v)} = \log \frac{|t(w, v)|/|\mathcal{T}|}{|t(w)|/|\mathcal{T}| \cdot |t(v)|/|\mathcal{T}|} \quad (4.6)$$

where t is a tweet and $|t(w, v)|$ is the number of tweets that contain both words w and v . Figure 4.1 is an example of the constructed word co-occurrence graph with weighted nodes and weighted edges.

The word importance score is determined by both single word properties and the word co-occurrence property. And we evaluate the word importance score $fs(w)$ based on its relationship with one-hop neighbors $Nei(w)$ and their quality scores as follows:

$$fs(w) = rs(w) \cdot \frac{1}{|Nei(w)|} \sum_{v \in Nei(w)} sim(w, v) \cdot rs(v) \quad (4.7)$$

The average relationship with neighbor words is to estimate the average quality of neighbors and also to avoid common words with a large number of neighbors dominating the importance score.

4.2.2.3 Sentence Importance Score

Recall that all the candidate tweets contain at least one word in the event-related keyword set. However, not all of such tweets are related to the event because of the following two reasons:

- The keyword contained by the tweet may have multiple meanings and it happens to express the meaning that is irrelevant to the event of interest;
- Tweet is informal and short. Sometimes several uncorrelated things are compressed in one tweet simply to attract more attention.

As a result, we simply adopt the multiplication of word importance scores as sentence importance score I to estimate its coherence and relevance to the event of interest.

$$I(t_i) = \prod_{w_{i,j} \in t_i} fs(w_{i,j}) \quad (4.8)$$

Having all of the definitions of the basic components (*i.e.*, Equations 4.3, 4.4, 4.5, 4.7, 4.8), we can detail how to maximize Equation 4.2 by adjusting tweet labels:

$$\begin{aligned}
r(w) &= \frac{\sum_i \frac{1+q_i}{2} 1(w_{i,j})}{\sum_i \frac{1+q_i}{2}} \log \frac{\sum_i \frac{1-q_i}{2} \sum_i (\frac{1+q_i}{2} 1(w_{i,j}))}{\sum_i \frac{1+q_i}{2} \sum_i (\frac{1-q_i}{2} 1(w_{i,j}))} \\
1(w_{i,j}) &= \begin{cases} 1, & \text{if } w_j \in t_i \\ 0, & \text{if } w_j \notin t_i \end{cases} \\
c(w) &= \begin{cases} e^{(x-\alpha)}, & x = \frac{|\mathcal{T}(w)|}{\sum_i \frac{1+q_i}{2} 1(w_{i,j})} \leq \alpha; \\ e^{(\alpha-x)}, & x = \frac{|\mathcal{T}(w)|}{\sum_i \frac{1+q_i}{2} 1(w_{i,j})} > \alpha. \end{cases} \\
q_i &= y_i \cdot k_i
\end{aligned} \tag{4.9}$$

where y_i is the label for tweet t_i assigned by the classifier and the changing of tweet label is indicated by an indicator variable k_i same as in Equation 4.2. Both the values of $r(w)$ and $c(w)$ are affected by the overall tweet labels.

4.2.3 Determination of Changeable Tweets

Making all available tweets labels to be changeable would lead to high computational cost. Therefore, we only allow labels change for tweets that are most possible to increase sentence importance score. Either changing labels from negative to positive or the opposite could make Equation 4.2 increases. Consider a situation that a tweet is indeed related to the event and is labeled as positive by the temporary classifier, while its sentence importance score is lower than the average value. This would happen when the classifier is of low recall during the iteration process. At this time, by changing the label from positive to negative might lead to higher value for Equation 4.2. However, by doing so, it would generate the wrong label. Therefore, it is not that confident to change labels from positive to negative. On the contrary, if we change a tweet label from negative to

positive and it leads to higher objective value, then the tweet sentence importance score must be above the average value. This means the tweet is coherent and the words are important to the event. Therefore, it is confident to tell that the tweet is related to the event of interest based on the current information. Thus, the negative tweets predicted by the current classifier are changeable. Even so, the changeable tweets are still of a large number. Not all negative tweets deserve the check and we can further reduce the unnecessary traverse. Among *Neg*, tweets that are closer to the hyperplane of SVM [76] and contain top ranked keywords by quality score rs are more likely to be positive and would maximize the objective function in Equation 4.2. Recall that before the whole process, we cluster similar tweets together. Therefore, we consider all the tweets in one cluster to be changeable that satisfy the following conditions:

- All the tweets in the cluster are classified as negative by the classifier;
- At least one of the tweets in the cluster contains top ranked keywords according to rs ;
- The average distance for the cluster of tweets is among top H closest to the hyperplane of the SVM classifier.

The selected cluster(s) of tweets which could maximize the objective function would change labels from negative to positive, and be added into the training data.

Note that we add positive clusters of tweets into the training set from maximization step. To avoid imbalanced training data and also maintain a high precision, we ask human to label the same number of cluster(s) of tweets. If no cluster changes labels in current iteration, we query one cluster for human label. The top cluster(s) with average distance closest to the hyperplane of SVM is selected as query. This is according to uncertainty sampling [76] which is a commonly adopted query strategy in active learning to reduce human effort.

4.2.4 Dynamic Keyword Graph

Event develops along with time and new aspects about the event would appear. Therefore, we would like the query keywords to capture the newly appearing aspects. We hope the top ranked keywords could have some differences from the previous ones, so that the keyword set could be updated effectively. Specifically, we prefer keywords over time with the following properties:

- Words are newly appearing and with high importance scores;
- Words already appeared in the past time window but with increasing importance scores and better quality of neighbors.

Definitely, the newly appearing words having high importance scores with respect to the event of interest, would represent new aspects. Such words should be added into the keyword set. There is another kind of words which appeared before but now has higher importance score. This indicates the aspect depicted by these words attract more attention. However, we do not prefer words which are with high importance scores but have similar neighborhood relationships as those in the past time window. Such words appear in similar sentence environment along with time and would not help to cover more event-related tweets.

Therefore, we need to measure a word’s neighborhood structure similarity along with time and we call it word *configuration similarity*. Eger and Mehler [41] provide us a method to evaluate the semantic changes. At each time window, the similarities between each word and its neighbor words construct a similarity vector. The similarity between the similarity vectors at different time windows reflects the word semantic changes over time.

In our problem setting, the word graph changes over time as illustrated in Figure 4.2:

Figure 4.2: An example of word graph changes over time.

- Words appear and disappear;
- Edges appear and disappear;
- Edge weight changes between neighbor words;
- Quality score changes for single word.

Thus, we can not directly apply Eger and Mehler's [41] method because they assume the words' neighbors are static. In fact, only top ranked words by word importance score f_s are in consideration of the dynamic change, *i.e.*, top N , which are the core candidate representative keywords for the event of interest. To make the similarity vector comparable over two contiguous time windows, we fix a set of neighbors for each word, *e.g.*, the keyword set. Besides the set of fixed neighbors, there are other neighbors for each word as well. To measure the co-occurred extent of the fixed set of neighbors over all the neighbors, we add the ratio $\frac{\sum_{k \in KeywordSet} sim(w, k)}{\sum_{v \in Nei(w)} sim(w, v)}$ in the similarity vector, illustrated in the last column of Table 4.1.

According to Table 4.1, the configuration similarity could be calculated by cosine similarity of the two similarity vectors:

$$sim(w_{t-1}, w_t) = cosine(s(w)_{t-1}, s(w)_t). \quad (4.10)$$

Table 4.1: Example of word’s configuration similarity. $sim(w, v)$ and $sim(w, k)$ are calculated according to Equation 4.6. If there is no edge between word w and k , the edge weight $sim(w, k) = 0$. m is the number of words in keyword set.

Feature	k_1	k_2	$\dots$	k_m	Ratio
$s(w)_{t-1}$	$sim(w, k_1)_{t-1}$	$sim(w, k_2)_{t-1}$	$\dots$	$sim(w, k_m)_{t-1}$	r_{t-1}
$s(w)_t$	$sim(w, k_1)_t$	$sim(w, k_2)_t$	$\dots$	$sim(w, k_m)_t$	r_t

For newly appearing words, the configuration similarity value is 0.

Based on the preference of keywords selection, we propose a re-rank function for top ranked words by word importance score fs , to take word changes over time into consideration:

$$rank_d(w) = (1 - sim(w_{t-1}, w_t)) \cdot \frac{N - rank_s(w)_t}{N - rank_s(w)_{t-1}} \quad (4.11)$$

where $rank_s$ is the ranking position by fs and $(1 - sim(w_{t-1}, w_t))$ indicates the preference of words with dissimilar neighbors. However, this could also be the word that becomes less important than before. Thus the ranking position comparison $\frac{N - rank_s(w)_t}{N - rank_s(w)_{t-1}}$ is adopted to avoid such situation.

At the first time window, words that are added into keyword set are selected by word importance score fs . From the second time window onwards, top R words ranked by $rank_d(w)$ are added in keyword set.

Overall, our method is based on the expectation and maximization framework. In expectation step, we leverage SVM classifier to predict the tweets labels. We adopt the simple linear kernel SVM which has time complexity $O(n^2)$, where n is the number of training samples. In maximization step, we first calculate single word properties and words co-occurrence property separately. The time complexity is $O(w)$ and $O(w^2)$ respectively, where w is the vocabulary size. Then, we need to maximize the objective function, *i.e.*, Equation 4.2, by changing tweets labels. The single word properties and words co-occurrence property are needed to re-calculate at each round of tweet label

changing. As the result, the time complexity is $2^H * O(w^2)$, where H is the number of changeable tweets. Since we have constraints to set the changeable tweets H to be a small number, the number 2^H can be considered as a constant a . Eventually, the time complexity of maximization step is $a * O(w^2)$. The expectation and maximization process iterates for several rounds, denoted by b , until relatively stable results are obtained. The time complexity for the whole process becomes $b * a * O(w^2)$. Experiments suggest that the number of iterations b is smaller than 25. The re-ranking process only takes linear time complexity. In short, the proposed method is efficient.

4.3 Experiments

4.3.1 Dataset

We use the two real-life datasets collected in chapter 3, *Chile Earthquake* and *Indian Flood*. Besides, we collect one more event of data:

- *NBA All-Star Game*: The 2016 NBA All-Star Game was an exhibition basketball game played on February 14, 2016.

We use *EQ*, *Flood* and *NBA* to denote the three events. For the *NBA* event, we collect data from Feb 14th to 16th.

We split all tweets in the Twitter stream into different time windows because of the different characteristics of the three events. Statistics are reported in Table 4.2. Specifically, we set the time window size for *EQ* to be 8 hours, and one day for *Flood* the same as in chapter 3. The time window for *NBA* is 12 hours.

The collection and ground truth labeling process for *NBA* data is the same as stated in Section 3.3.1 of chapter 3. For *NBA*, we determine the event relevant tweets as long as the content is related to NBA 2016 but not focus on specific aspects. Besides, the cleaning process for *NBA* data is also the same as before. Meanwhile, the initial seed

Table 4.2: Number of tweets $\mathcal{T}$ fall within the time window in the simulated Twitter stream (*i.e.*, 1% sample of the entire Twitter), and number of positive tweets $\mathcal{T}_P$ in each time window on the three datasets.

EQ	$\mathcal{T}$	$\mathcal{T}_P$	Flood	$\mathcal{T}$	$\mathcal{T}_P$	NBA	$\mathcal{T}$	$\mathcal{T}_P$
t_1	79,726	280	day_1	1,613,579	193	t_1	162,599	345
t_2	244,529	513				t_2	719,089	1,894
t_3	364,964	373				t_3	538,070	2,088
t_4	499,713	1,153	day_2	1,618,825	236	t_4	720,170	2,517
t_5	246,540	280	day_3	1,783,940	116	t_5	678,860	966
t_6	342,098	168				t_6	709,626	756

keywords for the three real-life event datasets are extracted from Wikipedia the same as Section 3.3.2 in chapter 3.

4.3.2 Baseline Methods

As reviewed in Section 2.1.2, the most commonly adopted methods for event-related text identification are classification and topic model. Meanwhile, topic model can also be applied for keyword extraction. Thus, we compare the proposed method *SSEM* with five state-of-the-art methods: ADP [100], Latent Dirichlet Allocation (LDA) [17], Dictionary Learning (DL) [63], Targeted Topic Modeling (TTM) [149], and Active Learning (AL). Except *ADP*, all the other baseline methods are applied on the cleaned tweets in the keyword filtered candidate tweet set. We will detail the processing for *ADP* shortly.

The *LDA*, *DL*, and *TTM* methods all need a topic number to be assigned, and would return several clusters of tweets and corresponding keywords under different topics. We take the top-ranked keywords of the most relevant topic(s) from *LDA* and *DL*, and top-ranked keywords from every topic of *TTM* are all considered relevant to the specified events. Then we add them into the keyword set K to collect tweets in the next time window. The number of selected keywords from the chosen topic in each time window is the same as the number of keywords extracted by our method. If more than one cluster

is selected as event-related in these baseline methods, *e.g.*, 2, then top $R/2$ keywords from each cluster will be selected as keywords.

LDA We consider each tweet as an input document for *LDA*, and the same for *DL* and *TTM*. We empirically evaluate various topic numbers and select the one with the best topic grouping results. For each topic number, among the resulting topics, we select the candidate relevant topic(s) according to top-ranked keywords by manual checking. The manual checking process is the same for *DL*.

DL The objective function in dictionary learning is as follows [63]:

$$l(X, D_T) = \min_{\Psi \in \mathbb{R}^{k \times m}} \frac{1}{2} \|X - D_T \Psi\|_2^2 + \lambda \|\Psi\|_1$$

where X is a word $\times$ document matrix, D_T is considered as a word $\times$ topic matrix, and Ψ is regarded as a topic $\times$ document matrix. We assign each word to the topic where the value of the row in D_T is the largest. The assignment of documents to topics is the same. When selecting top-ranked words, we rank all the words in one topic according to their corresponding values in D_T .

TTM This method requires documents of one domain but with different aspects as input [149]. Because we can not assign all the tweets into specific domains, we take the tweets selected by keywords as one domain. The aspect-specified keywords also need to be predefined. Note that, the model does not require comprehensive keywords to describe the target aspect. We select the most representative words as aspect-specified keywords.¹ Experiments in the paper [149] show the quality of keywords, but no result of the aspect-related documents is reported. In the model, there is a status variable $r \in \{0, 1\}$ which indicates the document’s relevance to the targeted aspect, where $r = 1$

¹We set *earthquake*, *quake*, *tsunami*, *magnitude*, *Chile* for the *EQ* event, and *flood*, *flooding*, *floods*, *Indian* for the *Flood* event. Keywords for the *NBA* event is changed for each time window, and the whole set of best-performance keywords are *NBA*, *star*, *dunk*, *dunks*, *#nbaallstar*, *@nbaallstar*, *kobe*, *steph*, *contest*. Each time a subset of these words would be used to train the model.

means the document is relevant to the aspect and $r = 0$ otherwise. Thus we identify event-related documents by variable r .

ADP We follow the setting in [100]. A well-defined set of queries Q on the given event is used to capture the static part of the event. Tweets containing words from the set of queries are considered as positive training data P_t . Negative training tweets N_t are randomly selected from the rest of tweets. A TF-IDF score is adopted to rank words in P and top x are selected as event-representative keywords E . Tweets in N_t containing E are removed. A classifier trained by P_t and N_t is used to classify tweets in the next time window, and the classifier is periodically retrained by new tweets. Note that the query keyword set Q is not updated along time.² We consider tweets extracted in the first 6 hours of each time window for the *EQ* event as training data candidates, and tweets in the remaining 2 hours for testing. For the *NBA* event, tweets from the first 9 hours of each time window are used as the training data, and those from the remaining 3 hours are used for testing. Since the event-related tweets for the *Flood* event are extremely unevenly distributed over time, we divide each time window to extract training data from the time period covering the first 80% positive tweets. Tweets from the remaining time period in a time window are used as test data. This is simply to obtain a relatively good results.

