A Study on Web Document Summarization by Exploiting Its Social Context

Nguyen Minh Tien

Japan Advanced Institute of Science and Technology

Doctoral Dissertation

A Study on Web Document Summarization by Exploiting Its Social Context

Nguyen Minh Tien

Supervisor: Associate Professor Nguyen Le Minh

School of Information Science
Japan Advanced Institute of Science and Technology

March, 2018

To my wife, my son, and my family. Without whom I would never have completed this dissertation.

Abstract

Text summarization is a challenging task of artificial intelligence and natural language processing, in that it reduces the size of an input document (or a set of documents) while preserving its meaning. The task has a long history dating back to 1950s. It mainly falls into two directions: extraction and abstraction. While extraction selects important information from a document, abstraction generates summaries, which are close to the writing style of humans. Even text summarization has extensively investigated by many studies in both directions, outputs of summarization systems are still far from human satisfaction, especially with abstractive summarization.

In the context of social media, users can freely reveal their viewpoints on an event and topics mentioned in a Web document published by a news provider. They tend to discuss an event by writing their comments on the web interface of a provider or posting relevant information on their timeline on social networks, e.g. Twitter. Such information has two important characteristics: (i) it reflects the content of an event and (ii) it includes viewpoints of readers. This observation suggests an interesting idea that the relevant social information of a Web document can be exploited to improve summarization. While traditional text summarization has deeply investigated, exploiting the support from social information for Web document summarization is still in an early stage, which requires more research. The objective of this thesis is to improve the quality of extractive summarization for single Web documents[2] by exploiting their social context.

First, we introduce three unsupervised ranking models, which formulate relationships between Web documents and their social context. The first model uses many lexical similarity features to measure the importance of a sentence and a user post in a mutual support fashion. More precisely, we encode the intra-information and inter-information of a sentence (or a user post) in a model, which ranks to extract summaries. The intuition behind this model is that important sentences include essential words or phrases which also appear in representative user posts. We show that by using a simple greedy selection method, this model obtains competitive results with state-of-the-art systems in term of ROUGE-scores. We also highlight that the number of social messages affects the importance estimation. The second model takes advantage of semantic similarity between sentences and user posts with an assumption that sentences and user posts share common topics denoted in the form of common words or phrases, which are in a variation writing style. From this, we present another ranking model, which combines intra-information and inter-information under a semantic similarity calculation. By using a greedy or an

[2]We consider main documents as single ones.

integer linear programming method, we show that this model obtains the best results in many cases. Among aspects affecting this model, we point out that the number of important words significantly influences the extraction of summaries. The third model explores the nature of sentences and user posts in sharing hidden topics presented in the form of common words or phrases. It encodes sentences and user posts in a unified ranking algorithm, which uses our proposed non-negative matrix co-factorization. It measures the importance of sentences and user posts by estimating their influence on hidden topics in term-sentence matrices. Experimental results indicate that this model obtains promising ROUGE-scores. We show that a joint optimization algorithm produces better results than an individual one.

Second, we present two learning-to-rank models to estimate the importance of sentences and user posts. We exploit social context by introducing many indicators extracted from three channels: local features, user-generated features, and third-party features for training summarizers. Following a supervised learning-to-rank algorithm which uses a greedy or majority voting method for sentence selection, we show that our models achieve the best results in many cases. Our analyses indicate that local features extracted from sentences in primary documents play an important role and features collected from their social context support local ones to improve the quality of the importance estimation step. Among our features, we point out that those from relevant Web articles are very useful in measuring the importance of sentences. We find that since sentences differ from user posts in term of writing style, different features should be used when modeling them.

Finally, we adapt deep learning for our task because it recently has achieved impressive results in many research fields, including text summarization. However, there are very little studies in applying this technique to our task. We first describe a well-known basic deep learning model, Convolutional Neural Network, for classification. Based on that, we adapt and extend it for our ranking purpose. Our model formulates relationships among $n-$grams in a sequence to enrich its representation. It also uses many our features to integrate social context into the ranking step. By doing that, our model obtains improvements compared strong baselines. Analyses show that using all features is inefficient due to the conflict when combining many different ones.

We apply our models to the task of sentence and highlight extraction on three datasets in two languages, English and Vietnamese. Promising results indicate that they can be viable alternative to extraction-based systems. Our findings and results contribute to the literature of text summarization as well as the task of summarizing Web documents by taking advantage of their social context.

Keywords: social context summarization, ranking, feature extraction, integer linear programming, deep leaning.

Acknowledgements

I had three unforgettable years in pursuing the Ph.D. at JAIST with an interesting text summarization task. Although this time is challenging, it is also a wonderful part of my life. The small contributions of this thesis to the literature remind that this study cannot be completed without the support of many people.

First, I would like to deeply thank my supervisor, Associate Professor Nguyen Le Minh, for his support and motivation. He gave me a lot of valuable comments, advice, and discussion, which foster me to approach my research problem from the starting point. I found many interesting ideas for my research from his suggestion. He always encourages and takes care me in research with great enthusiasm and encouragement. I really admire not only his knowledge on artificial intelligence and natural language processing but also the freedom of his perfect guidance balance. During three years, I learned a lot of useful research experiences which benefit my career in the future. Without his guidance and inspiration, I would have never finished my research.

I appreciate useful comments from committee members: Professor Satoshi Tojo, Associate Professor Kiyoaki Shirai, Associate Professor Shinobu Hasegawa, and Professor Tomoko Matsui. Through the discussion, professors pointed out the limited points of my research in several aspects and provided suggestions for improving my thesis.

I would like to give a big thank to my minor research advisers, Associate Professor Dam Hieu Chi and Ph.D. Pham Tien Lam, who guided and showed me how to apply machine learning to materials science with fundamental knowledge and impressive detail-oriented skills. I would like to show my great appreciation to my co-authors: Duc-Vu Tran, Chien-Xuan Tran, Dac Viet Lai, Tran Viet Cuong, Viet-Anh Phan, Truong-Son Nguyen, Tien-Huy Nguyen, Phong-Khac Do, Danilo S. Carvalho, Thi-Dung Nguyen, who debated and gave me many valuable comments and suggestions for my research, including technical support. Among my co-authors, I would like to especially thank Associate Professor Ha Quang Thuy and Associate Professor Nguyen Tri Thanh for their advice during my master course. Without their help during my entrance examination, I could probably not study at JAIST and could not pursue this study. I want to thank Associate Professor Asanobu Kitamoto, who supervised me during my internship at National Institute of Informatics. Under his supervision, I learned a lot of research experiences and skills, which facilitate my research. I really acknowledge ideas and suggestions from Associate Professor Nguyen Xuan Hoai, who suggested me to adapt matrix factorization for my problem. I want to thank Associate Professor Phan Xuan Hieu, who encouraged me in pursuing Ph.D. I would like to give a spacial thank to (i) Associate Processor Le Dinh Duy, who introduced me to Professor Nguyen Le Minh. Without his introduction, I might not have a chance to study at JAIST and work with Professor Nguyen Minh Le. And (ii) Ph.D. Kien Nguyen, who supported me during my entrance examination. I have to thank the mentors of the writing Lab at JAIST: John Blake, Natt Ambassah, Dubhgan Hinchey, Bill Holden and my mentor of ECIR 2016, Ph.D. Preslav Nakov, who gave many useful comments on the draft versions of my papers.

I must thank my collaborators in Nguyen's Laboratory for their useful comments during weekly seminars. I enjoyed fascinating discussion with their clever minds, which inspire several interesting ideas for my research. I love the friendly working atmosphere in our laboratory, in that members can freely discuss research questions and assist together to find out appropriate solutions. I learned many things from my collaborators. It is hard to find a laboratory with much fun like this.

I want to thank all members of Vietnamese Football Club (VIJA). Thanks to this club, I could maintain my physical health to avoid stress in doing my research. By playing football at every weekend, I could balance my life and recover my energy for pursuing Ph.D. during three challenging years.

Last but not least, I would not be who I am today without the love, encouragement, and unconditional support from my family. Especially, I would like to express the deepest appreciation to my wife, Nguyen Thi Hien, for her love and inspiration during my Ph.D. She takes care my son and always be on my side for sharing happiness and sorrow during the difficult years. She encourages me to look forward to a wonderful future. I owe my parents, Nguyen Duc Minh and Trinh Thi Loan, who fostered my love and cultivated my knowledge. Without their unconditional support, I cannot study and cannot pursue my research. I also would like to thank my younger sister, Nguyen Thi Phuong Mai, and my parents in law, who always help and encourage me.

I appreciate anonymous reviewers who commented on my papers. And there are also many other people I have to thank, but it is hard to list all their names. I keep your name in my mind and hope for your understanding.

The work in this dissertation was supported by Vietnamese Government Scholarship (911) and partly covered by Doctoral Research Fellow (DRF) of JAIST.

Nguyen Minh Tien.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Text Summarization

Text summarization is a reduction process which extracts salient information within (a) document(s) to generate an essential subset of textual content for users' usage while preserving its meaning (Ani Nenkova, 2011). Outputs of a summary system, in fact, facilitate to assist with user's information needs and benefit many natural language processing applications such as Web search or highlight generation. For example, search engines, e.g. Google or Bing show the snippet information of relevant pages corresponding to a search query; or online news providers such as Yahoo News[1] usually present an extracted summary corresponding to each article for readers. This information provides a quick view for readers in catching the main content of a document. Such beneficial use demands high-quality text summarization systems.

Text summarization has been widely investigated in many studies which technically fall into two directions: supervised and unsupervised learning. The first direction needs annotated data to usually train a classifier, which can decide whether a sentence should be included in a summary. In training, supervised methods use pre-defined hand-crafted features extracted from training data to train a model which can be used to predict unseen inputs (Cheng and Lapata, 2016; Chopra et al., 2016; Ren et al., 2017; Shen et al., 2007; Wei and Gao, 2014; Yang et al., 2011; Zhang et al., 2017). This approach is appropriate if we have high-quality annotated data and appropriate features. However, annotated data, in fact, would be unavailable in many cases and defining suitable features for a specific domain is also a challenging task. This inspires the second direction with many unsupervised learning methods (Erkan and Radev, 2004; Li et al., 2016; Lin and Bilmes, 2011; Wei and Gao, 2015; Woodsend and Lapata, 2012). They differ from supervised learning ones in that they do not require training data and thus are easy to adapt to new domains.

There are three major steps in a summarization system: data processing, summarization, and post-processing. Figure 1.1 summarizes the process of a summarization system with the idea is derived from Zitouni (2014).

[1]https://www.yahoo.com/news/

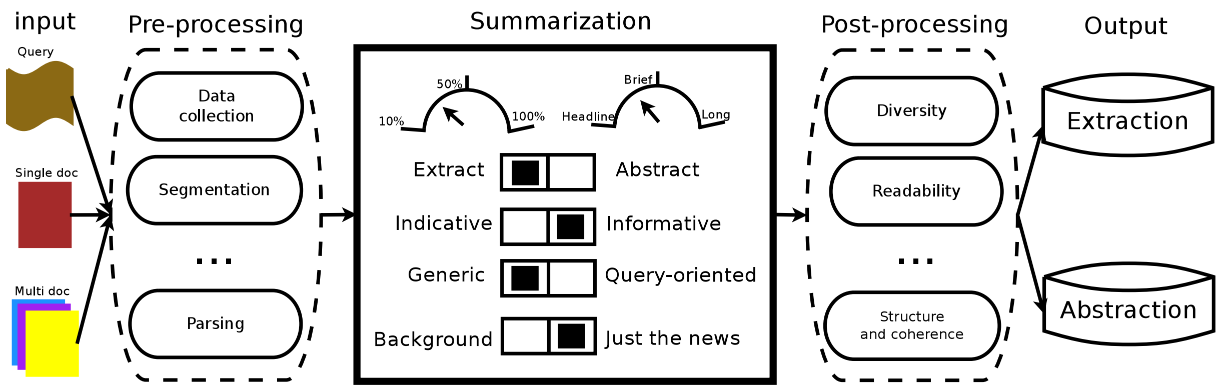


Figure 1.1: The overview of a summarization system.

From Figure 1.1 we can observe that the input of a summarization system is a single or multiple documents with or without a query. (i) The first step processes input data into an appropriate form for summarization by using several smaller modules such as segmentation or parsing. (ii) Based on different purposes, there are four main genres of summarization: extraction vs. abstraction, indicative vs. informative summarization, generic vs. query-oriented summarization, or background vs. just-the-news summarization. For the first genre, on the one hand, extraction extracts a small number of sentences in a document (or a collection) by measuring their importance. One of advantages is that summaries are grammatical because summarizers usually keep whole sentences as final outputs. However, it also eliminates other valuable information. On the other hand, abstraction generates summaries which do not fully use words or phrases in an original document. An example of abstraction is that CNN[2] provides highlights, which are short sentences written by humans for capturing main information of a document. It addresses the diversity issue of extraction, in which abstractive outputs cover all information in a document. However, it also has several challenges such as grammaticality or readability (see Section 2.1.2). While extraction achieves promising results, outputs of abstraction are still far from humans' satisfaction. For the second genre, an indicative summary normally contains the metadata (a description) of a document. Card catalog entries and movie trailers are examples of this genre. An informative summary includes informative parts of an article. Therefore, it has to incorporate pertinent elements to convey core information and omit unnecessary one (see Kan et al. (2001) for detail). For the third genre, generic summarization mainly summarizes salient information without considering users' needs. By contrast, query-oriented summarization outputs summaries based on their interests. For the fourth genre, while background summarization provides explanatory materials such as time, place for readers who lack prior knowledge of a document or domains, just-the-news only focuses on capturing the newest information by supposing that readers already know the existing event (Zitouni, 2014). (iii) The final step usually does some operations such as merging or removing near-duplicate sentences to ensure several constraints of evaluation such as diversity, readability, or grammaticality.

[2]http://edition.cnn.com

Text summarization is a challenging task of Artificial Intelligence (AI) and Natural Language Processing (NLP). Challenges come from three major aspects: language, data processing, and evaluation. For the first aspect, as we know, natural languages themselves are of complicity, which includes the ambiguity of its constituent elements such as words and sentences. It makes confusion in reading even with humans. For example, two people can have different viewpoints regarding the same sentence; or the same sentence has different meanings when it is put in different contexts. This explains that Document Understanding Conference provides four different references generated by four annotators to avoid the subjective issue when evaluating summarization systems. For the second aspect, summarization is at a text understanding level, which utilizes several low-level operations such as word segmentation, parsing, named entity recognition, etc. For English, their performance is promising but they are still developed in several languages such as Vietnamese. Finally, evaluation is not a trivial task because there are several constraints we have to consider in an evaluation scenario (refer Section 2.1.2 for more detail).

There are two major components in an extractive summarization system, named sentence scoring and sentence selection. The scoring estimates the importance of a sentence by measuring its constituent elements such as words, phrases, or trees. The estimation can use unsupervised learning approaches (Banerjee et al., 2015; Erkan and Radev, 2004; Li et al., 2016; Lin and Bilmes, 2011) or supervised learning ones (Cao et al., 2015b; Ren et al., 2017; Rush et al., 2015; Shen et al., 2007). The selection extracts summaries base on their importance. There are two main methods of the selection. The first one uses a greedy algorithm to select top-ranked sentences with the consideration of diversity. The second one uses a re-ranking algorithm to re-rank sentences. Integer Linear Programming (ILP) is an example (Cao et al., 2015b; Li et al., 2016; Woodsend and Lapata, 2010, 2012). It selects summaries based on their concepts defined by constraints which maximize global optimization. In the literature, researchers have focused on these components showing that they are essential for extractive summarization.

1.2 Summarization with Social Information

Summarization using social information is a subtask of text summarization, which considers both local information inside Web documents and their relevant social information, e.g. comments or tweets in the summarization process. The motivation of this task comes from the fact that in the context of social media, users can freely discuss events and topics mentioned in a Web document by posting their messages. For instance, Yahoo News provides both news articles and a Web interface, where readers can write their comments on an event such as *"Germanwings crash families could seek damages in the U.S.: lawyer"*. In the same time, they can post their messages on social networks such as Twitter or Facebook. After writing and posting comments, other readers can follow the event by tracking up-to-date information from both channels. Figure 1.2 shows this mutual relationship.

These user posts, also called as social information (Nguyen and Nguyen, 2016; Wei and Gao, 2014; Yang et al., 2011), have two important characteristics. Firstly, they reflect

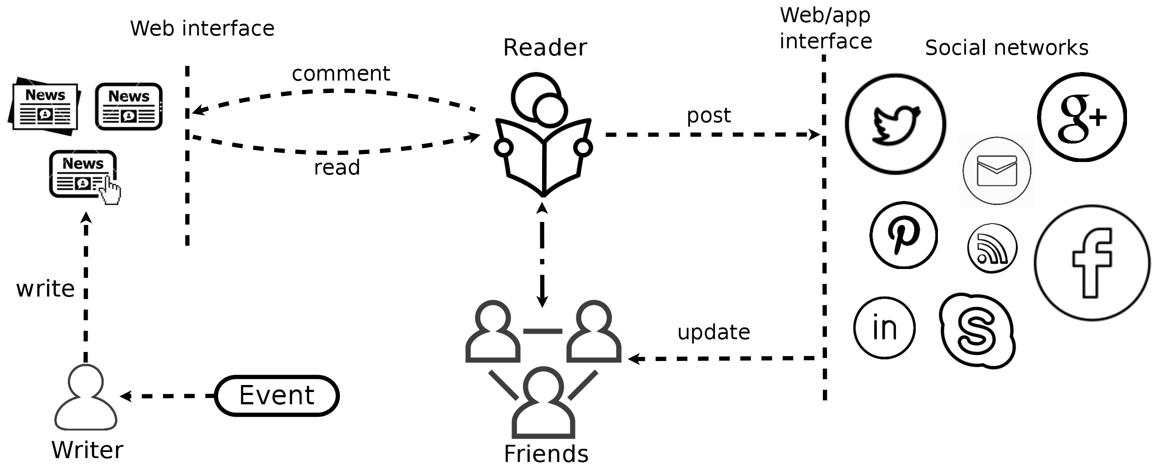


Figure 1.2: A generic scheme of the relationship between news and social media.

event content in a primary document by sharing salient words or phrases with sentences in the form of derivation or variation. Secondly, many well-written user posts can be directly extracted to enrich the meaning of selected sentences. This observation raises a challenging task of how to exploit relevant social information to enrich the importance estimation of sentences, which helps to improve the summarization of Web documents. A more formal definition can be seen in Section 2.2.1.

Summarization of Web documents by utilizing their social context is a challenging task of NLP. As mentioned, it is a subtask of text summarization, so it shares three main obstacles: the complexity of languages, data processing, and evaluation with text summarization. More importantly, its challenges also come from the concept of social context. First, with the fast growth of social media, social context is more and more complicated. It includes different kinds of data (social messages, links, relevant documents) with several factors such as users, user-generated content, social networks, and implicit networks (reply and forward). This makes a difficulty to formally define the concept of social context. Second, relationships between documents and their context are implicit and complicated; therefore, modeling these relations in a unified framework is a non-trivial task. Finally, social context is a useful data channel, providing additional valuable information which may be unavailable in main documents. However, it also contains inevitable noise (tweets are an example), which challenges to qualitatively analyze and quantitatively validate a proposed model on real-world datasets.

Compared to text summarization which only considers the inherent information of a document, summarization using social context integrates data from two channels: internal and external information. While internal information helps to model relationships of sentences inside a document, external information takes advantage of social context to support internal one in a mutual reinforcement fashion. As a result, the importance estimation of main sentences needs to be adapted. In addition, the integration of social context into the summarization process allows to incorporate various dynamic relationships, such as follower-followee relationships between users or retweeting and replying relationships between tweets (or comments, posts) from various kinds of data. This provides an opportunity to exploit rich information from social media for summarization.

Research questions Inspired by the fact that there are relationships between main documents and their social information, we state two research questions:

- **Question 1:** Whether the social information of a Web document can be used to enrich summarization? The enrichment is in two levels: (i) social information supports the scoring step to improve the importance estimation of sentences and (ii) well-written extracted social messages provide additional information which may not be usually available in primary documents.
- **Question 2:** If the social information of a Web document can be used to enrich summarization, hence which are appropriate ways to integrate such information into the summarization process?

1.3 Contributions

The objective of our research is to investigate the integration of social context to improve the importance estimation of sentences and user posts, which benefits Web document summarization. Based on the two research questions, we introduce several models which are concerned with several aspects of integration in three major axes. Figure 1.3 provides a snapshot of our thesis. We consider deep learning as an individual direction because its computation is quite different from other ones.

Under three exes, we investigate two problems which match with two steps of extractive summarization: (i) improving the importance estimation of sentences (or user posts) by exploiting social context and (ii) improving the selection of summarization. While the estimation and selection have widely investigated in traditional text summarization, they are in an early stage of the task of summarizing Web documents by using their social context with several studies (Gao et al., 2012; Li et al., 2016; Wei and Gao, 2014, 2015; Yang et al., 2011). For the first problem, we believe that the importance estimation plays an important role in a summarization system and thus improving this aspect benefits summarization systems in distinguishing summary and non-summary sentences. For the second problem, we argue that the performance of a summarization system can be still improved even it outputs high-quality estimation. In this aspect, the selection can be seen as re-ranking, which receives outputs from the estimation to re-rank sentences. Below we summarize main contributions of this thesis.

Estimating the importance of sentences by exploiting social context We first explore the importance estimation of sentences because it is a primary component of extractive summarization. To do that, we introduce six models which range in three directions: unsupervised, supervised, and deep learning, in which each model tries to formulate an aspect of relationships between Web documents and their social context.

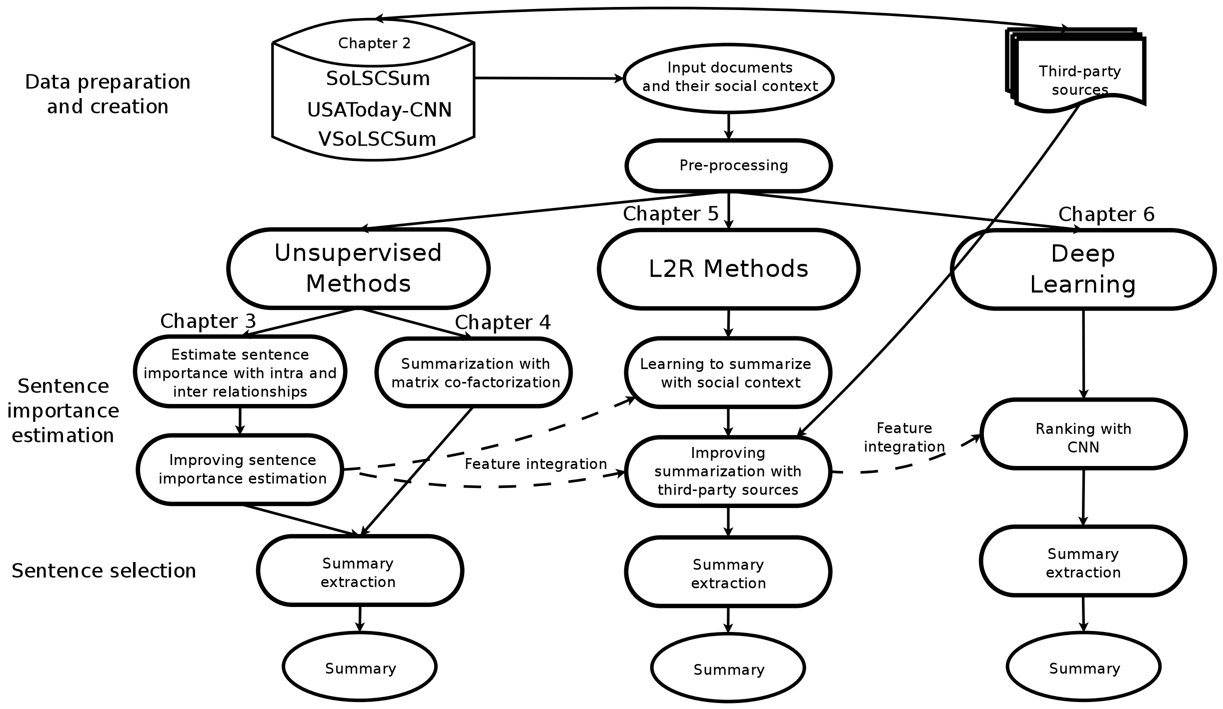

Data preparation and creation
Chapter 2
SoLSCSum
USAToday-CNN
VSoLSCSum
Input documents and their social context
Third-party sources
Pre-processing
Chapter 5
Chapter 6
Unsupervised Methods
L2R Methods
Deep Learning
Chapter 3
Chapter 4
Sentence importance estimation
Estimate sentence importance with intra and inter relationships
Summarization with matrix co-factorization
Learning to summarize with social context
Improving sentence importance estimation
Feature integration
Improving summarization with third-party sources
Feature integration
Ranking with CNN
Sentence selection
Summary extraction
Summary extraction
Summary extraction
Summary
Summary
Summary

Figure 1.3: The map of our thesis. Dot lines connecting chapters show extension.

Promising results of previous unsupervised methods (Gao et al., 2012; Li et al., 2016; Wei and Gao, 2015) inspire us to first examine the estimation by introducing three unsupervised ranking models which are in the first path of Figure 1.3. Different from prior work, here we argue that the importance of a sentence can be estimated by combining its intra-relations and inter-relations. The intra-relations of a sentence tell us how it is important in a document and intra-relations measure its importance in social context. To model these relations, we investigate a rich set of features for calculating similarity between two texts (Nguyen et al., 2015). They are used by our first model with promising results (Nguyen and Nguyen, 2016, 2017). From this, we extend the first model to the second one by using semantic similarity to present inter-relations (Nguyen et al., 2018a). This extension formulates the behavior of readers, who tend to borrow salient words in sentences to create their social messages. In this aspect, inter-relations measure the importance of a sentence by using semantic similarity among words instead of using lexical-based similarities. The third model captures common topics between documents and their social context. The motivation is that social messages usually reflect the main content of an original document by sharing common words or phrases denoted in the form of common hidden topics. To exploit these topics, we present a matrix co-factorization method which estimates the importance of sentences based on their influence on topics in a joint optimization fashion (Nguyen et al., 2017a). We organize these models in two chapters in the same path in Figure 1.3 because they are unsupervised learning methods.

For the second direction, we explore the estimation by presenting two supervised learning models shown in the second path of Figure 1.3. The main idea of this direction is that we formulate the sentence scoring step in form of learning to rank and use feature engineering to improve the estimation. Our models learn from training data to estimate the importance of sentences in testing data. To define features, we first include indicators from prior work (Svore et al., 2007; Wei and Gao, 2014) and then present new features organized in two classes: local and social features. Especially, we consider relevant news articles of primary documents as a factor of social context and then design a small set of features (third-party features) for taking advantage of such factor. By combining these indicators, our models obtain promising results (Nguyen et al., 2018b, 2016d, 2017b,c).

The final direction adapts deep learning for estimating the importance of sentences. To do that, we first present a model which is a variation of Convolution Neural Networks. We employ such networks to learn representation from data automatically and then integrate social context denoted in the form of features derived from the second direction. By using logistic regression, our model outputs scores of sentences for the selection step.

We also investigate several aspects which help to understand the operation of our models. We show the contribution of each feature and feature groups by two methods. First, we perform an ablation experiment that removes one feature at a time. This experiment measures the influence of each feature in our models. Second, we also do the ablation based on feature groups to estimate the contribution of each group. Besides such kind of investigation, we also observe other important aspects such as the relationship between the number of social messages and summarization performance, the impact of topics, training time, etc. Our investigation benefits the literature in providing useful information which facilitates researchers in dealing with such kind of summarization.

During the estimation, we present Web documents and their social context in a mutual reinforcement relationship. It means that when estimating the importance of sentences, social information is exploited to enrich information in sentences. Similarly, sentences are also considered as social information when doing the estimation of social messages. In this view, our representation is different from previous work (Gao et al., 2012; Li et al., 2016; Wei and Gao, 2014, 2015; Yang et al., 2011).

Improving the quality of sentence selection Sentence selection is the second step of an extractive summarization system. In the literature, many studies use a greedy algorithm to extract top-ranked sentences as a summary. In this thesis, we first also employ this method to select summaries. Besides, we explore two other methods for selecting important sentences. The first method is an extension of a simple sentence selection model based on ILP Woodsend and Lapata (2010). We adapt it to our task by presenting new constraints to integrate social context. Instead of directly using Cosine scores from its ranking step, it receives estimation from our models to re-rank sentences. We argue that summary sentences can be simultaneously selected with representative user posts. To do that, we introduce a new objective function to simulate our argument in a unified model (Nguyen et al., 2018a). For defining constraints, we consider concepts as sentences instead of using bi-grams (Li et al., 2016; Woodsend and Lapata, 2010). The second method is a kind of system combination (Hong et al., 2015). We utilize the efficiency from three different learning to rank summarizers to do majority voting on their outputs (Nguyen et al., 2018b). In our setting, these methods are a type of re-ranking. We believe that using re-ranking can potentially profit the selection of summaries.

Datasets The bottleneck of social context summarization is that released datasets do not contain both annotation and social information. Let's take well-known datasets from Document Understanding Conference (DUC)[3] as an example. DUC 2001, 2003, and 2004 are standard datasets to evaluate summarization systems. They contain news articles and standard references written by humans. However, they do not include social information. This challenges systems which use such information to produce summaries. In social context summarization, Yang et al. (2011) created a dataset containing news articles and tweets by using a crowdsourcing system. However, it is now unavailable. Other datasets which contain Web documents and tweets are also released (Cao et al., 2015a; Gao et al., 2012; Wei and Gao, 2014). They, however, are created without human annotation, e.g. 0 and 1, challenging supervised methods, e.g. Support Vector Machines (SVM).

To facilitate supervised methods, we release three datasets in two languages, English and Vietnamese (Nguyen et al., 2016a,c, 2017c). One of them is derived from Wei and Gao (2014). Our datasets have two essential characteristics: (i) all sentences and user posts are manually labeled by humans and (ii) they include both documents and user posts in that they can be used in a mutual reinforcement fashion for improving the estimation.

1.4 Dissertation Structure

We organize the structure of this thesis into seven parts, in which four major chapters base on three directions in Figure 1.3. We summarize each chapter as follow.

Chapter 1 is an introductory chapter. We first mention text summarization and then introduce Web document summarization using social context with motivation, research questions, followed by our contributions, and dissertation structure.

Chapter 2 provides preliminaries used for this thesis. We start by showing a brief history of text summarization, and then we describe evaluation, including ROUGE, and significant test used to compare summarization systems. We next present our summarization task using social information, which includes necessary definitions, literature review, and data preparation. The data preparation bases on two papers [9, 12].

Chapter 3 presents two unsupervised models, which exploit the social context of a document to estimate the importance of sentences. We start with the first model for modeling intra-relations and inter-relations in Section 3.1 and introduce its extension in Section 3.2. We provide the detail of experiments and discussion in each section. This chapter bases on four papers [1, 4, 10, 11].

[3]http://duc.nist.gov/data.html

Chapter 4 introduces our matrix co-factorization model for capturing common topics between documents and their social context. We first show a brief description of non-negative matrix factorization and then describe our model which encodes common hidden topics into a joint optimization algorithm. We also provide experimental results and discussion. This chapter bases on one paper [6].

Chapter 5 focuses on investigating features for two learning to rank models. We describe our first model which uses features extracted from user posts in Section 5.1. We next introduce an extended model of the first one in Section 5.2, in which it considers third-party sources as a new factor of social context. We also report experimental results corresponding to each model. This chapter relies on five papers [2, 3, 7, 8, 14].

Chapter 6 describes our deep learning model which bases on Convolutional Neural Networks for estimating the importance estimation. We first describe the background of these networks and then introduce our model which combines both local and social information for improving the representation of sentences. We also report results, discussion, and analyses of to our model. This chapter bases on two papers [5, 13].

Chapter 7 shows conclusions and main findings of this research. It also reveals future directions in exploiting social information to enrich the quality of Web document summarization.