Note that we do not set the query set of *TTM* and *ADP* to be the same as the initial seed keyword set, because poorer results are observed from these two methods using seed keywords. The reason could be noises introduced by the general words like *people*, *area*, *time*. The presented query sets for these two baseline methods are those leading to the best performance after an extensive tuning the query sets for these two methods.

²We set the query set of *Chile*, *earthquake*, *quake*, *tsuman*, *magnitude* for the *EQ* event, and *flood*, *flooding*, *floods*, *Indian* for the *Flood* event. For the *NBA* event, the keyword set is *dunk*, *dunks*, *all-star*, *aaron*, *gordon*, *kobe*, *steph*. All of them give best experimental results.

Figure 4.3: Performances comparison of the proposed method SSEM and baseline methods DL, LDA, TTM, ADP, and AL, on EQ.

Figure 4.4: Performances comparison of the proposed method SSEM and baseline methods DL, LDA, TTM, ADP, and AL, on Flood.

Figure 4.5: Performances comparison of the proposed method SSEM and baseline methods DL, LDA, TTM, ADP, and AL, on NBA.

AL This is a variation of our proposed method, which does not include the label inference step from keywords. Thus this is a supervised setting. We adopt SVM as the classifier and uncertainty sampling as the querying strategy. The keyword generation method is the same as in *SSEM*. To make a fair comparison, the input tweets are also clustered by incremental clustering and each time the cluster with minimum average distance to the hyperplane is selected to ask for human labeling. The process stops after a pre-defined number of iterations.

4.3.3 Evaluation on Event-related Tweets

Figure 4.3.2, 4.4 and 4.5 are the comparison results of *SSEM* and baseline models. Based on F_1 measure, our method outperforms all baselines on all the three datasets. With a deeper look at precision and recall, we observe that *SSEM* also achieves better results than other methods most of the time. The *AL* model achieves the second best F_1 result,

and *SSEM* performs better than *AL* all the time on the three datasets. This result suggests that the tweet label inference from keywords, *i.e.*, the maximization step, does help to improve the model performance. Looking at precision and recall, we observe that *SSEM* outperforms *AL* most of time as well. The third best method is *DL* by F_1 measure, and it is better than *TTM* and *LDA*. The performance of *ADP* is rather sensitive to the workload. This is because they consider all tweets containing the query keywords as positive, while this is not always true on the noisy data from Twitter. When the keyword set covers most of event-related tweets, recall would be very high while precision is low as shown on the *EQ* and *NBA* events in Figure 4.3.2 and Figure 4.5. If the keyword set cannot cover all aspects of an event, the recall would be rather low but the precision is fairly high, as in the case of *Flood* event in Figure 4.4. Thus, we cannot directly consider tweets containing keywords as event-related. Human labels and other label inference processes (*i.e.*, label inference from keyword) are needed to acquire high quality data.

Figure 4.6 shows one example of human label comparison over the number of clusters on the three events.³ Observe that, at almost all times, *SSEM* achieves stable results earlier than the *AL* method. In other words, our *SSEM* method requires fewer human labels than *AL* to achieve the same performance. Therefore, *SSEM* demonstrates the capability of reducing human labels by leveraging keyword information.

4.3.4 Evaluation on Keywords Generation

Tables 4.3, 4.4, and 4.5 list example keywords of one time window over the three events. Note that in the first time window for all three events, the top-ranked keywords for *AL* and *SSEM* are ranked by the importance score fs (see Section 4.2.2.2 for the definition of fs), which is the single time window ranking. From the second time window on,

³For space limitation, we cannot show the comparison results on all time windows.

Figure 4.6: Changing of F_1 measures of AL and SSEM, with different number of human labeled clusters on the three events EQ, Flood and NBA.

the top ranked keywords are generated by considering keyword graph changes. Results from all three tables are generated by considering keywords' dynamic changes over time. Some topic-model based methods are with fewer keywords because they generate the same keywords in different topics.

From the results, some important keywords appear all the time along the development of the event, such as *earthquake*, *chile* for the *EQ* event, *flood*, *flooding* for the *Flood* event, and *NBA*, *dunk* for the *NBA* event. This kind of keywords could be produced by all baseline methods and ranked higher than other words. However, generating such kind of keywords all the time does not help introduce new aspects of events. To make the newly added keywords effective on collecting event-related data, our keyword generation method tries to rank such kind of words lower. Emerging words are given higher ranks. Results from the three tables demonstrate the desired results, *e.g.*, *terremoto*, *@reuters* for *EQ*, *disaster*, *#infloodwetrust* for *Flood*, and *#dunkcontest*, *#legend* for *NBA* are

Table 4.3: Example keywords extracted on EQ in time t_3 . Keywords in bold are **NOT related** to the event of interest.

Seed	earthquake, chile, damage, magnitude, people, area, epicenter, building, time, fault, region, utc
DL	magnitude(67), panama(41), earthquake(200), hits(28), 6.2(20), chile(258), 8.2(57), tsunami(38), coast(46), quake(56), download (11), prison(15), female(17), #prayforchile(62), @matthunter123 (15)
LDA	chile(163), earthquake(142), tsunami(76), quake(66), magnitude(67), hits(28), panama(41), coast(44), death(24), massive(22), hit(19), japan(20), female(18), ice (11), damage(20)
TTM	earthquake(214), chile(282), #prayforchile(37), quake(91), panama(37), hits(28), focus (16), tsunami(46), today (23), following (30), 6.2(20), coast(40), death(19), magnitude(64)
ADP	earthquake(125), chile(116), magnitude(60), 8.2(42), #prayforchile(19), tsunami(44), panama(32), coast(31), following (227), hits(116), 6.2(19), chilean(10), usgs(10), #earthquake(13), aftershocks(13)
AL	magnitude(71), #chileearthquake(23), quake(86), #chile(19), #tsunami(19), #earthquake(27), 8.2(65), earthquake(202), @reuters(13), @matthunter123(14), preliminary(6), panama(39), inmates(10), terremoto(6), #prayforchile(78)
SSEM	fs @matthunter123(14), @reuters(13), declined (14), #prayforchile(78), #chileearthquake(23), preliminary(6), toll(15), panama(39), earthquake(202), chile(266), terremoto(6), inmates(10), lindsay (14), quake(86), 6.2(19)
	ReRank m8.2(7), #chileearthquake(23), magnitude(71), quake(86), #earthquake(27), terremoto(6), #tsunami(19), @reuters(13), @matthunter123(14), preliminary(6), panama(39), inmates(10), 6.2(19), iquique(10), earthquake(202)

extracted and rank higher. That is the proposed keyword generation method could produce keywords representing the emerging aspects of an event and important user accounts as well, *e.g.*, @reuters, @bleacherreport. This is also a desirable feature to depict a developing event. However, the other methods could not produce such useful keywords and would generate some irrelevant words.

The *fs* and *ReRank* in Tables 4.3, 4.4 and 4.5 give a comparison example of word

Table 4.4: Example keywords extracted on Flood in time day_2 . Keywords in bold are **NOT related** to the event of interest.

Seed	rainfall, floods, people, damage, water, part, area, displace, land, river
DL	#stopsciology (50), november(57), december(25), #weather(23), flooding(52), superb (42), craze (42), flood(3), great (1), partly (1)
LDA	river(13), blocked(14), flood(23), sam (3), shows (3), owblfn (13), ahmedabad(13), stopsciology (23), fatehbaad (29)
TTM	@karunainsan12(18), flood(42), flooding(53), indian(30), flooded(26), haryana(13), explanation (14), superb (42), near (15), blocked(14)
ADP	indianapolis (24), #othercityofficials (33), area (408), ikea (83), indiana (26), flooded(24), plans (536), store (759), bloodflood(17), ii (301)
AL	#msgonthetopinharyana, #stopsciology , zeal(29), #bansciology (28), #floodthefeed(28), disaster(14), #infloodwetrust(11), flooding(53), flood(42), rain(13)
SSEM	fs #stopsciology (28), boundaries(29), zeal(29), disaster(14), flooding(53), marineland (14), #bansciology (28), #infloodwetrust(11), @voice_ot_orcas(14), #msgonthetopinharyana(29)
	ReRank #msgonthetopinharyana(29), #stopsciology (28), zeal(29), #bansciology (28), #floodthefeed(28), disaster(14), #infloodwetrust(11), rain(13), flooding(53), flood(42)

ranking by the importance score fs and re-ranking results by considering keywords' dynamic changes. Observe that, before considering word changes over time, top ranked words by importance score fs include those core keywords of each event or words that have already been added before, *e.g.*, *#prayforchile*, *NBA*. After considering the keyword graph changes over time, more relevant keywords are introduced, *e.g.*, *#tsunami*, *iquique*, *#dunkcontest*.

4.4 Summary

We propose a semi-supervised expectation and maximization mechanism to collect high-quality tweets related to an event of interest from the Twitter stream. We make the first

Table 4.5: Example keywords extracted on NBA in time t_4 . Keywords in bold are **NOT related** to the event of interest.

Seed	NBA, all-star, basketball, game, MVP, Toronto, Raptors, celebrity, three-point, dunk	
DL	back(390), aaron(109), @nba(349), kobe(824), nba(991), dunks(80), jeans (56), curry(109)	
LDA	dunk(414), nba(656), double(6), lebron(19), curry(105), jordan(28), weekend(94), contest(248), points(77), aaron(108)	
TTM	star(1344), dunk(554), game(733), kobe(496), nba(862), wars(206), contest(266), #nbaallstarto(511), best(184)	
ADP	#contestentry(2), @bigjuicyjack (1), @1035kissfm (1), #kissfm-chicagolovestroye (1), #contest(103), win(1019), lavine(5), best(1381), back(1264), slam(11)	
AL	@ballislife(6), sabres(6), @cbssportsnba(6), tracy(37), dragonball(6), spurs(51), @double0ag(19), evander(15), retirement(25), @fullcourt-prez(7)	
SSEM	fs	kobe(590), @nba(349), star(650), nba(823), @bleacherreport(157), bryant(255), #nbaallstarto(511), #legend(15), tracy(37), sabres(6)
	ReRank	#legend(15), @ballislife(6), honors(35), #nbaallstarto(511), #dunkcontest(7), bryant(255), sabres(6), tracy(37), #zachlavine(6), @bleacherreport(157)

attempt to leverage word properties to help identify event-related tweets, which could reduce human annotation effort and maintain high performance in terms of F_1 measure. By considering the word graph changing over time, we can generate keywords with few overlap with historical ones. Existing works do not consider words' dynamic changes. Since keywords that could depict the emerging aspects of events are preferred, one can also refer to the top-ranked keywords to learn about the event development. Our proposed method outperforms state-of-the-art methods on both event-related tweets identification and keywords generation.

Chapter 5

Tweet-aware News Summarization by Attention and Reconstruction

5.1 Introduction

In chapter 3 and chapter 4, we propose a supervised method and a semi-supervised method to collect event-related tweets from Twitter Stream. As we discussed before, it is not practical to present all the collected data to readers, which is of great amount. A better way is to summarize the data and provide the core information to users first. If users are interested in the event and want to learn more about the detailed information, they can explore the complete set of data further. Since tweets are noisy, casually written and over detailed sometimes, we turn to news articles linked by the collected tweets with URLs. We consider the news documents linked by the tweets as event-related as well.

Note that not all of the web pages linked by the positive tweets we identified are news documents. They could be other types of linked resources like another tweet, YouTube videos, maps and so on. We prepare a list of common words to distinguish the linked web pages ¹. We expand the short link in tweets and check the full link. If the link contains any word in the list, we consider it as a news page. The collected results are shown in Table 5.1. Actually, web pages without any word in the list in URLs could also be news

¹The list of words used to check news web pages news: abc, bbc, cnn, nytimes, gov.

Table 5.1: Number of unique news linked by URLs in event-related tweets.

Time window		Number of URLs linking to news	Number of link-able URLs	Number of URLs	Positive tweet number
EQ	t_1	52	100	137	280
	t_2	93	168	251	513
	t_3	101	149	215	373
	t_4	117	202	321	1,153
	t_5	58	96	140	280
	t_6	23	52	78	168
Flood	t_1	8	39	60	193
	t_2	5	24	41	236
	t_3	7	17	25	116
NBA	t_1	11	127	169	345
	t_2	18	388	425	1,894
	t_3	41	816	937	2,088
	t_4	55	933	1,117	2,517
	t_5	11	153	183	966
	t_6	7	125	142	756

articles. Thus, the real number of news documents should be more than the reported number in Table 5.1. Note that, the tweets we collected are only 1% of the total stream of tweets. Thus, there should be more tweets containing the URLs of news.

As we can see from Table 5.1, many news articles could be collected through URLs in event-related tweets. We can construct summary from the news articles for the event of interest. However, we do not have reference summaries as ground truth to evaluate the generated summaries. Usually, the reference summaries are constructed manually by people with domain knowledge. It is costly and time consuming to generate such reference summaries. This is also the reason of no big dataset for summarization task. To avoid constructing reference summaries with bias on our collected data, we conduct the summarization task on existing datasets.

From the collected event-related tweets in chapter 3 and chapter 4, it is observed that such tweets do contain much useful information. We do not want to lose information in tweets. Therefore, we aim to construct summarization by considering information reported in a news article as well as its linking tweets, *i.e.*, tweets with URLs linking to this news article, in this chapter. The task is different from existing single document summarization, *e.g.*, [31, 43, 126], because we need to explore auxiliary information, from linking tweets, as well. Only few researches propose comment-oriented summarization [60] or reader-aware summarization [80]. But these works focus a lot on users' attentions or opinions. We agree that the focuses for users revealed in tweets could also be cared about by others. On the other hand, some complementary information in news not mentioned in tweets should also be summarized. Thus, the summary could be general to all the people.

The model is conducted with data provided by [152]. Before proposing the specific model, we first make a comprehensive analysis of data in Section 5.2, so that we can have a better understanding of the relationships between news and tweets. According to the observations in Section 5.2, the attention and reconstruction based model is proposed in Section 5.3, which does not rely on any reference summary for the model training. Extensive experiments are conducted in Section 5.4, which includes baselines comparisons, sentence selection strategies comparisons and case studies. Section 5.5 concludes the model.

5.2 Data Analysis

News articles and their linking tweets provide us an opportunity to explore relevant auxiliary information for news summarization. As reviewed earlier, leveraging tweets to enhance news article summarization has gained significant research interests [22, 80, 152,

Table 5.2: Quantitative evaluation: usefulness of linking tweets for news summarization.

Method	ROUGE-1		
	Precision(%)	Recall(%)	F1(%)
LexRank	19.74	43.53	26.64
Tweets-TFIDF	25.84	38.41	29.67

163]. Before we present our proposed solution, we conduct data analysis on news and their linking tweets to better understand their relationships.

From a glance of tweets, we find that linking tweets are mixed with comments (*i.e.*, readers’ opinions), summaries, references, and unrelated content or noises. To utilize tweets for summarization, a comprehensive data analysis is necessary. We conduct analysis on the dataset provided by [152]. This dataset contains 121 news and their linking tweets, where the extremely short tweets (*i.e.*, shorter than 5 tokens) and the tweets that are suspected copies of news titles and highlights are removed by the authors. The number of remaining tweets linking to each news article ranges from 200 to 22,000. As tweets are noisy, we perform further cleansing by removing # hashtag sign, @ account, URLs and other symbols. To avoid redundancy, we further remove retweets or near duplicate tweets. The cleaned dataset used in our analysis (and our experiments) consists of 14,692 tweets linking to 121 news articles.