References and publications follow Chapter 7.

Chapter 2

Preliminaries

This chapter provides fundamental knowledge for reading this thesis. We first review text summarization and its evaluation in Section 2.1. We next introduce definitions of our summarization task using social context, related work, and preparation for our models in Section 2.2.

2.1 Text Summarization Overview

2.1.1 A brief history

As mentioned, text summarization is a challenging task, which has intensely studied in a long history. While it is impossible to discuss all related work, this section tries to make a picture of text summarization by reviewing major directions.

DUC and TAC Document Understanding Conference (DUC)[1] between 2001 and 2007 and Text Analysis Conference (TAC)[2] in 2008 and 2009 are prior major venues of text summarization. They were organized by National Institute of Technology (NIST), which provides several datasets for evaluating summarization systems in both generic and focused-document summarization. NIST aims to deal with multi-document summarization by defining an event as a topic which includes ten documents. Documents on the same topic have to mention the same event. Each topic also consists of 3-4 references written by humans. Automatic summarization systems produce summaries up to a certain length.[3] These datasets have widely used in text summarization.

Traditional methods To the best our understanding, Luhn (1958) is the first researcher who states the task of text summarization. The author extracts summaries by measuring their significant components, e.g. high-frequency content words or sentence

[1]http://duc.nist.gov/data.html

[2]https://tac.nist.gov/data/

[3]The number of words is 100 or the length of a summary is 665 bytes.

location. In last decades, many researchers applied machine learning to text summarization. The authors define it as an extraction task presented in the form of a binary classification problem, in which label 1 denotes summary, and 0 represents non-summary sentences. For example, Yeh et al. (2005) used classification and latent semantic analysis (LSA) to build summarizers. The first approach utilizes a set of features for ranking the position of sentences to emphasize their difference. The second method uses the semantic matrix of a document to extract summaries. The highest F-score of these methods is 0.49 with 30% compression. Shen et al. (2007) exploited the sequence aspect in a document by proposing a set of features, e.g. Cosine similarity of a sentence with previous or next sentences (N=1,2,3) and then employed Conditional Random Fields (CRF) (Lafferty et al., 2001) to train a binary classifier for selecting summary sentences. This method achieves a ROUGE-2[4] of 0.483 and a F-score of 0.419 on DUC 2001. Hong and Nenkova (2014) improved the estimation of word importance for selecting sentences. The authors argue that the importance of sentences can be measured by observing their constituent words with many hand-crafted features. Their regression model achieves competitive results on DUC datasets. Later, Hong et al. (2015) presented a pipeline framework for combining summarizers. It includes two steps: candidate extraction and combination. A supervised selection model helps to obtain promising performance on DUC and TAC datasets.

Another approach for text summarization is graph-based methods. Among those, LexRank (Erkan and Radev, 2004) and TextRank (Mihalcea and Tarau, 2004) are perhaps the most popular. They build a similarity graph, in that its vertices are either sentences or phrases and edges are semantic or lexical similarity among text unit vertices. More precisely, LexRank creates a sentence similarity graph, then extracts important sentences based on the concept of eigenvector centrality. TextRank uses text structure inside documents and creates a central key phrase graph to extract summaries.

Lin and Bilmes (2011) designed a set of functions for document summarization. Each function guarantees representativeness and diversity. By using monotone nondecreasing and submodular functions, this method is the best in term of Recall and F-score over DUC 2004 to 2007. Woodsend and Lapata (2010, 2012) formulated sentences in the form of concepts and defined an objective function with a set of constraints for generating highlights of Web documents. The authors represent concepts as phrases extracted from dependency trees, from that each constraint covers a summary aspect. The ROUGE-1 and F-score of this method is 0.25 on a story highlight extraction dataset collected from CNN.[5] Wang et al. (2008) presented a model which bases on symmetric non-negative matrix factorization for multi-document summarization. This model first computes sentence-sentence similarities using semantic analysis for constructing a similarity matrix. Subsequently, it analyzes the matrix for grouping sentences into clusters. The most informative sentences from each group are selected to form a summary. Their method achieves improvements on DUC 2005 and 2006.

All traditional methods are different from our models in which they only consider internal information of a document such as sentences while ignoring its relevant information

[4]https://en.wikipedia.org/wiki/ROUGE_(metric)

[5]http://edition.cnn.com

such as user posts or related Web documents. In the context of social media, we argue that the related information of a Web document can be exploited to improve the importance estimation of sentences. We make a different view of summarization by integrating the social context of a Web document into the summarization process.

Deep learning methods The recent success of deep learning attracts researchers in applying this technique to extractive and abstractive summarization with several sophisticated models. For example, Cao et al. (2015b) developed a ranking framework relying on Recursive Neural Networks to rank sentences for multi-document summarization. It captures the hierarchy of sentences from phrases extracted from parsed trees. The authors use regression which allows their framework to learn hidden features from data. These features are combined with surface features extracted from raw texts. Summaries are extracted by using a greedy or an ILP-based method with very competitive ROUGE-scores on DUC 2001, 2002, and 2004. Chopra et al. (2016) presented a neural attention model for sentence summarization. It uses an attentive encoder followed by a conditioned Recurrent Neural Network (RNN). Results show the efficiency of this model. Cheng and Lapata (2016) proposed a hierarchical model for extracting sentences and words. It uses a Convolutional Neural Network (CNN) for constructing the representation of each sentence, followed by an RNN to capture the vector representation of a document based on its sentence vectors. In decoding, the label of each sentence and word is assigned by training the network to maximize the likelihood of all sentence labels given in the input document vector. The authors mention that their model can extract sentences and words. Results indicate that the sentence extractor obtains competitive results, but the word extractor acquires low scores due to ROUGE-scores computation. Zhang et al. (2017) adapted CNN for extractive summarization by presenting a multiview concept. Their model uses word embedding to map sentences into a deeper representation level and uses a multiview CNN to learn features for ranking sentences. Experimental results on DUC datasets show that this model is better than advanced methods. Ren et al. (2017) present a deep learning model, which uses Bi-CNN Cao et al. (2015c) for modeling each sentence and LSTM Hochreiter and Schmidhuber (1997) for modeling relationships among sentences for extractive summarization. By using ranking, this model is competitive on DUC datasets. These methods, however, are similar to traditional ones in which they ignore social information from readers. We enrich summarization in the literature by considering a challenging task, which integrates the social context of a Web document to improve the importance estimation, which benefits to produce high-quality summaries.

2.1.2 Summarization evaluation

Summarization evaluation is a challenging task because there are several aspects influencing the performance of a summarization system. A high-quality summary depends on the context of the task and the intention of audiences. For instance, the same summary can have different scores for extractive and abstractive summarization. The essentiality of evaluation is still an open problem, which inspires a number of considerable studies.

Manual evaluation

Manual evaluation requires human involvement to estimate the quality of summaries in several aspects such as content or grammaticality. After reading an original document with an instruction, annotators judge each summary, e.g. giving each summary a score from 1 to 5 without respect to any particular task or goal. A summarization system is better than others if it achieves a higher average rating score.

There are several criteria which make a high-quality summary. For example, they are adjusted based on requirements of DUC or TAC in each year; however, the instruction has to include at least four standard conditions:

- *Informativeness*: estimates the amount of useful information in a summary which helps to infer the original event. It only requires annotators to focus on words themselves rather than linguistic quality.

- *Grammaticality*: supports the first criteria, in which it balances between information in summaries and the grammaticality among constituent elements in each summary. It reduces the score of sentences if they break some rules such as ungrammaticality, capitalization errors, or make a difficulty in reading.

- *Non-redundancy*: states that summaries should be no repetition. It considers both sentence and word (or phrase) levels. For example, researchers usually use a Cosine similarity to remove sentences; those are similar in term of content when using a greedy method for selecting summaries.

- *Structure and coherence*: ensure a summary is well-structured and well-organized. By combining with the grammaticality, they require that a summary is easy to read in both grammatical and structural aspects.

Many researchers have employed manual evaluation to evaluate abstractive summarization systems (Banerjee et al., 2015; Bing et al., 2015; Chopra et al., 2016; Tan et al., 2017a; Woodsend and Lapata, 2010, 2012). Recently, crowdsourcing platforms such as Amazon Mechanical Turk[6] also facilitate such kind of evaluation.

Automatic evaluation

Manual evaluation provides several viewpoints regarding a summary. It has proved to be efficient in evaluating abstractive summarization systems. However, it still exists two issues. Firstly, it requires humans, which, in many cases, are costly and labor-expensive. Secondly, judgment is subjective; even annotators are given explicit instructions. These issues inspire automatic evaluation metrics. Among those, ROUGE (Recall-Oriented Understudy for Gisting Evaluation) (Lin and Hovy, 2003) is perhaps the most popular for evaluating extractive summarization systems.

[6]https://www.mturk.com/mturk/welcome

ROUGE is the most common method for comparing summarization systems. It computes $n-$grams word overlapping between references and extracted summaries.

$$ROUGE - N = \frac{\sum_{s \in S_{ref}} \sum_{gram_n \in s} Count_{match}(gram_n)}{\sum_{s \in S_{ref}} \sum_{gram_n \in s} Count(gram_n)} \tag{2.1}$$

where n is the length of n-gram, $Count_{match}(gram_n)$ is the maximum number of n-grams co-occurring in a candidate summary and references, $Count(gram_n)$ is the number of n-grams in references. Table 2.1 shows an example.

Table 2.1: An example of a reference and an output.

Reference	The cat was under the chair.
Output	The tiny little cat was found under the big funny bed.

From this table, we can compute ROUGE-1 and ROUGE-2 as the following:

- ROUGE-1: counts uni-grams word overlapping between the output and the reference. From the reference we have {*the, cat, was, under, the, chair*} and from the output we have {*the, tiny, little, cat, was, found, under, the, big, funny, bed*}. The number of uni-gram word overlapping is 5 and the number of uni-gram in the reference is 6. Therefore, ROUGE-1 recall is $\frac{5}{6} = 0.83$.

- ROUGE-2: counts bi-grams word overlapping between the output and the reference. The bi-grams of the reference are {*the cat, cat was, was under, under the, the chair*} and the bi-grams of the output are {*the tiny, tiny little, little cat, cat was, was found, found under, under the, the big, big funny, funny bed*}. The number of bi-grams overlapping are 2 and the number of bi-grams in the reference is 5. Therefore, ROUGE-2 recall is $\frac{2}{5} = 0.4$.

In practice, researchers usually use the `ROUGE-1.5.5` toolkit, which provides variety options with more complicated combinations than the original $n-$grams word overlapping model. For example, ROUGE-SU4 counts uni-grams and pairs of words up to four intervening words. In fact, the length of summaries and references can be different,[7] therefore, the toolkit also provides F-score to balance between recall and precision.

ROUGE-scores are appropriate to compare summarization systems based on word overlapping, but it is inefficient for abstractive summarization. This is because summaries from abstractive summarization systems may not completely be similar to references in term of words. However, all models in this thesis are extractive; we, therefore, select ROUGE as the metric for our evaluation.

[7]It is different from DUC and TAC, which usually constraint both summaries and references within 100 words or 665 bytes.

Semi-automatic evaluation

As mentioned, manual and ROUGE evaluation exist some issues. On the one hand, manual evaluation provides evidence to measure the quality of a system in several aspects (including grammaticality), but it is labor-expensive. On the other hand, ROUGE does not requires humans, but it relies on word overlapping without any consideration of structured or grammatical aspect. Therefore, Nenkova and Passonneau (2004) presented Pyramid, which attempts to bridge the gap between manual and ROUGE evaluation. It first requires *semantic content units*, e.g. phrases, sentences, which are manually extracted from references. After that, it scores systems based on relevant facts of units in summaries. They reported that it shows considerably stronger correlation than ROUGE.

Significant test

A common problem when comparing two summarization systems A and B based on their ROUGE-scores is that whether A's ROUGE-scores are statistical significant better than B's ROUGE-scores. A pair $t-$test, in this case, can be used to answer this question. It compares two population means of A and B in the same setting. In the evaluation of summarization, ROUGE-scores of A can be paired with ROUGE-scores of B.

Suppose we have a dataset split into 5-folds (observation samples $n = 5$), $RG1_A$ is the ROUGE-1 of systems A, $RG1_B$ is the ROUGE-1 of system B. To test the null hypothesis that the true mean difference is zero, we can follow the following procedure:[8]

- Compute the difference of ROUGE-scores of each fold: $d_i = RG1_A - RG1_B$.
- Calculate the mean difference, $\bar{d}$.
- Compute the standard deviation of the differences, s_d, to calculate the standard error of the mean difference, $SE(\bar{d}) = \frac{s_d}{\sqrt{n}}$.
- Calculate the $t-$statistic $T = \frac{\bar{d}}{SE(\bar{d})}$.
- Use $t-$distribution tables to compare with the value of T to show the $p-$value for the pair $t-$test.

Table 2.2 shows an example of ROUGE-1 between system A and system B.

Following the procedure, the mean and standard deviation of the differences are $\bar{d} = 0.030$, $s_d = 0.024$, then $SE(\bar{d}) = \frac{s_d}{\sqrt{n}} = \frac{0.024}{\sqrt{5}} = 0.011$.
So, we have: $t = \frac{\bar{d}}{SE(\bar{d})} = \frac{0.030}{0.011} = 2.727$ on $df = 4$ (degree of freedom). The value of df in $t-$distribution tables[9] gives $p = 0.0512$, indicating that system A is nearly significantly better than system B. In fact, if $p \leq 0.05$ we conclude that system A achieves a statistical significant improvement over system B. A 95% confidence interval for the true mean difference is:

$$\bar{d} \pm t^* \times \frac{s_d}{\sqrt{n}}, \text{ or, equivalently } \bar{d} \pm (t^* \times SE(\bar{d})) \tag{2.2}$$

[8]Another example can be seen at: http://www.statstutor.ac.uk/resources/uploaded/paired-t-test.pdf
[9]http://www.sjsu.edu/faculty/gerstman/StatPrimer/t-table.pdf

Table 2.2: An example of pairwise t-test of ROUGE-1 between two systems with $n = 5$.

Fold	System A	System B	Difference
1	0.401	0.378	0.023
2	0.325	0.282	0.043
3	0.367	0.375	-0.008
4	0.398	0.361	0.037
5	0.427	0.370	0.057

where t^* is the 2.5% point of the t-distribution on $n-1$ degrees of freedom. Applying to the above example, we have a mean difference is 0.030. The 2.5% point of the t-distribution with 4 degrees of freedom is 2.776. The 95% confidence interval for the true mean difference is: $0.030 \pm (2.776 \times 0.011) = 0.030 \pm 0.030536 = (-0.0005, 0.060)$. This confirms that even the difference score is nearly significant, the gap is small. We can be 95% sure that the true mean increases somewhat better 0.06%. In this thesis, the pairwise $t-$test is conducted by using a *scipy* package.[10]

2.2 Summarization with Social Context

This section provides a background of Web document summarization using social context. It first defines definitions and next reviews the literature.

2.2.1 Definitions

Social information is defined as relevant user posts (comments or tweets) generated from readers after reading a Web document. Formally, given a document d and a set of users $U = \{u_1, ..., u_n\}$, who read d, social information of d is UP_d created by $d \xrightarrow[U]{posting} C$ or $d \xrightarrow[U]{posting} T$, where C or T is a set of comments or tweets, $\xrightarrow[U]{posting}$ presents that C or T is produced by U. In this thesis, the terms of social information, user posts, user-generated content are mutually used with the same meaning.

Social context There are several definitions of social context corresponding to different kinds of additional data used to support sentences in main documents (Amitay and Paris, 2000; Delort et al., 2003; Sun et al., 2005; Svore et al., 2007). We adapt the definition of Yang et al. (2011) to define the social context of a Web document. Given a Web document d, its social context is C_d presented by $\langle S_d, UP_d, U_d \rangle$, where S_d is a set of sentences in a document d, UP_d is a set of relevant tweets or comments of d written by users U_d. It is possible to consider U_d as a factor of the context; however, it is eliminated because it is an implicit factor. We leave the exploration of using U_d as a future task.

[10]https://docs.scipy.org/doc/scipy-0.19.0/reference/generated/scipy.stats.ttest_ind.html

The summarization task is to select important sentences and representative user posts by using C_d giving a primary document d. While selecting important sentences is widely used in many extractive methods, extracting representative user posts makes a different view compared to traditional ones. Representative user posts are those that also reflect the most important content of a document. The intuition of this extraction is that while extracted sentences include salient information, user posts can also provide new information from users, which may be unavailable in main documents. The definition of our summarization task indicates that the importance estimation is now for both sentences and user posts, instead of only for sentences as the traditional task.

Table 2.3 shows an example. We can observe that: (i) selected sentences can be seen as a summary because it contains essential information of the event and (ii) extracted tweets also reflects the event and includes the viewpoint of readers. Sentences and tweets share common topics denoted by bold words, but they do not entirely match in term of content. Also, from extracted tweets, we can obtain additional information which is unavailable from extracted sentences such as the time of the event, which happened last week. By combining with the time in the primary document, i.g. Friday, we can relatively infer the correct time of this event, which happened on Friday, last week.

2.2.2 Literature review

Using the support from social media for summarization has been previously studied by several approaches based on different kinds of social information. As far as we know, Amitay and Paris (2000) are the first researchers who pick sentences from the hyperlinks of a Web document as a summary. The authors build *InCommonSense*, which contains two steps: hypertext retrieval and description selection. In the retrieval step, given a target document d, their system retrieves a set of related hypertext based on four distinct patterns represented in the form of links. In the selection step, a classifier is built to select the best description from retrieved hypertext. They use humans to evaluate the output of InCommonSense compared again search engine results. The mean (the average values from 1 to 5 rated by users) of InCommonSense is 4.71 compared to 4.14 of AltaVista-style (top X words from the document), and 4.13 of Google-style (query terms are highlighted, and surrounding context is taken). The system, however, only selects one sentence from linked texts, that may lack the content of final summaries. Late, Delort et al. (2003) considered whole linked articles as the context of a primary Web document instead of using paragraphs including hyperlinks as Amitay and Paris (2000). The authors proposed two context summarization algorithms based on similarity measurements. The first method combines the content and context of a document and the second one only considers the context. The best result is 0.45 in term of similarities in the content and context summarization. However, similar to Amitay and Paris (2000), their system extracts sentences from context segments; therefore, they may not completely capture the content of a Web document compared to internal sentences.

Sun et al. (2005) introduced a system which uses the help of click-through data retrieved from search engines to extract salient sentences in a Web document. This study bases on an assumption that query keywords from users typed on search engines usually reflect

Table 2.3: A summary example generated from the document *"President shares story of hero who tended to friend in theater shooting"*.

Highlights
President Obama tells how a woman helped her wounded friend in the Aurora theater.
Allie Young was shot in the neck but is going to be fine because of her friend, Obama says.
Stephanie Davies pulled Young into an aisle and put pressure on her friend's wound, he says.
Obama: Wounded Young urged friend to run, but she refused.
Sentences
Seconds after a gunman shot Allie Young in the neck in a crowded Colorado theater, a friend, Stephanie Davies, pulled her into an aisle, put pressure on her wound and dialed 911.
Young told her friend to run, but Davies refused.
President Barack Obama gave that account Sunday night in Aurora, praising Davies and others for heroism amid chaos and bloodshed in the shooting that killed 12 and wounded 58.
The women had settled in alongside others `early Friday` for a midnight showing of "The Dark Knight Rises," the latest installment in the Batman series.
Suddenly a gunman threw canisters only a few feet from where the pair sat.
Young, 19, instinctively stood to act or warn others. A shot ripped into her neck. She collapsed, blood spurting from the wound, Obama said.
Instead of running or hiding, Davies, 21, pulled Young into the aisle and put pressure on the wound with one hand and dialed 911 with the other, Obama said.
"I don't know how many people at any age would have the presence of mind that Stephanie did, or the courage that Allie showed," the president said. "hey represent what's best in us, and they assure us that out of this darkness, a brighter day is going to come."
When the SWAT team arrived, Davies helped carry Young to an ambulance.
Because of Davies' actions, Young is going to be fine, the president said.
Obama flew to Aurora on Sunday evening and met with families at a local hospital for more than two hours before delivering brief remarks to the media.
Tweets
shares story of a 21-year-old who saved her best friend & life in
President shares story of hero who tended to friend in theater shooting: President Obama tells how a woman helpe.
Top Story via CNN: Theater hero saved friend from dying - Speaking in Colorado after last week & theater shooting.
Obama talks about theater shooting hero (via cnn) so unreal! There & a real hero.
President shares story of hero who tended to friend in theater shooting What an awesome friend to have.
Stories of heroism during Colorado shooting via CNN heal amidst stories of terror.
Theater hero saved friend from dying: Speaking in Colorado after last week & theater shooting, President Obama.
Woman helps wounded friend in theater via cnnbrk.
& shares story of hero who tended to friend in theater shooting & via cnn.
More heroic accounts from CNN about the President & visit to this weekend:
Sent from the CNN App for Android Obama talks about theater shooting hero.
Speaking in Colorado after last week & theater shooting, President Obama highlighted the heroic acts of Stephani
Obama talks about theater shooting hero. Amazing story !!!
Obama talks about theater shooting hero (via cnn) this is one amazing friend.
VIDEO BarackObama talks abt hero Stephanie Davies helped friend Allie Young.
....
Sentence selection
Seconds after a gunman shot **Allie Young** in the neck in a crowded **Colorado** theater, a friend, **Stephanie Davies**, pulled her into an aisle, put pressure on her wound and dialed 911.
Instead of running or hiding, **Davies**, 21, pulled **Young** into the aisle and put pressure on the wound with one hand and dialed 911 with the other, Obama said.
President Barack Obama gave that account Sunday night in Aurora, praising **Davies** and others for heroism amid chaos and bloodshed in the shooting that killed 12 and wounded 58.
"I don't know how many people at any age would have the presence of mind that Stephanie did, or the courage that Allie showed," the **president** said.
Tweet extraction
VIDEO **BarackObama** talks abt hero **Stephanie Davies** helped friend **Allie Young**.
Stories of heroism during **Colorado** shooting via CNN heal amidst stories of terror.
Speaking in Colorado after `last week` & theater shooting, President Obama highlighted the heroic acts of Stephani
President shares story of hero who tended to friend in theater shooting: **President Obama** tells how a woman help.

the content of a Web document. From this, the authors present two methods using an adaptation of significant words (Luhn, 1958) and latent semantic analysis (Gong and Liu, 2001). ROUGE-1 is 0.55 on DAT1 and 0.20 on DAT2 , two their datasets. This method, however, faces two challenging issues: (i) there are no links from a new Web page to the older ones and (ii) pages which Web users click on may be irrelevant to their interests.

User-generated content such as comments has also used to enhance the estimation in extracting summaries. Delort (2006) clustered comments by using feature vectors and selects summary sentences based on their links with clusters. This method achieves 50% extracted matches corresponding to post summaries. Hu et al. (2008) extracted representative sentences that best represent topics discussed among readers in blog posts. The authors first derive salient words denoted in three graphs: topic, quotation, and mention from comments. Summary sentences are next selected by calculating the distance from each sentence to these graphs. This method acquires a ROUGE-1 of 0.64 and an NDCG[11] of 0.60 on their datasets. This approach, however, only picks up sentences in a blog post while ignoring relevant information from comments. Lu et al. (2009) studied the rated aspect summarization of short comments to help users for better understanding the discussions of a target object. The authors proposed a model containing three steps: aspect discovery and clustering, aspect rating prediction, and representative phrase extraction. Their method gets a precision of 0.592 and a recall of 0.637 in phrase extraction on their dataset. Since this approach is used to extract summaries of a target entity, adapting it for Web document summarization is still an open question due to the existence of many entities in a Web document, e.g. person or organization name.

Social messages, e.g. tweets from Twitter have also widely exploited to support the estimation. Yang et al. (2011) denoted sentences and tweets in a dual wing factor graph model to formulate their relationships. The authors employed SVM (Cortes and Vapnik, 1995) and CRF (Lafferty et al., 2001) as preliminary steps to generate weights of the graph. For extraction, they use a ranking method, which approximates an objective function to select both important sentences and tweets as a summary. The ROUGE-1 and ROUGE-2 of this model are 0.615 and 0.500 on five collected datasets. Wei and Gao (2014) train a learning to rank summarizer with rich features represented in three groups: local sentence, local tweet, and cross features for news highlight extraction. It selects top m sentences and tweets after ranking to form a summary. ROUGE-1 is 0.292 and 0.295 for sentence and tweet extraction, respectively. By contrast, several unsupervised summary methods have also proposed. For example, Gao et al. (2012) introduced a cross-collection topic-aspect modeling, which is used as a preliminary step to create a bipartite graph. The graph is exploited by co-ranking to select sentences and tweets for multi-document summarization. ROUGE-1 is 0.55 for sentence and 0.67 for tweet selection. However, the diversity of topics in tweets from social users could negatively affect this model. Wei and Gao (2015) address the need of training data (Wei and Gao, 2014; Yang et al., 2011) by proposing a model which is a variation of LexRank (Erkan and Radev, 2004). It uses auxiliary tweets for building a heterogeneous graph of random walks, which can be used to rank sentences for single-document summarization. Its ROUGE-1 is 0.298 when combing summary sentences and tweets. Li et al. (2016) proposed an ILP-based extraction model by exploring relevant public posts from Facebook of news articles to

[11] https://en.wikipedia.org/wiki/Discounted_cumulative_gain

improve the importance estimation and selection. The authors define different methods to embed information in public posts to estimate bi-grams weights used by ILP models. Experimental results on news articles and Facebook public posts show the efficiency of this method.

2.3 Preparation

2.3.1 Datasets

As mentioned, well-known datasets for document summarization are DUC and TAC. However, they challenge our task due to the lack of social information. We, therefore, prepared three other ones in two languages, English and Vietnamese.

SoLSCSum

We created a dataset with our motivation that there are few published datasets for our task. Here we summarize the creation of SoLSCSum two steps: data collection and annotation. More detailed instruction can be seen in Nguyen et al. (2016c).

Data collection To create this dataset, we first crawled up-to-date news articles from Yahoo News in May 2015. We select Yahoo News because it allows readers to discuss events by writing their comments, which are more formal than tweets from Twitter. Our crawler collects pages from this site and parses them to obtain raw data by removing HTML tags. Due to our purpose, we only extract title, sentences, and comments of a document. In parsing, sentences and comments are split into single ones. We also ignore links in comments. We finally retrieve 157 open-domain articles along with 3462 sentences and 25,633 comments.

Data annotation We asked two annotators to annotate this dataset in two rounds: data annotation and cross-validation. For annotation, annotators follow our instruction to give a label (summary or non-summary) for each sentence and comment.

- Each selected sentence or comment has to reflect the content of a document.
- The number of selected instances are no less than six for sentences and 15 for comments (less than 30% of average sentences per document). The total number is no more than 40, including both sentences and comments.

The final label of a sentence was made based on the cross-validation of annotators. For example, given a sentence is already marked as a summary label by the first annotator, if the second one also gives the same decision, it final label is summary. If they conflict, the label of the prior one is kept. The Cohen's Kappa[12] between two annotators is 0.5845 with 95% confidence, indicating that their agreement is moderate.

[12] http://graphpad.com/quickcalcs/kappa1.cfm

After validating, we used sentences and comments with summary labels to form references. There are three reasons behind our formulation. First, Yahoo News does not provide highlights to form references. Second, selecting important sentences as references naturally reflect our task, which is extractive summarization. Finally, creating abstractive references is a labor-expensive task, which is out of this thesis.

USAToday-CNN

We derived a dataset used for new highlight extraction from Wei and Gao (2014) for our problem. It contains 121 events along with 455 highlights and 78,419 tweets. Documents were collected from USAToday and CNN and their relevant tweets were crawled from Twitter. Since this dataset includes many similar tweets, before annotating, we removed near-duplicate tweets by using Simpson method:

$$simp(A, B) = 1 - \frac{|S(A) \cap S(B)|}{min(|S(A)|, |S(B)|)} \tag{2.3}$$

where $S(A)$ and $S(A)$ are the set of words of A and B. Two tweets t_i and t_j are similar if $simp(t_1, t_2) \leq 0.25$.

We asked two annotators to give weak labels for each sentence and tweet via our annotation page.[13] Weak labels (0 and 1) mean that they are only used for training, not for evaluation because the evaluation uses highlights as references. To facilitate annotators, we provided a Cosine score for each sentence and tweet. It is the maximal score between this sentence and highlights. Annotators make a decision based on both content reflection and Cosine suggestion. We used the same instruction of SoLSCSum, but the number of sentenced sentences or tweets are no fewer than four (each document usually includes 3-4 highlights) and no more than 15 in total. Cohen's Kappa between annotators is 0.617 with 95% confidence interval, showing that their agreement is good.

VSoLSCSum

VSoLSCSum is a Vietnamese dataset created for validating our models in a non-English language. We followed the creation of SoSLCSum to create this dataset. The more detailed instruction can be seen in Nguyen et al. (2016a).

Data collection We first identified 12 special events appearing on Vietnamese Web sites such as Vietnamnet[14] in September 2016. We empirically assigned a noun phrase keyword which reflects a main object for each event. Based on these keywords, our crawler retrieves their relevant Web documents and removes HTML tags to obtain raw data. Main information of a document includes its title, abstract, sentences, and comments. In parsing, sentences and comments are also tokenized.[15] The dataset consists of 141

[13]http://150.65.242.97/doc-sum-annotator/annotate
[14]http://vietnamnet.vn
[15]http://mim.hus.vnu.edu.vn/phuonglh/softwares/vnTokenizer

open-domain articles along with 3760 sentences and 6926 comments in 12 events. Table 2.5 shows the statistics of this dataset.

Data annotation We used the same two-step procedure in creating SoLSCSum for annotating this dataset. We asked social users to involve annotation via a website.[16] Finally, five native Vietnamese speakers involved. Similar to USAToday-CNN, we also provide an indicator based on Cosine similarity computed with the abstract for each sentence and comment for guiding annotators. We used the same instruction of SoLSCSum but adjust the number of selected sentences and comments. They are no fewer than six sentences and six comments. The total number is no more than 30.

The final label of a sentence or comment was created based on majority voting among annotators. For example, given a sentence, each annotator makes a binary decision indicating whether it is a summary candidate (summary) or not (non-summary). If three annotators agree yes then it is labeled by 3. The label of each sentence or comment ranges from 1 to 5 indicating the agreement among annotators (1: very poor, 2: poor, 3: fair, 4: good; 5: perfect). References are those which receive at least three agreements from annotators (3/5), resulting 2448 references in total. In training supervised methods, we mapped labels into binary classes. Sentences with labels in the range of 3, 4, 5 were converted to summary and labels in range 0, 1, 2 were converted to a non-summary.

To ensure the quality of this dataset, we asked two other native Vietnamese people to vote each sentence or comment, which were already labeled. The Cohen's Kappa between them is 0.685 with 95% confidence interval.

2.3.2 Data observation

We summarize the information of our datasets in Table 2.4. It indicates that the number of user posts is large enough to support the importance estimation. On SoLSCSum and VSoLSCSum, references include both sentences and user posts, which are manually selected by humans. The reference type of SoLSCSum and VSoLSCSum is extraction while it is a type of abstraction on USAToday-CNN.

Table 2.4: Information of our datasets.

Dataset	**#doc**	**#sentences**	**#user posts**	**#refs**	**ref type**	**language**	**label**
SoLSCSum	157	3,462	25,633	5,858	extraction	English	Yes
VSoLSCSum	141	3,760	6,926	2,448	extraction	Vietnamese	Yes
USAToday-CNN	121	6,413	78,419	455	highlight	English	Yes

We examined a further investigation by observing word overlapping between sentences and user posts. We consider a token as a single word separated by a space. Note that word overlapping without stopwords is not counted on VSoLSCSum because there is no well-known stopword list in Vietnamese at the moment.

[16]http://150.65.242.91:9080/vn-news-sum-annotator/annotate

Table 2.5: Statistical observation; s: sentences, c: comments, and t: tweets.

Dataset	**Observation**	**sentences (%)**	**user posts (%)**
SoLSCSum	% Token overlapping	s/c: 13.26	c/s: 42.05
	% Token overlapping (stopword removal)	s/c: 8.90	c/s: 31.21
VSoLSCSum	% Token overlapping	s/c: 37.712	c/s: 44.820
USAToday-CNN	% Token overlapping	s/t: 22.24	t/s: 16.94
	% Token overlapping (stopword removal)	s/t: 15.61	t/s: 12.62

The observation from Table 2.5 shows that: (i) there exists common words or phrases between sentences and user posts, and (ii) readers tend to use words or phrases appearing in sentences to create their comments or tweets, e.g. 31.21% of word overlapping. This observation suggests four hypotheses:

- *Representation*: summary sentences of a document contain important information.
- *Reflection*: representative tweets or comments written by readers reflect document content as well as summary sentences.
- *Generation*: readers tend to borrow salient words or phrases appearing in a document to create their social messages, e.g. tweets or comments.
- *Common topic*: sentences and social messages mention some common topics represented in the form of common words.

2.3.3 Data segmentation

We segmented all datasets into folds as a preliminary step for training summarization methods in next chapters. We divided SoLSCSum into ten parts for 10-folds cross-validation. For USAToday-CNN and VSoLSCSum, we split them into five parts 5-folds cross-validation. We also used these partitions for unsupervised learning methods.

2.3.4 Evaluation procedure

We used ROUGE-scores for evaluation. We fixed $m = 6$ as the number of extracted sentences and user posts on SoLSCSum and VLSCSum (less than 30%) and $m = 4$ on USAToday-CNN because each document contains 3-4 highlights.

To compute ROUGE-scores, we employed `ROUGE-1.5.5` by using `pyrouge`[17] with parameters `‘‘-c 95 -2 -1 -U -r 1000 -n 2 -w 1.2 -a -s -f B -m"`. We used ROUGE-1, ROUGE-2, and ROUGE-W as major metrics. Due to the different length of references and summaries, we utilized F-score because it balances the trade-off between recall and precision. We separately evaluated extracted sentences and comments (Gao et al., 2012; Wei and Gao, 2014, 2015; Yang et al., 2011) instead of combining them. We leave the evaluation of combination as a feature task.

[17]https://github.com/andersjo/pyrouge

Chapter 3

Modeling Intra-relations and Inter-relations

This chapter is our first effort to investigate the integration of social information to improve the importance estimation of sentences and user posts. We organize it into two sections corresponding to two models. The first model in Section 3.1 formulates intra-relations and inter-relations between sentences and user pots by using similarity features. It encodes these relationships in a mutual reinforcement fashion used to estimate the importance of sentences and user posts. We published our work in Nguyen et al. (2015); Nguyen and Nguyen (2016, 2017). The second model in Section 3.1.5 is an extended version of the first one. It pushes the calculation of inter-relations from lexical to semantic similarity, from the sentence to word level while keeping the formulation of intra-relations. It also presents a new ILP method for extracting summaries. This model bases on our work in Nguyen et al. (2015, 2018a). In next sections, we describe our first model, which uses similarity features to integrate social information into the summarization process. We next introduce our second model, which exploits semantic similarity between sentences and user posts. We also show experimental results of our models on three datasets in two languages, English and Vietnamese, with discussion and analyses. ROUGE-scores indicate that our models obtain promising results compared to strong methods.

3.1 Ranking with Similarity Features

A Web document (called document) contains a set of sentences organized in a appropriate order to make a story. Therefore, they encode inter-relations indicating their relative importance. The importance of a sentence shows that it usually includes important information. For example, the importance of a sentence s_i can be measured by using a similarity score, e.g. Cosine, computed with remaining ones. In this view, a summary receives a higher score than others. The content of a summary sentence is also mentioned in many user posts showing that it also receives the considerable amount of attention from readers. It is defined as inter-relations. From this, we present a score-based raning model for formulating mutual relationships between sentences and user posts.

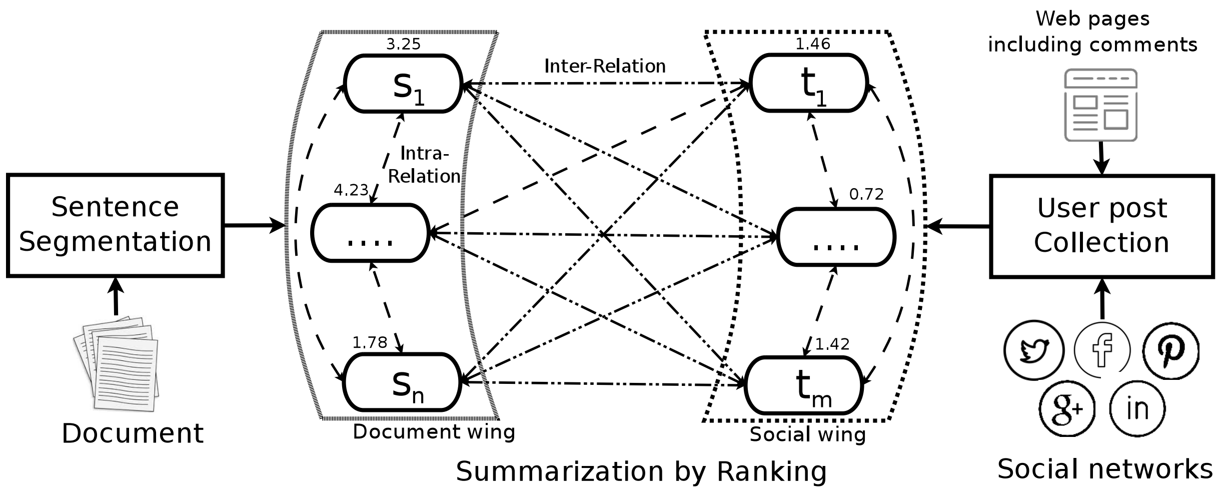


Figure 3.1: The overview of summarization using intra-relations and inter-relations

Figure 3.1 describes our model, in which, s_i and t_j present a sentence and user post. Lines connecting sentences (or user posts) are intra-relations and lines connecting sentence-user-post pairs are inter-relations. The weight of each node, e.g. 3.25 at s_1, is its importance calculated by a set of our features. In our view, sentences and user posts are presented in a mutual reinforcement way. For example, we take user posts as social information when modeling a sentence. Similarly, we also consider sentences as social information when estimating the importance of a user post. After scoring and ranking, our model selects top m ranked sentences and user posts as a summary.

3.1.1 Feature extraction

To integrate the social context of a Web document into the summarization process, a similarity score, e.g. Cosine can be used; however, using a single measurement may not efficient enough to completely capture the similarity aspect of sentence-user-post pairs. For example, a sentence and user post may not share common words (due to content variation), which may negatively affect the calculation of Cosine. Therefore, we introduce a set of similarity features represented in the form of three groups: distance, lexical, and semantic features to compute the similarity between sentences and user posts. They are grouped based on their characteristics. Table 3.1 shows our features.

Distance features cover the distance aspect of a sentence-user-post pair, showing that a summary sentence should be close to an important user post. We consider word and character levels of a sentence-user-post pair by using various distance features. For example, a Manhattan distance covers pairs those share common words and a Levenshtein distance based on characters treats pairs; those are content variation. Eqs. (3.1), (3.2)

Table 3.1: Features for computing similairties between two texts.

Feature group	**Feature name**
Distance	Manhattan distance between two texts.
	Euclidean distance between two texts.
	Cosine similarity between two texts.
	Word matching (wmc) of two texts.
	Dice coefficient of two texts.
	Jaccard coefficient of two texts.
	Levenshtein between two texts.
	JaroWinkler distance between two texts.
	Damerau-Levenshtein distance betweet two texts.
Lexicon	The longest common substring of (s_i, t_j) (LSC).
	Inclusion-exclusion of (s_i, t_j).
	% words of S appearing in T (p(s_i,t_j)).
	% words of T appearing in S (p(t_j, s_i)).
	Word overlap coefficient (woc) of (t_j, s_i).
	Smith Waterman of (s_i, t_j).
Semantics	Semantic similarity of (s_i, t_j) based on word level.

and (3.3) define Manhattan, Euclidean, and Cosine.

$$manhattan(\vec{x},\vec{y}) = \sum_{i=1}^{n} \mid x_i - y_i \mid; \tag{3.1}$$

$$euclidean(\vec{x},\vec{y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{3.2}$$

$$cos(\vec{x},\vec{y}) = \frac{\vec{x}.\vec{y}}{\parallel \vec{x} \parallel . \parallel \vec{y} \parallel} \tag{3.3}$$

where n is the cardinality of words appearing in s_i and t_j; x_i and y_i are the frequency of each word in s_i and t_j; $\vec{x}$ and $\vec{y}$ are two same size vectors.

Equation (3.4) represents the word matching coefficient.

$$wmc(s_i, t_j) = comWord(s_i, t_j) \tag{3.4}$$

where *comWord()* returns the number of common words between s_i and t_j.

Equations (3.5) and (3.6) define Dice and Jaccard distances.

$$dice = \frac{2. \mid X \cap Y \mid}{\mid X + Y \mid}; \tag{3.5}$$

$$jaccard = \frac{\mid X \cap Y \mid}{\mid X \cup Y \mid} \tag{3.6}$$

where X is a set of words in s_i; and Y is a set of words in t_j.

Equation (3.7) denotes the JaroWinkler distance of two texts.

$$d_j(s_i,t_j) = \begin{cases} 0 & \text{if } m=0 \\ \frac{1}{3}\left(\frac{m}{|s_i|} + \frac{m}{|t_j|} + \frac{m-t}{m}\right) & otherwise \end{cases} \tag{3.7}$$

where $|s_i|$ and $|t_j|$ are the number of characters in s_i and t_j, m is the number of matching characters, and t is a half number of transpositions.

Suppose s_i can be represented by a and t_j can be denoted by b, Eq. (3.8) defines a Damerau–Levenshtein distance.

$$d_{a,b}(i,j) = \begin{cases} max(i,j) & \text{if } min(i,j)=0 \\ min\begin{cases} d_{a,b}(i-1,j)+1 \\ d_{a,b}(i,j-1)+1 \\ d_{a,b}(i-1,j-1)+1_{(a_i \# b_j)} \\ d_{a,b}(i-2,j-2)+1 \end{cases} & \text{if } i,j>1 \text{ and } a_i=b_{j-1} \text{ and } a_{i-1}=b_j \\ min\begin{cases} d_{a,b}(i-1,j)+1 \\ d_{a,b}(i,j-1)+1 \\ d_{a,b}(i-1,j-1)+1_{(a_i \# b_j)} \end{cases} & otherwise \end{cases} \tag{3.8}$$

where $1_{(a_i \# b_j)}$ equals 0 if $a_i = b_j$ or equals 1, otherwise. $d_{a,b}(i-1,j)+1$ is the deletion from a to b; $d_{a,b}(i,j-1)+1$ is the insertion from a to b; $d_{a,b}(i-1,j-1)+1_{(a_i \# b_j)}$ corresponds to a match or mismatch, depending on whether respective symbols are the same; $d_{a,b}(i-2,j-2)+1$ corresponds to a transposition between two successive symbols.

Equation (3.9) presents a Levenshtein distance.

$$lev_{a,b}(i,j) = \begin{cases} max(i,j) & \text{if } min(i,j)=0 \\ min\begin{cases} lev_{a,b}(i-1,j)+1 \\ lev_{a,b}(i,j-1)+1 \\ lev_{a,b}(i-1,j-1)+1_{(a_i \# b_j)} \end{cases} & otherwise \end{cases} \tag{3.9}$$

where $1_{(a_i \# b_j)}$ equals 0 if $a_i = b_j$ or equals 1, otherwise. $lev_{a,b}(i,j)$ is the distance between the first i characters of a and the first j characters of b.

Lexical features They capture the word overlapping aspect of a sentence and a user post. A summary sentence and a representative user post usually share common words (the *generation* hypothesis), indicating their content is similar. Eq. (3.10) represents the longest common substring of two texts.

$$lcs(s_i,t_j) = \frac{len(maxComSub(s_i,t_j))}{min(len(s_i), length(t_j))} \tag{3.10}$$

where *len()* returns the length of a string; *maxComSub()* returns a number of maximum common sub string between s_i and t_j.

Equation (3.11) shows the inclusion_exclusion coefficient.

$$\text{inclusion-exclusion}(s_i, t_j) = \frac{comWord(s_i, t_j)}{len(s_i) + len(t_j)} \tag{3.11}$$

where $comWord()$ returns the number of common words between s_i and t_j, $len()$ returns the number of words in s_i or t_j.

Equation (3.12) defines the percentage of word overlapping of s_i in t_j.

$$p(s_i, t_j) = \frac{k}{len(s_i)} \tag{3.12}$$

where k is the number of common words denoted by $w = \{w_1, w_2, ..., w_k\}$ between s_i and t_j; *len()* counts the number of words in s_i. The percentage of word overlapping of t_j in s_i is also defined by changing the role of s_i and t_j.

Equation (3.13) calculates the word overlap coefficient.

$$woc(s_i, t_j) = \frac{wmc(s_i, t_j)}{min(len(s_i), len(t_j))} \tag{3.13}$$

where $wmc()$ is the word matching coefficient of two texts s_i and t_j defined in Eq. (3.4), $len()$ returns the number of words in a text.

Smith-Waterman has widely used in sequence alignment, which determines similar regions between two strings. This feature compares the segments of all possible length between a sentence s_i and a user post t_j. Eq. (3.14) computes this feature.

$$h_{a,b}(i,j) = \left\{ \begin{array}{cc} 0 & \\ H(i-1, j-1) + s(a_j, b_j) & match/mismatch \\ max_{k \geq 1} H(i-k, j) + W_k & deletion \\ max_{l \geq 1} H(i, j-l) + W_l & insertion \end{array} \right\} \begin{array}{l} 1 \leq i \leq m \\ 1 \leq j \leq n \end{array} \tag{3.14}$$

where $H(i, 0) = 0, 0 \leq i \leq m$; $H(0, j) = 0, 0 \leq j \leq n$; a, b are strings over s_i and t_j; m and n are the length of s_i and t_j; $s(a, b)$ is a similarity function; $H(i, j)$ is the maximum similarity-score between a suffix of $a[1...i]$ and $b[1...j]$; W_i is a gap-scoring scheme.

Semantic features Since distance and lexical features rely on lexicons; therefore, they may be limited to capture the semantics of a sentence-tweet/comment pair. For example, if a sentence contains a *"police"* word and a tweet includes an *"officer"* word, they should be closer than those which include *"police"* and *"child"* words. To deal with this issue, we integrated a semantic feature represented in the form of *Word2Vec* (Mikolov et al., 2013). Word2Vec[1] takes a large dataset as an input for training and then produces word vectors. In training, it first generates a vocabulary from the dataset and maps each word into a high-dimensional vector space, in which each vector represents the meaning of a word with its context. The context of a word is the number of its surrounding words

[1]https://code.google.com/p/word2vec/

(usually called by window size). After training, we can calculate the similarity between two words, e.g. Cosine similarity between two words: *"police"* and *"officer"*. In practice, we trained a Word2Vec model on 1 billion words from Google by using SkipGram model, the vector dimension of 300 with the window size of 7 for English. We also trained a Vietnamese Word2Vec model on 4 million news articles collected form BaoMoi[2] with the same setting.

Given a Word2Vec model, Eq. (5.10) calculates the semantic similarity of two sentences.

$$sentSim(s_i, t_j) = \frac{\sum_{w_i}^{N_s} \sum_{w_j}^{N_c} w2vSim(w_i, w_j)}{N_s + N_c} \tag{3.15}$$

where N_s and N_c are the number of words in s_i and t_j (stopword removal); $w2vSim()$ returns the semantic similarity between two words.

3.1.2 Social context integration

We applied our features to estimate the importance of each sentence and user post in Figure 3.1. More precisely, we present two methods named *score-based Inter-Wing* and *score-based Dual-Wing*.

Score-based inter-wing In this method, our model computes the score of a sentence or a user post by using auxiliary information from the other side. For example, the score of sentence s_i is calculated by using complementary user posts from social context. Eq. (3.16) explains this calculation.

$$score(s_i) = \frac{1}{m} \sum_{j=1}^{m} interScore(s_i, t_j) \tag{3.16}$$

where $s_i \in S$, $t_j \in T$, S is a set of sentences and T is a set of user posts; $interScore(s_i, t_j)$ returns a similarity score between sentence s_i and user post t_j; m is the number of user posts corresponding to each document. Eq. (3.17) calculates the inter-score.

$$interScore(s_i, t_j) = \frac{1}{F} \sum_{k=1}^{F} f_k(s_i, t_j) \tag{3.17}$$

where F contains 16 features; $f_k()$ is a similarity function calculated by each k^{th} feature. We treated these features equally to ensure no bias among them. Similarly, the score of a user post is also computed in the same mechanism.

$$score(t_j) = \frac{1}{n} \sum_{i=1}^{n} interScore(t_j, s_i) \tag{3.18}$$

where n is the number of sentences in a document d.

[2]https://www.baomoi.com

Score-based dual-wing In this method, our model estimates the importance of a sentence by using two indicators: an intra-score and an inter-score. The intra-score captures the relationship of a sentence s_i with remaining ones in the same document, and the inter-score represents the relationship of this sentence with auxiliary user posts. For example, the score of s_i is calculated by using s_1 to s_n; at the same time, its score is also computed by using complementary user posts t_1 to t_m. The final score of s_i is a linear combination of these indicators.

$$score(s_i) = \delta * \sum_{k=1}^{n} intraScore(s_i, s_k) + (1 - \delta) * \sum_{j=1}^{m} interScore(s_i, t_j) \tag{3.19}$$

Similarly, our model also computes the score of a user post with the same mechanism in Eq. (3.20).

$$score(t_j) = \delta * \sum_{k=1}^{m} intraScore(t_j, t_k) + (1 - \delta) * \sum_{i=1}^{n} interScore(t_j, s_i) \tag{3.20}$$

where δ is a balanced parameter which controls the contribution of social information; n and m are the number of sentences and user posts. Note that $intraScore(s_i, t_j)$ is also computed by Eq. (3.17). Section 3.1.5 shows the selection of balanced parameter δ.

3.1.3 Sentence selection

Our model extracts summaries by selecting sentences and user posts which have highest scores. After scoring and ranking, our model loops on ranked sentences and user posts and picks up top ones. This process stops when the number of selected sentences reaches to m. We ignore sentences and user posts those are fewer than five words because they are short (Erkan and Radev, 2004). Eq. (3.21) presents our ranking algorithm.

$$S_r \leftarrow ranking(S); \quad T_r \leftarrow ranking(T) \tag{3.21}$$

where *ranking()* returns a list of sentences or user posts in a decreased weight order. After ranking, top m ranked sentences and user posts are selected to form a summary.

3.1.4 Baselines

We compared our model to several methods of text summarization and summarization using social information. They are categorized into two groups: non-social context methods and social context ones.

Non-social context methods We compared our model to well-known baselines of text summarization. **Lead-m** chooses the first m sentences as a summary (Nenkova, 2005) who reported that it is a strong baseline which is better than many systems participated DUC competition. In our setting, it was not used in selecting tweets or comments. **LexRank**[3] is a common graph-based method for extractive summarization (Erkan and Radev, 2004). It builds a stochastic graph for computing the relative importance of textual units of an input document then ranks sentences based on their importance. **SVM** is a powerful supervised learning method for classification (Cortes and Vapnik, 1995). Researchers usually train a binary classifier on training data and apply it on testing sets to extract summaries, e.g. sentences labeled by 1. For implementation, we used LibSVM[4] with an RBF kernel and five basic features from Yang et al. (2011). Due to imbalanced data that the number of non-summary user posts is much larger than that of summary ones, we applied the weighting to reduce the bias of SVM on the non-summary class. **CRF** (Lafferty et al., 2001) is also employed for summarization to exploit the sequence aspect among sentences in a document (Shen et al., 2007). This method formulates the summarization process as a sequence labeling task, in which summary sentences are labeled by 1 and non-summary sentences are labeled by 0. To implement a CRF-based summarizer, we used a Mallet sequence tagging[5] with a set of basic features in Shen et al. (2007). We formulated SVM and CRF as a binary classification problem. Note that these methods were separately trained and applied to sentences and user posts.

Social context methods We also compared our model to state-of-the-art methods of social context summarization. **cc-TAM** builds a cross-collection topic-aspect modeling as a preliminary step to generate a bipartite graph used for co-ranking to select sentences and tweets for multi-document summarization (Gao et al., 2012).[6] Wei and Gao (2015) introduced **HGRW**, which is a variation of LexRank named Heterogeneous Graph Random Walk. The authors utilize the help of tweets to support sentences in building graphs used by LexRank to rank sentences. Wei and Gao (2014) presented **RB CCF**, which exploits set of local and cross features trained by RankBoost implemented in RankLib[7] for training two L2R models for sentences and tweets. Cross features are extracted from tweets. In this thesis, for our re-implementation, we ignored unnecessary features, e.g. hashtags and URLs because they are not usually available in comments.

3.1.5 Results and Discussion

In this section, we present ROUGE-scores of our model against baselines and advanced models, followed by analyses, which help to understand several aspects of our model.

ROUGE-scores

We report our comparison with baselines followed by ROUGE-scores of advanced methods.

[3]https://code.google.com/p/louie-nlp/source/browse/trunk/louie-ml/src/main/java/org/louie/ml/lexrank/?r=10

[4]http://www.csie.ntu.edu.tw/∼cjlin/libsvm/

[5]http://mallet.cs.umass.edu/sequences.php

[6]We acknowledge Gao et al. (2012) for sharing their code.

[7]https://people.cs.umass.edu/∼vdang/ranklib.html

Comparison with non-social context methods The general trend of ROUGE-scores from Table 3.2 indicates that our model obtains competitive results, in which it is the best on USAToday-CNN and VSoLSCSum, and is comparable on SoLSCSum. It is significantly better SVM and CRF, two well-known methods for summarization in some cases (values denoted by † with $p \leq 0.05$). Promising results support our idea in combining intra-relations and inter-relations for improving the importance estimation.

Table 3.2: Our model vs. baselines; **bold** is the best value; *italic* is the second best. RG stands for ROUGE. † means our model is significantly better. Methods with * are supervised learning.

Dataset	**Method**	**Sentences**			**User posts**		
		RG-1	RG-2	RG-W	RG-1	RG-2	RG-W
SoLSCSum	Lead-*m*	0.345	*0.322*	*0.170*	—	—	—
	LexRank	0.327	0.243†	0.138	**0.210**	**0.115**	**0.085**
	SVM*	0.325	0.263	0.147	0.152†	0.089†	0.062†
	CRF*	**0.393**	**0.379**	**0.187**	0.091†	0.075†	0.037†
	Inter-Wing	0.352	0.277	0.154	*0.209*	**0.115**	*0.083*
	Dual-Wing	*0.357*	0.280	0.156	0.180	*0.098*	0.070
USAToday-CNN	Lead-*m*	0.249	**0.106**	*0.172*	—	—	—
	LexRank	0.251	0.092	0.163	0.193	0.068	0.128
	SVM*	*0.261*	0.106	0.171	*0.221*	**0.084**	**0.149**
	CRF*	0.186†	0.088	0.114†	0.190	0.065	0.119
	Inter-Wing	**0.268**	*0.104*	**0.173**	0.215	0.074	0.137
	Dual-Wing	0.252	0.085	0.163	**0.227**	*0.080*	*0.146*
VLSCSum	Lead-*m*	0.495	0.420	0.214	—	—	—
	LexRank	0.506	0.432	**0.219**	0.348†	0.198†	0.127†
	SVM*	0.497	0.440	0.208	0.374	0.212†	0.140
	CRF*	0.422†	0.357†	0.172†	0.111†	0.062†	0.041†
	Inter-Wing	**0.532**	**0.463**	*0.218*	**0.443**	**0.277**	**0.170**
	Dual-Wing	*0.531*	*0.457*	*0.218*	*0.409*	*0.234*	*0.153*

Score-based Inter-Wing achieves good results in some cases, and the Dual-Wing method is better in other cases. For example, the Inter-Wing method surpasses the Dual-Wing method for document summarization on USAToday-CNN, i.e. 0.268 vs. 0.252 of ROUGE-1; however, for user post extraction, Dual-Wing calculation is better. For sentence selection, many tweets are derived by using the title of a Web document (readers copy the title to create their tweets) on USAToday-CNN, or comments usually include salient phrases appearing in sentences; therefore, adding information from documents (dual wing) is unnecessary. For user post selection, the score of a user post in the Dual-Wing method is calculated by an accumulative fashion; therefore, it tends to select longer social sentences compared to inter-wing computation. However, their gaps are small on SoSLCSum and USAToday-CNN, i.e. 0.074 vs. 0.080 of ROUGE-2 on USAToday-CNN. On VLScSum, the Inter-Wing method is better. Their results reveal that sentences from a document can also be taken as additional information in scoring user posts.

Results from Table 3.2 indicate that our methods are better than SVM and CRF, two supervised learning methods. This shows the efficiency of our model and features. On SoLSCSum, CRF is the best for sentence selection because it exploits the sequential dependencies among sentences in documents with sophisticated features, leading to big margins compared to our model. However, on USAToday-CNN and VSoLSCSum, it is inefficient because its features may be inappropriate. For example, some features cannot be extracted on VSoLSCSum, e.g. indicator words due to the limitation of language sources and tools in Vietnamese. For user post extraction, CRF is the worst because the sequence aspect does not implicitly exist in comments or tweets. SVM acquires competitive results on USAToday-CNN, but it is not the best on other datasets. The reason may be similar to CRF, in which its features may be not suitable. LexRank is quite stable on three datasets, in which it is the best for comment extraction on SoLSCSum. This is because, with a sophisticated scoring method, it can effectively extract salient sentences. However, in other cases, our model surpasses LexRank due to the integration of social information, the same conclusion with Wei and Gao (2014, 2015); Yang et al. (2011). Note that it is challenging for a method to obtain the best results on all datasets. Lead-m is a strong baseline because it simulates the behavior of writers by picking up m first sentences. It confirms that many participated systems of DUC can not significantly pass this method (Nenkova, 2005).

Comparison with social context methods We report the comparison of our model with advanced methods in Table 3.3.

Table 3.3: Our model vs. advanced methods.

Dataset	Method	Sentences			User posts		
		RG-1	RG-2	RG-W	RG-1	RG-2	RG-W
SoLSCSum	cc-TAM	0.306†	0.238†	0.136	0.054†	0.022†	0.024†
	HGRW	**0.379**	0.204†	**0.167**	**0.209**	0.115	**0.084**
	RB CCF*	*0.360*	**0.283**	*0.158*	*0.190*	*0.098*	0.077
	Inter-Wing	0.352	0.277	0.154	*0.209*	**0.115**	*0.083*
	Dual-Wing	*0.357*	0.280	0.156	0.180	*0.098*	0.070
USAToday-CNN	cc-TAM	0.229	0.077†	0.145†	**0.249**	*0.089*	*0.152*
	HGRW	**0.279**	*0.098*	**0.177**	**0.242**	0.088	**0.157**
	RB (CCF)*	0.221	0.070†	0.140†	0.233	**0.091**	0.132
	Inter-Wing	**0.268**	*0.104*	**0.173**	0.215	0.074	0.137
	Dual-Wing	0.252	0.085	0.163	**0.227**	*0.080*	*0.146*
VLSCSum	cc-TAM	0.488†	0.377†	0.201	0.301†	0.167†	0.111†
	HGRW	**0.570**	*0.479*	*0.233*	*0.454*	*0.298*	**0.173**
	RB CCF	*0.561*	**0.494**	**0.235**	**0.471**	**0.308**	0.168
	Inter-Wing	**0.532**	**0.463**	*0.218*	**0.443**	**0.277**	**0.170**
	Dual-Wing	*0.531*	*0.457*	*0.218*	*0.409*	*0.234*	*0.153*

ROUGE-scores indicates that our model is competitive with state-of-the-art methods. For example, our Inter-Wing method is the best for comment extraction on SoLSCSum, or it is very competitive for sentence selection on USAToday-CNN. HRGW achieves the best ROUGE-scores over three datasets even it is an unsupervised method. It is significantly better than LexRank because it is an extension of LexRank by exploiting social information. This again validates the support of social information in the scoring step. Results of RankBoost CCF is quite stable because it is a learning-to-rank method, which uses many sophisticated features (Wei and Gao, 2014). Interestingly, cc-TAM obtains poor results because it is designed for multi-document summarization, while our dataset is created for single-document summarization.

Feature contribution analysis

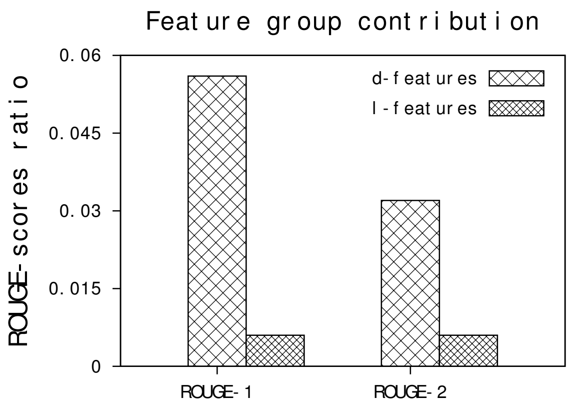
We investigated the contribution of features in our proposed model. In order to do that, we ran our score-based Inter-Wing method by using leave-one-out test. Each feature was removed and all remaining ones ($n-1$) were kept. Feature influence was measured by the minus of this method using all features with the method using $n-1$ features on USAToday-CNN. Table 3.4 presents top six effective features.

Table 3.4: Top six effective features on USAToday-CNN; * is Inclusion-exclusion coefficient; *italic* denotes lexical-based features.

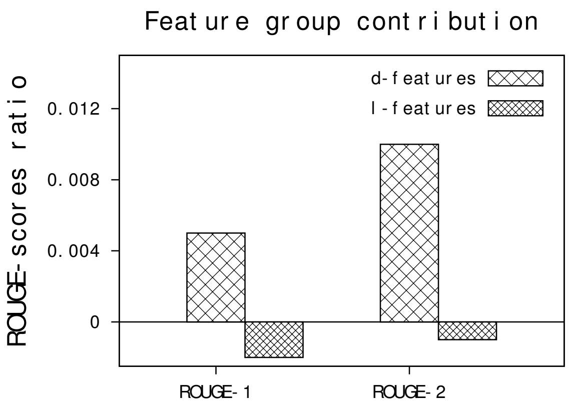
Feature	**Document**		**Feature**	**Tweet**	
	ROUGE-1	ROUGE-2		ROUGE-1	ROUGE-2
dice	0.1 x 10^{-3}	0.6 x 10^{-3}	dice	0.9 x 10^{-3}	0.3 x 10^{-3}
overlap	0.1 x 10^{-3}	0.5 x 10^{-3}	manhattan	0.9 x 10^{-3}	0.1x10^{-3}
*in-ex coeffi**	0.8 x 10^{-3}	0.5 x 10^{-3}	*in-ex coeffi**	0.9x10^{-3}	0.2x10^{-3}
jaccard	0.1 x 10^{-3}	0.000	jaccard	0.9x10^{-3}	0.1x10^{-}3
cosine	0.1 x 10^{-3}	0.5 x 10^{-3}	*matching*	0.7x10^{-3}	0.000
w2v	0.000	0.2 x 10^{-3}	w2v	0.9x10^{-3}	0.1x10^{-3}

Table 3.4 indicates that both distance and lexical features affect our model. For sentence selection, lexical features play an important role. This shows that summary sentences include salient common words or phrases. Among them, Dice coefficient, Inclusion-exclusion coefficient, and Jaccard positively affect the extraction. For tweet extraction, distance features are more important than lexical ones (only inclusion-exclusion coefficient and matching appear). The Word2vec feature has no contribution for ROUGE-1 of sentence selection; however, for ROUGE-2 and tweet extraction, it positively contributes our model.