Observation 1. *Linking tweets could suggest or highlight important information reported in a news article.*

We first quantitatively evaluate ‘the extent of usefulness’ of linking tweets for news summarization. We choose a simple way of evaluation based on word importance estimated from tweets. Specifically, we apply the traditional TFIDF weighting scheme to evaluate importances of words in tweets. Given a news article, we consider each linking

Figure 5.1: Distribution of documents with respect to the corresponding ratio of overlapping words, against tweets vocabulary and news vocabulary respectively.

tweet as one ‘document’ and calculate the TFIDF scores for words in each tweet.² Using the obtained word importance scores, we calculate the average word importance score over each news sentence as sentence salience score. If a word does not appear in any tweet, the importance score is set to 0. We then select top 4 sentences with highest sentence salience scores as the summary for news. In short, we estimate word salience scores based on tweets, then compute salience score of a news sentence by averaging its contained word scores. Top ranked sentences by salience scores are extracted as news summary. We name this simple summarization method as Tweets-TFIDF.

To evaluate the effectiveness of Tweets-TFIDF, we compare the resultant summary with the one generated by LexRank, a classic extractive summarization method [43]. As reported in Table 5.2, Tweets-TFIDF outperforms LexRank by 3 points with respect to ROUGE1-F1 score. This result suggests that the linking tweets could capture important information reported in news articles.

Observation 2. *Linking tweets do not cover all the information reported in news article.*

²All the linking tweets to the same article form the document collection for IDF computation.

Next, we investigate ‘to what extent’ the information reported in a news article is covered by its linking tweets. Since word is the smallest unit conveys meaningful information, we again test the information coverage at word level. In this evaluation, we simply calculate the percentage of overlapping words which appear in both news and linking tweets, over the vocabulary of linking tweets and that of news, respectively. The calculation is based on word surface form. That is, we do not take semantic meaning into consideration to avoid making the problem over complicated. Nevertheless, stop words are removed from both news and linking tweets before the calculation. Figure 5.1(a) plots the number of news articles with respect to the ratio of overlapping words against the vocabulary of linking tweets. The ratio values are mapped to predefined bins. For instance, there are 15 news articles where the ratio of overlapping words against the vocabulary of each article’s linking tweets are in the range of 10% to 20%. Figure 5.1(b) plots the distribution by the ratio of overlapping words against the vocabulary of news articles.

Figure 5.1(b) shows that, for most news articles, the overlapping words with linking tweets only take up no more than 30% of the vocabulary of news article. We argue that a lot of information reported in news are not revealed in their linking tweets. From Figure 5.1(a), we observe that the overlapping words do not reach more than 50% of the unique words in linking tweets most of the time. The remaining words may reveal opinions, additional related information, or unrelated noises.

In short, we show that (i) linking tweets do highlight important information reported in news, and (ii) linking tweets do not fully cover the information reported in news articles. Based on these two observations, we design our model to summarize news by considering both information highlighted by tweets and important information contained in news but not mentioned in tweets.

Figure 5.2: Framework for summarization by attention and reconstruction.

5.3 The Proposed Method: ARS

We name our proposed solution **ARS** (for Attention and Reconstruction based Summarization). Our proposed framework is illustrated in Figure 5.2. The model consists of three components: *attentive information extraction*, *reconstruction regularization*, and *sentence selection*. The attentive information extraction module addresses information highlighted in linking tweets on news content through attentive sequence-to-sequence model. By Observation 2, linking tweets do not cover all the information in news and may also include noises. To cover the news information that are not mentioned in linking tweets, and also to reduce the impact of noise, we propose to use reconstruction regularization. The last step in the framework is to extract sentences as summary.

5.3.1 Attentive Information Extraction

The linking tweets are expected to be generated based on their linked news article. To model the correlation between them, we adopt a sequence-to-sequence framework [34]. Specifically, a news article and a linking tweet form a pair, and we feed the news as input for encoder and the tweet for decoder.

We adopt bidirectional Gated Recurrent Unit (GRU) [36] as basic unit for encoder, as denoted by *RNN encoder* in Figure 5.2. Each news article is composed of a sequence of sentences $D = \{s_q | q \in [1, |D|]\}$, and each sentence consists of a sequence of words $s_q = \{x_i | i \in [1, |\mathcal{V}|\]\}$. Here, we define vocabulary $\mathcal{V}$ to include all the words from both news articles and linking tweets. We concatenate sentences in a news article in order as one sequence, so that the input for each encoding step is a word representation. Each word is represented by a distributed word vector mapped by a word embedding matrix $\mathbf{E}$. The word embedding matrix is constructed from Glove [119] and pre-trained Word2Vec [105] on both news and linking tweets. More specifically, if a word appears in the vocabulary of Glove, the embedding comes from Glove. Otherwise, the embedding is from pre-trained Word2Vec. At each encoding step, we can get the hidden representations on both directions, and we concatenate them as the hidden vector at each step $\mathbf{h}_i^{enc} = [\vec{\mathbf{h}}_i^\top; \overleftarrow{\mathbf{h}}_i^\top]^\top$. The hidden representation of the last step is considered as encoder contextual vector $\mathbf{c}$.

The decoder also utilizes GRU to generate each of the corresponding tweet, denoted by *RNN decoder* in Figure 5.2. The predicted words share the same embedding matrix $\mathbf{E}$ as encoder inputs. At the training stage, we need to feed words in tweet as inputs for the decoding process, to learn the decoding targets. We add a symbol $\langle s \rangle$ before the first word y_1 of each tweet to represent the beginning of a sentence. The decoding target is the tweet itself and we add a symbol $\langle s / \rangle$ after the last word y_n of the tweet to denote the end of a sentence. At each step of the decoding process, the hidden representation is generated from previous output, previous hidden representation, and the encoder contextual vector

$\mathbf{h}_j^{dec} = f(y_{j-1}, \mathbf{h}_{j-1}^{dec}, \mathbf{c})$. At the first step of decoding process, the encoder contextual vector $\mathbf{c}$ acts as the previous hidden representation. At the training stage, the previous output is the previous word in decoder input. The decoder targets are used to calculate the loss with predicted results.

The recurrent neural network suffers from remembering long sequence information. Attention mechanism [5] is usually introduced to alleviate the heavy burden and address different relevance between each encoder state and decoder state. Instead of generating decoder hidden states according to a static encoder contextual vector $\mathbf{c}$ at each decoding step, a different encoder contextual vector $\mathbf{c}_j$ is produced based on the relationship between each encoder hidden state and the previous decoder hidden state as follows:

$$\begin{aligned}
p(y_j | y_1, \dots, y_{j-1}, \mathbf{x}) &= g(y_{j-1}, \mathbf{h}_j^{dec}, \mathbf{c}_j) \\
\mathbf{h}_j^{dec} &= f(y_{j-1}, \mathbf{h}_{j-1}^{dec}, \mathbf{c}_j) \\
\mathbf{c}_j &= \sum_{i=1}^m \alpha_{ij} \mathbf{h}_i^{enc} \\
\alpha_{ij} &= \frac{\exp(\mathbf{e}_{ij})}{\sum_{k=1}^m \exp(\mathbf{e}_{kj})} \\
\mathbf{e}_{ij} &= \mathbf{v}_a^\top \tanh(\mathbf{W}_e \mathbf{h}_i^{enc} + \mathbf{W}_d \mathbf{h}_{j-1}^{dec})
\end{aligned} \tag{5.1}$$

where $\mathbf{W}_e$, $\mathbf{W}_d$ and $\mathbf{v}_a$ are trainable parameters. The probability of the predicted word in step j is acquired from the softmax operation on the final layer representation of decoder state j , whose dimension is the same as vocabulary size $|\mathcal{V}|$. The objective function for sequence-to-sequence framework could be defined as negative log likelihood:

$$\mathcal{L}_s = - \sum_{(\mathbf{X}, \mathbf{Y}) \in D} \log p(\mathbf{Y}' | \mathbf{X}; \theta). \tag{5.2}$$

From attention mechanism, at each decoding step, we can obtain the alignment relationship α_{ij} between each decoded word in tweet x_j with each word in news x_i , by

considering contextual information which is encoded in hidden state. Therefore, the attention weights of words in news could be considered as their relative salience scores at each position with different context. We assume the relative importance of a word with the same surface form at different positions is different, because of the different context. Therefore, we propose a position-dependent word salience score, instead of estimating a constant single salience score for each word. To obtain a synthetic salience score for each word x_i in news at different positions, we sum the relative salience scores of x_i over all of the decoding steps (*i.e.*, over all the words x_j in tweets):

$$\beta_i = \sum_{j=1}^n \alpha_{ij} \quad (5.3)$$

The importance of each word to different tweets are different. Here, we aim to obtain a synthetic salience score which matters for news summarization. Therefore, we do not record the relative word salience scores at the training stage. After the model converges to a stable state, we feed each news document in the inference process. The obtained attention weights at each decoding step is used to construct the synthetic position-dependent salience score β_i for word x_i with respect to the whole set of linking tweets. In fact, word salience score should not be estimated solely based on the correlation between linking tweets and news article, as discussed earlier in Section 5.2. In our model, the position-dependent word salience scores are further refined by the reconstruction regularization module, to be detailed next.

5.3.2 Reconstruction Regularization

Observation 2 states that the linking tweets do not cover all the information in news article and also contain noises. To capture the information in news that are not covered by linking tweets and also to reduce the effect of noises, we propose reconstruction regularization. This is inspired by the idea of reconstruction [58, 91, 99] in the sense that

we should be able to reconstruct the rough story of a news article from its summary. Specifically, the reconstruction regularization consists of two parts: *news reconstruction* and *tweet-news reconstruction*.

News Reconstruction. This is an autoencoder process that encodes news into hidden representation, then decodes from the representation vector to the news itself. The process is used to generate comparison hidden vector for the tweet-news reconstruction regularizer, to be detailed shortly. We apply variational autoencoder [67] which is an efficient approach to rebuild the input through reparameterization trick [124]. We make the variational encoder share the same embedding matrix $\mathbf{E}$ and the same RNN encoder with the previous sequence-to-sequence framework as illustrated in Figure 5.2. By doing so, the estimation of word salience scores would not be dominated by the linking tweets, as it will be affected by news reconstruction regularizer. The encoder contextual vector $\mathbf{c}$ is fed into variational decoder.

The objective for variational autoencoder is to reproduce original input from the hidden representation, formulated as:

$$p_{\theta}(\mathbf{x}) = \int p_{\theta}(\mathbf{x}|\mathbf{z})p_{\theta}(\mathbf{z})d\mathbf{z} \quad (5.4)$$

where $\mathbf{z} \in \mathbb{R}^K$ is the hidden representation which can reconstruct the input $\mathbf{x}$. We refer further derivation of the objective function to the tutorial [39]. By introducing a recognition model $q_{\varphi}(\mathbf{z}|\mathbf{x})$ to approximate the posterior $p_{\theta}(\mathbf{z}|\mathbf{x})$, we can get the lower bound of the objective as follows:

$$\begin{aligned} \mathcal{L}_v &= \log p_{\theta}(\mathbf{x}) - D_{KL}[q_{\varphi}(\mathbf{z}|\mathbf{x}) \parallel p_{\theta}(\mathbf{z}|\mathbf{x})] \\ &= \mathbb{E}_{\mathbf{z} \sim q}[\log p_{\theta}(\mathbf{x}|\mathbf{z})] - D_{KL}[q_{\varphi}(\mathbf{z}) \parallel p_{\theta}(\mathbf{z})]. \end{aligned} \quad (5.5)$$

where D_{KL} denotes Kullback-Leibler divergence. We make the value $D_{KL}[q_{\varphi}(\mathbf{z}|\mathbf{x}) \parallel p_{\theta}(\mathbf{z}|\mathbf{x})]$ to be small and the parameters θ and φ could be jointly learned [39].

Based on parameters introduced in our model, we rewrite above loss function as:

$$\begin{aligned}\mathbb{E}_{\mathbf{z} \sim q}[\log p_{\theta}(\mathbf{x}|\mathbf{z})] &= - \sum_{\mathbf{x} \in D} \log p(\mathbf{X}'|\mathbf{X}; \theta, \varphi) \\ D_{KL}[q_{\varphi}(\mathbf{z}) \parallel p_{\theta}(\mathbf{z})] &= -\frac{1}{2} \sum_{i=1}^K (1 + \log(\sigma_i^2) - \mu_i^2 - \sigma_i^2)\end{aligned}\tag{5.6}$$

As in existing works [66, 67], we assume both prior and posterior of the latent variables follow Gaussian distribution:

$$p_{\theta}(\mathbf{z}) \sim \mathcal{N}(0, \mathbf{I}), q_{\varphi}(\mathbf{z}|\mathbf{x}) \sim \mathcal{N}(\mathbf{z} : \boldsymbol{\mu}, \boldsymbol{\sigma}^2 \mathbf{I}).\tag{5.7}$$

Mean $\boldsymbol{\mu}$ and standard deviation $\boldsymbol{\sigma}$ can be computed from the encoder contextual vector $\mathbf{c}$, through a linear layer:

$$\begin{aligned}\boldsymbol{\mu} &= W_{c\mu} \mathbf{c} + b_{c\mu} \\ \log(\boldsymbol{\sigma}^2) &= W_{c\sigma} \mathbf{c} + b_{c\sigma}\end{aligned}\tag{5.8}$$

The latent vector $\mathbf{z}$ is obtained through reparameterization trick [124] from the estimated $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$:

$$\mathbf{z} = \boldsymbol{\mu} + \varepsilon \boldsymbol{\sigma}.\tag{5.9}$$

where $\varepsilon \sim \mathcal{N}(0, \mathbf{I})$ is a noise sampled from Gaussian distribution.

Having latent vector $\mathbf{z}$, we are able to calculate the conditional probability $p_{\theta}(\mathbf{x}|\mathbf{z})$ through a decoding process. Here, we also adopt the attentive GRU framework as decoder, which is denoted as *RNN VAE decoder* in Figure 5.2. Note that each of the hidden states in encoder is calculated the same as contextual vector $\mathbf{c}$ to $\mathbf{z}$ as in Equation 5.9 when performing attention alignment in VAE decoding process. The probability of output word in prediction process is estimated similar to Equation 5.1.

Tweet-News Reconstruction. By exploring the correlations between news and linking tweets, the important information in news revealed by tweets could be captured through attention mechanism. To some extent, we consider tweets being pseudo summaries for

news, although they are not true summaries for the reasons of partially covering information reported in news and of containing noises. However, we can make them approximate summaries in hidden vector space. Again based on the idea of reconstruction, we push the hidden representations of each linking tweet and news closer. Since we can reconstruct news input from its hidden representation, we should also be able to rebuild most of the news information from similar tweet representation. By doing so, we aim to capture the information in news that are not mentioned in tweets and to alleviate the effect of noises in tweets.