We also observe the contribution of feature groups. Since the influence of semantic similarity is already shown in Table 3.4, we only observed the effectiveness of distance and lexical groups by runing the Inter-Wing method. Values on the y-axes were computed by the minus of the method using all groups with the model using distance or lexical features.



(a) Feature group in document



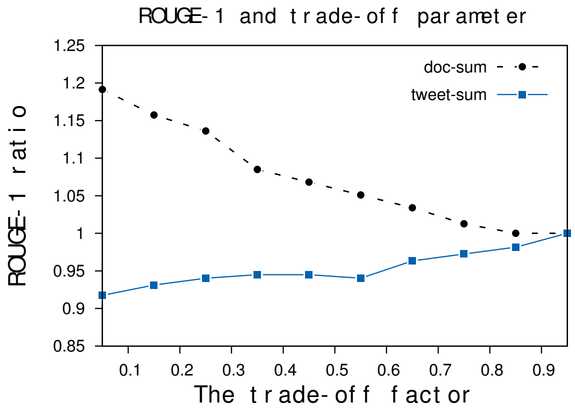
(b) Feature group in tweet

Figure 3.2: The contribution of feature groups in score-based Inter-Wing.

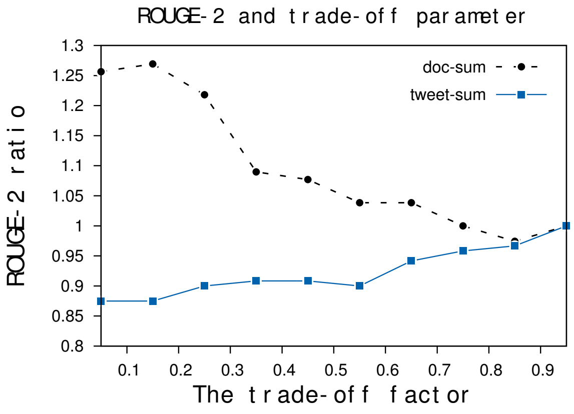
Figure 3.2 shows that for sentence selection, both groups positively contribute our model, in which distance features have large weights than lexical ones. This trend is consistent in Figure 3.2b for tweet selection. This supports results in Table 3.4, in which the number of distance features is larger than that of lexical ones, and concludes that distance features play an important role in our model. In Figure 3.2b, lexical features slightly negatively affect our model because although each lexical feature has a positive impact, combining them may lead a conflict. Note that negative values are tiny.

Tuning trade-off parameter

We investigated the impact of the balanced parameter in Eqs. (3.19) and (3.20) by adjusting δ in [0.05, 0.95] with a jumping step of 0.1 on USAToday-CNN, running score-based Dual-Wing. We divided its scores at each tuning point for the scores at $\delta = 0.95$. We used this as a normalization because margins among tuning points are tiny.



(a) The performance in ROUGE-1



(b) The performance in ROUGE-2

Figure 3.3: The adjustment of δ with score-based Dual-Wing.

Figure 3.3 indicates that the performance of document and tweet summarization is in a reverse form. When increasing δ, the performance of sentence extraction decreases while the performance of tweet summarization raises. Auxiliary information benefits our model until a certain turning point. When δ is close to 0.85, our model seems to be converging. To balance, $\delta = 0.85$ was empirically selected for all datasets. Note that the change is slightly different among tuning points because the score of a sentence is computed by using many features; therefore, the role of δ may be saturated.

Tweet-size analysis

We observed the impact of tweet-size in score-based Inter-Wing by adjusting the similarity threshold in the pre-processing step on USAToday-CNN. Recall that we removed near-duplicate tweets by using a Simpson calculation before doing annotation (Section 2.3.1). However, our model does not require labels from the annotation. Therefore, we ran the model on the whole dataset. In order to do that, we tuned the similarity threshold of Eq. (2.3) in [0.05, 0.95] with a jumping step of 0.1 and conducted two observations.

Tweet-size and running time When running, we recorded the running time corresponding to each tuning point; then plotted it on Figure 3.4.

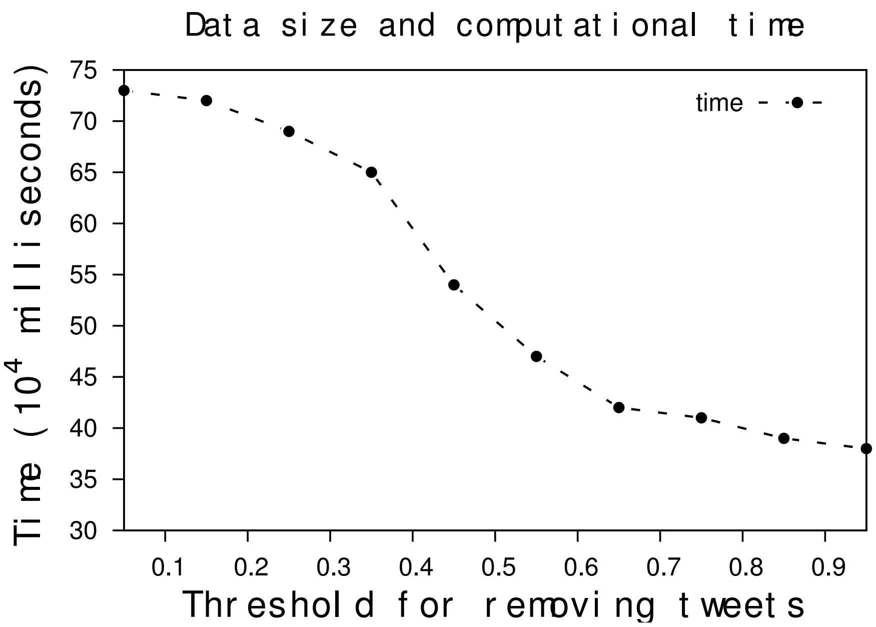
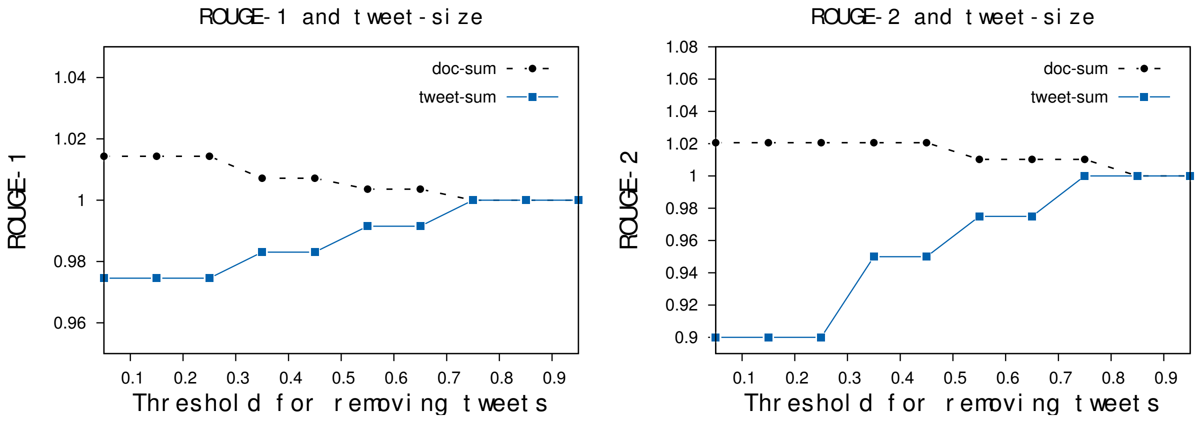


Figure 3.4: The relationship between tweet-size and computational time.

The trend in Figure 3.4 shows that when increasing the similarity threshold (removing more near-duplicate tweets), the computational time decreases. It slightly falls from 0.05 to 0.35 and significantly drops from 0.35 to 0.65. It is understandable that when removing tweets, our model can speed up the calculation. In remaining points, the trend indicates that removing more tweets slightly speeds up our system. It means that there are not many similar tweets when tuning the threshold from 0.65 to 0.95.

Tweet-size and ROUGE-scores We also observed ROUGE-scores in each tuning point to show the relationship of tweet-size and performance. Because margins among tuning points are small, then we used a normalization as in Section 3.1.5 by dividing the scores of each data-size point for those at 0.95. Finally, we visualized normalized scores on Figure 3.5.



(a) Performance of ROUGE-1 (b) Performance of ROUGE-2

Figure 3.5: The relationship between ROUGE-scores and tweet-size.

In Figures 3.5a and 3.5b, we can see that the performance of document summarization decreases while the performance of tweet extraction raises when removing near-duplicate tweets. This is because, for sentence selection, removing tweets reduces the support from social information. For tweet extraction, the removal improves ROUGE-scores because our score-based Inter-Wing can correctly select summaries in a smaller set rather than in the original set. Besides, recall that many tweets are directly borrowed documents by copying their titles; therefore, the removal helps to keep salient ones, which reflect their content. To balance, the threshold was fixed at 0.25.

Hypothesis validation

We validated our hypotheses in Section 2.3.2 by simulating the running of score-based Dual-Wing. Table 3.5 shows an example, in which S1 and T2 are summary sentences while S2 and T1 are non-summary. The scores generated from the running are visualized on Figure 3.6.

Figure 3.6 reveals that summary sentences, i.e. S1 and T2 receive higher scores than non-summary ones, i.e. S2 and T1. This validates our hypotheses stated in Section 2.3.2. In addition, T1 and T2 contain important information of the Boston bombing event. This supports *representation* and *reflection* hypotheses. We also observe that sentences and

Table 3.5: An example of the Boston Bombing (24) on USAToday-CNN.

Sentences	**Tweets**
[S1] Police have identified Tamerlan Tsarnaev as the dead Boston bombing suspect	[T1] Who is Tamerlan Tsarnaev, 26, the man ID&#39 as the dead #BostonBombing
[S2] The brothers had been living together on Norfolk Street in Cambridge	[T2] Before his death Tamerlan Tsarnaev called an uncle andasked for his forgiveness. Said he is married and has a baby

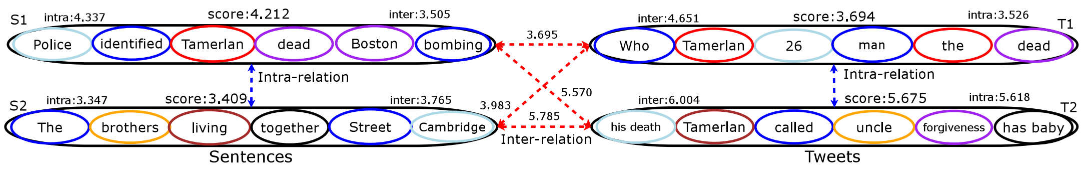


Figure 3.6: A running example generated by score-based Dual-Wing.

tweets share common words, e.g. *"Tamerlan"*, *"bombing"*, *"dead"* supporting *generation* and *common topic* hypotheses.

Table 3.6: A summary example from the Boston Bombing (24) on USAToday-CNN.

Highlights
Police identified Tamerlan Tsarnaev, 26, as the dead Boston bombing suspect. Tamerlan studied engineering at Bunker Hill Community College in Boston. He was a competitive boxer for a club named Team Lowell.
Summary Sentences of the two methods
S1: Tamerlan Tsarnaev, the 26-year-old identified by police as the dead Boston bombing suspect, called his uncle Thursday night and asked for forgiveness, the uncle said. S2: Police have identified Tamerlan Tsarnaev as the dead Boston bombing suspect. S3: Tamerlan attended Bunker Hill Community College as a part-time student for three semesters, Fall 2006, Spring 2007, and Fall 2008. S4: He said Tamerlan has relatives in the United States and his father is in Russia.
Summary tweets of Inter-Wing
T1: Before his death Tamerlan Tsarnaev called an uncle and asked for his forgiveness. Said he is married and has a baby. T2: I proudly say I was the 1st 1 to write this on twitter. Uncle, Tamerlan Tsarnaev called, asked for forgiveness. T3: So apparently the dead suspect has a wife & baby? And beat his girlfriend enough to be arrested? (same woman?). T4: Tamerlan Tsarnaev ID'd as dead Boston blast suspect - USA Today - USA TODAY, Tamerlan Tsarnaev ID'd as dead.
Summary tweets of Dual-Wing
T1: I proudly say I was the 1st 1 to write this on twitter. Uncle,Tamerlan Tsarnaev called, asked for forgiveness. T2: So apparently the dead suspect has a wife & baby? And beat his girlfriend enough to be arrested? (same woman?). T3: Before his death Tamerlan Tsarnae called an uncle and asked for his forgiveness.Said he is married and has a baby. T4: #BostonMarathon bomber Tamerlan called uncle couple of hours before he was shot dead said 'I love you and forgive me.

Output observation

We examined the output from our model on USAToday-CNN. From Table 3.6 we observe that our methods yield the same output for document summarization. The content of S1, S2, and S3 completely relates to highlights, which mention the death of Tamerlan Tsarnaev at the Boston bombing event or attending information in his college. This is because they contain important words; hence our methods can select correctly. By contrast, S4 mentions his father information, which is slightly relevant to the event.

For tweet extraction, our methods output three similar tweets, and the remaining one is different. Summaries include the same tweets, i.e. T1 of score-based Inter-Wing and T3 of score-based Dual-Wing; while other ones are different, leading to different ROUGE-scores between two methods. Extracted tweets are quite relevant to the event but do not directly mention the death of Tamerlan Tsarnaev, e.g. T2. This is because T2 also consists of salient information which challenges the ranking of our methods and thus it leads to an incorrect selection.

Irrelevant data may negatively affect our model because the score of a sentence or tweet is calculated by an accumulative mechanism; therefore, non-important ones can also achieve a high score, e.g. T2. They do not directly mention the death of Tamerlan Tsarnaev but receives a lot of attention from readers. All sentences and tweets in Table 3.6 contain keywords suggesting that semantic similarities among salient phrases on both sides can be exploited to improve the importance estimation.

3.2 Improving Summarization with Semantic Similarity and ILP

As observed in Section 3.1.5, sentences and user posts share common topics denoted in form of words or phrases. Let's take Table 3.7 as an example. We observe that the sentence and comment share several common words in bold. Words in the comment are the same with those in the sentence (*"police" or "officers"*) or are in a variation form such as *"shots"* and *"fire"*. We denote this as *word variation*, which comes from the fact that readers tend to borrow salient information in sentences to create their posts.

Table 3.7: An example of common words or phrases between sentences and user posts.

[S]: The 26-year-old man, identified as Usaamah Rahim, brandished a **knife** and advanced on **officers** working with the Joint Terrorism Task Force who initially tried to retreat before opening **fire**, Boston **Police** Superintendent William Evans told reporters.
[C] If I had been one of the **police officers** I would have whispered 3 times "**drop the knife**" then quickly **fired** several **shots** at his sternum.

The insight of this section is that we formulate the word variation to integrate social context for improving the importance estimation. To do that, we introduce our model in Figure 3.7. It includes three components: topical word generation, sentence scoring (the graph), and sentence selection. The role of generation is to generate salient words

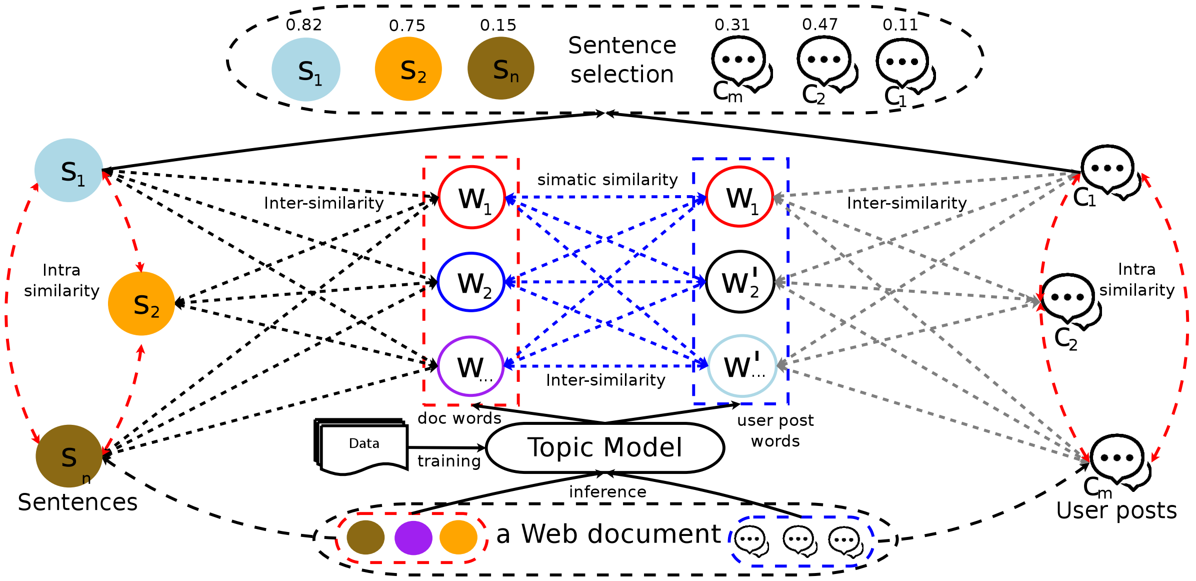


Figure 3.7: The summarization model, in which coloured circles represent words.

used to present the word variation. For scoring, we adopt our last model by presenting sentences and user posts in a mutual reinforcement graph, which allows to estimate the importance of sentences and user posts by combining their intra-relations (red lines) and inter-relations (black and blue lines). We keep intra-relations and extend inter-relations by pushing them from lexical to semantic levels. Instead of using many lexical features, we form the variation by using semantic similarity among salient words on two sides. After modeling, the importance of a sentence is measured by combining these relations. For selection, besides employing a greedy method, we present a novel ILP model for selecting summaries. We summarize the scoring process of our model in Figure 3.8.

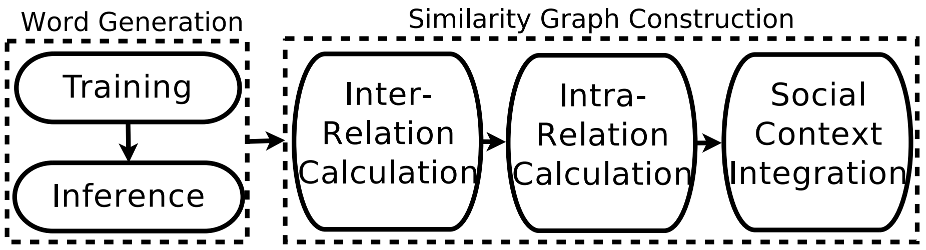

Word Generation
Training
Inference
Similarity Graph Construction
Inter-Relation Calculation
Intra-Relation Calculation
Social Context Integration

Figure 3.8: The work flow of sentence scoring.

We next show topical word generation and then describe our score calculation.

3.2.1 Topical word generation

Our model formulates the word variation, in which readers tend to borrow salient words or phrases in sentences to create their posts. To do that, salient words of each document were first generated. It is possible to create these words by using TF-IDF; however, they may not capture the topic aspect hidden in each document. We, therefore, apply Latent Dirichlet Allocation (LDA)[8] (Blei et al., 2003) to take into account word distribution over topics. The generation was done in two steps: training and inference.

Training

An obvious way is to train an LDA model on our datasets and then infer to obtain the weight of topical words. To do that, we divided each dataset into two smaller parts: documents and their user posts, then we formed them into two new sets. We next separately trained two LDA models on documents and their user posts. The reason for this is that sentences and user posts have different characteristics; hence, training a common LDA model may negatively influence the inference. For training, topic and word distribution parameter are $\alpha = \beta = 0.01$ with 1,000 iterations; the topic number $k = 30$. The analysis of selection is shown in Section 3.2.4. After training, we used a word-topic-weight matrix M_{WT} and document-topic distribution matrix M_{DT} for the inference. In practice, we can train an LDA model on the large amount of data as a global source for inferring new documents. However, it is not a major task in this thesis, and we leave this setting for future work when we implement an online summarization system.

Inference

The inference receives matrices from the training to infer hidden topics with their topical words (including weights). Due to our purpose to connect documents and their user posts, we divided each document d into two parts: d_s (sentences) and d_c (comments or tweets), then we formed each part as a smaller document, that is an input for inference. Given a small single document d_s (or d_c), the inference first chooses top t closet topics of d_s. The closet topics of d_s are those that have the highest values in the document-topic distribution matrix. With each topic, it selects top w words, which have the highest weights in the word-topic-weight matrix. As a result, the number of topical words for each small document is $|t| \times |w|$. In practice, the number of topics $|t| = 5$ and the number of topical words $|w| = 5$ were empirically fixed. As a result, each small document d_s or d_c has 25 topical words resulting 625 words on both sides for constructing the graph.

We summarize the inference in Algorithm 1. It first loops over documents in D, then achieves the document distribution of document d_s (or d_c) over topics from the document-topic distribution matrix M_{DT} for word-weight-normalization. Next, top five closest topics of d_s are selected from the document-topic distribution. After that, a loop over selected topics is conducted to select top five topical words. For each closest topic, the algorithm selects a set of words with their weights and computes a document-topic value from $dTopic$. Finally, top five topical words of each topic are retrieved. The weight of a word is normalized by the document-topic value. The computational time of this algorithm is $O(|D||N||W|)$. Note that the topical words of d on the social side were also generated in the same mechanism. Table 3.8 shows an example of topical words and their weight.

[8]http://mallet.cs.umass.edu

Algorithm 1: Document inference algorithm. d denotes for both d_s and d_c.

```
Data: A document distribution matrix $M_{DT}$ and word distribution matrix $M_{WT}$ over
      $k$ topics; the number of closet topic $N$ and the number of topical words $W$.
Result: Topical words and its distribution value.
$lstDocWord = null$;
for ($d$ in $D$) do
    $dTopic \leftarrow$ getting the document distribution of $d$ from $M_{DT}$;
    $topNTopic \leftarrow$ getting top $N$ closest topics from $dTopic$;
    $tmp = \{\}$;
    for ($idx = 0; idx < topNTopic.length; idx++$) do
        $tmp \leftarrow$ getting top $W$ topical words of each topic $idx$ from $M_{WT}$;
        $dtValue \leftarrow dTopic(idx)$;
        for ($w$ in $tmp.keys()$) do
            $tmp[w] = dtValue * tmp[w]$;
        end
    end
    $lstDocWord \leftarrow tmp$;
end
```

Table 3.8: Topical words and their weights of Boston man shot by police (121).

police	officers	investigation	guilty	islamic	security	states
2.161	0.603	0.294	$1.55*10^{-4}$	$1.38*10^{-4}$	$1.17*10^{-4}$	$8.28*10^{-5}$

Topical words benefit our model in simulating the word variation hypothesis, in which summary sentences or user posts should contain salient words. These words also provide an insightful method to integrate social information into the summarization process by using semantic similarity among topical words. By using these words, our model can effectively build semantic similarity graphs which are used for ranking.

3.2.2 Sentence scoring

We define a Dual Wing Semantics Similarity Graph (DWSSG) for modeling relationships of sentence-user-post pairs. A *DWSSG* is $G_d = (C_d, V_d, E_d, W_d)$, in which C_d is the social context (a document and its comments or tweets) of a document d; V_d is a set of vertices containing two types of node: topical words and sentences (or user posts); E_d includes inter-relations and intra-relations; W_d is a weight matrix. Next sections describe the calculation of these relations.

Inter-score calculation

We present a sentence-user-post relationship by using semantic similarity calculated through topical words due to word variation from readers, e.g. *"officer"* and *"police"*. Eq. (3.22) defines the calculation of inter-relations:

$$inter_{sc}(s_i, C_d) = topic_{sc}(s_i) + sim(s_i, C_d) \tag{3.22}$$

where $topic_{sc}()$ returns the score of sentence s_i regarding topical words generated from LDA; the $sim()$ function models the word variation hypothesis by calculating the semantic similarity of s_i and its social context C_d. Eq. (3.22) consists of two components: the topic score of s_i and the similarity of s_i with its context. It shows that a summary sentence should contain important topics presented by topical words on one side, e.g. document, and has a high similarity with its social information in C_d. Given a set of topical words $T_d = \{w_1, ..., w_k\}$ of d_s inferred by LDA, Eq. (4.9) computes the topic score of s_i:

$$topic_{sc}(s_i) = \frac{\sum_{j=1}^{k} weight(w_j)}{|s_i|} \text{ if } w_j \in s_i \tag{3.23}$$

where $weight()$ returns the word weight of w_j (normalized in [0, 1]) in s_i;[9] and i is the index i^{th} of the sentence. Eq. (3.24) presents the $sim()$ function:

$$\begin{split} sim(s_i, C_d) = \frac{1}{W_s} \sum_{h}^{W_s} \sum_{j}^{T_d} \sum_{k}^{T_c} \sum_{p}^{C} \sum_{q}^{W_c} (wSim(w_h, w_j) + wSim(w_j, w_k) + \\ wSim(w_k, w_q) + topic_{sc}(c_p)) \end{split} \tag{3.24}$$

where W_s is the number of words in sentence s_i; T_d is the number of topical words of a small document d_s; T_c is the number of topical words of a small document d_c; C is a set of user posts; W_c is a set of words in a user post c_p; and $wSim()$ returns the semantic similarity between two words calculated by *Word2Vec* (Mikolov et al., 2013).

Equation (3.24) connects a sentence s_i in a document d and its social context by using word semantic similarity via salient words on two sides. It indicates that a summary sentence in Eq. (3.22) should not only have a high similarity with user posts but also cover important topics mentioned on the social side.

Intra-score calculation

It is possible to utilize word-word similarities again to calculate intra-relations; however, there are many common words among sentences; thus, we adopted intra-relations in Section 3.1.2 for this calculation. Table 3.9 shows features adopted from Section 3.1.1.

After defining features, Eq. (3.25) computes the intra-score of a sentence (or user post):

$$intra_{sc}(s_i) = \frac{1}{n} \sum_{j=1}^{n} simScore(s_i, s_j) \tag{3.25}$$

where s_i and s_j are two sentences; $simScore()$ returns a similarity value between s_i and s_j; n is the number of sentences. Eq. (3.25) indicates that a summary sentence has a

[9]all scores are computed with stopword removal.

Table 3.9: The features; S: a sentence, C: a comment or tweet.

Features	Description
Manhattan	Manhattan distance between S and C.
Euclidean	Euclidean distance between S and C.
Cosine	Cosine similarity between S and C.
Word matching	Number of common words between S and C.
Dice coefficient	Dice distance between S and C.
Jaccard coefficient	Jaccard distance between S and C.
JaroWinkler	Jaro-Winkler distance between S and C.
Damerau-Levenshtein	Damerau-Levenshtein distance between S and C.
Levenshtein distance	Levenshtein distance between S and C.
LSC(S, C)	Longest common substring between S and C.
Inclusion-exclusion	The division of common words for total length of S and C.
% words of S in C	% of words in S appearing in C.
% words of C in S	% of words in C appearing in S.
Word overlap coefficient	Division of word matching for minimum length of (S, C).

higher score than other ones on the same side. The *simScore*() function is computed by the following equation:

$$simScore(s_i, s_j) = \frac{1}{F}\sum_{k=1}^{F} f_k(s_i, s_j) \tag{3.26}$$

where F is the number of features in Table 3.9; $f_k()$ is a similarity function calculated by each k^{th} feature. Note that this score is also normalized in [0, 1].

Social context integration

The social context of a document is integrated into the scoring by using a linear combination of intra-score and inter-score as in Eq. (3.27).

$$score(s_i) = \delta * intra_{sc}(s_i) + (1 - \delta) * inter_{sc}(s_i) \tag{3.27}$$

where δ is a balanced parameter which controls the contribution of social information. The sensitivity analysis of δ is shown in Section 3.2.4. The score of a sentence contains two parts: an intra-score and an inter-score. The first score presents intra-relations and the second score simulates the word variation. Note that Eqs. (3.22), (4.9), (3.24), (3.25), (3.26), and (3.27) were also applied to calculate the score of a user post in the same mechanism. For example, the score of a comment or tweet is computed by Eq. (3.28):

$$score(c_j) = \delta * intra_{sc}(c_j) + (1 - \delta) * inter_{sc}(c_j) \tag{3.28}$$

After scoring, sentences and user posts have scores, which indicate their importance. Their scores were used to select summary sentences and user posts in the next section.

3.2.3 Sentence selection

This section presents two methods for sentence selection. The first method is a simple greedy method, which bases on scores of sentences and user posts generated from the graph. The second one is a novel ILP model, which defines a set of constraints to select summary sentences and user posts in a simultaneous fashion.

Score-based selection

This method is named as SowsRank, which uses scores from the graph for ranking to select summaries. After ranking, it iteratively dequeues one sentence (or one user post) from the sorted list and appends it to form a summary. The iteration stops if the number of selected sentences reaches a certain value, e.g. $m = 6$.

ILP-based selection

The score-based method selects summary sentences and user posts individually; therefore, it may ignore relationships of sentences and user posts in the selection step. Therefore, we utilized a kind of re-ranking by presenting a novel ILP model for simultaneously extracting summaries. In order to do that, we start with a simple ILP model used for sentence extraction in Woodsend and Lapata (2010).

$$\max_{x} \sum_{i \in S} f_i x_i \tag{3.29}$$

$$\text{s.t.} \sum_{i \in S_t} x_i \geq 1 \qquad \forall t \in T \tag{3.30}$$

$$\sum_{i \in S} x_i \leq N_S \tag{3.31}$$

$$x_i \in \{0, 1\}; \qquad \forall i \in S \tag{3.32}$$

The simple ILP model selects summary sentences which maximize an objective function in Eq. (3.29). f_i is the average Cosine score of a sentence (or user post) i^{th} with remaining ones in the same side. For example, when computing the score of a sentence s_i, all other sentences in a document d containing s_i are used. Eq. (5.6) defines this calculation.

$$score(s_i) = \frac{1}{n} \sum_{j=1}^{n} cos(s_i, s_j) \tag{3.33}$$

where n is all other sentences in d; $cos()$ returns a Cosine similarity between two sentences using the bag-of-words model. Similarly, all other user posts are used to score a comment or tweet c_j. The constraint in Eq. (3.30) states that a summary sentence has to include at least one important word, which owns a high TF-IDF value. The TF of a word w is computed on the document level, and IDF is calculated on the sentence level. We consider each sentence as a single document and count the number of sentences containing the word

w as DF. In practice, we select top 10 important words, which are two times greater than Woodsend and Lapata (2010). The constraint in Eq. (3.31) restricts the number of summary sentences. $x_i \in \{0, 1\}$ in Eq. (3.32) is a binary variable denoting whether a sentence is selected or not.

The simple ILP model exists two potential issues. Firstly, it uses Cosine similarity scores calculated on one side (for sentences or user posts), then it ignores the social context of a Web document. Secondly, it separately selects summary sentences and user posts. Based on their relationships we argue that they can be jointly selected. Therefore, we introduce an extended ILP model with four significant improvements compared to the basic one. Firstly, it selects summary sentences and user posts simultaneously to ensure content consistency. Secondly, it replaces Cosine values by scores generated from the DWSSG graph to integrate social information into the ILP model. In addition, it collects important words from both sides instead of only using words from documents as Woodsend and Lapata (2010). This is because user posts also include other important information. Finally, it considers two similar sentences (or user posts) should not be included in a final summary. This is to tackle the diversity aspect in summarization (Cao et al., 2015b,c; Ren et al., 2017). Before showing our ILP model, we first present notations in Table 3.10.

Table 3.10: Notations used in SowsRank ILP.

Notation	Meaning
f_i	The score of a sentence i^{th} DWSSG.
f_j	The score of a user post j^{th} DWSSG.
f_{ij}	The ROUGE-1 F-score between sentence x_i and user post c_j.
T_{SC}	A set of words with the highest TF-IDF scores (important words) on both sentences and user posts.
S	A set of sentences of a document d.
C	A set of comments or tweets corresponding to a document d.
N_S	The number of summary sentences.
N_C	The number of summary user posts.
$sim(a, b)$	Cosine similarity of two sentences or user posts.
v	A Cosine similarity threshold.
x_i	Indicator variables for sentences in main documents.
y_j	Indicator variable for user posts.
a_{ij}	Indicator variable of ROUGE-1 F-score between sentence i^{th} and user post j^{th}.
i	Index of a sentence i^{th}.
j	Index of a user post j^{th}

Given notations in Table 3.10, Eqs. (3.34)-(3.43) describe our ILP model:

$$\max_{x,y,a} \sum_{i \in S} \sum_{j \in C} (f_i x_i + f_j y_j + f_{ij} a_{ij}) \tag{3.34}$$

$$\text{s.t.} \sum_{i \in S_t} x_i \geq 1 \text{ and } \sum_{j \in C_k} y_j \geq 1 \qquad \forall t, k \in T_{SC} \tag{3.35}$$

$$if(sim(x_i, x_g) \geq v) : x_i + x_g \leq 1 \qquad \forall i, g \in S \tag{3.36}$$

$$if(sim(y_j, y_h) \geq v) : y_j + y_h \leq 1 \qquad \forall j, h \in C \tag{3.37}$$

$$x_i - a_{ij} \geq 0 \qquad \forall i \in S \text{ and } \forall j \in C \tag{3.38}$$

$$y_j - a_{ij} \geq 0 \qquad \forall j \in C \text{ and } \forall i \in S \tag{3.39}$$

$$\sum_{i \in S} x_i \leq N_S \quad \text{and} \quad \sum_{j \in C} y_j \leq N_C \tag{3.40}$$

$$x_i \in \{0, 1\} \qquad \forall i \in S \tag{3.41}$$

$$y_j \in \{0, 1\} \qquad \forall j \in C \tag{3.42}$$

$$a_{ij} \in \{0, 1\} \qquad \forall i \in S \text{ and } \forall j \in C \tag{3.43}$$

The objective function in Eq. (3.34) models exactly the score calculation in Figure 3.7, in which it selects summary sentences and user posts simultaneously by using their scores generated from SowsRank. This method is named as SowsRank ILP. The objective function includes three components: (i) the score f_i of a sentence, (ii) the score f_j of a user post, and (iii) the similarity score f_{ij} between them computed by ROUGE-1 F-1. For each sentence, the model tries to select user posts, which maximize their scores computed by SowsRank and similarity scores between this sentence and user posts. Adding $f_{ij}a_{ij}$ ensures that user posts highly related to summary sentences should be also selected.

The constraint in Eq. (3.35) states that summary sentences and user posts have to contain at least one important word. It is possible to use topical words generated from the LDA model; however, we use important words extracted by using TF-IDF inside documents and their social context to form T_{SC} as the same setting of Woodsend and Lapata (2010). In addition, topical words may come from different documents, which limit Eq. (3.35). Compared to the original ILP model in Eqs. (3.29)-(3.32), our new model argues that a summary should include important words from both sentences and user posts due to the word variation. In addition, the number of comments or tweets is usually larger than that of sentences; therefore, the number of important words collected from user posts should also be larger than that on sentences. Suppose that T_S and T_C are sets of top important words on sentences and comments or tweets, we empirically fixed $|T_S| = \frac{2}{3}|T_C|$. We merged T_S and T_C into a single set named T_{SC}. If a word appears twice in both T_S and T_C, we kept one. By adding words from user posts, our new ILP model again utilizes the support of social information in the summarization process.

Equations (3.36) and (3.37) consider diversity in the selection step. They indicate that two selected sentences or user posts have to be different in term of content defined by a Cosine similarity threshold. Eqs. (3.38) and (3.39) support the third component in Eq. (3.34), stating that a sentence and user post with a high similarity should be selected.

Constraints in Eq. (3.40) restrict the number of summary sentences ($N_S = N_C = m$). All indicator variables in Eqs. (3.41)-(3.43) are in $\{0, 1\}$.

We applied this model to re-rank sentences and user posts. Those satisfy the objective function and constraints were selected and put into summaries.

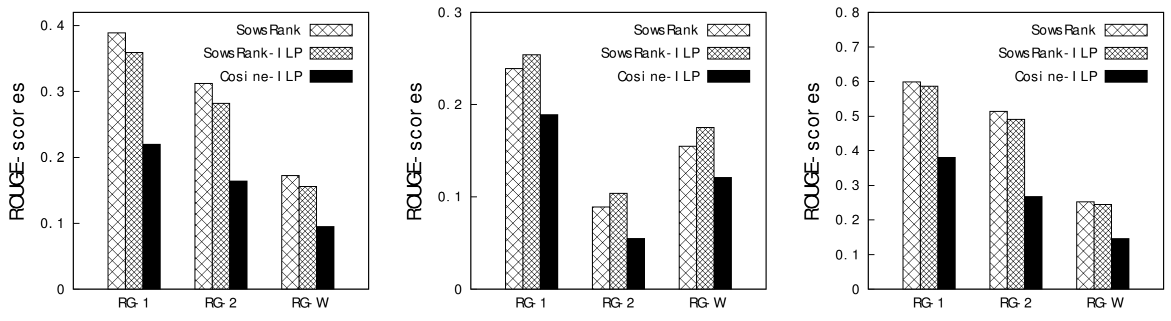
3.2.4 Results and Discussion

To measure the efficiency of our model, we first show ROUGE-scores of our model compared to strong baselines. The comparison answers two questions: (i) whether the performance of our model can compare to other methods and (ii) whether our approach is efficient. We next present observation for showing several aspects of our model.

ROUGE-scores

This section first shows ROUGE-scores of our model compared to Cosine ILP. It next reports the comparison with non-social context and social context methods.

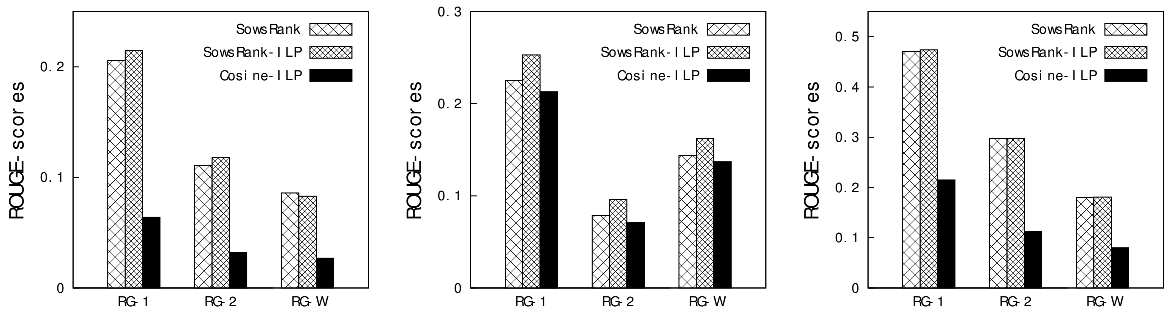
Our model vs. Cosine ILP We first compared our model with Cosine ILP, which our model bases on. We report their ROUGE-scores in Figures 3.9 and 3.10. From these figures, we observe that SowsRank and SowsRank ILP significantly outperform Cosine ILP in all metrics, for sentence and user post extraction. Significant improvements come from two factors. Firstly, our model exploits social context for both scoring and selection. Secondly, our SowsRank ILP uses several new constraints, which ensure to select high-quality summaries. On the other hand, Cosine ILP is limited because it does not consider social context in the scoring step and it uses a small set of important words on one side. ROUGE-scores also indicate that our ranking method outputs better scores for our ILP model than Cosine similarity. It is understandable that the ranking encodes inter-relations by using semantic similarity, which can handle the word variation.



(a) SoLSCSum (b) USAToday-CNN (c) VSoLSCSum

Figure 3.9: Our model vs. Cosine ILP for sentence selection.

We conducted a pairwise t-test to confirm significant improvements of our model with Cosine ILP. To do that, we observed ROUGE-scores of these methods and employed the test on each fold. $p-$values in Table 3.11 statistically confirm the improvements of our



(a) SoLSCSum (b) USAToday-CNN (c) VSoLSCSum

Figure 3.10: Our model vs. Cosine ILP for user post extraction.

model compared to Cosine ILP. Our model significantly surpasses the basic one in almost all cases, e.g. on SoLSCSum and VSoLSCSum. On USAToday-CNN, Cosine ILP is competitive for sentence selection. A possible reason is that Cosine calculation is efficient on this dataset, in which it can effectively measure the importance of each sentence by summing scores from other ones. For tweet extraction, our model is better.

Table 3.11: The pairwise t-test of our model vs. Cosine ILP, in which SR means SowsRank. **Bold** is significant with $p \leq 0.05$.

Data	Method	Sentence			User posts		
		RG-1	RG-2	RG-W	RG-1	RG-2	RG-W
SoLSCSum	SR	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
	SR ILP	**0.0000**	**0.0000**	**0.0001**	**0.0000**	**0.0000**	**0.0000**
USAToday-CNN	SR	0.3228	0.5724	0.5422	0.4065	0.4546	0.4189
	SR ILP	**0.0129**	**0.0243**	**0.0142**	**0.0345**	0.0558	**0.0240**
SoVLSCSum	SR	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0001**
	SR ILP	**0.0000**	**0.0001**	**0.0001**	**0.0002**	**0.0000**	**0.0006**

Comparison with non-social context methods We next compared our model with non-social context methods shown in Section 3.1.4. The general trend in Table 3.12 indicates that SowsRank and SowsRank ILP achieve competitive results, especially on USAToday-CNN and VSoLSCSum. In many cases, our methods obtain significant improvements (denoted by † with $p \leq 0.05$) compared to baselines, e.g. 0.215 vs. 0.152 of ROUGE-1 for comment extraction on SoLSCSum, 0.162 vs. 0.128 of ROUGE-W for tweet summarization on USAToday-CNN. This again confirms the efficiency of our model and constraints.

SowsRank significantly outperforms baselines in several cases, e.g. 0.514 vs. 0.440 of ROUGE-2 for sentence selection on VSoLSCSum. In other cases, it still surpasses baselines, e.g. for comment extraction on VSoLSCSum or for ROUGE-2 of sentence selection on SoLSCSum. This indicates that a sophisticated scoring method with a simple

Table 3.12: Summarization performance; **bold** is the best value; *italic* is the second best. RG stands for ROUGE.

Dataset	Method	Sentences			User posts		
		RG-1	RG-2	RG-W	RG-1	RG-2	RG-W
SoLSCSum	Lead-m	0.345	*0.322*	0.170	—	—	—
	LexRank	0.327†	0.243†	0.138†	*0.210*	*0.115*	*0.085*
	SVM*	0.325†	0.263†	0.147†	0.152†	0.089	0.062†
	CRF*	**0.393**	**0.379**	**0.187**	0.091†	0.075†	0.037†
	SowsRank	*0.389*	0.312	*0.172*	0.206	0.111	**0.086**
	SowsRank ILP	0.359	0.282	0.156	**0.215**	**0.118**	0.083
USAToday-CNN	Lead-m	0.249	**0.106**	*0.172*	—	—	—
	LexRank	0.251	0.092	0.163	0.193†	0.068†	0.128†
	SVM*	**0.261**	**0.106**	0.171	0.221	*0.084*	*0.149*
	CRF*	0.186	0.088	0.114	0.190†	0.065†	0.119†
	SowsRank	0.239	0.089	0.155	*0.225*	0.079	0.144
	SowsRank ILP	*0.254*	*0.104*	**0.175**	**0.253**	**0.096**	**0.162**
VLSCSum	Lead-m	0.495†	0.420†	0.214†	—	—	—
	LexRank	0.506†	0.432†	0.219†	0.348†	0.198†	0.127†
	SVM*	0.497†	0.440†	0.208†	0.374†	0.212†	0.140†
	CRF*	0.422†	0.357†	0.172†	0.111†	0.062†	0.041†
	SowsRank	**0.599**	**0.514**	**0.252**	*0.471*	*0.297*	*0.180*
	SowsRank ILP	*0.587*	*0.491*	*0.245*	**0.474**	**0.298**	**0.181**

ranking approach can extract high-quality summaries. This is because the score of a sentence or a user post is computed in an accumulative fashion, in which a score is modeled by two parts: an intra-score and an inter-score. In this view, outputs from SowsRank cover salient information on both document and social sides. In addition, the integration of social context by using semantic similarity among topical words benefits the scoring and selection steps. For example, on SoLSCSum, SowsRank acquires sufficient improvements, e.g. 0.389 vs. 0.357 compared to score-based Dual-Wing in Table 3.12, that models sentence-tweet pairs by using lexical similarity features. On USAToday-CNN, SowsRank produces competitive results, e.g. 0.239 vs. 0.261 of ROUGE-1 for sentence selection. This is because highlights in USAToday-CNN are created in an abstract way, in that writers read the content of a Web document, then generate 3-4 summaries. In this sense, only using the scoring may not completely cover the abstract aspect of summaries. However, even with this challenge, it is still the second best of ROUGE-1 for tweet extraction.

SowsRank ILP achieves sufficient improvements compared to SowsRank in some cases, e.g. 0.254 vs. 0.239 of ROUGE-1 for sentence summarization and is the best for tweet extraction on USAToday-CNN. Our constraints treat the abstract aspect in generating highlights from sentences by requiring that extracted sentences and tweets have to include important words on both sides and to be selected simultaneously. In this mechanism, tweets produced by the word variation from readers support to pick up important sentences. The definition of new concepts boosts the performance of sentence selection. On SoLSCSum and VSoLSCSum, SowsRank ILP achieves very competitive results, in which it is the best in extracting comments on VSoLSCSum. The general trend of SowsRank and SowsRank ILP indicates that if SowsRank extracts high-quality summaries, adding more constraints is unnecessary, for example, on SoLSCSum. On the other hand, if the scoring step is not efficient enough, putting new constraints profits sentence selection. In practice, we can use both methods to improve the quality of summarization. The trend also shows that SowsRank may be more appropriate for extractive summarization (on SoLSCSum and VSoLSCSum) whereas SowsRank ILP is suitable for abstractive summarization (USAToday-CNN).

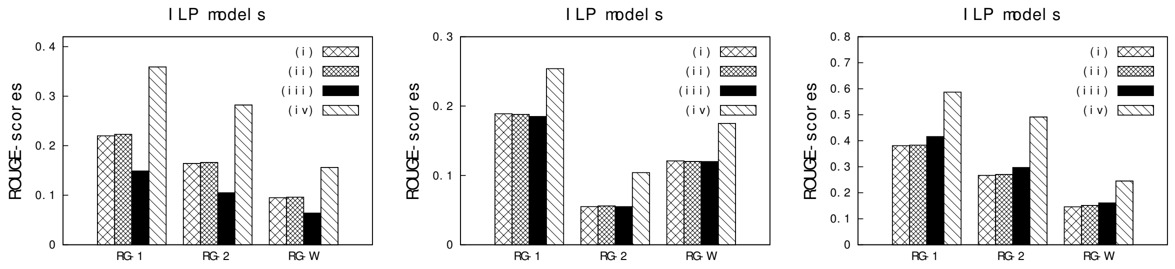
Comparison with social context methods We challenged our model to state-of-the-art methods stated in Section 3.1.4. Table 3.13 summarizes our comparison.

Table 3.13: Our model vs. advanced methods; **bold** is the best value; *italic* is the second best. RG stands for ROUGE.

Dataset	Method	Sentences			User posts		
		RG-1	RG-2	RG-W	RG-1	RG-2	RG-W
SoLSCSum	cc-TAM	$0.306^{\dagger}$	$0.238^{\dagger}$	$0.136^{\dagger}$	$0.054^{\dagger}$	$0.022^{\dagger}$	$0.024^{\dagger}$
	HGRW	*0.379*	0.204	*0.167*	*0.209*	**0.115**	*0.084*
	RB CCF*	0.360	*0.283*	0.158	$0.190^{\dagger}$	0.098	$0.077^{\dagger}$
	SowsRank	**0.389**	**0.312**	**0.172**	0.206	*0.111*	**0.086**
	SowsRank ILP	0.359	0.282	0.156	**0.215**	**0.118**	0.083
USAToday-CNN	cc-TAM	0.229	0.077	0.145	*0.249*	*0.089*	0.152
	HGRW	**0.279**	*0.098*	**0.177**	0.242	0.088	*0.157*
	RB CCF*	0.221	0.070	0.140	0.233	0.091	0.132
	SowsRank	0.239	0.089	0.155	0.225	0.079	0.144
	SowsRank ILP	*0.254*	**0.104**	*0.175*	**0.253**	**0.096**	**0.162**
VLSCSum	cc-TAM	$0.488^{\dagger}$	$0.377^{\dagger}$	$0.201^{\dagger}$	$0.301^{\dagger}$	$0.167^{\dagger}$	$0.111^{\dagger}$
	HGRW	0.570	$0.479^{\dagger}$	$0.233^{\dagger}$	0.454	*0.298*	0.173
	RB CCF*	0.561	0.494	$0.235^{\dagger}$	*0.471*	**0.308**	0.168
	SowsRank	**0.599**	**0.514**	**0.252**	*0.471*	0.297	*0.180*
	SowsRank ILP	*0.587*	*0.491*	*0.245*	**0.474**	*0.298*	**0.181**

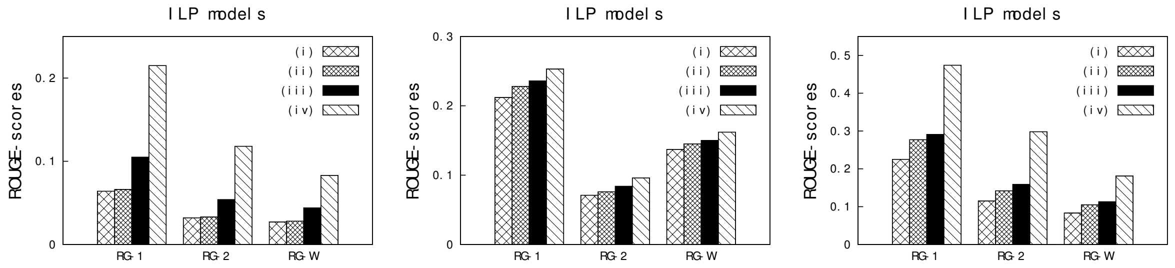
ROUGE-scores indicate that SowsRank is competitive on SoLSCSum and VSoLSCSum, whereas SowsRank ILP outperforms most of methods on USAToday-CNN. We can observe that HGRW is a powerful method (Wei and Gao, 2015). It is understandable that HGRW integrates social information based on a variation of LexRank. Therefore, it is better than most of the methods in Table 3.13. cc-TAM is the second best in extracting tweets on USAToday-CNN but obtains poor results on other ones. It is possible that cc-TAM is designed for multi-document summarization (Gao et al., 2012) but all our datasets are for single-document summarization. Also, the noise of social information may affect its topic model. RankBoost CCF achieves stable and competitive ROUGE-scores because it utilizes many sophisticated features in the form of learning to rank. However, our model is still comparable even it is a supervised learning method.

ILP models We investigated summarization performance using ILP by comparing four versions of our model: (i) Cosine ILP, (ii) Simple SowsRank ILP, (iii) SowsRank ILP using top 10 words in T_{SC}, and (iv) SowsRank ILP. Cosine ILP is the baseline of sentence extraction. Simple SowsRank ILP is a variation of Cosine ILP with the same setting, but it uses scores from our scoring step instead of using Cosine scores. SowsRank ILP (T_{SC}-10) is our model in Eqs. (3.34)-(3.43) but it only uses top 10 words in T_{SC} (the same setting with (i) and (ii)). And finally, SowsRank ILP is the full setting of our model.



(a) ROUGE-1 (b) ROUGE-2 (c) ROUGE-W

Figure 3.11: ROUGE-scores of ILP models for sentence selection.



(a) ROUGE-1 (b) ROUGE-2 (c) ROUGE-W

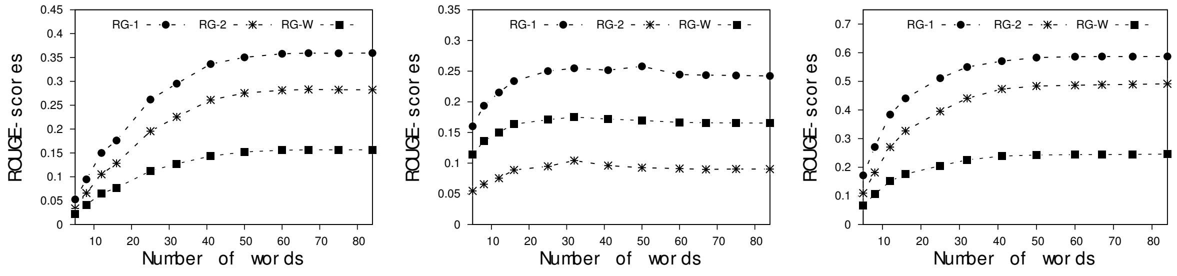
Figure 3.12: ROUGE-scores of ILP models for comment extraction.

Results in Figures 3.11 and 3.12 show that SowsRank ILP with the usage of important words from both sides significantly outperforms other methods. There are two reasons behind this. Firstly, it selects important sentences and user posts simultaneously instead of using an individual selection as Cosine ILP or Simple SowsRank ILP. Secondly, it uses a larger number of important words from both sides instead of only using 10 important words from one side (documents or user posts). This leads to an argument that the number of words in T_{SC} extremely affects our ILP model. Results of SowsRank ILP (T_{SC}-10) and SowsRank ILP also support this conclusion, in which SowsRank ILP acquires significant improvements compared to a model, which only uses top 10 words. This also supports the *generation* hypothesis stated in Section 2.3.2, in that readers tend to borrow words

because the contribution of social information is limited with a large value of δ (see Eq. (3.27)). Therefore, we empirically set $\delta = 0.85$ for three datasets.

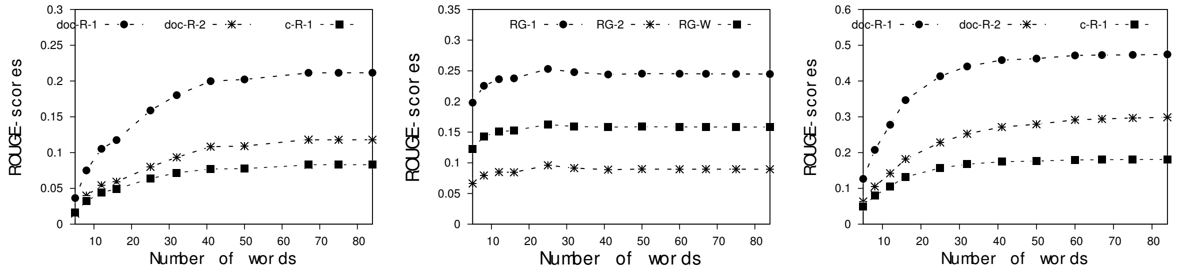
Important word observation

As our assumption in Section 3.2.3, the number of important words (those have high TF-IDF scores) significantly affects our SowsRank ILP. We, therefore, analyzed their effectiveness in T_{SC} by tuning T_{SC} in [5, 84]. As showed in Eq. (3.35), T_{SC} was created by combining T_S (from sentences) and T_C (from user posts). More precisely, we adjusted T_C in $\{3, 5, 7, 10, 15, 20, 25, 30, 35, 40, 45, 50\}$ and calculated the number of words in T_S by using $|T_S| = \frac{2}{3}|T_C|$. For example, given top three important words from user posts, we select top two words from sentences to create five words in T_{SC}. We plot the trend of ROUGE-scores on Figures 3.15 and 3.16.



(a) SoLSCSum (b) USAToday-CNN (c) VSoLSCSum

Figure 3.15: Tuning important words on documents.



(a) SoLSCSum (b) USAToday-CNN (c) VSoLSCSum

Figure 3.16: Tuning important words on user posts.

Figures 3.15 and 3.16 reveal that the number of words in T_{SC} significantly influences SowsRank ILP. The summarization performance is very low when our model uses a small set of words in T_{SC}. Adding more words boosts its ROUGE-scores until a certain number. After that, the performance is stable or slightly decreases. For example, the trend of ROUGE-scores is consistent in Figures 3.15a and 3.15c, in which SowsRank ILP obtains poor results with $|T_{SC}|$ =5. By increasing the size of $|T_{SC}|$ from 5 to 50, SowsRank ILP achieves significant improvements. Subsequently, its performance is quite stable. In

Figure 3.15b and 3.16b our model reaches the top of ROUGE-scores with $|T_{SC}| =25$. After that, its ROUGE-scores tend to decrease. We, therefore, empirically chose $|T_{SC}| = 75$ for SoLSCSum and VSoLSCSum, and $|T_{SC}| = 25$ for USAToday-CNN.

Word overlapping observation

As shown in Table 2.5 of Section 2.3.2, sentences and user posts share common words or phrases. We used extracted sentences and comments of SowsRank on SoLSCSum to observe word overlapping, in which each token was considered as a single word. We split sentences and comments to create two separate sets of words. Subsequently, we counted the percentage of common words between these sets. Table 3.15 summarizes our observation.

Table 3.15: Word overlapping observation; *sents* are sentences, *coms* are comments.

# Extracted sents/coms	% sents share common words	% comments share common words	% similar words in sents/coms	# similar words in coms/sents
948	41.66	60.12	62.76	69.13

Results from Table 3.15 again validate data observation in Table 2.5. Firstly, there is a considerable number of sentences and comments sharing common words. For example, 41.66% of extracted sentences share common words with comments. This number is even larger for comments with 60.12%. Secondly, extracted summaries share common words, i.e. 62.76% common words in sentences over comments and 69.13% common words in comments over sentences.

We also validated common topics between extracted sentences and comments by checking common topical words generated by LDA in Section 3.2.1. To do that, all topical words of documents and their comments were first collected (they already inferred by LDA). Because they were used for each small document d_s and d_c, then they were also considered as topical words of extracted sentences and comments. Suppose we have two sets of topical words, TW_d for selected sentences and TW_c for extracted comments, we count word overlapping between these sets. Table 3.16 shows counting numbers.

Table 3.16: The observation of topical words generated by LDA.

# topical words	% similar words in TW_d apppearing in TW_c	# similar words in TW_c appearing in TW_d
3,925	42.21	44.84

Results from Table 3.16 confirm the *common topic* hypothesis in Section 2.3.2, in which both selected sentences and extracted comments share topical words, i.e. 42.21% words in sentences appearing in comments and 44.84%, vice versa. These results also support the *word variation* hypothesis, in which readers borrow topical words from sentences to create their comments. The number of common words in Table 3.16 also shows that our method is efficient in dealing with sentences and comments which lack *the same* words. In Table

3.16, although comments and sentences share 44.84% of common words, our model can still calculate their similarities because it utilizes semantic relationships among words.

Error analysis

This section illustrates the summary of the document *"Boston man shot by police was target of terrorism probe"*[10] generated from SowsRank and SowsRank ILP on SoLSCSum. In Table 5.17, SowsRank selects one correct sentence and comment (denoted by [+]) which mention the Boston man shot by police event. This is because summary sentences contain important words and also take high intra-scores, e.g. 0.887 of S1 and 0.670 of C1 in Figure 3.17. As a result, they are selected by the selection. Also, several words can be inferred by using Word2Vec among selected sentences. For example, *"police"* and *"cops"* are very close, e.g. 0.47; as a result, our model can select S1 and C1, those cover these words on both sides. The ILP model shares the same selected sentences because they receive high scores from SowsRank and own common words with C1 and C2, e.g. *"cops"*, *"Boston"*. Therefore, the similarity ROUGE-1 F-score (as shown in Eq. (3.34)) of S1 with C1 and C2 is larger than other ones.

Table 3.17: A summary example; two summaries are shown instead of six. [+] means that this sentence is in the references and [-] means that it is is not in the references.

SowsRank Summarization	
Sentences	**Comments**
[+]S1: The 26-year-old man, identified as Usaamah Rahim, brandished a knife and advanced on officers working with the Joint Terrorism Task Force who initially tried to retreat before opening fire, Boston Police Superintendent William Evans told reporters.	[+]C1: If I had been one of the police officers I would have whispered 3 times "drop the knife" then quickly fired several shots at his sternum.
[-]S2: Boston Police said in a statement on their website that "as part of this ongoing investigation, Boston Police and State Police made an arrest this evening in Everett.	[-]C2: Once again, the police did what they're supposed to do when threatened by an armed Muslim wacko who wasn't he was supposed to be doing.
SowsRank ILP Summarization	
Sentences	**Comments**
[+]S1: The 26-year-old man, identified as Usaamah Rahim, brandished a knife and advanced on officers working with the Joint Terrorism Task Force who initially tried to retreat before opening fire, Boston Police Superintendent William Evans told reporters.	[+]C1: Either those cops weren't switched on enough to grasp the scope of the threat or Boston PD needs to review their procedures for addressing these types of threats.
[-]S2: This suspect is in the process of being booked, fingerprinted and interviewed.	[-]C2: Once again, the police did what they're supposed to do when threatened by an armed Muslim wacko who wasn't he was supposed to be doing.

On the other hand, for SowsRank, non-important sentences, e.g. S2 and C2 also include topical words challenging our method. For example, S2 contains *"Boston"* and *"police"*; therefore, it obtains a high inter-score, i.e. 0.735 (Figure 3.17). However, these sentences are still relevant to the event. In addition, C2 mentions the religion of Rahim, i.e. *muslim*; therefore, it receives a lot of attention from readers leading a high intra-similarity score, i.e. 0.675. SowsRank ILP shares the same mistake with SowsRank in the case of C2, in which it is difficult to judge whether C2 is important even with humans. While C2 is still relevant to the event, S2 extracted by the ILP model does not provide any information which helps to infer the event. In this case, our constraints are limited.

[10]https://www.yahoo.com/news/boston-man-shot-police-target-terrorism-probe-officials-022407784.html?ref=gs

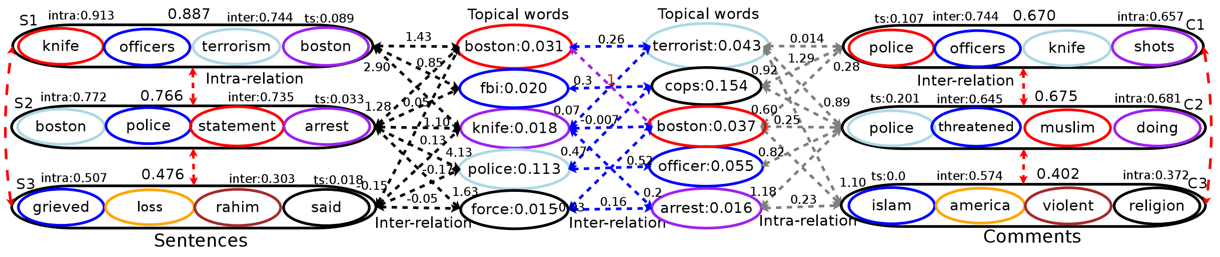


Figure 3.17: A running example of document 121th generated from SowsRank; *intra* is intra-score and *inter* is inter-score; *ts* is topical scores; sentences and comments are represented by a set of words. *S1*, *S2*, *C1*, and *C2* are taken from Table 5.17 whereas *S3* and *C3* are taken from the original article. The score of each topical word (0.031) is a word weight inferred by LDA, the score between two words (0.26) is a word similarity, and the score between a word and a sentence (1.43) is a word-sentence similarity.

We also show a running example generated by SowsRank on extracted sentences and comments in the first part of Table 5.17. Figure 3.17 presents the running process, which reveals important points. Firstly, summary sentences contain salient words and also own high intra-scores, e.g. S1 and C1. As a result, they are selected by the ranking (0.887 of S1 and 0.670 of C1). Secondly, non-summary sentences, e.g. S2 and C2 also include topical words, which challenge our method. In this sense, our method is limited. For example, S2 contains *"boston"* and *"police"*; therefore, it obtains a high inter-score, i.e. 0.735. In addition, irrelevant sentences, e.g. S3 and C3 do not include important words; hence, they achieve lower scores, e.g. 0.476 of S3 and 0.402 of C3. As a result, they are not included in the summary.