We take $\mathbf{z}$ as the representation for news which is used to reconstruct the news itself. We formulate tweet representation from hidden vector $\mathbf{h}_n^{dec}$ of the last step in tweet decoder, which represents the tweet target. To make the two hidden vectors comparable, we apply a linear layer on $\mathbf{h}_n^{dec}$:

$$\mathbf{d}_z = W_h \mathbf{h}_n^{dec} + b_h \quad (5.10)$$

where $\mathbf{d}_z$ is taken as tweet representation. We then measure the cosine similarity between two representations and aim to reduce the distance between them. We call it tweet-news reconstruction regularizer:

$$\mathcal{L}_r = 1 - \cos(\mathbf{d}_z, \mathbf{z}). \quad (5.11)$$

The training process for the whole model works in multi-task learning schema because the sequence-to-sequence module and variational autoencoder module are trained together with different targets. We also add the tweet-news reconstruction regularizer. Then the loss function for the whole model become as follows:

$$\mathcal{L} = \mathcal{L}_s + \mathcal{L}_v + \lambda \mathcal{L}_r \quad (5.12)$$

where λ is a hyperparameter and we experimentally set the value.

5.3.3 Sentence Selection

With two reconstruction regularizers, word salience scores are estimated not only by correlations between news and linking tweets, but also by the news information not mentioned in linking tweets. Therefore, we consider that word salience scores capture important information from both linking tweets and news at different positions. We average salience scores of all words (except stop words) in a sentence as sentence salience score:

$$w_s = \frac{1}{|s|} \sum_{i=1}^{|s|} \beta_i. \quad (5.13)$$

We then select the top 4 sentences with highest sentence salience scores to be summary for the news article. This is one of the most simple sentence selection strategies. We evaluate some alternative strategies in our experiments (see Section 5.4.4).

5.4 Experiments

In this section, we conduct experiments to evaluate our summarization solution against baseline methods. We also evaluate different sentence selection strategies under our framework. Lastly, we perform detailed analysis of our solution and provide case studies.

5.4.1 Dataset and Model Implementations

Dataset. The dataset used in experiments is provided by Wei *et al.* [152], which includes news articles, human generated reference summaries, and linking tweets. The news articles were collected from CNN and USAToday. Reference summaries are the associated highlights. There are 121 news documents in the dataset and one reference summary is associated with each news. Each reference summary (*i.e.*, highlights of a news article) consists of 3 to 4 sentences. The number of linking tweets for each news article is in the range of 200 to 22,000.

We clean the linking tweets as stated in Section 5.2. After cleaning, we have 121 news and 14,692 tweets. Following [134], we do not feed all news sentences into our model. Instead, we take the first 20 sentences if a news article is longer than that. Our results show that the first 20 sentences are sufficient to construct summaries, and including more sentences does not bring noticeable improvements.

Implementation Details. We set the hidden units for GRU cell as 128, so the dimension of hidden vector after concatenating both directions is 256. Adam [65] is adopted as optimizer. We set 0.001 as learning rate and decay the value by 0.96 after 1,000 iterations. To reduce the vanishing and exploding gradient problems for the RNN training [13], we apply gradient norm clipping strategy [117] and set the bounded norm as 1. Our model is implemented on TensorFlow³ and runs on single GPU.⁴

We apply 5-fold cross validation to estimate the converging training epoch for our model. One epoch refers to training all examples in training set once. During each epoch, we randomly shuffle training data to accelerate model convergence [73]. For each training process, we adopt early stopping rule to decide when to stop the training process. Specifically, we use minibatch stochastic gradient descent training scheme [12] and set batch size to 16. Each iteration is training one batch of data. At every 200 iterations, we calculate the loss on validation data and compare the results with the previous minimum validation loss. The number of iterations for minimum validation loss is recorded as the optimum number of iterations. If this number is 600 iterations smaller than the current number of iterations, the training process stops [12]. The epoch for optimum number of iterations is recorded for each cross validation model. We take the average optimum epoch over 5 training processes as the required number of training epoch, which is 12 in our settings.

³Version 1.3 https://www.tensorflow.org/versions/r1.3/api_docs/python/

⁴Tesla P100, with 3,584 Cuda cores and 16G GPU memory.

Table 5.3: Results of ARS and all baselines. All ROUGE scores reported in this chapter are with significance test and are in 95% confidence interval with at most ± 0.25 of reported results calculated by the official ROUGE script with default settings. Best F1 scores are in bold and second best are underlined.

Method	ROUGE-1		ROUGE-2		ROUGE-L	
	Precision(%)	Recall(%)	F1(%)	Precision(%)	Recall(%)	F1(%)
Supervised methods						
LSF-CCF	23.90	39.80	29.20	8.90	15.50	<u>11.00</u>
LTF-CCF	32.00	29.50	29.50	11.80	10.50	10.50
Get-to-the-Point	26.39	35.84	29.73	8.53	12.07	9.81
Unsupervised methods						
Lead4	23.47	46.12	30.13	8.33	16.96	10.83
LexRank	19.74	43.53	26.64	5.70	13.15	7.79
Sentence ILP	20.90	29.30	23.80	5.80	8.80	6.80
Phrase ILP	21.50	28.10	23.60	6.10	8.60	6.90
HGRW-ST	25.80	37.60	29.80	-	-	-
Tweets-TFIDF	25.84	38.41	29.67	8.81	14.49	10.59
RAVAESum	22.78	43.55	29.17	7.10	14.44	9.30
The proposed solution ARS, and its variants						
ARS/R	21.47	39.89	27.17	6.33	12.53	8.18
ARS/TNR	23.23	46.87	<u>30.24</u>	8.17	17.11	10.76
ARS	24.39	47.42	31.42	8.82	17.80	11.52
						28.66

5.4.2 Baseline Methods

We compare our proposed solution **ARS** with both unsupervised and supervised methods, listed below.

Lead4. The first several sentences usually perform good summarization of news articles as analyzed in [109, 151]. We select the first 4 sentences as summary.

LexRank [43] is a typical degree-based graph model for unsupervised extractive summarization. We use it as one of the baselines.

Phrase ILP is proposed by Woodsend [156]. The method jointly performs selection and compression on extracted phrases from parse trees and dependency graphs. Integer linear programming is used to formulate constraints and generate final summaries.

Sentence ILP is a variant of Phrase ILP method, and the extracted unit is sentence instead of phrase.

HGRW-ST is proposed in [153]. This is a random walk method on heterogeneous graph where both news sentences and linking tweets are vertices.

Tweets-TFIDF is the method we utilized to generate results for Observation 1. Top 4 sentences with highest importance scores are selected as summary.

RAVAESum [80] is the most similar work to ours. We re-implement their model. They use VAE to reconstruct news and comments (*i.e.*, linking tweets in our setting) separately on different spaces. They explore the relationships between news and comments through multiplication of their representation matrices. The coefficient value is obtained by an average pooling over the relationship matrix. The magnitudes over rows in the weight matrix which is utilized to recover news in different spaces are considered as sentence salience scores. Next, ILP is performed on phrases to construct the summary. However, it is not clear how to get phrase salience scores from sentence salience scores from the

original paper [80]. Thus, we select top 4 sentences with highest sentence salience scores as summary. The number of training epoch is set to be the same as ours, *i.e.*, 12.

LSF-CCF and LTF-CCF are reported in [152]. LSF-CCF model extracts summary sentences by a pair-wise ranking model RankBoost [46] with a set of hand-crafted features from news and correlations between news and tweets. LTF-CCF adopts the same ranking method but with local features in tweets and correlations between news and tweets.

Get-to-the-Point [134] is a state-of-the-art method for single document summarization. We perform abstractive summary generation by using their pre-trained model⁵ on our test data, and compare the generated texts with reference summaries.

ARS/R is a variant of our ARS method. ARS/R refers to the attentive information extraction module, without adding the two reconstruction (R) regularizers. The word salience scores and sentence salience scores are generated similar as in ARS. Top 4 sentences with highest sentence salience scores are extracted as summary.

ARS/TNR is a variant of ARS without adding tweet-news reconstruction (TNR) regularizer $\mathcal{L}_r$ in Equation 5.12. The other settings remain the same.

ARS is the proposed method. Parameter λ for tweet-news reconstruction regularizer in Equation 5.12 is empirically set to 0.2.

5.4.3 Performance Comparison

We evaluate all methods by using pyrouge package,⁶ which is a python wrapper of the official ROUGE metrics [87]. The results of ARS and all baselines are reported in Table 5.3.

ARS outperforms all baselines in terms of ROUGE-1, ROUGE-2 and ROUGE-L scores. Although we utilize a simple sentence selection strategy, our model achieves

⁵<https://github.com/abisee/pointer-generator>

⁶<https://pypi.python.org/pypi/pyrouge/0.1.0>

good results. By comparing ARS with ARS/R and ARS/TNR, we claim that all the components in ARS together yield the best performance. With only the attentive information extraction module, the performance is poorer than the simple Tweets-TFIDF method. This result is consistent with Observation 2, that linking tweets do not cover all the information reported in news articles. By adding news reconstruction regularizer, a big performance improvement is observed. With the sharing RNN encoder, news reconstruction regularizer helps on word bi-directional hidden vectors learning, which further affects word salience scores estimation. In other words, by considering both correlations between news and linking tweets, and news content, the learned word salience scores reflect more comprehensive information for summarization. Combining attentive information extraction module and the two regularizers (*i.e.*, news reconstruction regularizer and tweet-news reconstruction regularizer), ARS achieves the best performance. Thus, we could consider tweet-news regularizer do help on reducing noises in tweets and capturing information in news but not mentioned in linking tweets. This result implies the effectiveness of the reconstruction regularization.

The most simple method Lead4 could produce rather good results due to the inherent writing structure of news. To further analyze the differences between our proposed method ARS and Lead4, we calculate the overlapping ratio of the extracted summaries by the two methods. Only 38.02% of the extracted summaries by ARS is the same as Lead4. We also note that the average summary length of ARS is 90.0 words, while that is 92.6 words for Lead4. This means that ARS selects more concise sentences to summarize the document with better ROUGE scores.

As observed from Table 5.3, ARS outperforms both supervised and unsupervised models which do not consider tweets in addition to news. To some extent, linking tweets help on highlighting important information in news articles. More importantly, our model does not require reference summaries as training data, compared to Get-to-the-Point. At

the same time, we do not need to spend effort on hand-crafting features as in LSF-CCF and LTF-CCF methods. The Phrase ILP method also investigates fine-grained salience scores at word or phrase level based on hand-crafted features on news articles. In ARS, we compute different salience scores for each word at different positions in contrast to their way of computing word or phrase salience scores. Results show that our model achieves much better performance. By investigating correlations between news article and linking tweets attentively other than matrices multiplication, and position-dependent word salience estimation, ARS could achieve better results than RAVAESum.

5.4.4 Sentence Selection Strategies

In document summarization, Maximum Marginal Ranking (MMR) is a widely adopted re-ranking method which balances summary relevance and redundancy [24]. In our ARS, we select top ranked sentences with highest sentence salience scores, without considering redundancy. We now investigate alternative sentence selection strategies by considering information redundancy through various MMR implementations.

Specifically, MMR re-ranks sentence s_i in document D to get final summary S as follows:

$$MMR = \arg \max_{s_i \in D/S} [\xi \text{sim}(s_i, q) - (1 - \xi) \max_{s_j \in S} \text{sim}(s_i, s_j)] \quad (5.14)$$

where q is query, *e.g.*, centroid of a document to be summarized. In general, we can take sentence initial ranking scores as $\text{sim}(s_i, q)$ and $\max_{s_j \in S} \text{sim}(s_i, s_j)$ represents information redundancy. Thus, we investigate different sentence selection strategies through combinations of different initial ranking methods and different ways to compute information redundancy.

- *Initial Ranking Methods: **SSR v.s. TSR***. Besides ranking sentences by sentence salience scores (SSR), the similarity scores between sentences and news title (TSR),

Table 5.4: Results of using different sentence selection strategies. Best F1 are in bold and second best underlined.

Method	ROUGE-1			ROUGE-2			ROUGE-L		
	Precision(%)	Recall(%)	F1(%)	Precision(%)	Recall(%)	F1(%)	Precision(%)	Recall(%)	F1(%)
SSR-TOP	24.39	47.42	31.42	8.82	17.80	11.52	22.26	43.30	28.66
SSR-MMR-TFIDF	24.32	42.28	30.10	7.77	14.53	9.87	21.81	38.03	27.01
SSR-MMR-PDWSS	24.27	44.82	<u>30.77</u>	8.18	15.97	<u>10.57</u>	21.88	40.56	<u>27.75</u>
TSR-TOP-TFIDF	23.51	42.50	29.34	7.40	14.11	9.42	21.32	38.67	26.63
TSR-TOP-PDWSS	23.29	43.59	29.66	7.83	15.23	10.08	21.15	39.72	26.97
TSR-MMR-TFIDF	23.65	41.23	29.27	6.91	12.81	8.76	21.18	37.09	26.24
TSR-MMR-PDWSS	23.84	42.20	29.58	7.65	14.03	9.63	21.55	38.26	26.76

could also be used to generate an alternative initial ranking. Since many studies take news title as a summary of news article [35, 126], we compute the cosine similarity between each news sentence and its title to produce a similarity-based ranking.

- *Sentence Representation: **PDWSS v.s. TFIDF***. To calculate information redundancy, we need a proper way to represent sentences. Recall that our proposed model generates position-dependent word salience scores (*i.e.*, PDWSS). To test the effectiveness of the scoring method, we take them as a weighting scheme similar to TFIDF and project the value to bag-of-words vector space. We compare this weighting scheme with the classic TFIDF method.

TOP denotes the simple selection of top ranked sentences, in contrast with MMR. The results of 7 sentence selection strategies are reported in Table 5.4. When the initial ranking method is SSR, the simple selection method TOP, denoted by SSR-TOP, achieves the best performance on all measures. Note that, this selection strategy does not consider information redundancy. One possible reason is that the word salience scores are estimated in position-dependent manner, by considering contextual information. Thus, information redundancy is indirectly considered when estimating salience scores. SSR-MMR-PDWSS also performs slightly better than SSR-MMR-TFIDF, suggesting that the proposed position-dependent word salience score is effective.

When the initial ranking method is TSR, TOP and MMR produce comparable results on the same sentence representation. Comparing PDWSS and TFIDF, PDWSS sentence representation gives slightly better results than TFIDF. We argue that PDWSS conveys richer information than TFIDF. Overall, SSR ranking method performs better than TSR, which further suggests the effectiveness of position-dependent word salience estimation.

PDWSS Example. To further illustrate the effectiveness of PDWSS, we provide an example to visualize the position-dependent word salience scores in Figure 5.3. The

Figure 5.3: Heat map of two examples of position-dependent word salience scores.

two sentences are sampled from the resultant summaries by our model. Word “Boston” appears in both sentences. We can observe that word salience score for “Boston” in the first sentence is larger than that in the second sentence. The main focus of the first sentence is “Boston University student”, while the main focus of the second sentence is “twin blasts Monday’s Marathon”. PDWSS could highlight key focuses here with consideration of different contexts.

5.4.5 Model Sensitivity to Parameter λ

Here we report the model sensitivity to parameter λ in Equation 5.12, with different sentence selection strategies. We vary the value of parameter λ from 0 to 1.0 with 0.2 as a step. Note that TSR-TOP-TFIDF and TSR-MMR-TFIDF in Table 5.4 have nothing to do with λ . Thus, we compare the other 5 methods, and plot their performances against λ in Figure 5.4.

From the figure, our method achieve best result when λ is 0.2. However, other than TSR-TOP-PDWSS, the other methods are more sensitive to the value of λ . One possible reason is that the position-dependent word salience scores estimate the relative relationships among words but not the exact values. Regardless the actual value at each position,

Figure 5.4: Model sensitivity to parameter λ on different sentence selection strategies.

the relationship between the title and each news sentence is stable, which leads to less sensitive to λ for TSR-TOP-PDWSS. The other methods depend on exact values, hence are more sensitive to λ .