3.3 Conclusion

In this chapter, we show our first effort to investigate the integration of social context into the summarization process. We introduce two unsupervised ranking models, which consider two different viewpoints of the integration. Our models formulate mutual relationships of sentences and user posts by modeling their intra-relations and inter-relations, which are presented by a rich set of features, from the lexical to semantic level. Besides employing a simple greedy method, we also introduce a new ILP model used as re-ranking for improving the selection step. We validate the efficiency of our models on three datasets in two languages, English and Vietnamese. Experimental results show that our models achieve very competitive ROUGE-scores over strong methods. To make better observation, we put ROUGE-scores of our models in Table 3.18. It shows that SowsRank is better on SoLSCSum and VLSCSum while SowsRank ILP is more appropriate on USAToday-CNN. Score-based Inter-Wing and Dual-Wing are not the best on SoLSCSum and VSoLSCSum. A possible explanation is that inter-relations are not well captured by using our features. We suggest that a deeper analysis of feature contribution should be conduct with more sophisticated features, e.g. tree edit distance.

Table 3.18: The performance of our models. RG stands for ROUGE.

Dataset	Method	Sentences			User posts		
		RG-1	RG-2	RG-W	RG-1	RG-2	RG-W
SoLSCSum	Inter-Wing	0.352	0.277	0.154	*0.209*	*0.115*	*0.083*
	Dual-Wing	0.357	0.280	*0.156*	0.180	0.098	0.070
	SowsRank	**0.389**	**0.312**	**0.172**	0.206	0.111	**0.086**
	SowsRank ILP	*0.359*	*0.282*	*0.156*	**0.215**	**0.118**	*0.083*
USAToday-CNN	Inter-Wing	**0.268**	**0.104**	*0.173*	0.215	0.074	0.137
	Dual-Wing	0.252	0.085	0.163	*0.227*	*0.080*	*0.146*
	SowsRank	0.239	*0.089*	0.155	0.225	0.079	0.144
	SowsRank ILP	*0.254*	**0.104**	**0.175**	**0.253**	**0.096**	**0.162**
VLSCSum	Inter-Wing	0.532	0.463	0.218	0.443	0.277	0.170
	Dual-Wing	0.531	0.457	0.218	0.409	0.234	0.153
	SowsRank	**0.599**	**0.514**	**0.252**	*0.471*	*0.297*	*0.180*
	SowsRank ILP	*0.587*	*0.491*	*0.245*	**0.474**	**0.298**	**0.181**

We also analyze several aspects of our models to provide the better understanding of their operations. For the first model, we point out that: (i) different features should be used for sentences and user posts and (ii) the number of user posts affects the estimation. For the second model, analyses show two points. (i) Adding constraints is unnecessary when the scoring achieves good results, but in other cases, the constraints benefit in extracting summaries. (ii) The number of important words plays an important role in the ILP method because it considers summaries have to include atleast one important word. If the number of these words is small, they can not cover important sentences.

Chapter 4

Exploiting Common Topics with Matrix Co-factorization

This chapter is our second effort to investigate the integration of social information to improve the importance estimation of sentences and user posts. In this section, we introduce a model which exploits common topics between sentences and user posts. The insight of our model comes from the fact that sentences in a Web document and its user posts share common hidden topics denoted in the form of common words or phrases. Unlike our previous methods which base on hand-crafted features, this approach ranks sentences and user posts based on their importance affecting topics. To do that, we formulate sentence-user-post relationships in a share topic matrix, which presents their mutual reinforcement support. Our newly proposed matrix co-factorization algorithm computes the score of each sentence and comment and extracts top m ranked sentences and m comments as the summarization. For evaluation, we confirm the efficiency of our model on three datasets in two languages, English and Vietnamese. Experimental results indicate that our model achieves promising results compared to baselines and advanced methods. We published our work in Nguyen et al. (2017a). In next sections, we first show a short description of non-negative matrix factorization, document representation, and a basic model. We next introduce our matrix co-factorization, followed by experimental results and conclusion.

4.1 Non-negative Matrix Factorization

Non-negative Matrix Factorization (NMF)[1] is a useful method based on matrix computation to capture latent structure from non-negative data (Lin, 2007). NMF receives an input matrix X and factorizes X into to matrices W and H so that all elements in three matrices are non-negative. Suppose $X \in R^{n\times m}$ with $X_{ij} \geq 0$ and a pre-defined positive integer value $r < \min(n, m)$ (for document summarization, $r = k$ is the number of topics), NMF finds two non-negative matrices $W \in R^{n\times r}$ and $H^{r\times m}$ so that:

$$X \approx WH \tag{4.1}$$

[1]https://en.wikipedia.org/wiki/Non-negative_matrix_factorization

A usual approach to find W and H is to minimize the difference between X and WH.

$$\min_{W,H} \qquad f(W,H) \equiv \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{m}(X_{ij}-(WH)_{ij})^2 \tag{4.2}$$

$$\text{subject to} \qquad W_{ia} \geq 0, H_{bj} \geq 0 \qquad \forall i,a,b,j. \tag{4.3}$$

In term of optimization, Eq. 4.3 is a standard bound-constrained optimization problem. We also note that the sum part of Eq. (4.2) can be written as:

$$\sum_{i=1}^{n}\sum_{j=1}^{m}(X_{ij}-(WH)_{ij})^2 = ||X-WH||_F^2 \tag{4.4}$$

where $||\cdot||$ is the Frobenius norm. For optimization, NMF uses an iterative procedure to modify initial values of W and H so that their product approaches A. There are several techniques such as multiplicative update methods (Lee and Seung, 2001), alternating non-negative least squares (Lee and Seung, 2001; Lin, 2007), or gradient approaches (Lee and Seung, 2001; Lin, 2007) which can be employed to optimize Eq. (4.2). With its nice properties, NMF is usually used to reduce data dimension, which is beneficial in many problems such as finding bias vectors of images (Lee and Seung, 1999) or clustering for document summarization (Wang et al., 2008).

4.2 Document Representation

The first step of a NMF-based summarization system is to convert a document into a matrix, which can be used by decomposition. Given a document d with t terms and n sentences, we can use a term-sentence matrix X to represent d as Gong and Liu (2001). In this model, $X(i,j)$ is the weight of term t_i in sentence s_j computed by term frequency (TF). However, for our purpose, this representation is deficient since $X(i,j)$ is only calculated on the sentence level, which ignores the document level. We, therefore, adopt an approach of Nakov et al. (2001), which considers both local and global weights of a term.

$$X(i,j) = L(i,j) \times G(i) \tag{4.5}$$

where $L(i,j)$ is a local weight (TF) and $G(i)$ is a global weight (document inverse frequency - IDF). In other words, Eq. (4.5) can be referred as traditional TF-IDF. This representation is employed for both basic and advanced models given below.

4.3 Basic Model

We start with a basic model using NMF (Lee and Seung, 1999) to summarize documents (Gong and Liu, 2001; Lee et al., 2009; Park et al., 2006; Wang et al., 2008). Given a term-sentence matrix X, NMF finds two non-negative matrices W and H which approximate X, as described by Lin (2007).

$$X \approx WH \tag{4.6}$$

where W is a topic matrix, H is a sentence weight matrix presenting the influence or importance of sentences to topics given in W. To find W and H, an optimization procedure is needed for minimizing the following error function:

$$E = ||X - WH||_F^2 \tag{4.7}$$

If each column of X represents a sentence (an object), NMF approximates its linear decomposition in the bases of k topics (where k is the column size of W). The weight, WS_j, of each sentence is calculated as follows.

$$WS_j = \sum_{i=1}^{k} H_{ij} \times weight(H_i) \tag{4.8}$$

$$weight(H_i) = \frac{\sum_{q=1}^{n} H_{iq}}{\sum_{p=1}^{k} \sum_{q=1}^{n} H_{pq}} \tag{4.9}$$

where k is a chosen topic number and n is the number of sentences. After having the weight score of each sentence, WS_j, we can rank all sentences based on their scores on the matrix H and extract top m sentences for a summary.

4.4 Advanced Model with Matrix Co-factorization

The basic model only uses sentences to create a term-sentence matrix. It ignores the social context of a Web document, that can be used to enrich information of sentences. Here, we assume that user posts and sentences in a document share hidden topics denoted in the form of common words or phrases. Let's take Table 4.1 as an example.

Table 4.1: An extraction example

[S]: **Families** of the **victims** of the **Germanwings** crash are considering filing a **claim for damages** in the United States if they cannot reach agreement with parent airline Lufthansa in Germany, a lawyer representing the families said on Sunday.
[C]: I'm sad for the people who are no longer on their life of this **Germanwings** strategy, but I don't know how much **money relatives** want for the flesh of **who die** on **Germanwings** case.

We obverse that the sentence and comment share common or inferred words in bold, e.g. *"victims"* ~ *"who die"*, *"Germanwings"*, *"families"* ~ *"relatives"*. These terms form hidden topics presenting the nature of relationships between sentences and user posts. Also, as observed in Section 2.3.2, 42.05% words in comments are from sentences on SoLSCSum and 44.82% on VSoLSCSum. We, hence, present a co-factorization method for the representation of common topics between main documents and their user posts.

Figure 4.1 describes our model. It first maps a document and its user posts into two matrices X_1 and X_2, which share a hidden topic matrix W. The decomposition produces two matrices H_1 and H_2, which are analyzed by our new matrix co-factorization algorithm to estimate the importance of sentences and user posts. Our model finally analyzes two column-matrices H_1 and H_2 to extract top m ranked sentences and user posts as a summary. By introducing our matrix co-factorization model, we also validate the *common topic* hypothesis stated in Section 2.3.2.

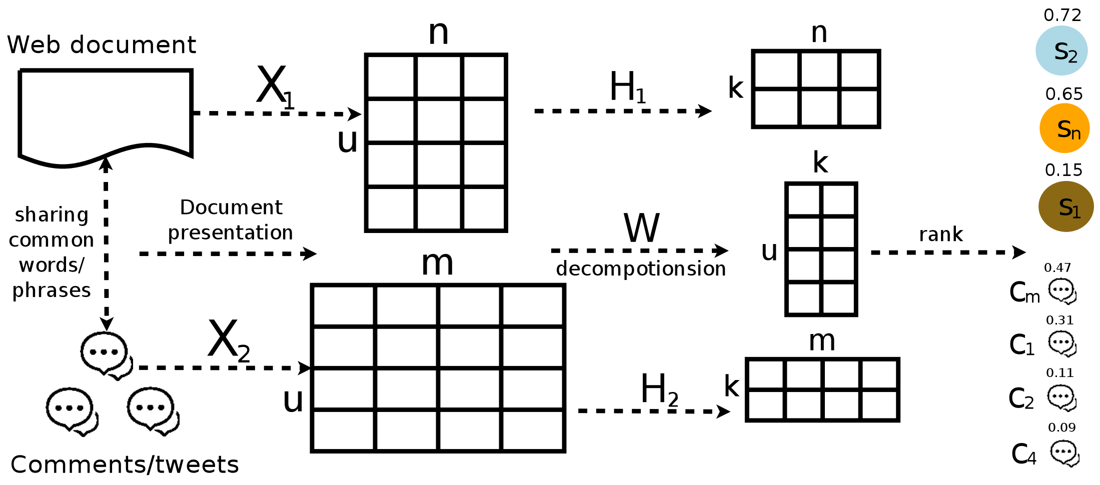


Figure 4.1: Our co-factorization summarization model.

Our model shares the idea of using matrix factorization for summarization with Wang et al. (2008); however, we extend this approach to our task by taking advantage of relevant user posts for single Web document summarization. Our model is different from prior NMF (Gong and Liu, 2001; Lee et al., 2009; Park et al., 2006; Wang et al., 2008) in that it integrates user posts into the ranking process and presents a matrix co-factorization algorithm for extracting summaries. We organize our model in two steps: matrix creation and non-negative matrix co-factorization, which are given bellow.

4.4.1 Matrix creation

Given a document d with n main text sentences and m user post sentences, we used two term-sentence matrices: X_1 and X_2 for presenting the main text and user posts respectively. To create these matrices, we first merged sentences and user posts into a single set and generate a term dictionary from this set by using word lemmatization, stemming, and stopword removal from `NLTK`.[2],[3] Suppose the size of the dictionary is u, hence we could present $X_1 = u \times n$ and $X_2 = u \times m$ ($n \ll u$ and $m \ll u$). We applied Eq. (4.5) to each component of X_1 and X_2 to compute their weights. The TF of a term w was computed on the sentence level, whereas its IDF was calculated on the document level. For example, if w appears in sentences, we consider each sentence as a single document and count the number of sentences including w as its document frequency (DF). If w appears in both document d and its user posts, two DF values are separately

[2]http://www.nltk.org

[3]We did not do lemmatization and stemming on VSoLSCSum since there are no effective algorithms for Vietnamese at the moment.

calculated for each side. This creation is the first level to exploit user posts for creating a term-sentence matrix, which encodes share hidden topics into the estimation.

4.4.2 Non-negative matrix co-factorization (NMCF)

This section introduces our matrix co-factorization. We first describe a model, in which X_1 and X_2 share the same number of topic k. Based on that, we present two versions of this model, in which the number of topics in X_1 and X_2 is different.

Case 1: the same topic number

As mentioned, documents and their social information share common topics, which can be the same or different in term of number. In this case, we assume that the number of topics in documents and their social context is the same. In fact, this assumption requires that all information posted by readers is relevant to topics in main documents.

Given X_1 and X_2, we can independently apply NMF to X_1 and X_2. However, as in the aforementioned assumption that they share hidden topics, and thus we argue that they can be jointly optimized in a unified co-factorization framework. Let k be the number of share hidden topics between main text sentences and user posts of a document d; we can rewrite Eq. (4.6) to represent the co-factorization of X_1 and X_2 as follows.

$$X_1 \approx WH_1 \tag{4.10}$$

$$X_2 \approx WH_2 \tag{4.11}$$

where $W \in R^{u \times k}$ is the matrix of share hidden topics; $H_1 \in R^{k \times n}$ is the sentence weight matrix, which represents the influence of main text sentences to hidden topics in W; and $H_2 \in R^{k \times m}$ is the user post weight matrix, which represents the influence of user posts to W. The sensitivity analysis of selecting topic number k is shown in Section 4.5.2.

For our co-factorization, the error function in Eq. (4.7) can be redefined as follows.

$$E = ||X_1 - WH_1||_F^2 + ||X_2 - WH_2||_F^2 \tag{4.12}$$

The new error function includes two components: one for main text sentences and the other one for user posts. The optimization procedure has to consider both components to achieve global optimization instead of optimizing only one component as Eq. (4.7). By jointly optimizing Eq. (4.12), this is the second level of utilizing user posts for our model. Eq. (4.12) also reveals an important characteristic of our model, that is, sentences and user posts are formulated in a mutual reinforcement support. To optimize Eq. (4.12), we adapted the gradient algorithm (Lee and Seung, 2001; Lin, 2007) for our co-factorization as it has been shown to be efficient in the literature. Eqs. (4.13)–(4.15) describe the gradient calculation in our proposed optimization algorithm.

$$\frac{\bigtriangledown E}{\bigtriangledown W} = (WH_1 - X_1)H_1^T + (WH_2 - X_1)H_2^T \tag{4.13}$$

$$\frac{\bigtriangledown E}{\bigtriangledown H_1} = W^T(WH_1 - X_1) \tag{4.14}$$

$$\frac{\bigtriangledown E}{\bigtriangledown H_2} = W^T(WH_2 - X_2) \tag{4.15}$$

with update rules as the following:

$$W = W \odot \frac{X_1H_1^T + X_2H_2^T}{WH_1H_1^T + WH_2H_2^T} \tag{4.16}$$

$$H_1 = H_1 \odot \frac{W^TX_1}{W^TWH_1} \tag{4.17}$$

$$H_2 = H_2 \odot \frac{W^TX_2}{W^TWH_2} \tag{4.18}$$

where $\odot$ is the Hadamard product. To optimize the objective function in Eq. (4.12), we use an iterative algorithm, which is described in Algorithm 2.

Algorithm 2: Computing error rate in our optimization algorithm.

```
   Data: Matrices $W$, $H_1$, and $H_2$.
   Result: The weights of these matrices.
1  1. Initialize $W \geq 0$, $H_1 \geq 0$, and $H_2 \geq 0$ ;
2  2. for $t = 1, 2, \ldots$ do
3      Update $W, H_1, H_2$ by using Eqs. (4.16), (4.17), and (4.18) ;
4      Compute error rate $E$ by using Eq. (4.12) ;
5      if $(E < \epsilon)$ then
6          break ;
7      end
8  end
9  Output: $W, H_1$, and $H_2$ ;
```

Our algorithm first initializes W, H_1, and H_2 with a constraint that all values ≥ 0. In each iteration, it computes values of W, H_1, and H_2 based on Eqs. (4.16), (4.17), and (4.18) and then calculates an error value in Eq. (4.12). The algorithm stops if the error value $\leq \epsilon$ or the number of iterations is larger than a certain value. For implementation, we fix $\epsilon = 0.01$ and $inter = 1000$. In practice, to ensure the uniqueness of our NMCF, we normalize W and H (H_1 and H_2) with L_1 or L_2.
Normalization L_1:

$$w_{ij} = \frac{w_{ij}}{\sum_i w_{ij}} \tag{4.19}$$

$$h_{ij} = h_{ij} \sum_i w_{ij} \tag{4.20}$$

or L_2:

$$w_{ij} = \frac{w_{ij}}{\sqrt{\sum_i w_{ij}^2}} \tag{4.21}$$

$$h_{ij} = h_{ij}\sqrt{\sum_i w_{ij}^2} \tag{4.22}$$

The effects of using L_1 or L_2 are analyzed in Section 4.5.3.

Case 2: $k_1 > k_2$

The model in Eqs. (4.10) and (4.10) strictly requires X_1 and X_2 sharing a topic matrix W, which has the same topic number. It is somewhat uncommon because in practice, the topic number in documents and their user posts may be different. We, therefore, relax our NMCF in Eqs. (4.10)–(4.15) by considering the number of topic in documents and their user posts is different. Suppose k_1 is the topic number in a document d and k_2 is the topic number in its user posts, this condition leads to two cases: (i) $k_1 > k_2$ described in this section and (ii) reversely, $k_1 < k_2$ mentioned in next section.

By setting $k_1 > k_2$ we consider the topic number in a document d is larger than that in its user posts. Suppose the share matrix $W \in R^{u\times k_1}$, $H_1 \in R^{k_1\times n}$, and $H_2 \in R^{k_2\times m}$, the representation of X_1 and X_2 in Eqs. (4.10) and (4.11) can be rewritten as follows.

$$X_1 \approx WH_1 \tag{4.23}$$

$$X_2 \approx WIH_2 \tag{4.24}$$

where $I \in R^{k_1\times k_2}$ is a matrix with values on the diagonal line equal 1. The intuition of this formulation is that we consider topics in user posts are a subset of topics in sentences. Suppose $k_2 \subset k_1$, hence the topic matrix of X_2 is a set of k_2 vectors of W. Picking first k_2 vectors of W leads to the topic matrix of X_2 as WI. Values on the diagonal line of I are 1 because we would like to keep the same weights of W for sentences and user posts. The objective function is now re-defined as Eq. (4.25).

$$E = ||X_1 - WH_1||_F^2 + ||X_2 - WIH_2||_F^2 \tag{4.25}$$

followed by the gradient optimization algorithm.

$$\frac{\bigtriangledown E}{\bigtriangledown W} = (WH_1 - X_1)H_1^T + (WIH_2 - X_2)(IH_2)^T \tag{4.26}$$

$$\frac{\bigtriangledown E}{\bigtriangledown H_1} = W^T(WH_1 - X_1) \tag{4.27}$$

$$\frac{\bigtriangledown E}{\bigtriangledown H_2} = W^T(WIH_2 - X_2) \tag{4.28}$$

with update rules as the following:

$$W = W \odot \frac{X_1H_1^T + X_2(IH_2)^T}{WH_1H_1^T + WH_2(IH_2)^T} \tag{4.29}$$

$$H_1 = H_1 \odot \frac{W^TX_1}{W^TWH_1} \tag{4.30}$$

$$H_2 = H_2 \odot \frac{(WI)^T X_2}{(WI)^T (WI) H_2} \tag{4.31}$$

We apply Algorithm 2 to optimize this model. We can observe that X_2 now has an additional element I, which denotes the relationship between k_1 and k_2 while X_1 is the same with the original NMCF model. Update rules also consider I as an aspect of the optimization algorithm. This model also uses normalization by using L_1 and L_2.

Case 3: $k_1 < k_2$

In this case, our model considers the topic number in a document d is smaller than that in its user posts. By moving the matrix I from X_2 to X_1, their representation can be rewritten as the following:

$$X_1 \approx WIH_1 \tag{4.32}$$

$$X_2 \approx WH_2 \tag{4.33}$$

with a new objective function.

$$E = ||X_1 - WIH_1||_F^2 + ||X_2 - WH_2||_F^2 \tag{4.34}$$

Following definitions in Eqs. (4.26)–(4.31) we can move the matrix I from X_2 to X_1 to define new gradient optimization with new update rules in the same mechanism. The objective function in Eq. (4.34) is also optimized by using Algorithm 2.

Sentence selection

After our optimization procedure finds optimal solutions, we applied Eqs. (4.8) and (4.9) to W and H_1 to select m important sentences (having highest weights); and to W and H_2 to extract m representative comments. Our selection employs a simple greedy algorithm as previous sections. After scoring and ranking, our model loops on ranked sentences and user posts and dequeues one sentence and puts it into the summary. This process stops when the number of selected sentences (or user posts) reaches to m.

4.5 Results and Discussion

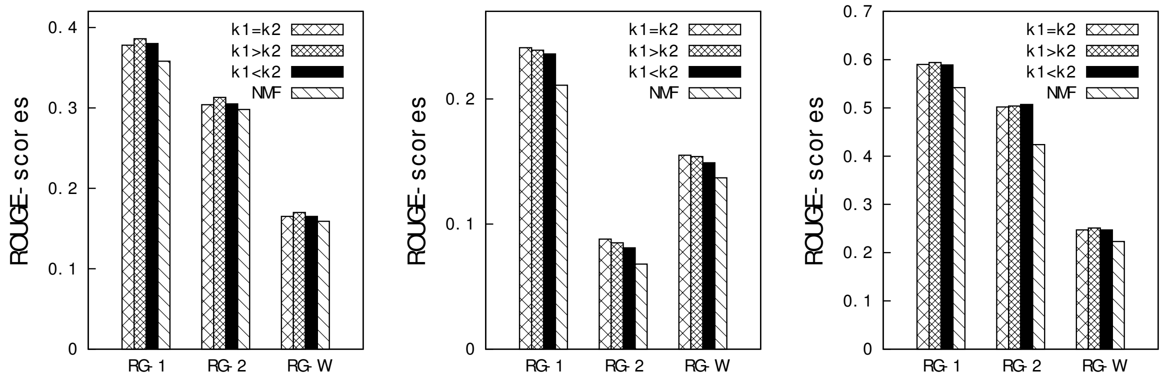
This section presents experimental results to validate the efficiency of our matrix co-factorization. It first reports ROUGE-scores of our NMCF and baselines. It next shows analyses regarding several aspects of our NMCF.

4.5.1 ROUGE-scores

This section shows our comparison in three paragraphs: NMCF vs. NMF, comparison with non-social context, and social context methods.

NMCF vs. NMF

We first compared our NMCF with the basic model based on NMF. Figures 4.2 and 4.3 summarizes ROUGE-scores of our comparison. It is clear from these figures that our NMCF achieves sufficient improvements over NMF in most all cases (p-values ≤ 0.05). For example, for sentence extraction on VSoLSCSum in Figure 4.2c, the performance of our model is 5% higher than NMF of ROUGE-1 and 7% of ROUGE-2. This trend is consistent with comment extraction in Figure 4.3, in which there are big margins between our methods and NMF. This confirms the efficiency of our model in exploiting mutual information between user posts and sentences. In Figure 4.2a, NMCF slightly outperforms NMF, e.g. 0.378 vs. 0.358 of ROUGE-1. This is because NMF also utilizes the advantage of matrix factorization, which has shown to be efficient for document summarization (Gong and Liu, 2001; Lee et al., 2009; Park et al., 2006; Wang et al., 2008). Also, sentences themselves contain important information for summarization; hence, adding more data from user posts slightly improves ROUGE-scores. However, for comment extraction, the support from sentences boosts ROUGE-scores of our model with large margins.



(a) SoLSCSum (b) USAToday-CNN (c) VSoLSCSum

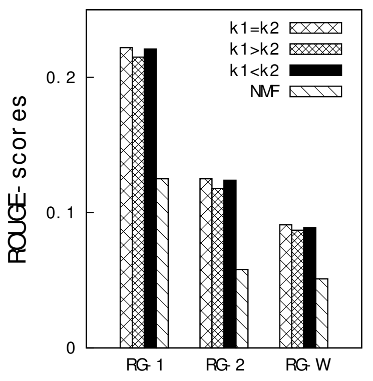
Figure 4.2: NMCF vs. NMF for sentence selection.

We conducted statistical tests (pairwise $t-$test) to confirm the significance of these improvements. p-values in Table 4.2 statistically validate the efficiency of our model, in which it is significantly better than NMF in almost all cases. For sentence extraction on SoLSCSum, NMF is competitive with our model in ROUGE-1 with $p-$value $= 0.2868$.

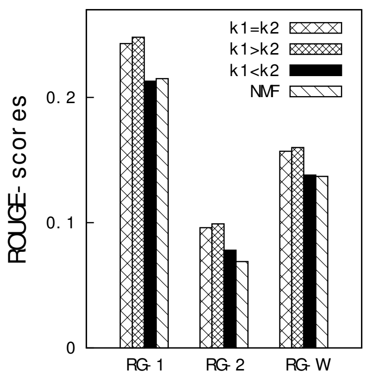
Comparison with non-social context methods

We also compared our NMCF with non-social context methods described in Section 3.1.4. Table 4.3 reports their ROUGE-scores.

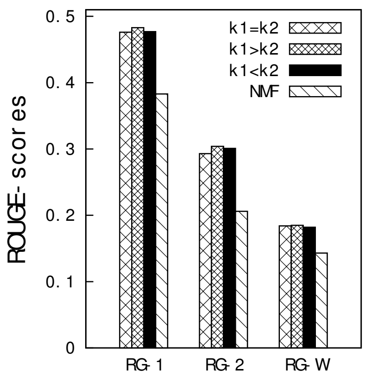
Table 4.3 indicates that our NMCF surpasses strong methods in almost all cases. For example, it acquires very competitive ROUGE-scores on VSoLSCSum and it is the best in extracting user posts on SoLSCSum and USAToday-CNN. There are big margins between our methods and the second best method, e.g. 0.590 vs. 0.506 (significant improvements



(a) SoLSCSum



(b) USAToday-CNN



(c) VLSCSum

Figure 4.3: NMCF vs. NMF for user post extraction.

Table 4.2: The pairwise t-test of NMCF and NMF. **Bold** is significant with $p \leq 0.05$.

Data	Method	Sentence			User posts		
		RG-1	RG-2	RG-W	RG-1	RG-2	RG-W
SoLSCSum	$k_1 = k_2$	0.2868	0.3277	0.6711	**0.0000**	**0.0000**	**0.0000**
	$k_1 > k_2$	0.1374	0.1018	0.4011	**0.0000**	**0.0000**	**0.0000**
	$k_1 < k_2$	0.2806	0.3362	0.6615	**0.0000**	**0.0000**	**0.0000**
USAToday-CNN	$k_1 = k_2$	0.1361	0.1609	0.2013	**0.0143**	**0.0140**	**0.0037**
	$k_1 > k_2$	0.1607	0.2499	0.1939	**0.0057**	**0.0092**	**0.0017**
	$k_1 < k_2$	0.2115	0.3857	0.3678	—	0.2617	0.8512
VSoLSCSum	$k_1 = k_2$	**0.0384**	**0.0008**	**0.0000**	**0.0006**	**0.0000**	**0.0296**
	$k_1 > k_2$	**0.0507**	**0.0017**	**0.0000**	**0.0002**	**0.0000**	**0.0223**
	$k_1 < k_2$	0.0631	**0.0004**	**0.0001**	**0.0003**	**0.0000**	**0.0333**

are denoted by † with $p-$values ≤ 0.05). This trend is consistent in other cases. This is because our methods use the support of social context and the advantage of matrix co-factorization. For the first aspect, user posts help to enhance the information of sentences in document representation. A bigger dictionary created from sentences and user posts enriches the representation of term-sentence matrices, which improves the quality of the scoring step. For the second aspect, scoring with matrix co-factorization takes advantage of share topic matrices to effectively rank sentences and comments. For sentence selection, our method is competitive on SoSLCSum and USAToday-CNN. On SoLSCSum, CRF with sophisticated features is a strong baseline. On USAToday-CNN, SVM and Lead-m obtain the best results. It is because SVM is also a supervised method, which uses several suitable features. Lead-m outperforms many systems participated in DUC (Nenkova, 2005). However, in other cases, our NMCF surpasses these methods.

Table 4.3: NMCF vs. basic methods; **bold** is the best value; *italic* is the second best. RG stands for ROUGE.

Dataset	Method	Sentences			User posts		
		RG-1	RG-2	RG-W	RG-1	RG-2	RG-W
SoLSCSum	Lead-m	0.345	*0.322*	*0.170*	—	—	—
	LexRank	$0.327^\dagger$	$0.243^\dagger$	$0.138^\dagger$	0.210	0.115	0.085
	SVM*	$0.325^\dagger$	$0.263^\dagger$	0.147	$0.152^\dagger$	$0.089^\dagger$	$0.062^\dagger$
	CRF*	**0.393**	**0.379**	**0.187**	$0.091^\dagger$	$0.075^\dagger$	$0.037^\dagger$
	NMCF ($k_1 = k_2$)	0.378	0.304	0.165	**0.222**	**0.125**	**0.091**
	NMCF ($k_1 > k_2$)	*0.386*	0.313	*0.170*	0.215	0.118	0.087
	NMCF ($k_1 < k_2$)	0.380	0.305	0.165	*0.221*	*0.124*	*0.089*
USAToday-CNN	Lead-m	0.249	**0.106**	**0.172**	—	—	—
	LexRank	*0.251*	*0.092*	0.163	$0.193^\dagger$	$0.068^\dagger$	$0.128^\dagger$
	SVM*	**0.261**	**0.106**	*0.171*	0.221	0.084	0.149
	CRF*	$0.186^\dagger$	0.088	$0.114^\dagger$	$0.190^\dagger$	$0.065^\dagger$	$0.119^\dagger$
	NMCF ($k_1 = k_2$)	0.241	0.088	0.155	*0.243*	**0.096**	*0.157*
	NMCF ($k_1 > k_2$)	0.239	0.085	0.154	**0.248**	**0.099**	**0.160**
	NMCF ($k_1 < k2$)	0.236	0.081	0.149	0.213	0.078	0.138
VLSCSum	Lead-m	$0.495^\dagger$	$0.420^\dagger$	$0.214^\dagger$	—	—	—
	LexRank	$0.506^\dagger$	$0.432^\dagger$	$0.219^\dagger$	$0.348^\dagger$	$0.198^\dagger$	$0.127^\dagger$
	SVM*	$0.497^\dagger$	$0.440^\dagger$	$0.208^\dagger$	$0.374^\dagger$	$0.212^\dagger$	$0.140^\dagger$
	CRF*	$0.422^\dagger$	$0.357^\dagger$	$0.172^\dagger$	$0.111^\dagger$	$0.062^\dagger$	$0.041^\dagger$
	NMCF ($k_1 = k_2$)	*0.590*	0.502	*0.247*	0.476	0.293	*0.184*
	NMCF ($k_1 > k_2$)	**0.594**	*0.504*	**0.251**	**0.483**	**0.304**	**0.185**
	NMCF ($k_1 < k_2$)	0.589	**0.507**	*0.247*	**0.477**	*0.301*	0.182

Comparison with social context methods

We challenged our model with social context methods stated in Section 3.1.4. Table 4.4 shows their comparison. Again, ROUGE-scores from this table validate the efficiency of our model, in which it is the best in almost all cases. For comment extraction on SoLSC-Sum, our methods are the best following by HGRW because it exploits user posts to score sentences by using a random walk ranking algorithm (Wei and Gao, 2015). However, our methods are still better than HGRW. RankBoost CCF obtains competitive results because it uses many sophisticated domain-dependent features to model sentence-comment relationships. Note that, our model is unsupervised, which is domain-independence. ROUGE-scores of cc-TAM are quite poor because it is designed for multi-document summarization (Gao et al., 2012) while all the datasets are for single-document summarization.