5.4.6 Case Studies

News article summarization. We now show example summaries of a news article in Table 5.5, including human generated reference summary, the extracted summary by our proposed model ARS, and example tweets that highlight readers’ focuses. Note that, the order of sentences in extracted summary is the ranking order by sentence salience scores. The sentence id $s\#$ indicates the position of sentence in original news article, where $\#$ is the position number.

Comparing reference summary and the extracted summary, we note that the extracted sentences $s1$ and $s2$ coincide with part of human generated summary. Although $s3$ and $s9$ do not appear in the reference summary, they reflect readers’ focuses and many linking tweets mention these two points (*e.g.*, $t1$ and $t2$). By analyzing the correlations between news article and linking tweets with attentive information extraction, we indeed capture

Table 5.5: Summaries of a sampled news article.

Reference Summary	r1: The plane which crashed on runway 28L at SFO was a Boeing 777. r2: More than 300 passengers and crew were abroad. r3: NTSB Chairman says investigators looking at all possibilities seek cockpit voice data recordings. r4: At least five in critical condition
Extracted Summary	s1: SAN FRANCISCO An Asiana Airlines flight from Seoul crashed on landing at San Francisco’s airport Saturday killing two passengers injuring more than 160 and and forcing dozens of frightened passengers and crew to scamper from the heavily damaged aircraft before it was engulfed in smoke and flames. s3: Officials said 123 escaped without injury and 181 were hospitalized or treated for injuries. s2: There were 307 on board including 16 crew. s9: The Boeing 777 appeared to have touched down tail-first and short of the runway
Sample Linking Tweets	t1: Two dead 181 hurt in San Francisco air crash t2: Asiana Airlines crash landing at San Fran airport Sat Touched down tail-1st amp short of runway 2 dead 168 hurt

readers’ focuses. Such kind of summary should be more meaningful to other readers as well. At the same time, summary generated by our method maintains the key information reported in news.

Tweets Summarization. A by-product of our method is that it could be potentially extended to summarize linking news. Recall that we select news sentences by sentence salience scores, which are the average word salience scores excluding stop words in a sentence. Word salience score is calculated by a summation of attentions over decoding steps as shown in Figure 5.2. However, we cannot compute similar word salience score on tweets, because the word salience score is position-dependent. Further, we cannot simply reverse the process of estimating word salience score in news to make it work on tweets, because of the constraints in Equation 5.1. Instead, we can select the maximum attention weight over the word sequences in news as word salience score in tweet at each

Table 5.6: Example summary of linking tweets.

News	Four-state tornado outbreak kills 1 - CNN.com
Tweets	<p>t1: Kansas Oklahoma Iowa Missouri are in bull’s-eye Baseball-sized hail wind gusts and tornado teamfollowback.</p> <p>t2: Storm spotters report tornado on ground in northern Oklahoma City Oklahoma metro area near Edmond OK</p> <p>t3: Twister kills 1 in Oklahoma 1 death in Pottawatomie County State of emergency in Oklahoma Tornado touches dow</p>

position. Then tweet salience score can be calculated by averaging word salience scores within it. Top ranked tweets with highest tweet salience scores are extracted as tweets summary, *i.e.*, to summarize all linking tweets to the same news article.

We do not have reference summaries to evaluate the tweet summaries. Here, we show an example tweet summary in Table 5.6, with news title and the 3 top-ranked (cleaned) tweets. The first tweet introduces the basic situation of the tornado which synthesizes the description in news article. The second tweet provides tornado related information out of news content. The third tweet rephrases the news title and suggests more details stated in the article. This tweet summary is consistent with our Observations 1 and 2. In short, by incorporating tweets for news document summarization, our model also provides a potential approach for linking tweets summarization.

5.5 Summary

We propose an unsupervised extractive summarization model, *i.e.*, not requiring reference summaries for the model training. Our model extracts sentences from news articles to reflect readers’ focus points in linking tweets, and the core information reported in news. The model consists of three modules: attentive information extraction, reconstruction regularization, and sentence selection. We explore the relationships between linking

tweets and news article through an attentive sequence-to-sequence model. The information in news article that are not mentioned in linking tweets is taken care of through the reconstruction mechanism. Specifically, we push the hidden representations for tweets and news closer. The position-dependent word salience scores are learned from both linking tweets and news article. The proposed model outperforms state-of-the-art supervised and unsupervised models on benchmark dataset. Besides news summarization, our model could be potentially extended for tweet summarization.

Chapter 6

Aspect-driven Multi-Document Summarization

6.1 Introduction

Chapter 5 presents the summarization model for single news document and its linking tweets. However, in real life, multiple news documents and their linking tweets describing one event or topic is the more common situation. Therefore, a multi-document summarization should be more practical. In this chapter, we discuss the summarization for multiple news documents and their linking tweets.

Multi-document summarization is more challenging than single document summarization, due to information diversity, redundancy and noise. We need to take both sequential context information in single document and the relationships among multiple documents into consideration. Graph-based models are typical frameworks for multi-document summarization [43, 104, 120]. Random walk method is adopted to evaluate the importances of sentences. However, this type of models does not perform well when the graph is not well connected. Thus, Wan *et al.* [145] claim that the multiple documents describe different topics of the underlying topic and not all of them are equally important. They propose to group sentences into clusters and perform random walk on each cluster separately. However, their model is solely based on statistical information. The

sequential context information within each document is neglected and not considered. In this chapter, we propose a model which could deal with sequential context knowledge in single document and the semantic relationships among documents in the set with the same underlying topic. The details are presented in Section 6.3. Before that, we define the preliminary and display the brief data analysis in Section 6.2. Experiments are conducted in Section 6.4 and we concludes this chapter in Section 6.5.

6.2 Preliminary

We define our problem as follows. Given a set of documents under the same topic, $D = \{d_1, d_2, \dots, d_n\}$, we get a pool of sentences from these documents, denoted by S . Besides, we also have a set of comments $C_i = \{c_{i,1}, c_{i,2}, \dots, c_{i,h}\}$ about each document d_i . Both document d_i and comment $c_{i,j}$ consists of a sequence of sentences, *i.e.*, $d_i = \{s_1, s_2, \dots, s_{|d_i|}\}$ and $c_{i,j} = \{r_1, r_2, \dots, r_{|c_{i,j}|}\}$. We aim to extract document sentences from S to summarize the set of documents which could also address readers' focuses contained in comments at the same time.

To make better use of comments, we explore the relationships between news documents and their comments based on dataset provided by Li *et al.* [80]. From this dataset, we observe that the number of comments are not uniformly distributed over news documents of the same topic. Some news documents are associated with a large number of comments, while some have no comment at all. Since the documents share the same topic, comments about one document may also be related to other documents in one set. Meanwhile, it is also common for users to make comments after getting through several topic-relevant articles. Thus, comments for one document could also contain information from related news. To verify our assumption, we simply calculate the vocabulary overlap between news and comments. We calculate the ratio of overlapping vocabulary of each news and its comments, over news vocabulary, and the average number is 0.345. Against

news and comments pair, we calculate the same ratio of all the news and comments under the same topic over news vocabulary, and the average number is 0.447. Therefore, by considering comments as related to the whole document set seems to be more reasonable than document and comment set pair. Thus, the inputs of our model could be denoted by a set of news documents $D = \{d_1, d_2, \dots, d_n\}$ with their sentences S and a set of comments $C = \{c_1, c_2, \dots, c_m\}$ with a pool of comment sentences R . Later, we will use boldface of each notation to represent their embeddings.

6.3 The Proposed Method: ADS

Our model considers contextual information along sequence of sentences in single document and the relationships among documents under the same topic. To tackle sentences semantic correlations in single document, we adopt a hierarchical RNN structure, which can inherently capture the sequential context information. From a global view of the set of documents, it is natural to conclude that they describe several aspects or sub-topics of the underlying topic. By distributing sentences into appropriate aspects, we can deal with the complex relationships among documents in a simpler manner. To capture readers' interests about the topic, our model incorporates readers' comments to discover attractive aspects of the underlying topic. However, it is common to find that not all the aspects about a topic are equally important. Through attention operation, we estimate aspects' importances. For each aspect, we also obtain the relative saliences of each sentence with respect to aspects. Then our model constructs summaries based on the sentences-aspects-topic hierarchical structure. By multiplying aspect salience scores with sentence relative salience scores, we are able to obtain an overall news sentences ranking. Top ranked sentences are selected as summary for news documents within length limit.

Figure 6.1: Hierarchical structure for sentence and document representations construction.

6.3.1 Document Representation

We start representation generation from words, and pre-train word embeddings $\mathbf{W} \in \mathbb{R}^{|V| \times u}$ by word2vec [105] on both news and comments. The representations for sentences and documents are learned in hierarchical structure as shown in Figure 6.1. The embeddings of words $\mathbf{w}_i$ in each sentence is fed in a simple bidirectional RNN encoder-decoder model [34] in order. GRU [36] is adopted as the basic RNN unit. Each sentence acts as both encoder input and decoder target. Thus, the encoder-decoder could be considered as a RNN auto-encoder. Here, we aim to learn sentence representation $\mathbf{S} \in \mathbb{R}^{|S| \times u}$ without losing word sequence information. The concatenated hidden vectors on both directions at the last step of encoder is adopted as sentence embedding, $\mathbf{s} = [\overrightarrow{\mathbf{h}}_{|s|}^\top; \overleftarrow{\mathbf{h}}_{|s|}^\top]^\top$. The loss function for the RNN encoder-decoder is:

$$\mathcal{L}_e = - \sum_{X \in S \cup R} \log p(X'|X; \theta) \quad (6.1)$$

Note that we get sentence embeddings for both news $\mathbf{s}$ and comments $\mathbf{r}$ from the RNN encoder-decoder structure. Words in both news and comments share the same word embedding $\mathbf{W}$. The sentence embeddings for comments are $\mathbf{R} \in \mathbb{R}^{|R| \times u}$ where $\mathbf{r} = [\overrightarrow{\mathbf{h}}_{|r|}^\top; \overleftarrow{\mathbf{h}}_{|r|}^\top]^\top$.

Then news sentence embeddings $\mathbf{s}$ are fed into the next level bidirectional RNN encoder sequentially as in document, to generate document representation $\mathbf{D} \in \mathbb{R}^{|D| \times u}$.

Figure 6.2: Saliency estimation for news sentences and aspects.

Note that, no decoding process exists from sentence to document. At each encoding step for document embedding, we can obtain the hidden vectors on both directions for each sentence, which contain rich sequential context information. We concatenate them as context enriched sentence embedding $\tilde{\mathbf{S}} \in \mathbb{R}^{|S| \times u}$, where $\tilde{\mathbf{s}} = [\overrightarrow{\mathbf{h}}_{s_i}^\top; \overleftarrow{\mathbf{h}}_{s_i}^\top]^\top$. Again, we take the concatenated hidden vectors on both directions at the last step as document embedding $\mathbf{d} = [\overrightarrow{\mathbf{h}}_{|d|}^\top; \overleftarrow{\mathbf{h}}_{|d|}^\top]^\top$.

Now, the sequential context information among sentences in single document are solved by the hierarchical RNN inherently.

6.3.2 Aspect Representation

We construct aspects for the underlying topic from sentences in news and readers' comments. Recall that we generate document embeddings with bidirectional RNN to take care of contextual information. To be distinguished from the view of document, we just take sentence representations generated from word embeddings to construct aspect representations. In other words, we only consider single sentence without contextual information for aspects formation, like a typical topic model [17].

As we do not know the number of aspects contained in the document set and readers' comments, we set the number of latent aspects to k . That is, we assume there could be k latent aspects to be learned automatically from news and comments. With both news and comments representations $\mathbf{E} = [\mathbf{S}; \mathbf{R}] \in \mathbb{R}^{(|S|+|R|) \times u}$, we calculate aspect embeddings as follows:

$$\mathbf{a} = \text{ReLU}((\mathbf{H}\mathbf{e} + \mathbf{b}_h)\mathbf{U} + \mathbf{b}_u) \quad (6.2)$$

where $\mathbf{H} \in \mathbb{R}^{k \times (|S|+|R|)}$, $\mathbf{U} \in \mathbb{R}^{u \times u}$, $\mathbf{b}_h \in \mathbb{R}^k$ and $\mathbf{b}_u \in \mathbb{R}^k$ are trainable parameters. Thus, the aspect representation is $\mathbf{A} \in \mathbb{R}^{k \times u}$.

We hope that each of the aspect embeddings could represent a unique sub-topic for the underlying topic, which means that the overlap across different aspects should be small. Inspired by Luxburg *et al.* [143], we constrain the aspect embeddings to be orthogonal with each other:

$$\mathcal{L}_r = \|\mathbf{A}\mathbf{A}^\top - \mathbf{I}_{k \times k}\| \quad (6.3)$$

6.3.3 Salience Estimation for Aspect

We consider the underlying topic from two perspectives as shown in Figure 6.2. First, the set of documents is an explicit composition of topics. Then, the underlying topics are implicitly composed of latent aspects. Therefore, either from documents' view or from aspects' view, we should be able to get similar topic representations.

From document to topic. From the hierarchical RNN in Section 6.3.1, we obtain document representation $\mathbf{d}$. Again, we adopt bidirectional RNN to encode document embeddings to topic representation $\mathbf{t}_d$, which is the concatenation of hidden vectors on both directions at the last step. Here, the document sequences do not affect the final performance from our experiments. In fact, one can also try CNN to get topic vector. Our experiments suggest RNN performs better than CNN.

From aspect to topic. As we stated before, not all the aspects are equally important to the underlying topic. Some aspects are more salient than others. We aim to leverage saliences of aspects to filter less important information or noises to generate a distilled topic representation $\mathbf{t}_a$. This is different from the topic representation constructed from document $\mathbf{t}_d$, which encodes all the information into topic vector.

To calculate saliences of aspects, we apply attention mechanism [5, 98] on each aspect representation $\mathbf{a}_i$ with respect to the topic representation $\mathbf{t}_d$ which is generated from documents:

$$\alpha_i = \frac{\exp(\mathbf{e}_i)}{\sum_{k=1}^m \exp(\mathbf{e}_k)} \quad (6.4)$$

$$\mathbf{e}_i = \mathbf{v}^\top \tanh(\mathbf{H}\mathbf{a}_i + \mathbf{U}\mathbf{t}_d)$$

where $\mathbf{H}$, $\mathbf{U}$ and $\mathbf{v}$ are trainable parameters. Thus, we feed the weighted aspect embedding $\alpha_i \mathbf{a}_i$ to a bidirectional RNN as above. The concatenation of hidden vectors on both directions at the last step is the topic representation $\mathbf{t}_a$ generated by aspects. Both topic representations $\mathbf{t}_a$ and $\mathbf{t}_d$ should be similar with each other, because they denote the same underlying topic. Since the two embeddings are from identical structure as Siamese network [70], we adopt contrastive loss [51] which can estimate saliences of aspects efficiently through backpropagation:

$$\mathcal{L}_t = \beta * \frac{1}{2} * (\max\{0, 1 - \mathbf{t}_d \mathbf{t}_a\})^2 + (1 - \beta) * \frac{1}{2} * (\mathbf{t}_d - \mathbf{t}_a)^2 \quad (6.5)$$

where β is a hyper-parameter and we set it to 0.5.