ROUGE-scores on DUC 2004

We confirmed NMCF's efficiency on DUC 2004. Our objective is to show the adaptation of our model on a standard dataset rather than obtaining the best results on this dataset,

Table 4.4: ROUGE-scores of our model and advanced methods; **bold** is the best value; *italic* is the second best. RG stands for ROUGE..

Dataset	**Method**	**Sentences**			**User posts**		
		RG-1	RG-2	RG-W	RG-1	RG-2	RG-W
SoLSCSum	cc-TAM	$0.306^{\dagger}$	$0.238^{\dagger}$	$0.136^{\dagger}$	$0.054^{\dagger}$	$0.022^{\dagger}$	$0.024^{\dagger}$
	HGRW	0.379	0.204	*0.167*	0.209	0.115	0.084
	RB* CCF	0.360	0.283	0.158	$0.190^{\dagger}$	0.098	$0.077^{\dagger}$
	NMCF ($k_1 = k_2$)	0.378	0.304	0.165	**0.222**	**0.125**	**0.091**
	NMCF ($k_1 > k_2$)	**0.386**	**0.313**	**0.170**	0.215	0.118	0.087
	NMCF ($K_1 < k_2$)	*0.380*	*0.305*	0.165	*0.221*	*0.124*	*0.089*
USAToday-CNN	cc-TAM	0.229	0.077	0.145	**0.249**	0.089	0.152
	HGRW	**0.279**	**0.098**	**0.177**	0.242	0.088	*0.157*
	RB* (CCF)	0.221	0.070	0.140	0.233	0.091	0.132
	NMCF ($k_1 = k_2$)	*0.241*	*0.088*	*0.155*	0.243	*0.096*	*0.157*
	NMCF ($k_1 > k2$)	0.239	*0.085*	*0.154*	*0.248*	**0.099**	**0.160**
	NMCF ($k_1 < k2$)	0.236	0.081	0.149	0.213	0.078	0.138
VLSCSum	cc-TAM	$0.488^{\dagger}$	$0.377^{\dagger}$	$0.201^{\dagger}$	$0.301^{\dagger}$	$0.167^{\dagger}$	$0.111^{\dagger}$
	HGRW	*0.570*	0.479	$0.233^{\dagger}$	0.454	0.298	0.173
	RB* CCF	0.561	0.494	$0.235^{\dagger}$	0.471	**0.308**	0.168
	NMCF ($k_1 = k_2$)	*0.590*	0.502	*0.247*	0.476	0.293	*0.184*
	NMCF ($k_1 > k_2$)	**0.594**	*0.504*	**0.251**	**0.483**	*0.304*	**0.185**
	NMCF ($k_1 < k_2$)	0.589	**0.507**	*0.247*	*0.477*	0.301	0.182

which lacks user posts. Also, we would like to observe margins between NMCF and strong methods. Since there are tiny gaps among our methods; we, hence, only show results of NMCF with $k_1 = k_2$.

DUC 2004 contains 50 topics, in which each topic has 10 articles and four references written by humans. Since it has no social information, we adapted it to our model by using a one-versus-all setting. We kept one article as a primary document and formed nine remaining ones as relevant information. We applied NMCF to each primary document to select two sentences, resulting 20 sentences in total. To select summaries, we employ a simple greedy algorithm. Sentences fewer than five words were first removed because they are fairly short for summarization (Erkan and Radev, 2004) and the rest was sorted in decreasing order based on their Cosine scores.

$$score(s_i) = \frac{1}{|s_i|} \sum_{j=1}^{m} cos(s_i, s_j) \tag{4.35}$$

where m is all other sentences in a topic. We repeatedly dequeued one sentence from the sorted list and append it to form a summary if it is non-redundant with a Cosine threshold = 0.75. The iteration stops if the summary reaches a length constraint.

We report three basic methods: PROB, LLR, MRW presented in Hong and Nenkova (2014) and three advanced methods: (i) MD-ILP, an abstractive ILP-based summarizer

(Banerjee et al., 2015); (ii) REGSUM, a regression-based model with hand-crated features (Hong and Nenkova, 2014); and (iii) CRSum, a deep learning model based on sentence context (Ren et al., 2017) with ROUGE-1, 2, and 4 recall. Due to the setting of DUC, the evaluation uses a length constraint with a parameter `‘‘b 665’’` (665 bytes).

Table 4.5: ROUGE-scores on DUC 2004.

Method	ROUGE-1	ROUGE-2	ROUGE-4
PROB†	0.3514	0.0817	0.0106
LLR†	0.3460	0.0756	0.0083
MRW†	0.3578	0.0815	0.0099
MD-ILP†	—	**0.1199**	—
REGSUM†	*0.3857*	0.0975	**0.0160**
CRSum†	**0.3953**	*0.1060*	—
NMF	0.3557	0.0762	0.0107
NMCF	0.3734	0.0846	*0.0132*

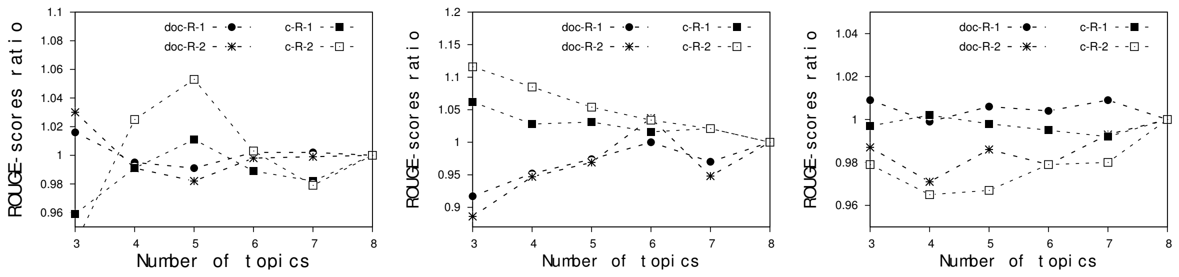
ROUGE-scores in Table 4.5 indicate that our method outputs competitive results on DUC 2004. It outperforms three basic models: PROB, LLR, MRW. The ROUGE-scores of NMF are also promising, but our method is still better than NMF. However, there are rather gaps between NMCF and state-of-the-art methods. For example, CRSum is the best of ROUGE-1 and REGSUM is the best of ROUGE-4. This is understandable that they are supervised learning. CRSum exploits the context surrounding a sentence in word and sentence levels by using Bi-CNN (Cao et al., 2015c) and a version of LSTM (Graves et al., 2013), respectively. The final vector representation of a sentence is concatenated with several surface features such as its position. REGSUM uses sophisticated hand-crafted features to estimate the importance of words, which can be used to measure the importance of sentences. By contrast, our model is unsupervised learning developed for social context summarization but not for multiple-document summarization. In another case, MD-ILP is the best of ROUGE-2 because it exploits informativeness and linguistic aspects with a set of constraints. However, adapting these methods for our task is still an open question. On the other hand, our method can be flexibly adapted to domains which include user posts (Figures 4.2, 4.3, and Tables 4.3, 4.4) or do not include user posts as DUC 2004 in Table 4.5 with very competitive results.

4.5.2 Topic analysis

We analyzed the influence of topic numbers in our model corresponding to three settings in the following paragraphs.

Case 1: the same number of topics

We consider the sensitivity analysis of k in Eqs. (4.10) and (4.11) in by tuning topic number k in [3, 8] with a jumping step of 1. The number of topics outside this rage is too small or large. More precisely, we normalized Figure 4.4 by dividing ROUGE-scores in each tuning point for those at $k = 8$.



(a) SoLSCSum (b) USAToday (c) VSoLSCSum

Figure 4.4: ROUGE-scores with various k.

Figure 4.4 shows that the topic number k affects our model. In Figure 4.4a, increasing k reduces ROUGE-scores. Our model reaches a peak at $k = 5$; then its performance slightly decreases at $k = 8$. This trend is quite similar to Figure 4.4b in which our model obtain better results with smaller topic numbers. By contrast, in Figure 4.4c, the trend is reversed, in which increasing k improves ROUGE-scores. The trend in Figure 4.4b is inconsistent, in that increasing k raises ROUGE-scores of sentence selection, but the performance of tweet extraction decreases. The general trend from these figures indicates that our model operates well at $k = 5, 6$. Gaps between lower and upper bounds are insignificant (around 0.05) showing that changing k slightly changes ROUGE-scores.

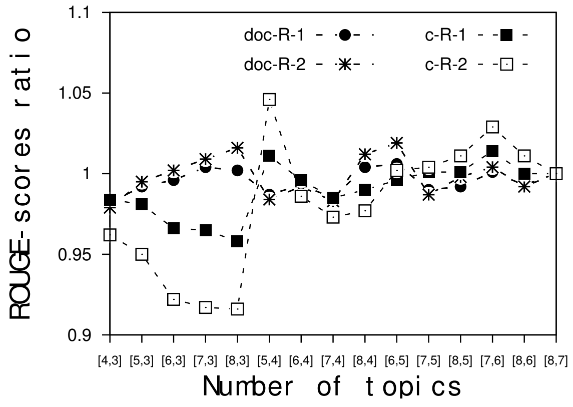
Case 2: $k_1 > k_2$

This setting considers the number of topics in document is larger than that in their user posts. To investigate the influence of k_1 and k_2, we tuned k_1 in [4, 8] and k_2 in [3, 7] so that $k_1 > k2$. In each pair of tuning point, we observed its ROUGE-scores and plotted them on Figure 4.5.

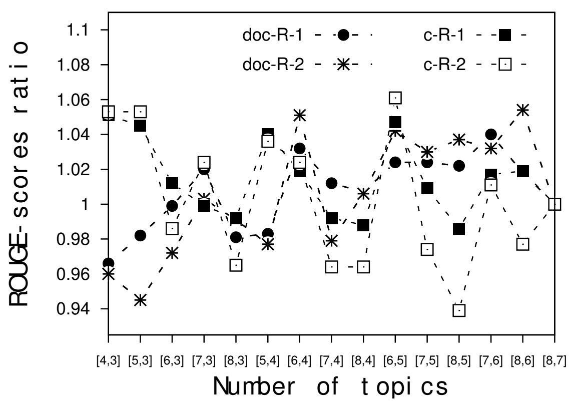
From Figure 4.5 we see that all lines are fluctuant. The trend in Figure 4.5a seems to increase while it tends to decrease in Figure 4.5c. In Figure 4.5b, it is hard to conclude the general direction. However, our model obtains the best results with $k_1 = 6$ and $k_2 = 5$. For some cases in which the number of topics is very different, e.g. [8, 3], results are not as good as pairs which have similar topics such as [6, 5]. A possible reason is that our model divides documents into too specific topics while it splits user posts into too general ones. As a result, it is biased when computing weights in H_1 and H_2. This observation also explains that the previous model with the same topic number ($k = 5, 6$) can achieve competitive ROUGE-scores in several cases.

Case 3: $k_1 < k_2$

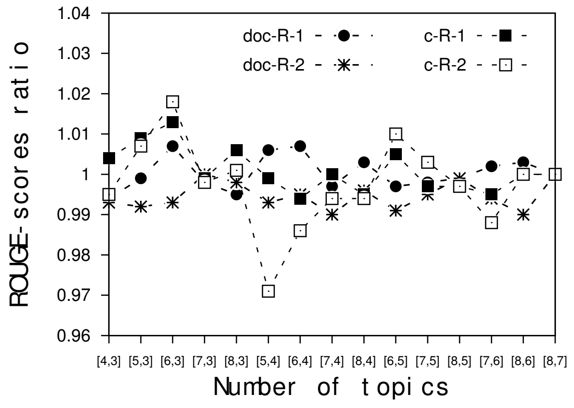
We also examined the influence of topic number in the setting of $k_1 < k_2$. We used the same tuning processing in the second case but changed the constraint which requires that $k_1 < k_2$. We visualize ROUGE-scores after normalizing on Figure 4.6.



(a) SoLSCSum



(b) USAToday



(c) VSoLSCSum

Figure 4.5: ROUGE-scores with various k pairs with $k_1 > k_2$.

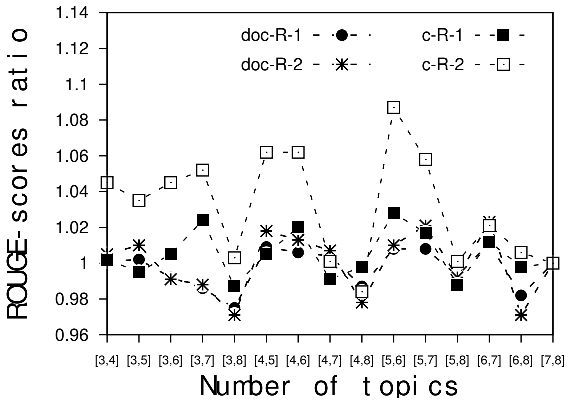
The observation is similar to the second case in that our model outputs better results with topic pairs which are close in term of number. Therefore, to balance ROUGE metrics, we suggest that pairs created by combining values in [4, 5, 6] can be used.

4.5.3 Normalization observation

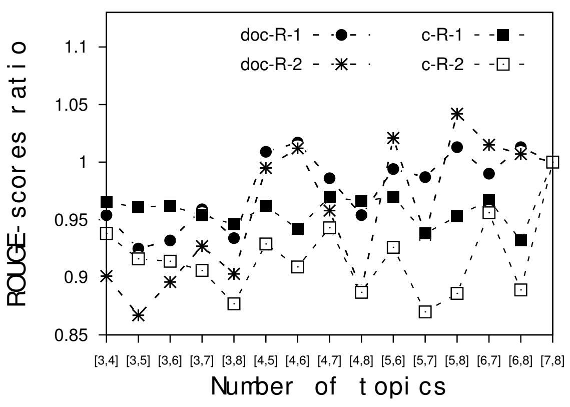
As mentioned, our model considers normalization L_1 or L_2 to avoid over-fitting during optimization. Due to tiny margins among our methods, we only report ROUGE-scores of NCMF with the same topic number in Table 4.6. From this table we can observe

Table 4.6: Norm observation. RG means ROUGE-scores.

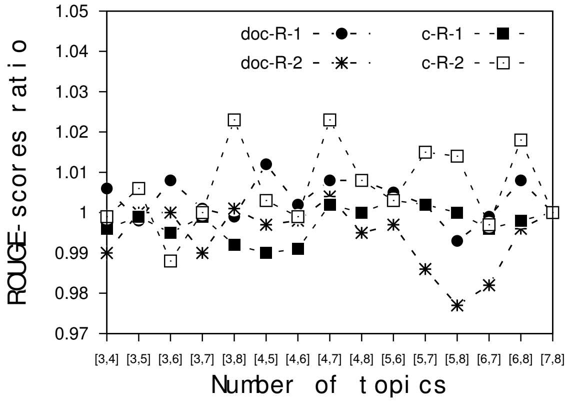
Norm	SoLSCSum				USAToday-CNN				VSoLSCSum			
	Sentence		Comment		Sentence		Tweet		Sentence		Comment	
	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2	RG-2	RG-2	RG-1	RG-2	RG-2	RG-2
L_1	0.378	0.304	**0.222**	**0.125**	**0.241**	**0.088**	**0.243**	**0.096**	0.590	0.502	0.476	0.293
L_2	**0.381**	**0.307**	0.214	0.115	0.212	0.069	0.237	0.092	0.590	**0.505**	**0.483**	**0.306**



(a) SoLSCSum



(b) USAToday



(c) VSoLSCSum

Figure 4.6: ROUGE-scores with various k pairs.

that normalization with L_2 outputs better summarization results than L_1 on SoLSCSum and VSoLSCSum. For example, ROUGE-scores of L_2 are higher than those of L_1 on VSoLSCSum. For comment extraction on SoLSCSum, L_1 is better. On USAToday-CNN, L_1 consistently surpasses L_2. To balance performance on three datasets, we selected L_1 as the normalization.

4.5.4 Error rate in optimization

We observed the trend of error reduction in our optimization algorithm. Figure 4.7 shows that the error is asymptotic to 0 when increasing the number of iterations. It significantly reduces after 200 iterations. After that, it slightly falls and is nearly 0 after 1000 iterations. It is understandable that the optimization algorithm finds a better optimal solution with a large number of iterations, but it takes a long time of coverage. Based on this observation, we fixed the iterations of the algorithm within 1000.

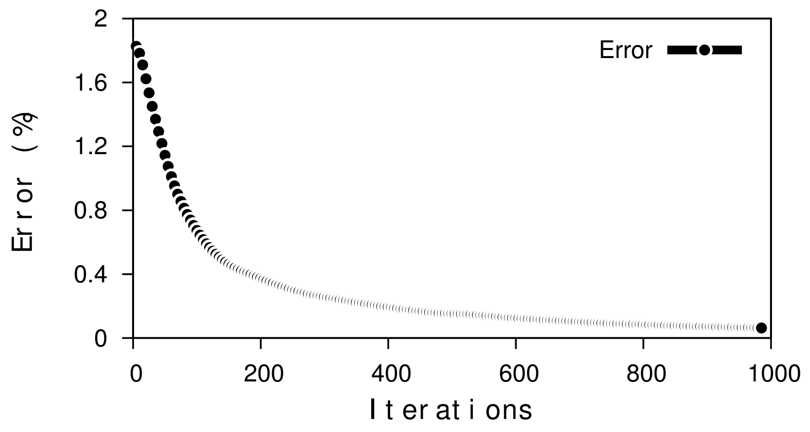


Figure 4.7: The convergence of optimization algorithm.

4.5.5 Output analysis

We examined extracted sentences and comments of our model and NMF on SoLSCSum. From Table 5.17, we can see that our model extracts one correct sentence and comment. The sentence locates in the second position. It contains much important information of the event *"Germanwings crash families could seek damages in the U.S.: lawyer"* such as the type of event (*"Germanwings crash"*), the situation (deal with the company for the damage). In many cases, reading S1 can provide enough salient information about this event. The comment C1 also reflects the content of this event. By reading C1, we can partly understand the event, in which the relatives of victims try to claim with the company for their money. Interestingly, C1 includes reader's opinions regarding victims. However, topics mentioned in comments are usually diverse, then they bring the noise, such as C2. Our model also extracts incorrect sentences (S2 and C2). While S2 is relevant to the event, it is hard to conclude C2 belongs to this event. However, they include many words, which appear in summary sentences, such as *"airline"*, *"US"*, or *"claim"*. Also, as our expectation, extracted sentences share many common words because we present the nature of sentences and user posts in a share hidden topic-matrix. In this case, many common words help to improve the representation of a document.

NMF shares one correct sentence (S1) with our model. It is understandable that NMF is also competitive in summarizing documents. It selects the second one, which is relevant to the event, but not important at this moment because the event passed. This explains that our model outperforms NMF in sentence selection in Section 4.5.1. For comment extraction, NMF extracts incorrect comments (two of those are shown in Table 5.17). This explains the reason that our model significantly surpasses NMF in Figure 4.3a. Extracted sentences and comments share few common words because it does not exploits the share of common topics in the summarization process.

Table 4.7: A summary example generated from the document 31st on SoLSCSum; two summary sentences and comments are shown instead of six. Sentences with [+] means that they are also in the references; [-] means that they are not in the references.

NMCF Summarization
Sentence selection
[+]S1: Families of the victims of the Germanwings crash are considering filing a claim for damages in the United States if they cannot reach agreement with parent airline Lufthansa in Germany, a lawyer representing the families said on Sunday.
[-]S2: "If the airline is not prepared to do so, however, we will look seriously at making a claim in the United States," said Giemulla, adding that he was representing 21 families including those of the German school children who died.
Comment extraction
[+]C1: I'm sad for the people who are no longer on their life of this Germanwings strategy, but I don't know how much money relatives want for the flesh of who die on Germanwings case.
[-]C2: The only ones who should be allowed to take this to a US court are the US citizens.
NMF Summarization
Sentence selection
[+]S1: Families of the victims of the Germanwings crash are considering filing a claim for damages in the United States if they cannot reach agreement with parent airline Lufthansa in Germany, a lawyer representing the families said on Sunday.
[-]S2: Nearly half of the victims of the Germanwings Barcelona to Duesseldorf flight were German, with the remaining passengers hailing from a range of countries, including Spain, Australia and Argentina.
Comment extraction
[-]C1: Julie to Ann (who is holding a cabbage): "I would have taken a ride with anyone except Peter Kramer, Freddie—maybe Joe.
[-]C2: Don't understand why a US court would have authority over European citizens.

4.6 Conclusion

In this chapter, we show our second effort to investigate the integration of social context into the summarization process. We introduce our matrix co-factorization model, which considers common topics between sentences and user posts. The insight behind our model is that it formulates common topics, which include common words or phrases generated from the word variation behavior of readers. Common topics are exploits in two levels: document representation and our matrix co-factorization algorithm. We confirm the efficiency of our model on three datasets in two languages, English and Vietnamese. Experimental results show that our model obtains very competitive ROUGE-scores over strong methods. We also analyze several aspects of our model to provide the better understanding of its operations. We point out that that a joint optimization algorithm outputs better ROUGE-scores than an individual one.

Chapter 5

Learning to Summarize by Utilizing Social Context

This chapter investigates a different approach to incorporate the social context of a document into the estimation. Instead of using unsupervised learning, we develop two supervised ranking models. The key idea of these models is that we integrate human knowledge denoted in the form of features to improve the estimation. To do that, we define informative indicators for measuring the importance of a sentence (or a user post). Our indicators are extracted from three channels: local information, user-generated information, and third-party sources. They are learned by a learning to rank (L2R) algorithm to do the estimation. For extraction, besides using a greedy method, we also introduce our voting algorithm, which combines outputs of L2R models for extracting summaries.

We also report experimental results of each model with discussion and analyses. ROUGE-scores indicate that our models achieve competitive results compared to strong methods over three datasets, in English and Vietnamese. We organize this chapter in two sections. Section 5.1 bases on our work in Nguyen et al. (2016d, 2017c), which presents a L2R model for exploiting information from user posts. Section 5.2 is an extended version of the first one by considering both user posts and relevant news articles to enrich the representation of sentences and user posts. We published our work in Nguyen et al. (2018b, 2017b).

5.1 Learning to Rank Sentences with User Posts

Supervised learning methods formulate extraction as binary classification, which may not model the summarization process naturally due to the reading behavior of humans. In a common way, they create a summary by reading all sentences in a document, estimate informative information in each sentence, rank sentences based on their estimation, and then select top m sentences as a summary. Based on this observation, we define our summarization in the form of L2R, in which summaries are extracted by ranking sentences based on their informative information. Figure 5.1 presents our model.

When modeling a sentence, our model exploits a set of social features from user posts via a supporting channel to support local ones, e.g. the red dot line from user posts.

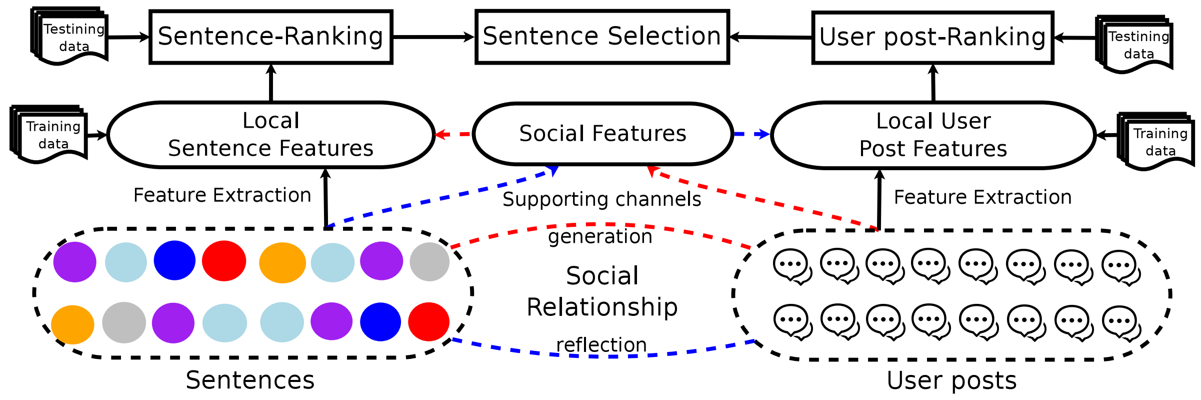


Figure 5.1: The overview of our L2R-based model, in which a document contains a set of sentences and user posts. A sentence is presented by a set of words denoted by circles; dot lines connect sentences, user posts to social features show mutual support.

Similarly, a set of social features from sentences is also used to enrich local features in modeling a user post. In this view, local and social information mutually support together in a reinforcement fashion. After modeling, two L2R-based models are separately trained for sentences and user posts. Finally, they select top m ranked sentences and user posts as a summary. In next sections, we first describe a basic model which uses RankBoost with basic features in Section 5.1.1. We next introduce our model with new features in Section 5.1.2. We final report the comparison of our model with baselines in Section 5.1.4.

5.1.1 Basic model

We start with a basic model shown in Wei and Gao (2014), who presented a summarization method, which integrates the support of tweets to improve the estimation of sentences with a set of local and cross features. For example, when modeling a sentence, cross features were extracted to support local ones which cover several aspects of a single sentence (or tweet), e.g. sentence position, importance based on uni-gram hybrid TF-IDF, etc. Cross features exploit mutual support from tweets when modeling a sentence, e.g. a maximal Cosine score between a sentence in a main document with its relevant tweets, etc. The detail of features is shown in Wei and Gao (2014). To train the model, they employed RankBoost (Freund et al., 2003).

Freund et al. (2003) introduced RankBoost, which bases on AdaBoost (Freund and Schapire, 1995) to perform pairwise classification on documents. These methods are only different in term of distribution, in which RankBoost considers the distribution on documents pairs while AdaBoost defines the distribution on individual documents. Here we summarize the learning algorithm of RankBoost, which is derived from Liu (2011).

In this algorithm, D_t is the distribution on documents pairs, f_t is a weak ranker selected at the i^{th} iteration, α_t is the weight for linearly combining weak rankers, and $(x_u^{(i)}, x_v^{(i)})$ is a document pairs in training data. The learning algorithm minimizes an exponential loss

Algorithm 3: The learning algorithm of RankBoost.

```
Data: training data in terms of document pairs.
Result: initial distribution $D_1$ on input document pairs.
1 for ($t = 1, ...., T$) do
2     Train a weak ranker $f_t$ based on distribution $D_t$ ;
3     Choose $\alpha_t$ ;
4     Update $D_{t+1}(x_u^{(i)}, x_v^{(i)}) = \frac{1}{Z_t} D_t(x_u^{(i)}, x_v^{(i)}) \exp(\alpha_t(f_t(x_u^{(i)}) - f_t(x_u^{(i)})))$ ;
5     where $Z_t = \sum_{i=1}^{n} \sum_{u,v:y_{u,v}^{(i)}=1} D_t(x_u^{(i)}, x_v^{(i)}) \exp(\alpha_t(f_t(x_u^{(i)}) - f_t(x_u^{(i)})))$ ;
6 end
7 Output: $f(x) = \sum_t \alpha_t f_t(x)$ ;
```

function.

$$L(f; x_u, x_v, y_{u,v}) = \exp(-y_{u,v}(f(x_u) - f(x_v))) \tag{5.1}$$

Algorithm 3 shows that RankBoost learns the optimal weak rank f_t and its coefficient α_t based on the current distribution of documents pairs in D_t. Freund and Schapire (1995) indicated that there are three ways to select α_t. Please refer this paper for more detail.

We argue that the performance of the basic model can be still improved. To do that, we extended it in two aspects: (i) proposing new features, which can capture more characteristics of a sentence and user post and (ii) employing a different learning algorithm. For the first aspect, it is understandable that adding more sophisticated features improves the importance estimation, which benefits to yield correct predictions on testing data. For example, features in Wei and Gao (2014) do not consider the sequence aspect, which plays an important role in summarization (Shen et al., 2007). For the second aspect, we employ Ranking SVM instead of using RankBoost because Ranking SVM bases on SVM, which has shown to be efficient for classification.

5.1.2 Our model with new features

We first utilize a different learning algorithm. We expect by doing this, our model achieves better results than using RankBoost. To train our L2R model, we used Ranking SVM Joachims (2006) because it has shown as one state-of-the-art L2R methods for ranking (Cao et al., 2007; Nguyen et al., 2016b). Ranking SVM[1] applies the characteristics of SVM Cortes and Vapnik (1995) to perform pairwise classification. In information retrieval, given n training queries $\{q_i\}_{i=1}^{n}$, their associated document pairs $(x_u^{(i)}, x_v^{(i)})$, and

[1]https://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html

the corresponding ground truth label $y^{(i)}_{(u,v)}$, Ranking SVM optimizes an objective function:

$$\min \frac{1}{2} \|w\|^2 + \lambda \sum_{i=1}^{n} \sum_{u,v:y^{(i)}_{u,v}} \xi^{(i)}_{u,v} \tag{5.2}$$

$$\text{s.t. } w^T(x^i_u - x^{(i)}_v) \geqslant 1 - \xi^{(i)}_{u,v}, \text{ if } y^{(i)}_{u,v} = 1 \tag{5.3}$$

$$\xi^{(i)}_{u,v} \geqslant 0,\ i = 1, ..., n \tag{5.4}$$

where $f(x) = w^T x$ is a linear scoring function, (x_u, x_v) is a pairwise and $\xi^{(i)}_{u,v}$ is the loss. For summarization, the document pair-wise is sentence-sentence or comment-comment (tweet-tweet) and our model it learns to optimize Eq. (5.2) to reach gold-standard references. In this thesis, for SoLSCSum and VSoLSCSum datasets, the pair-wise order is the label of each sentence or comment. For USAToday-CNN, the pair-wise order is a salient score between a sentence or tweet with ground-truth references suggested by Wei and Gao (2014). Given $H = \{h_1, h_2, ..., h_k\}$ is a set of highlights, Eq. (5.5) computes the salient score of a sentence or tweet.

$$score(s_i) = max\{score(s_i, h_j)\},\ j \in \{1, k\} \tag{5.5}$$

where $score()$ returns the ROUGE-1 F-score between s_i and h_j.

New local features

We improve the performance of our model by investigating new local features. Our features try to cover the *representation* hypothesis, which basic features of Wei and Gao (2014) may not completely consider. They cover four important aspects: length, sequence, topic covering and meaningless words.

Sentence length This feature bases on a hypothesis that a summary sentence usually contains more important information than non-summary ones. This feature counts the number of words in a sentence s_i.[2]

Sentence length before Next four features cover the sequence aspect of a document. In writing, writers arrange sentences in an appropriate order to create a story. A summary candidate is followed by several supporting sentences, which enrich its meaning. Given a sentence s_i, we consider a previous and a next sentence to capture the sequence aspect because further sentences are references or do not directly support the meaning of s_i. The sentence length before is the number of words in sentence s_{i-1} giving s_i. The value of this feature is 0 if s_i is the first sentence in a document.

Sentence length after is the number of words in s_{i+1} giving s_i. The value of this feature is 0 if s_i is the last sentence in a document.

[2]When extracting features, all stopwords were removed.

Cosine similarity before shares the sequence aspect with sentence length before and after features, but we use Cosine similarity. Given sentences s_{i-1} and s_i denoted by vectors $\overrightarrow{x}$ and $\overrightarrow{y}$ using the bag-of-words model, Eq. (5.6) defines their Cosine similarity.

$$cos(\overrightarrow{x}, \overrightarrow{y}) = \frac{\overrightarrow{x}.\overrightarrow{y}}{\| \overrightarrow{x} \| . \| \overrightarrow{y} \|} \tag{5.6}$$

where x_i and y_i are the frequency of each word in s_i and s_{i-1}; $\overrightarrow{x}$ and $\overrightarrow{y}$ are two same size vectors. The Cosine value of s_i is 0 if it is the first sentence in a document.