6.3.4 Saliency Estimation for Sentence

Note that we employ both news sentences $\mathbf{s}$ and comment sentences $\mathbf{r}$ to construct aspect representations $\mathbf{a}$. Readers' comments are diverse and noisy, and we do not want readers' comments to dominate the aspects. We only want to use comments to highlight readers' focuses that exist in document set. Although we apply saliency scores of aspects to filter

the information encoded in $\mathbf{t}_a$, noises could still affect the quality of the learned aspects. Therefore, we need to constrain on aspect representation learning.

We require the aspect representations to be able to reflect the set of documents. Therefore, we should also be able to construct aspect vectors solely based on news sentence embeddings. Similarly as in Section 6.3.3, we plan to construct the distilled aspect embeddings with weighted sentence embeddings. To distinguished from $\mathbf{s}$, we adopt the context enriched sentence embedding $\tilde{\mathbf{S}} \in \mathbb{R}^{|S| \times u}$ presented in Section 6.3.1.

We estimate salience score γ for sentences as in Equation 6.4, but with $\tilde{\mathbf{s}}$ and $\mathbf{a}$ as input:

$$\gamma_{i,j} = \frac{\exp(\mathbf{e}_{i,j})}{\sum_{k=1}^m \exp(\mathbf{e}_{i,k})} \quad (6.6)$$

$$\mathbf{e}_{i,j} = \mathbf{v}^\top \tanh(\mathbf{H}\tilde{\mathbf{s}}_i + \mathbf{U}\mathbf{a}_j)$$

Then we approximate aspect representation in a similar manner as in Equation 6.2. We have:

$$\mathbf{a}^{\tilde{s}_i j} = \text{ReLU}(\gamma_{i,j}\tilde{\mathbf{s}}_i\mathbf{H} + \mathbf{b}_h) \quad (6.7)$$

where $\mathbf{H} \in \mathbb{R}^{u \times u}$ and $\mathbf{b}_h$ are trainable parameters. Again we adopt contrastive loss:

$$\mathcal{L}_a = \beta * \frac{1}{2} * (\max\{0, 1 - \mathbf{a}\mathbf{a}^{\tilde{s}}\})^2 + (1 - \beta) * \frac{1}{2} * (\mathbf{a} - \mathbf{a}^{\tilde{s}})^2 \quad (6.8)$$

where β is hyper-parameter and we set it to 0.5. With backpropagation, salience scores for sentences could be learned. Thus, the overall training objective is:

$$\mathcal{L} = \mathcal{L}_e + \mathcal{L}_r + \mathcal{L}_t + \mathcal{L}_a. \quad (6.9)$$

We multiply the max salience score for each sentence over aspects $\max \gamma_i$ with the corresponding aspect salience score $\mathbf{a}_j$, where $j = \arg \max \gamma_i$, to get the final salience score $\max \gamma_i \mathbf{a}_j$ for a sentence. We extract the top ranked sentences as summary, for a given length limit (*e.g.*, the total number of words should be less than 100).

6.4 Experiments

6.4.1 Dataset and Experimental Settings

Dataset. We use dataset provided by Li *et al.* [80] to evaluate our proposed model. The dataset contains 45 topics, and each topic contains 10 news documents and 4 reference summaries generated by human. The length limit of the model summary is 100 words (separated by space). On average, each topic contains 215 comments and 940 comment sentences. Each news document contains an average of 27 sentences, and each sentence contains an average of 25 words.

Since there are only 45 topics, the small amount of data makes model training difficult. The good news is that many comments are associated with each topic. Thus, we randomly sample 50 comments for each topic when the number of associated comments is larger than 50, and repeat the sample process for 40 times. Then we get 1,800 training examples. This process could also relieve the bias of comments when generating aspects embeddings. At test stage, we randomly select a news and comments combination for each topic to estimate the model performance.

Experimental settings. Note that we use a single u to denote the hidden vector dimensions in the model description section, as it is a general description of hidden dimension to avoid confusion. In our implementation, we set word embedding dimension as 300. The hidden units of one direction for all GRU cells is set to 128, so the dimension of hidden vector after concatenating both directions is 256. The hidden dimensions of the linear operations in Equation 6.2 and 6.7 are set to 256 as well. We empirically set the number of aspects k to 5. We pre-train word embeddings on news and comments and make the embeddings immutable during the model training process.

We set batch size as 1, which represents data in one topic (*i.e.*, one set of documents). One iteration is training one batch of data and one epoch is training all the training

examples once. We use Adam [65] to optimize the model. The learning rate is set as 0.001 and the value will be decayed by 0.96 after 1,000 iterations. To reduce the vanishing and exploding gradient problems for RNN training [13], we apply gradient norm clipping strategy [117] and set the bounded norm as 1. We stop training when the loss stops decreasing more than 3 iterations. Our model is implemented by TensorFlow version 1.3¹ and runs on a single GPU.²

6.4.2 Baseline Methods

We compare our model with both traditional models and the state-of-the-art methods which take readers' comments into consideration.

Lead [151]: For news documents, the first several sentences usually perform good summarization for the document as analyzed in [109, 151]. The news sentences are ordered chronologically and top sentences are extracted one by one until meeting the length limit.

MEAD [120]: It summarizes clusters of news articles that are automatically grouped by a topic detection system. Then it uses information from the centroids of the clusters to select sentences.

LexRank [43] and **TextRank** [104]: Both methods are unsupervised graph-based model for document summarization. They perform PageRank on the sentence graph where sentence similarities are edge weights.

APSM [15]: We call the abstractive phrase selection and merging method as APSM. It extracts noun and verb phrases from contiguity tree first and estimate the phrase salience based on features. Then an integer linear programming process is employed to optimize the phrase selection.

¹https://www.tensorflow.org/versions/r1.3/api_docs/python/

²Tesla P100, 3,584 Cuda cores, 16G GPU memory.

RA-Sparse [81]: This model tackles reader-aware MDS problem. A sparse-coding-based method is used to calculate the salience of the news sentences by jointly considering news documents and reader comments.

RAVAESum and **RAVAESum-NC** [80]: RAVAESum reconstructs news and comments separately on different spaces. They explore the relationship between news and comments through multiplication of their representation matrices. The coefficient value is obtained by an average pooling over the relationship matrix. The magnitudes over rows in the weight matrix which is utilized to recover news in different spaces are considered as sentence salience scores. Further phrase salience scores could be estimated from sentence salience scores. Next, the integer linear programming operator is used to select optimal phrases for summary construction. RAVAESum-NC removes readers comments from input and trains the model solely based on news documents.

ADS: We call our proposed aspect-driven summarization and ADS for short. In our setting, we feed the first 25 words of each sentence and the first 27 sentences of each document into the model (*i.e.*, which are the average numbers in the dataset). Experimental results suggest no significant differences with that feeding all data into model.

ADS-NC: This is a variant of our proposed model. Here we do not include readers' comments. Thus the aspect embeddings are generated solely based on sentence embedding $\mathbf{s}$. Note that this is not the context enriched sentence embedding. All other settings are the same as ADS.

6.4.3 Performance Comparison

We evaluate all models with pyrouge package³, which is a python wrapper of the official ROUGE metrics [87]. We report F1 scores for ROUGE-1, ROUGE-2 and ROUGE-SU4 as shown in Table 6.1.

³<https://pypi.python.org/pypi/pyrouge/0.1.0>

Table 6.1: Results of the proposed methods and baselines. “R” indicates ROUGE. All the ROUGE scores reported in this work are with significance test and the values fall in 95% confidence interval with at most ± 0.25 of reported results calculated by the official ROUGE script with default settings.

Method	R-1	R-2	R-SU4
Lead	0.384	0.110	0.144
MEAD	0.402	0.141	0.171
TextRank	0.402	0.122	0.159
LexRank	0.425	0.135	0.165
APSM	0.422	0.157	0.188
RA-Sparse	0.442	0.157	0.188
RAVAESum-NC	0.437	0.162	0.189
RAVAESum	0.443	0.171	0.196
ADS-NC	0.443	0.163	0.177
ADS	0.456	0.181	0.190

From Table 6.1 we observe that our proposed model ADS outperforms all baseline methods on all of ROUGE-1, ROUGE-2 and ROUGE-SU4. The variant of our model ADS-NC achieves comparable performance with the best baseline methods RAVAESum on ROUGE-1 score, which demonstrates the superiority of our model.

Methods incorporating readers’ comments information, *i.e.*, RA-Sparse, RAVAESum, and ADS, generally perform better than those without comments. It could be directly observed from the results of RAVAESum and our ADS. For both methods, by adding reader comment information into the model, better performances are achieved. Therefore, we conclude that reader comments do help on summarization, although they are diverse and noisy. On the other hand, both RAVAESum and ADS are neural network based models, which encode semantic information into representations inherently. The difference is that our ADS does not take all news sentences as a whole set. ADS distributes sentences into different aspects for better structured representation. It is also our intuition that multiple documents describe several aspects of the underlying topic.

Table 6.2: Results on different sentence selection strategies, which are also with significance test as above.

Method	R-1	R-2	R-SU4
ADS-HIR	0.456	0.181	0.190
ADS-HIR-MMR	0.410	0.141	0.154
ADS-ASP	0.444	0.172	0.181
ADS-ASP-MMR	0.365	0.102	0.124

6.4.4 Comparison on Different Sentence Selection Strategies

Note that, we do not consider information redundancy when selecting the summary sentences. Maximum Marginal Relevance (MMR) [24] is a commonly adopted method to balance both salience and redundancy. We now compare the performances of our model with and without MMR applied.

Recall that we multiply the maximal sentence salience score over aspects with the corresponding aspect salience score to get the importance value of each sentence. However, we cannot ensure that at least one representative sentence from every important aspect is extracted and added into the final summary. Thus, we try another way of selecting sentences. We simply define the aspect with salience score larger than the average value $1/k$ (where k is the number of aspects) as an important aspect. Then the important aspects are sorted by aspect salience scores in descending order. We select sentence with the maximal sentence salience score with respect to each of the important aspects in order. If the length of extracted sentences does not exceed summary length limit, sentence with the second maximal salience score from each of the important aspects are added in order until reaching the length limit. To distinguish with previous global sentence selection strategy, we name this aspect respected approach as **ADS-ASP**. For easy comparison, we name the previous hierarchical salience multiplication method as **ADS-HIR** (*i.e.*,

Table 6.3: Example sentences.

Reference Summary	r1: Facebook launched an Internet.org app in Zambia to provide free access to Facebook and other online services including Wikipedia, Google Search, AccuWeather and websites offering health and other services. r2: Facebook aims to bring the Internet to people who are not yet online.
Extracted Summary	s1: Facebook launches Internet.org app to let users access basic Internet services for free. s2: Facebook’s Internet.org project is taking another step toward its goal of bringing the Internet to people who are not yet online with an app launching Thursday in Zambia.

ADS in Table 6.1). By incorporating with and without MMR for both strategies, we have four combinations as shown in Table 6.2.

From Table 6.2 we observe that MMR strategy leads to poorer performance for both ADS-HIR and ADS-ASP. The reason could be the writing style. When describing different or even same aspect of the underlying topic, representative words would appear multiple times to indicate the topic. Thus, when redundancy information is taken into consideration, such sentences would not be selected in final summary. One example is shown in Table 6.3.⁴ Two of the extracted sentences correspond to two sentences in reference summary. They describe the complementary information for one aspect of the topic, while redundancies exist. By applying MMR, **s2** is removed and not included in summary. On the other hand, ADS-HIR performs slightly better than ADS-ASP. This suggests that the simple definition of important aspect may not be proper, leading to sentences from unimportant aspects be selected in summary. But via hierarchical saliences, such cases are reduced.

⁴We do not show the complete reference and extracted summaries, but just a few sentences from them.

6.5 Summary

In this chapter, we propose an extractive multi-document summarization solution, which deals with both sequential context information in single document and relationships among documents. Hierarchical RNN is adopted to encode sequential context information in sentence and document embeddings. This helps on estimating sentence saliences because sentence importance is affected by its context knowledge. Relationships among documents are complex, because different possible relationships exist. Thus, we make an assumption that k aspects are described by the document set. Then we distribute sentences into appropriate aspects which group similar parts and distinguish different content across documents. We incorporate readers' comments for aspects construction to capture attractive aspects to readers. Then we estimate saliences of aspects by comparing the underlying topic representation constructed from documents and aspects, which should be similar. Analogously, sentence saliences are estimated by comparing aspect representations generated from news and comments with those generated from context enriched news sentences. The global sentence salience score is produced by multiplying the above two salience scores. Top ranked sentences are extracted as summary within length limit. Through extensive experiments, we show that the proposed model outperforms state-of-the-art neural network models, and traditional models.

Chapter 7

Conclusion and Future Work

In this chapter, we summarize the research works in this thesis first and discuss promising directions for future works.

7.1 Conclusion

Social media platforms provide us a new channel to learn about information, which is more convenient, fast, detailed and interactive compared to traditional media. However, the diverse, noisy and large volume make it difficult for general users to get access to desired tweets through keyword search function provided by Twitter platform. Especially for decision makers and people involved in or care about the events, a clean and complete set of tweets is highly preferred. We propose methods to collect clean and complete set of event-related tweets. To better display the retrieved data, we further propose unsupervised methods to summarize the data.

After reviewing existing approaches on collecting event-related data in chapter 2, we propose our first method in chapter 3. We adopt keyword search method to collect event-related tweet candidates from Twitter stream. Instead of preparing a static set of keywords which requires good domain knowledge, we propose to generate event representative keywords dynamically. The preferred properties for evaluating qualified keywords are *relevance*, *coverage*, and *evolvment*. We aim to generate the keywords that are not

only representative to the event of interest, but also can cover many relevant tweets. Besides, the keywords change together with event development. Having the candidate tweets collected by the generated keywords, an advanced identification on event-related or not should be done to remove noises. To achieve high precision on identification, we adopt classification approach while training data is required. To reduce human labeling effort, active learning together with uncertainty sampling method is employed to label only selective tweets. Experimental results on real-life events suggest better performance than state-of-the-art methods, and fewer labeled data is needed for training the model.

From results of previous work, we find some core keywords always rank at top along with event development. However, such keywords have already been added into keyword set and we hope some new words to rank higher. To achieve this goal, we propose to evaluate word graph changes dynamically and measure the configuration similarity in chapter 4. Therefore, we can obtain representative and novel keywords dynamically along with event evolvement. Actually, the generated keywords also contain rich information about event of interest. This inspires us to identify event-related tweets by leveraging keywords information, which could further reduce human labeling effort. Thus, a semi-supervised event-related tweets collection method is proposed. We estimate the relevance of a sentence to the event of interest based on the ranking score of words in the sentence, which is estimated based on the features of keywords. Through expectation and maximization, we can identify a tweet to be event-related or not through both human label and keywords importance. We compare the proposed method with state-of-the-art models and observe better performance. To achieve similar performance, our method requires less human labeling efforts than supervised approaches.

In chapter 5 and chapter 6, we research on data presentation. A concise and comprehensive summary of the clean and complete set of data is easier for the public to understand the event or topic. Since tweets are short, noisy and casually written, it is

not appropriate to select several tweets to construct the concise summary presenting to others. Therefore, we turn to well written news linked by tweets via URLs from where to extract sentences as the summary. We start from single news article together with tweets linking to it. This is because the focuses of readers could be different from those in the news report. Most of existing news summarization approaches ignore the points cared by readers. But these could also be cared about by the other readers. Thus, we aim to produce the summary which can express both readers' focuses and the key information in news article. The attention-based sequence-to-sequence model is applied to explore information mentioned in both news document and the linking tweets. A reconstruction mechanism is proposed to complement the missing information in news article not mentioned by linking tweets. A position-dependent word salience estimation is proposed other than the commonly seen constant value for each word. This benefits sentence salience estimation by considering context information. Experiments suggests the usefulness of the position-dependent word saliences. Without requiring reference summary to train the model, we can achieve even better results than state-of-the-art supervised methods.

We further explore the summarization on multiple documents in chapter 6. Multiple news documents describing the same event or topic is the common and natural scenario. Besides the sequential context information that is needed to be considered within single document, the complex relationships among documents are also needed to be resolved. However, the challenge is that, it is hard to define and distinguish different relationships among documents. Inspired by topic model, we assume the relationships as different aspects of the same topic and define them as k latent aspects. By doing so, sentences are assigned to different aspects. To further explore readers' focuses, tweets linking to the news articles are distributed to different aspects at well. Thus, the underlying topic could be represented from both document level and aspects level, which should be similar

with each other. The representations of sentences, documents and topic are generated from the hierarchical RNN model, which encodes the sequential contextual information inherently. We estimate saliences for sentences and aspects separately and produce the final score for sentences hierarchically. Testing on benchmark dataset, the proposed method outperforms the baselines.

To sum up, we do researches on desired data collection and the concise way to present data. Specifically, we propose supervised and semi-supervised methods on collecting clean and complete event-related tweets from Twitter stream through dynamically generated keywords. Then, we explore methods to present both key information in news and the focuses of readers through summary with limited length. The proposed methods in this dissertation overcomes the lack of data issue for deep learning based models on summarization with benefits from tweets.

7.2 Future Work

Our research works in this thesis are about event-related tweets collection and presenting summaries of news articles and their linking tweets. In other words, the works are about collecting desired data and presenting the large amount of data in a proper form, which refers to techniques in IR and NLP fields. We plan to continue working on either extending the proposed approaches and resolving possible drawbacks, or moving to other open issues. We briefly discuss the potential future directions as follows.

- For collecting tweets on Twitter Stream, our proposed methods are processed in batch mode. Specifically, we define a time window size and identify event-related tweets after collecting tweets in the time window. An advanced case is identifying tweets online which is more challenging. For online version, we accept tweets in real-time and cannot control the sequence of tweets arriving. A fast and efficient

event-related tweet identification method is needed for this purpose. If incorporating human labeling effort, the uncertainty sampling method cannot be applied directly. An appropriate and effective strategy for selecting tweets to label therefore is necessary. Besides, query keywords should also be updated. When to update query keywords also need to be decided in contrast with previous predefined time window size. The online version could update information immediately and are more useful in real events.

- In this thesis, we make research on extracting sentences from news articles and presenting them as summary for news and linking tweets. However, as we emphasized previously, sentences do not exist independently. They are supported by surrounding context. Inevitably, some specific coreference in extracted summary sentence could be less meaningful without relevant context. In addition, some clause could be redundant or fusing contextual sentences could compose more concise and informative summary sentence. Just take a real case reported in a news article as an example, *Oscar Pistorius, a Paralympic runner who blazed new terrain by competing in last summer's Olympics, is accused of killing his girlfriend, Reeva Steenkamp.* This could be shorten by removing the redundant information as *Oscar Pistorius is accused of killing his girlfriend, Reeva Steenkamp.* All these situations are not taken into consideration in current solutions. These could lead to more concise and coherent summary, and some researches also try to resolve the issues [15, 69] with machine learning techniques. However, the semantic coherence problem cannot be well resolved. Thus, they still deserve paying attention to develop solution techniques.
- Currently, the summary is constructed by leveraging news articles linked by URLs in tweets due to the well written content. Another direction is summarizing collected tweets directly. However, this requires overcoming many challenges. First,

tweets are noisy, casually written and mixed with abbreviations, slangs and misspellings. Converting the relatively casual writing style and wrong words to well format and correct references requires much research efforts. This is also helpful for natural language understanding. Second, tweets are short and information in the collection is piecemeal. Proper and efficient approaches to integrate fragment information together deserve to be investigated. Last but not least, lack of reference summary makes model training much challenging. One can manually construct corresponding reference summaries or resolve the problem from other angle without requiring reference summary. To achieve all above requirements, abstractive summarization is a better choice. This direction is tough but deserves researching because this could be generalized to many applications, *e.g.*, article comprehending, summarization for complicated meeting.

- The setting for researches in this thesis is that the event of interest is described by few words. This is a general case that we collect and summarize everything about the event. However, different people could care about different things. Someone could know about part of the event, and she wants to learn more about some other or specific piece of information. This calls for query-based information collection and summarization. The query may not be well-defined and sometimes even confusing. This requires further refining and define the information boundary.

List of Publications

- **Xin Zheng** and Aixin Sun: Collecting Event-related Tweets from Twitter Stream. *Journal of the Association for Information Science and Technology (JASIST)*, accepted 2018.
- **Xin Zheng**, Jialong Han, and Aixin Sun: A Survey of Location Prediction on Twitter. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, accepted in 2018.
- **Xin Zheng**, Aixin Sun, Sibor Wang and Jialong Han: Semi-Supervised Event-related Tweet Identification with Dynamic Keyword Generation. In *Proceedings of the 26th ACM Conference on Information and Knowledge Management (CIKM 2017)*, pages 1619-1628, 2017.
- **Xin Zheng**, Aixin Sun, Karthik Muthuswamy. Tweet-aware News Summarization by Attention and Reconstruction. **submitted to CIKM 2018**.
- **Xin Zheng**, Aixin Sun, Karthik Muthuswamy. Aspect-driven Multi-Document Summarization. **submitted to EMNLP 2018**.

References

- [1] Willyan Daniel Abilhoa and Leandro Nunes de Castro. A keyword extraction method from twitter messages represented as graphs. *Applied Mathematics and Computation*, 240:308–325, 2014.
- [2] James Allan, Ron Papka, and Victor Lavrenko. On-line new event detection and tracking. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 37–45, 1998.
- [3] Stuart Andrews, Ioannis Tsochantaridis, and Thomas Hofmann. Support vector machines for multiple-instance learning. In *Advances in Neural Information Processing Systems, NIPS*, pages 561–568, 2002.
- [4] Peter Auer. On learning from multi-instance examples: Empirical evaluation of a theoretical approach. In *Proceedings of the Fourteenth International Conference on Machine Learning, ICML*, volume 97, pages 21–29, 1997.
- [5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations, ICLR*, 2015.
- [6] Siddhartha Banerjee, Prasenjit Mitra, and Kazunari Sugiyama. Multi-document abstractive summarization using ILP based multi-sentence compression. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI*, pages 1208–1214, 2015.
- [7] Vladimir Batagelj and Matjaz Zaversnik. Generalized cores. *CoRR*, cs.DS/0202039, 2002.

- [8] Hila Becker, Dan Iter, Mor Naaman, and Luis Gravano. Identifying content for planned events across social media sites. In *Proceedings of the Fifth International Conference on Web Search and Web Data Mining, WSDM*, pages 533–542, 2012.
- [9] Hila Becker, Mor Naaman, and Luis Gravano. Beyond trending topics: Real-world event identification on twitter. In *Proceedings of the Fifth International Conference on Weblogs and Social Media, ICWSM*, 2011.
- [10] Slobodan Beliga, Ana Meštrović, and Sanda Martinčić-Ipšić. An overview of graph-based keyword extraction methods and approaches. *Journal of Information and Organizational Sciences*, 39(1):1–20, 2015.
- [11] Abdelghani Bellaachia and Mohammed Al-Dhelaan. Ne-rank: A novel graph-based keyphrase extraction in twitter. In *2012 IEEE/WIC/ACM International Conferences on Web Intelligence*, pages 372–379, 2012.
- [12] Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural Networks: Tricks of the Trade - Second Edition*, pages 437–478. 2012.
- [13] Yoshua Bengio, Patrice Y. Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.
- [14] Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. Jointly learning to extract and compress. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, ACL*, pages 481–490, 2011.
- [15] Lidong Bing, Piji Li, Yi Liao, Wai Lam, Weiwei Guo, and Rebecca J. 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 of the Asian Federation of Natural Language Processing, ACL, Volume 1: Long Papers*, pages 1587–1597, 2015.

- [16] Saroj K. Biswas, Monali Bordoloi, and Jacob Shreya. A graph based keyword extraction model using collective node weight. *Expert Systems with Applications*, 97:51–59, 2018.
- [17] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.
- [18] Praveen Bommanavar, Jimmy J. Lin, and Anand Rajaraman. Estimating topical volume in social media streams. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, pages 1096–1101, 2016.
- [19] Florian Boudin, Hugo Mougard, and Benoît Favre. Concept-based summarization using integer linear programming: From concept pruning to multiple optimal solutions. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 1914–1918, 2015.
- [20] Chris Buckley, Gerard Salton, James Allan, and Amit Singhal. Automatic query expansion using SMART: TREC 3. In *Proceedings of The Third Text REtrieval Conference, TREC*, pages 69–80, 1994.
- [21] Mark A. Cameron, Robert Power, Bella Robinson, and Jie Yin. Emergency situation awareness from twitter for crisis management. In *Proceedings of the 21st World Wide Web Conference, WWW*, pages 695–698, 2012.
- [22] Ziqiang Cao, Chengyao Chen, Wenjie Li, Sujian Li, Furu Wei, and Ming Zhou. Tgsum: Build tweet guided multi-document summarization dataset. In *Proceeding of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 2906–2912, 2016.
- [23] Ziqiang Cao, Wenjie Li, Sujian Li, and Furu Wei. Improving multi-document summarization via text classification. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 3053–3059, 2017.
- [24] Jaime G. Carbonell and Jade Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 335–336, 1998.

- [25] Jorge Carrillo de Albornoz, Enrique Amigó, Laura Plaza, and Julio Gonzalo. Tweet stream summarization for online reputation management. In *Proceedings of Advances in Information Retrieval - 38th European Conference on IR Research, ECIR*, pages 378–389, 2016.
- [26] Yllias Chali, Moin Tanvee, and Mir Tafseer Nayeem. Towards abstractive multi-document summarization using submodular function-based framework, sentence compression and merging. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing, IJCNLP, Volume 2: Short Papers*, pages 418–424, 2017.
- [27] Yllias Chali and Mohsin Uddin. Multi-document summarization based on atomic semantic events and their temporal relationships. In *Advances in Information Retrieval - 38th European Conference on IR Research, ECIR*, pages 366–377, 2016.
- [28] Moses Charikar, Chandra Chekuri, Tomás Feder, and Rajeev Motwani. Incremental clustering and dynamic information retrieval. *SIAM Journal on Computing*, 33(6):1417–1440, 2004.
- [29] Yan Chen, Hadi Amiri, Zhoujun Li, and Tat-Seng Chua. Emerging topic detection for organizations from microblogs. In *The 36th International ACM SIGIR conference on research and development in Information Retrieval*, pages 43–52, 2013.
- [30] Zhiyuan Chen, Arjun Mukherjee, Bing Liu, Meichun Hsu, Malú Castellanos, and Riddhiman Ghosh. Discovering coherent topics using general knowledge. In *22nd ACM International Conference on Information and Knowledge Management, CIKM*, pages 209–218, 2013.
- [31] 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, ACL*, 2016.
- [32] Xueqi Cheng, Jiafeng Guo, Shenghua Liu, Yanfeng Wang, and Xiaohui Yan. Learning topics in short texts by non-negative matrix factorization on term correlation matrix. In *Proceedings of the 13th SIAM International Conference on Data Mining*, pages 749–757, 2013.

- [33] Lee-Feng Chien, T.-I. Huang, and M-C. Chien. Pat-tree-based keyword extraction for chinese information retrieval. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 50–58, 1997.
- [34] Kyunghyun Cho, Bart van Merriënboer, Çağlar Gülçehre, 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, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734, 2014.
- [35] Sumit Chopra, Michael Auli, and Alexander M. Rush. Abstractive sentence summarization with attentive recurrent neural networks. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 93–98, 2016.
- [36] Junyoung Chung, Çağlar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.
- [37] Andras Csomai and Rada Mihalcea. Linking educational materials to encyclopedic knowledge. In *Proceedings of the 13th International Conference on Artificial Intelligence in Education, Building Technology Rich Learning Contexts That Work, AIED*, pages 557–559, 2007.
- [38] 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.
- [39] Carl Doersch. Tutorial on variational autoencoders. *CoRR*, abs/1606.05908, 2016.
- [40] H. P. Edmundson. New methods in automatic extracting. *Journal of the ACM*, 16(2):264–285, 1969.

- [41] Steffen Eger and Alexander Mehler. On the linearity of semantic change: Investigating meaning variation via dynamic graph models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL, Volume 2: Short Papers*, 2016.
- [42] Ehsan Elhamifar and M. Clara De Paolis Kaluza. Online summarization via submodular and convex optimization. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 1818–1826, 2017.
- [43] 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.
- [44] Feifan Fan, Yansong Feng, Lili Yao, and Dongyan Zhao. Adaptive evolutionary filtering in real-time twitter stream. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM*, pages 1079–1088, 2016.
- [45] Guy Feigenblat, Haggai Roitman, Odellia Boni, and David Konopnicki. Unsupervised query-focused multi-document summarization using the cross entropy method. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 961–964, 2017.
- [46] Yoav Freund, Raj D. Iyer, Robert E. Schapire, and Yoram Singer. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969, 2003.
- [47] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML*, pages 1243–1252, 2017.
- [48] 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.

- [49] Camille Guinaudeau and Michael Strube. Graph-based local coherence modeling. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL, Volume 1: Long Papers*, pages 93–103, 2013.
- [50] Maryam Habibi and Andrei Popescu-Belis. Diverse keyword extraction from conversations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL, Volume 2: Short Papers*, pages 651–657, 2013.
- [51] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR*, pages 1735–1742, 2006.
- [52] James Douglas Hamilton. *Time series analysis*, volume 2. Princeton university press Princeton, 1994.
- [53] Zellig S Harris. Distributional structure. *Word*, 10(2-3):146–162, 1954.
- [54] Kazi Saidul Hasan and Vincent Ng. Automatic keyphrase extraction: A survey of the state of the art. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL, Volume 1: Long Papers*, pages 1262–1273, 2014.
- [55] Taher H. Haveliwala. Topic-sensitive pagerank. In *Proceedings of the Eleventh International World Wide Web Conference, WWW*, pages 517–526, 2002.
- [56] Ruifang He and Xingyi Duan. Twitter summarization based on social network and sparse reconstruction. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.
- [57] Ruifang He, Yang Liu, Guangchuan Yu, Jiliang Tang, Qinghua Hu, and Jianwu Dang. Twitter summarization with social-temporal context. *World Wide Web*, 20(2):267–290, 2017.
- [58] 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*, 2012.

- [59] Thomas Hofmann. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence, UAI*, pages 289–296, 1999.
- [60] 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, 2008.
- [61] Lifu Huang, Hongjie Li, and Lian’en Huang. Comments-oriented document summarization based on multi-aspect co-feedback ranking. In *Proceedings of the Web-Age Information Management - 14th International Conference, WAIM*, pages 363–374, 2013.
- [62] Rishabh K. Iyer and Jeff A. Bilmes. Submodular optimization with submodular cover and submodular knapsack constraints. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems, NIPS*, pages 2436–2444, 2013.
- [63] Shiva Prasad Kasiviswanathan, Prem Melville, Arindam Banerjee, and Vikas Sindhwani. Emerging topic detection using dictionary learning. In *Proceedings of the 20th ACM Conference on Information and Knowledge Management, CIKM*, pages 745–754, 2011.
- [64] Samir Khuller, Anna Moss, and Joseph Naor. The budgeted maximum coverage problem. *Information Processing Letters*, 70(1):39–45, 1999.
- [65] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations, ICLR*, 2015.
- [66] Diederik P. Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems, NIPS*, pages 3581–3589, 2014.

- [67] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations ICLR*.
- [68] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
- [69] Kevin Knight and Daniel Marcu. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139(1):91–107, 2002.
- [70] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, volume 2, 2015.
- [71] Andreas Krause, Carlos Guestrin, Anupam Gupta, and Jon M. Kleinberg. Near-optimal sensor placements: maximizing information while minimizing communication cost. In *Proceedings of the Fifth International Conference on Information Processing in Sensor Networks, IPSN*, pages 2–10, 2006.
- [72] Henry Oliver Lancaster and Eugene Seneta. *Chi-Square Distribution*. Wiley Online Library, 2005.
- [73] Yann LeCun, Léon Bottou, Genevieve B. Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural Networks: Tricks of the Trade - Second Edition*, pages 9–48. 2012.
- [74] Heeyoung Lee, Angel X. 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.
- [75] Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne M. VanBriesen, and Natalie S. Glance. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 420–429, 2007.
- [76] David D. Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the 11th International Conference on Machine Learning, ICML*, pages 148–156, 1994.

- [77] Chen Li, Xian Qian, and Yang Liu. Using supervised bigram-based ILP for extractive summarization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL, Volume 1: Long Papers*, pages 1004–1013, 2013.
- [78] Cheng Li, Yue Wang, Paul Resnick, and Qiaozhu Mei. Req-rec: high recall retrieval with query pooling and interactive classification. In *The 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 163–172, 2014.
- [79] Decong Li, Sujian Li, Wenjie Li, Wei Wang, and Weiguang Qu. A semi-supervised key phrase extraction approach: Learning from title phrases through a document semantic network. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL, Short Papers*, pages 296–300, 2010.
- [80] 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, NFiS@EMNLP*, pages 91–99, 2017.
- [81] Piji Li, Lidong Bing, Wai Lam, Hang Li, and Yi Liao. Reader-aware multi-document summarization via sparse coding. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI*, pages 1270–1276, 2015.
- [82] 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*, pages 2081–2090, 2017.
- [83] 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*, pages 2091–2100, 2017.

- [84] 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 Thirty-First AAAI Conference on Artificial Intelligence*, pages 3497–3503, 2017.
- [85] Rui Li, Kin Hou Lei, Ravi Khadiwala, and Kevin Chen-Chuan Chang. TEDAS: A twitter-based event detection and analysis system. In *IEEE 28th International Conference on Data Engineering*, pages 1273–1276, 2012.
- [86] Zhenhui Li, Ding Zhou, Yun-Fang Juan, and Jiawei Han. Keyword extraction for social snippets. In *Proceedings of the 19th International Conference on World Wide Web, WWW*, pages 1143–1144, 2010.
- [87] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text Summarization branches out: Proceedings of the ACL-04 Workshop*, volume 8. Barcelona, Spain, 2004.
- [88] Hui Lin and Jeff A. Bilmes. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, ACL*, pages 510–520, 2011.
- [89] Wang Ling, Chris Dyer, Alan W. Black, and Isabel Trancoso. Two/too simple adaptations of word2vec for syntax problems. In *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1299–1304, 2015.
- [90] Feifan Liu, Deana Pennell, Fei Liu, and Yang Liu. Unsupervised approaches for automatic keyword extraction using meeting transcripts. In *Proceedings of Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, NAACL*, pages 620–628, 2009.
- [91] He Liu, Hongliang Yu, and Zhi-Hong Deng. Multi-document summarization based on two-level sparse representation model. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 196–202, 2015.

- [92] Pengqi Liu, Javad Azimi, and Ruofei Zhang. Automatic keywords generation for contextual advertising. In *23rd International World Wide Web Conference, WWW, Companion Volume*, pages 345–346, 2014.
- [93] Xiaohua Liu, Yitong Li, Furu Wei, and Ming Zhou. Graph-based multi-tweet summarization using social signals. In *Proceedings of the 24th International Conference on Computational Linguistics: Technical Papers, COLING*, pages 1699–1714, 2012.
- [94] Zhiyuan Liu, Wenyi Huang, Yabin Zheng, and Maosong Sun. Automatic keyphrase extraction via topic decomposition. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 366–376, 2010.
- [95] Zhiyuan Liu, Peng Li, Yabin Zheng, and Maosong Sun. Clustering to find exemplar terms for keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 257–266, 2009.
- [96] Hans Peter Luhn. A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of Research and Development*, 1(4):309–317, 1957.
- [97] Hans Peter Luhn. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2):159–165, 1958.
- [98] 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, EMNLP*, pages 1412–1421, 2015.
- [99] Shulei Ma, Zhi-Hong Deng, and Yunlun Yang. An unsupervised multi-document summarization framework based on neural document model. In *Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers, COLING*, pages 1514–1523, 2016.

- [100] Walid Magdy and Tamer Elsayed. Adaptive method for following dynamic topics on twitter. In *Proceedings of the Eighth International Conference on Weblogs and Social Media, ICWSM*, 2014.
- [101] Luís Marujo, Wang Ling, Isabel Trancoso, Chris Dyer, Alan W. Black, Anatole Gershman, David Martins de Matos, João Paulo da Silva Neto, and Jaime G. Carbonell. Automatic keyword extraction on twitter. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL, Volume 2: Short Papers*, pages 637–643, 2015.
- [102] Yutaka Matsuo and Mitsuru Ishizuka. Keyword extraction from a single document using word co-occurrence statistical information. *International Journal on Artificial Intelligence Tools*, 13(1):157–169, 2004.
- [103] Ryan T. McDonald. A study of global inference algorithms in multi-document summarization. In *Advances in Information Retrieval, 29th European Conference on IR Research, ECIR*, pages 557–564, 2007.
- [104] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing , EMNLP, A meeting of SIGDAT, a Special Interest Group of the ACL, held in conjunction with ACL*, pages 404–411, 2004.
- [105] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems, NIPS*, pages 3111–3119, 2013.
- [106] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 3075–3081, 2017.

- [107] Ramesh Nallapati, Bowen Zhou, Cícero Nogueira dos Santos, Çağlar Gülçehre, and Bing Xiang. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL*, pages 280–290, 2016.
- [108] George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions - I. *Mathematical Programming*, 14(1):265–294, 1978.
- [109] Ani Nenkova. Automatic text summarization of newswire: Lessons learned from the document understanding conference. In *Proceedings of The 20th National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference*, pages 1436–1441, 2005.
- [110] Ani Nenkova and Kathleen McKeown. A survey of text summarization techniques. In *Mining Text Data*, pages 43–76. Springer, 2012.
- [111] Alexandra Olteanu, Sarah Vieweg, and Carlos Castillo. What to expect when the unexpected happens: Social media communications across crises. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, pages 994–1009, 2015.
- [112] Olutobi Owoputi, Brendan O’Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith. Improved part-of-speech tagging for online conversational text with word clusters. In *Proceedings of Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, NAACL*, pages 380–390, 2013.
- [113] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: bringing order to the web. 1999.
- [114] Daraksha Parveen, Hans-Martin Rams, and Michael Strube. Topical coherence for graph-based extractive summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 1949–1954, 2015.

- [115] Daraksha Parveen and Michael Strube. Multi-document summarization using bipartite graphs. In *Proceedings of TextGraphs@EMNLP 2014: the 9th Workshop on Graph-based Methods for Natural Language Processing*, pages 15–24, 2014.
- [116] Daraksha Parveen and Michael Strube. Integrating importance, non-redundancy and coherence in graph-based extractive summarization. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI*, pages 1298–1304, 2015.
- [117] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning, ICML*, pages 1310–1318, 2013.
- [118] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. *CoRR*, abs/1705.04304, 2017.
- [119] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 1532–1543, 2014.
- [120] Dragomir R Radev, Hongyan Jing, and Malgorzata Budzikowska. Centroid-based summarization of multiple documents: Sentence extraction, utility-based evaluation, and user studies. In *Proceedings of the 2000 NAACL-ANLP Workshop on Automatic summarization-Volume 4*, pages 21–30. Association for Computational Linguistics, 2000.
- [121] Marc’Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. In *Proceeding of the International Conference on Learning Representations, ICLR*, 2016.
- [122] Pengjie Ren, Zhumin Chen, Zhaochun Ren, Furu Wei, Jun Ma, and Maarten de Rijke. Leveraging contextual sentence relations for extractive summarization using a neural attention model. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 95–104, 2017.

- [123] Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jarret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 1179–1195, 2017.
- [124] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic back-propagation and variational inference in deep latent gaussian models. In *International Conference on Machine Learning, ICML*, 2014.
- [125] Ellen Riloff and Wendy Lehnert. Information extraction as a basis for high-precision text classification. *ACM Transactions on Information Systems (TOIS)*, 12(3):296–333, 1994.
- [126] 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, EMNLP*, pages 379–389, 2015.
- [127] Mehdi Sadri, Sharad Mehrotra, and Yaming Yu. Online adaptive topic focused tweet acquisition. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, CIKM*, pages 2353–2358, 2016.
- [128] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of the 19th International Conference on World Wide Web, WWW*, pages 851–860, 2010.
- [129] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Tweet analysis for real-time event detection and earthquake reporting system development. *IEEE Transactions on Knowledge and Data Engineering*, 25(4):919–931, 2013.
- [130] Shinsaku Sakaue, Tsutomu Hirao, Masaaki Nishino, and Masaaki Nagata. Provable fast greedy compressive summarization with any monotone submodular function. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, Volume 1 (Long Papers)*, pages 1737–1746, 2018.
- [131] Gerard Salton and Chris Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988.

- [132] Gerard Salton, A. Wong, and Chung-Shu Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.
- [133] Surendra Sedhai and Aixin Sun. Hspam14: A collection of 14 million tweets for hashtag-oriented spam research. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 223–232, 2015.
- [134] 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, ACL 2017, Volume 1: Long Papers*, pages 1073–1083, 2017.
- [135] Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. Minimum risk training for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL, Volume 1: Long Papers*, 2016.
- [136] Shiqi Shen, Yu Zhao, Zhiyuan Liu, Maosong Sun, et al. Neural headline generation with sentence-wise optimization. *arXiv preprint arXiv:1604.01904*, 2016.
- [137] 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, ACL, Volume 1: Long Papers*, pages 1171–1181, 2017.
- [138] Antoine J.-P. Tixier, Polykarpos Meladianos, and Michalis Vazirgiannis. Combining graph degeneracy and submodularity for unsupervised extractive summarization. In *Proceedings of the Workshop on New Frontiers in Summarization, NFiS@EMNLP*, pages 48–58, 2017.
- [139] Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *The Journal of Machine Learning Research*, 2:45–66, 2002.

- [140] Denis Turdakov and Pavel Velikhov. Semantic relatedness metric for wikipedia concepts based on link analysis and its application to word sense disambiguation. In *Proceedings of the SYRCODIS 2008 Colloquium on Databases and Information Systems*, 2008.
- [141] Peter D. Turney. Learning algorithms for keyphrase extraction. *Information retrieval*, 2(4):303–336, 2000.
- [142] Sudha Verma, Sarah Vieweg, William J. Corvey, Leysia Palen, James H. Martin, Martha Palmer, Aaron Schram, and Kenneth Mark Anderson. Natural language processing to the rescue? extracting “situational awareness” tweets during mass emergency. In *Proceedings of the Fifth International Conference on Weblogs and Social Media, ICWSM*, 2011.
- [143] Ulrike von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.
- [144] Abraham Wald. Sequential tests of statistical hypotheses. *The Annals of Mathematical Statistics*, 16(2):117–186, 1945.
- [145] Xiaojun Wan and Jianwu Yang. Multi-document summarization using cluster-based link analysis. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 299–306, 2008.
- [146] Chen Wang and Sujian Li. Corankbayes: bayesian learning to rank under the co-training framework and its application in keyphrase extraction. In *Proceedings of the 20th ACM Conference on Information and Knowledge Management, CIKM*, pages 2241–2244, 2011.
- [147] Li Wang, Junlin Yao, Yunzhe Tao, Li Zhong, Wei Liu, and Qiang Du. A reinforced topic-aware convolutional sequence-to-sequence model for abstractive text summarization. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI*, pages 4453–4460, 2018.

- [148] 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, ACL, Volume 1: Long Papers*, pages 1384–1394, 2013.
- [149] Shuai Wang, Zhiyuan Chen, Geli Fei, Bing Liu, and Sherry Emery. Targeted topic modeling for focused analysis. In *Proceedings of the 22nd ACM International Conference on Knowledge Discovery and Data Mining, KDD*, pages 1235–1244, 2016.
- [150] Shuai Wang, Zhiyuan Chen, Bing Liu, and Sherry Emery. Identifying search keywords for finding relevant social media posts. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 3052–3058, 2016.
- [151] 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, COLING-ACL '98*, pages 1364–1368, 1998.
- [152] Zhongyu Wei and Wei Gao. Utilizing microblogs for automatic news highlights extraction. In *Proceedings of the 25th International Conference on Computational Linguistics: Technical Papers*, pages 872–883, 2014.
- [153] Zhongyu Wei and Wei Gao. Gibberish, assistant, or master?: Using tweets linking to news for extractive single-document summarization. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1003–1006, 2015.
- [154] Zhongyu Wei, Yang Liu, Chen Li, and Wei Gao. Using tweets to help sentence compression for news highlights generation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Volume 2: Short Papers*, pages 50–56, 2015.

- [155] Ian H. Witten, Gordon W. Paynter, Eibe Frank, Carl Gutwin, and Craig G. Nevill-Manning. KEA: practical automatic keyphrase extraction. In *Proceedings of the Fourth ACM conference on Digital Libraries*, pages 254–255, 1999.
- [156] Kristian Woodsend and Mirella Lapata. Automatic generation of story highlights. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL*, pages 565–574, 2010.
- [157] Roman V. Yampolskiy. Turing test as a defining feature of ai-completeness. In *Artificial Intelligence, Evolutionary Computing and Metaheuristics - In the Footsteps of Alan Turing*, pages 3–17. 2013.
- [158] Ming Yang, Melody Y. Kiang, and Wei Shang. Filtering big data from social media - building an early warning system for adverse drug reactions. *Journal of Biomedical Informatics*, 54:230–240, 2015.
- [159] Ping Yang, Yan Song, and Yang Ji. Tag-based user interest discovery through keywords extraction in social network. In *Proceedings of Big Data Computing and Communications - First International Conference, BigCom*, pages 363–372, 2015.
- [160] Wenfeng Yang. Chinese keyword extraction based on max-duplicated strings of the documents. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 439–440, 2002.
- [161] Shiren Ye, Tat-Seng Chua, Min-Yen Kan, and Long Qiu. Document concept lattice for text understanding and summarization. *Information Processing and Management*, 43(6):1643–1662, 2007.
- [162] Yisong Yue and Thorsten Joachims. Interactively optimizing information retrieval systems as a dueling bandits problem. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML*, pages 1201–1208, 2009.
- [163] Evi Yulianti, Sharin Huspi, and Mark Sanderson. Tweet-biased summarization. *Journal of the Association for Information Science and Technology, JASIST*, 67(6):1289–1300, 2016.

- [164] Deyu Zhou, Liangyu Chen, and Yulan He. An unsupervised framework of exploring events on twitter: Filtering, extraction and categorization. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 2468–2475, 2015.
- [165] Xiangmin Zhou and Lei Chen. Event detection over twitter social media streams. *The VLDB JournalThe International Journal on Very Large Data Bases*, 23(3):381–400, 2014.