Cosine similarity after calculates the similarity of s_i and s_{i+1} using Eq. (5.6). The value is 0 if s_i is the last sentence of a document.

By using these features, our model deals with the sequence aspect of a document, in which the importance of a sentence is measured based on its surrounding neighbors.

Local LDA score This feature bases on the *common topic* hypothesis. It states that a summary sentence should include topics presented by topical words. It is possible to use TF-IDF as topical words; however, they do not take into account the topical aspect denoted in the form of word distribution. Therefore, we again adopted LDA (Blei et al., 2003) to generate these words in two steps: training and inference.

- **Training**: The goal of training is to obtain word-topic-weight and document-topic distribution matrices used by inference. Its detail is shown in Section 3.2.1.

- **Inference**: We used the inference in Section 3.2.1. Given a small single document d_s (or d_c), we first chose top t closet topics of d_s. The closet topics are those that have the highest values in the document-topic distribution matrix. With each topic, we selected top w words, which have the highest weights in the word-topic-weight matrix. As a result, the number of topical words for each small document is $(|t| \times |w| = 5 \times 5 = 25)$.

Given a set of topical words of d_s named $T_d = \{w_1, ..., w_k\}$, Eq. (5.7) computes a local LDA score:

$$\textit{local-lda-score}(s_i) = \frac{\sum_{j=1}^{k} weight(w_j)}{n} \text{ if } w_j \in s_i \tag{5.7}$$

where $weight()$ returns the word weight of w_j, e.g. 0.45 in s_i (normalized in [0, 1]); n is the number of words in s_i.

Stop words The hypothesis of this feature is that a summary sentence should contain content words, which include important information, e.g. person name. Let len_{org} to be the length of an original sentence and len_{rmw} is the length of the sentence after removing stop words, Eq. (5.8) counts stop word number.

$$\textit{stop-word-count}(s_i) = len_{org}(s_i) - len_{rmw}(s_i) \tag{5.8}$$

New social features

Local features only capture the internal aspect of a sentence, but they ignore the support from user-generated content, which provides additional information from readers. To improve the quality of our model, we introduce social features, which cover three aspects: semantic similarity, topical covering, and lexical similarity between a sentence and auxiliary comments or tweets.

Semantic-based similarity This feature bases on the *generation* hypothesis, in which readers tend to use salient words in sentences to create their comments or tweets in a variation form, which is already discussed in Section 3.2. Here we re-show an example in Table 5.1 derived from Section 3.2.

Table 5.1: An example of the word variation between a sentence and comment.

The 26-year-old man, identified as Usaamah Rahim, brandished a **knife** and advanced on **officers** working with the Joint Terrorism Task Force who initially tried to retreat before opening **fire**, Boston **Police** Superintendent William Evans told reporters.
If I had been one of the **police officers** I would have whispered 3 times "**drop the knife**" then quickly **fired** several **shots** at his sternum.

In this example, we can observe that the comment and sentence share several common words (bold words), which are directly extracted from the sentence, e.g. *"knife"*, *"officers"*, or are derived in a variation, e.g. *"fired"*, *"shots"*. From this observation, we present sentence-user-post pairs by using a semantic similarity.

To exploit the semantic aspect, we utilized Word2Vec (Mikolov et al., 2013). It takes a large dataset as an input and produces the vectors of words in the vocabulary. Its training for English and Vietnamese is shown in Section 3.1.1. Given a Word2Vec model, Eq. (5.9) calculates the semantic similarity of a sentence and auxiliary comments (or tweets):

$$\textit{w2v-score}(s_i, UG_d) = \max_{j=1}^{m} \left(sentSim(s_i, c_j)\right) \tag{5.9}$$

where m is the number of user posts in UG_d, $sentSim()$ returns the semantic similarity of s_i and c_j and is calculated by Eq. (5.10):

$$sentSim(s_i, c_j) = \frac{\sum_{w_i}^{N_s} \sum_{w_j}^{N_c} w2vSim(w_i, w_j)}{N_s + N_c} \tag{5.10}$$

where N_s and N_c are the number of words in s_i and c_j after removing stop words; $w2vSim()$ returns the similarity between two word vectors.

Social LDA score This feature shares the characteristic of *Local LDA score*, in which a summary should also cover topics discussed among readers. Given a set of topical words named $T_d = \{w_1, ..., w_k\}$ inferred from d_c (topical words are inferred from user posts), the social LDA score is computed by the same mechanism with Eq. (5.7).

$$\textit{social-lda-score}(s_i) = \frac{\sum_{j=1}^{k} weight(w_j)}{n} \text{ if } w_j \in s_i \tag{5.11}$$

where n is the number of words in s_i, weights are derived from topical words on the social side. By combining two LDA scores, our model states that a summary sentence should not only cover topics written by writers on the document side but also include topics discussed among readers on the social side.

Distance-based similarity This feature tackles the *generation* hypothesis by formulating sentence-user-post pairs in the form of lexical similarity as described in Section 3.1.1. It treats a different aspect with *semantic-based similarity* by operating on word and lexical levels instead of on the semantic level. This feature bases on nine distance features in the right column of Table 5.2. Their detail can be seen in Section 3.1.1.

Table 5.2: The features; S: a sentence, C: a comment or tweet.

Distance features	**Lexical features**
Manhattan distance	The longest common sub string of S and C
Euclidean distance	Inclusion-exclusion coefficient
Cosine similarity	% words of S in C
Word matching	% words of C in S
Dice coefficient	Word overlap coefficient
Jaccard coefficient	—
Jaro coefficient	—
Damerau-Levenshtein	—
Levenshtein distance	—

The distance feature states that a summary sentence and user post should be closer than non-summary ones. To compute it, a similarity score, e.g. Cosine can be used; however, using a single measurement may not efficient enough to completely capture the similarity aspect of a sentence-user-post pair. For example, a sentence and comment may not share common words due to their content variation, which may negatively affect the Cosine calculation. We, therefore, used various distance features in word and character levels to tackle this issue. For example, the Manhattan distance covers pairs those share common words while the Levenshtein distance based on characters treats pairs; those are content variation. Eq. (5.12) computes the distance between a sentence and user posts.

$$dist(s_i, UG_d) = \max_{j=1}^{m} \left(distSim(s_i, c_j)\right) \tag{5.12}$$

where m is the number of user posts in UG_d; $distSim()$ returns the distance similarity of s_i and c_j and is computed by Eq. (5.13).

$$distSim(s_i, c_j) = \frac{1}{F}\sum_{k=1}^{F} f_n(s_i, c_j) \tag{5.13}$$

where F contains nine distance features in Table 5.2; $f_k()$ is a similarity function computed by each k^{th} feature.

Lexical-based similarity This feature also shares the *generation* hypothesis with the *distance-based similarity* but using word overlapping. It states that a summary sentence and user post should share common words.

$$lex(s_i, UG_d) = \max_{j=1}^{m} (lexSim(s_i, c_j)) \tag{5.14}$$

This feature is modeled in the same mechanism with the distance feature but using five lexical features in Table 5.2. Note that our features are also used for modeling user posts.

After extracting features, we train two L2R-based models, for sentences and user posts on training data and apply them to estimate the importance of sentences and user posts in testing data. After predicting, sentences with high scores are important while those with low scores are unimportant. For implementation, we used $C = 3$ with the linear kernel for Ranking SVM. We leave the tuning of C as a future task. We employed the simple greedy method in Section 3.1.3 for selection. After ranking, it dequeues one sentence and puts it into the summary. It stops when the number of selected sentences reaches to m.

5.1.3 Baselines

We first compared our model to the basic model described in Section 5.1.1. It was trained by using RankBoost with many sophisticated features (Wei and Gao, 2014). Although it is a basic model in this section, it is also an advanced method. Therefore, its ROUGE-scores are reported in the category of social context methods.

We also used the same non-social context and social context methods which are already mentioned in Section 3.1.4 for our comparison for our comparison.

5.1.4 Results and Discussion

In this section, we first present ROUGE-scores of our model against baselines and advanced models. We next show analyses to reveal several aspects of our model.

ROUGE-scores

This section reports results of our comparison with non-social context methods, followed by ROUGE-scores of social context methods.

Our model vs. non-social context methods Table 5.3 shows ROUGE-scores of our model and non-social context methods.

We can observe that our model acquires improvements, in which in some cases, it is significantly better than baselines (denoted by † with $p \leq 0.05$). For example, it is the best on VSoLSCSum and SoLSCSum except for sentence selection on SoLSCSum. On USAToday-CNN, its ROUGE-scores are also comparable with baselines. This is because, as mentioned, highlights of USAToday-CNN are generated by humans in an abstract way, which challenges our features. However, for tweet extraction, our model is the second

Table 5.3: Summary performance; **bold** is the best value; *italic* is the second best.

Dataset	Method	Sentences			User posts		
		RG-1	RG-2	RG-W	RG-1	RG-2	RG-W
SoLSCSum	Lead-m	0.345	*0.322*	*0.170*	—	—	—
	LexRank	0.327†	0.243†	0.138†	**0.210**	*0.115*	**0.085**
	SVM*	0.325†	0.263†	0.147	0.152†	0.089†	0.062†
	CRF*	**0.393**	**0.379**	**0.187**	0.091†	0.075†	0.037†
	Our model*	*0.381*	0.304	0.169	*0.209*	**0.122**	**0.085**
USAToday-CNN	Lead-m	0.249	**0.106**	**0.172**	—	—	—
	LexRank	0.251	*0.092*	0.163	0.193	0.068	0.128
	SVM*	**0.261**	**0.106**	*0.171*	**0.221**	**0.084**	**0.149**
	CRF*	0.186†	0.088	0.114†	0.190	0.065	0.119
	Our model*	*0.253*	0.084	0.153	*0.213*	*0.080*	*0.140*
VLSCSum	Lead-m	0.495†	0.420†	0.214†	—	—	—
	LexRank	0.506†	0.432†	0.219†	0.348†	0.198†	0.127†
	SVM*	0.497†	0.440†	0.208†	0.374†	0.212†	*0.140*
	CRF*	0.422†	0.357†	0.172†	0.111†	0.062†	0.041†
	Our model*	**0.582**	**0.527**	**0.249**	**0.482**	**0.319**	**0.183**

best, i.e. 0.217 vs. 0.221. Also, combining many indicators could yield a conflict, which reduces the performance of the ranking algorithm.

As discussed, CRF comparably performs other methods in selecting sentences, but it achieves very poor results in extracting comments or tweets. This is because, the sequence aspect may not explicitly exist in social messages; therefore, it limits CRF. SVM also obtains acceptable performance, especially for tweet extraction on USAToday-CNN. This shows that with five basic features, its performance can reach to models which use many features. Lead-*m* and LexRank can be compared to our model in some cases, showing that they are very strong baselines. However, ROUGE-scores of our model are better than those in almost all cases.

Comparison with social context methods Table 5.4 summarizes the comparison with advanced methods.

The trend in Table 5.4 is consistent with Table 5.3, in which our model is the best in almost all cases, except for USAToday-CNN. For example, its ROUGE-1 is better than HGRW for sentence extraction on VSoLSCSum, i.e. 0.582 vs. 0.570. This is because: (i) our model integrates human knowledge in the form of features and (ii) it is is a supervised learning method instead of ranking based on random walk graphs. This also shows that HGRW is a competitive method even it is unsupervised. For example, it is the best in almost all cases on USAToday-CNN. RankBoost (CCF) also comparably performs other methods showing the efficiency of features in Wei and Gao (2014), but our model is still better because of our extension (Section 5.1.1). cc-TAM achieves quite poor results because it is designed for multi-document summarization whereas the three datasets are

Table 5.4: Our model vs. advanced methods.

Dataset	Method	Sentences			User posts		
		RG-1	RG-2	RG-W	RG-1	RG-2	RG-W
SoLSCSum	cc-TAM	0.306†	0.238†	0.136†	0.054†	0.022†	0.024
	HGRW	*0.379*	**0.304**	*0.167*	**0.209**	*0.115*	*0.084*
	RB* CCF	0.360	0.283†	0.158	0.190†	0.098	0.077†
	Our model*	**0.381**	**0.304**	**0.169**	**0.209**	**0.122**	**0.085**
USAToday-CNN	cc-TAM	0.229	0.077	0.145	**0.249**	*0.089*	*0.152*
	HGRW	**0.279**	**0.098**	**0.177**	*0.242*	0.088	**0.157**
	RB* CCF	0.221	0.070	0.140	0.233	**0.091**	0.132
	Our model*	*0.253*	*0.084*	*0.153*	0.213	0.080	0.140
VLSCSum	cc-TAM	0.488†	0.377†	0.201†	0.301†	0.167†	0.111†
	HGRW	*0.570*	0.479†	0.233†	0.454	0.298†	*0.173*
	RB* CCF	0.561	0.494†	0.235†	*0.471*	*0.308*	0.168
	Our model*	**0.582**	**0.527**	**0.249**	**0.482**	**0.319**	**0.183**

for single document summarization. We also observe that it selects quite short sentences and user posts on the three datasets.

Feature contribution

We examined the contribution of features in our model. We first present the influence of each new feature by observing its weight, and next show the role of each feature group.

Feature weight We investigated the influence of each feature by averaging its weight generated in training our L2R model on SoLSCSum. Because the role of original features is already reported in Wei and Gao (2014), we only show the contribution of ours.

Feature weights in Tables 5.5 and 5.6 indicate that for sentence selection, local features, e.g. sentence length, Cosine similarity with a next and a previous sentence positively contribute our model while the sentence length with a next and previous sentence, local topical score have negative values. Social features, e.g. Word2Vec score and lexical similarity score also play an important role whereas auxiliary topical score is negative. This trend is similar to comment extraction. Interestingly, sentence length and the number of stop words are positive in selecting sentences but they are negative for comment extraction. It is understandable that long comments usually include redundant information, e.g. the opinion of readers. For the number of stop words, because comments or tweets are written in an informal style with noise, then counting stop words is inefficient.

Feature group contribution We further conducted an observation to show the contribution of each feature group in our model. To do that, new features were combined with basic ones in Wei and Gao (2014) to train our model with three settings: (i) using all features, (ii) using local features (new and old features), and (iii) using social features

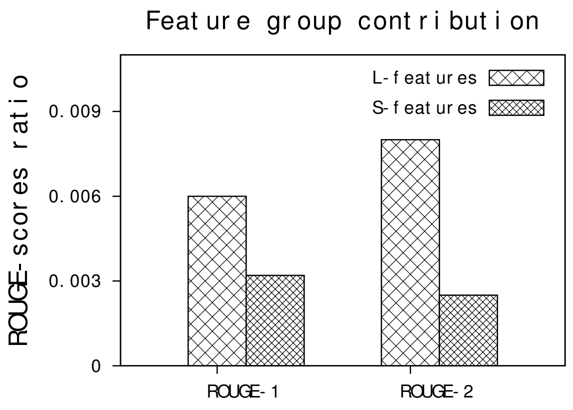
Table 5.5: Feature contribution for sentence selection.

Sentence selection			
Local features	**Weight**	**Social features**	**Weight**
Sent-length	1.533	Semantic-based score	1.768
Sent-length before	-0.283	Aux-LDA score	-0.101
Sent-length after	-0.251	Lexical-based sim	0.024
Cosine similarity before	1.039	Distance-based sim	0.216
Cosine similarity after	1.044	—	—
Local LDA score	-0.267	—	—
Function word	0.489	—	—

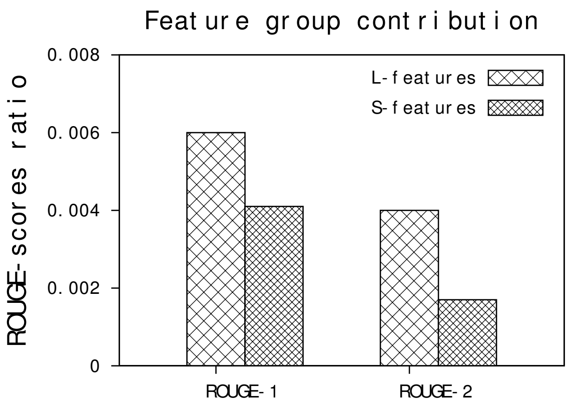
Table 5.6: Feature contribution for comment extraction.

Comment extraction			
Local features	**Weight**	**Social features**	**Weight**
Sent-length	-0.213	Semantic-based score	0.085
Sent-length before	-0.655	Aux-LDA score	-0.153
Sent-length after	-0.317	Lexical-based sim	2.197
Cosine similarity before	0.475	Distance-based sim	0.270
Cosine similarity after	0.217	—	—
Local LDA score	-0.415	—	—
Function word	-1.155	—	—

(new and old features). The influence of each group was defined as the ratio of ROUGE-scores computed by the minus ROUGE-scores of the first setting for the second and the third setting. The assumption behind this setup is that we expect the model using all features obtains better results than those using local or social features.



(a) Sentence selection.



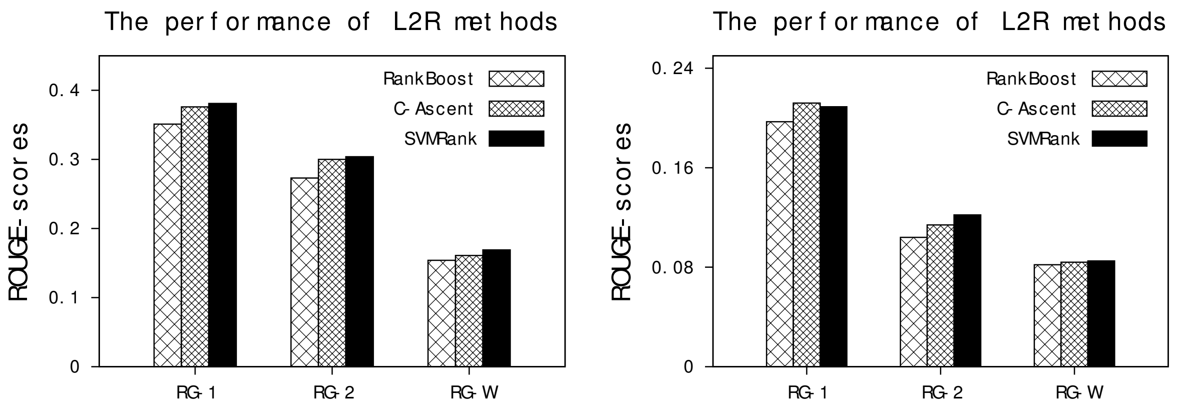
(b) Comment extraction.

Figure 5.2: Feature group obervation. ROUGE-W is not shown due to its tiny values.

Figure 3.2 shows that both local and social features contribute to our model with positive values. It means that when removing them, summarization performance decreases. Values of local features are larger than those of social features showing that the inherent information of each sentence or user post is more important than social information. It is understandable that our model uses many sophisticated features from a sentence as the main part and exploits additional features from comments as the support. Social features slightly affect the estimation of sentences with small values; however, for comment extraction, the contribution of social features increases. It means that additional from sentences benefits the estimation of user posts.

Summary performance with L2R methods

We conducted an observation of using different L2R methods to answer a question that which is an appropriate L2R method for our task. In order to do that, besides using our model, we ran RankBoost (Freund et al., 2003) (iteration of 300, metric is ERR10), Coordinate Ascent (Metzler and Croft, 2007) (random restart of 2, iteration of 25, tolerance of 0.001 with non-regularization). We used all features to train these models on SoLSCSum.



(a) Sentence selection

(b) Comment extraction

Figure 5.3: Our features with L2R methods.

ROUGE-scores in Figure 5.3 indicate that Ranking SVM is the best for both sentence and comment extraction. This is understandable that it inherits powerful characteristics from SVM to perform pair-wise ranking. For example, it can create correct margins for classification based on the help of margin maximization. In training, this property may help to reduce the over-fitting problem. RankBoost comparably performs Ranking SVM except for ROUGE-1 of comment extraction. This supports our idea stated in Section 5.1.1, in which we improve the basic model by not only adding new features but also employing a strong L2R approach. Coordinate Ascent is the second best in almost all cases. Compared to ROUGE-scores in Section 5.1.4, these methods still outperform baselines, suggesting that formulating the estimation in the form of L2R benefits sentence selection.

Output observation

We further analyzed summaries generated from our model on SoLSCSum. In Table 5.7, our model yields correct sentences (denoted by [+]) which mention the death of Usaamah Rahim at the Boston shooting event and the opinions of readers on this event. For sentence

Table 5.7: Extracted summaries of document 121^{th} on SoLSCSum dataset.

Selected sentences
[+]S1: Law enforcement officers in Boston shot dead a man on Tuesday who came at them with a large knife when they tried to question him as part of a terrorism-related investigation, authorities said, describing him as a "threat."
[-]S2: Boston Police said in a statement on their website that "as part of this ongoing investigation, Boston Police and State Police made an arrest this evening in Everett".
[+]S3: The 26-year-old man, identified as Usaamah Rahim, brandished a knife and advanced on officers working with the Joint Terrorism Task Force who initially tried to retreat before opening fire, Boston Police Superintendent William Evans told reporters.
[-]S4: Evans said officers had approached the man in a strip-mall parking lot without weapons drawn and opened fire only after he repeatedly advanced on them, leaving them in fear for their lives.
[+]S5: A man who identified himself on Twitter as Rahim's brother said the family was shocked by the shooting.
[+]S6: "The FBI and the Boston Police did everything they could to get this individual to drop his knife," Evans said.
Extracted comments
[+]C1: "Fear for your life" is exactly like a "sincerely held belief", there's absolutely nothing to weigh and no measurement possible to make such a determination.
[+]C2: If I had been one of the police officers I would have whispered 3 times "drop the knife" then quickly fired several shots at his sternum.
[+]C3: Either those cops weren't switched on enough to grasp the scope of the threat or Boston PD needs to review their procedures for addressing these types of threats.
[-]C4: Lawyers in a Union, lawyers in politics, they have made these unqualified sayings up, and its time to make them use more defined terms and refuse to accept escape path words that mean absolutely nothing.
[+]C5: Disturbed by the fact that they "didn't expect a reaction like this" and that they first retreated from this threat to themselves and others.
[+]C6: Yet his Iman brother was already claiming he was shot in the back with this hands in the air.

selection, by using the support from comments, our model selects four correct sentences. This is because they contain important information, i.e. the arrest of Boston Police and the description of Evans in the arrest mentioned in the document and its comments. As a result, our features can efficiently capture informative information in each sentence. However, it also picks up two incorrect ones (S_2 and S_4, denoted by [-]) because they have a similar length with the correct ones and also contain important information. This challenges our model and shows that our features are inefficient in such cases. However, S_2 and S_4 are still relevant to the event.

For comment extraction, we found that candidate summaries are long sentences and also share important phrases, e.g. *"drop the knife"*, *"cops"* and *"Boston"* with sentences. As a result, by using our features, information from sentences benefits the importance estimation of comments. However, our model also yields an incorrect comment (C_4) because it also has a similar sentence length. Extracted comments also show that they contain the opinions of readers (C_1 and C_5) and their suggested solutions (C_2 and C_3). Interestingly, C_6 provides new information (*"he was shot in the back with this hands in the* *air"*) of the arrest. This supports our argument in Section 2.2.1, which argues that user posts can provide additional information which may not be available in main documents.

5.2 Improving Summarization with Third-party Sources

The last section introduces a model which utilizes relevant user posts of Web documents to improve the quality of estimation. However, in the context of social media, we argue that the content of a main document is not only mentioned in its user posts but also can be found in related news articles published from different news providers. For example, after reading a Web document describing the Boston bombing event on Yahoo News, readers discuss this event by writing their comments on the website or posting tweets on their Twitter timeline. In the meanwhile, its content can also be published by different news providers, e.g. CNN in a writing variation form. Such relevant news articles can be found by using a search engine, e.g. Google with a search query including keywords or even the title of original documents. Figure 5.4 illustrates this schema.

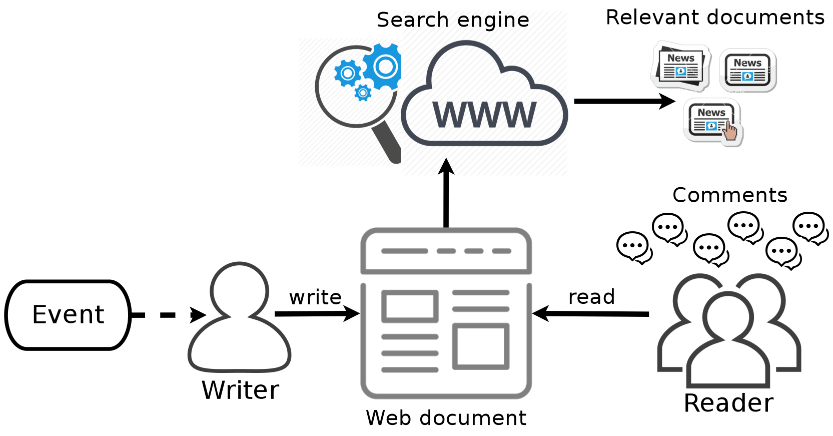


Figure 5.4: The schema of Web document, user-generated content, and relevant articles.

Figure 5.5 presents our model for taking advantage of mutual relationships between primary documents, their user posts, and their relevant articles. It includes three components: data collection, sentence scoring, and sentence selection. The collection collects main documents, their user posts, and relevant articles from different news providers. Its input is a combination of three elements collected from the collector. Our model learns to estimate the importance of sentences and user posts by employing a L2R algorithm, which uses information from three channels: local sentences, user posts, and relevant articles. For example, when modeling a sentence, it first extracts a set of local features. In the same time, it collects a set of features from user posts (the red line) and third-party sources (the blue line) to support local ones. When presenting a user post, similarly, social features from sentences (the red line) and third-party features from relevant documents (the blue line) are also used to support its local features. The selection takes scores from the scoring step and outputs a summary by using score-based or voting methods. Because the data collection is rather simple, we focus on describing scoring and selection steps.

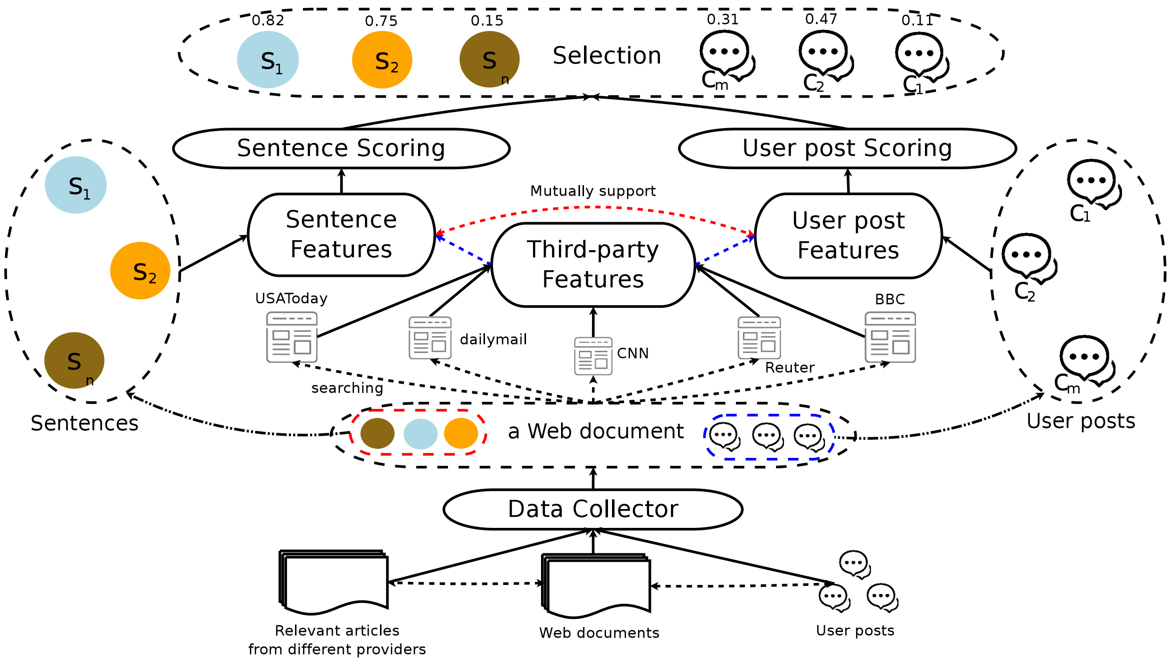


Figure 5.5: The overview of our model. Red and blue lines present the support of social context. Dashed lines from the document to news providers denote the searching process.

This model is an extended version of ours in the previous section. However, we improve the prior one with two significant improvements. Firstly, we consider a wider social context concept, which includes both user posts and relevant documents. The wider consideration allows to define novel features for estimating the importance of sentences and user posts. Secondly, we investigate the selection step by presenting a re-ranking method that bases on voting to extract summaries. In the following sections, we next show the new definition of social context and the collection of relevant articles. We then introduce our model, which uses several features from a basic one. Subsequently, we present our voting method and baselines. We final report our comparison.

5.2.1 New definition of social context

Since we consider relevant articles of primary documents as a new factor of social context; therefore, its definition in Section 2.2.1 needs to be modified.

User-generated content is defined as comments or tweets generated from readers after reading a Web document as in Section 2.2.1. This term is similar to user posts. We use this term for naming features extracted from user posts instead of using the name of social features. This is because social information now also includes relevant articles.

Third-party sources are a set of relevant news articles retrieved from a search engine by searching the title of original documents. Formally, let d is a primary document with its title t, $R = \{r_1, ..., r_m\}$ is m news providers, third-party sources of d is TS_d created by $d \xrightarrow[searching(t)]{R} TS_d$. We only select top ten results returned from a search engine because those outside this range are usually irrelevant. In this view, we consider third-party sources as a kind of global information.

Social context is an extension of the definition in Section 2.2.1, which includes third-party sources as an additional factor. The social context of a Web document d now is C_d presented by $\langle S_d, UG_d, TS_d, U_d \rangle$, where S_d is a set of sentences in document d, UG_d is user-generated content such as tweets or comments of d written by users U_d, and TS_d is relevant Web documents returned from a search engine by searching the title of d. By introducing a new definition, our model is flexible to integrate any additional sources related to main documents.

5.2.2 Data collection of third-party sources

As mentioned in Section 2.3.1, we already prepared three datasets in two languages for testing summarization methods. However, the new definition of social context requires to collect relevant news articles of primary documents.

SoLSCSum To create third-party sources of SoLSCSum, we retrieved relevant Web documents by searching titles of 157 primary documents. We chose Google search engine because it is more popular than others. Given an original document, after searching, top ten results appearing on the searching page were manually selected by removing duplicate Web documents. Unnecessary information, e.g. HTML tags was removed to obtain raw texts. We kept the order of relevant articles. Finally, we got 1570 relevant documents corresponding to 157 primary events. Note that in practice, we can build a search engine which crawls web pages from different news providers as a supplementary component of our model in Figure 5.5.

USAToday-CNN To create third-party information, we again employed Google to retrieve relevant articles of each original document via its title. Finally, we collected 1210 relevant documents corresponding to 121 primary ones.

VSoLSCSum With the same procedure, we retrieved 1410 relevant news articles corresponding to 141 primary documents. Table 5.8 presents the statistics of our datasets after collecting relevant documents. It indicates that the number of user-generated content and relevant articles is large enough to support sentences in main documents.

In next sections, we first describe a basic model, then introduce our new model, which take advantage of third-party sources to improve the quality of estimation.

Table 5.8: Statistical observation on our datasets.

Dataset	#doc	#sents	#comments/ tweets	#refs	#relevant docs
SoLSCSum	157	3,462	25,633	5,858	1,570
USAToday-CNN	121	6,413	78,419	455	1,210
VSoLSCSum	141	3,760	6,926	2,448	1,410

5.2.3 Basic model

We start by using a basic model for document summarization. Shen et al. (2007) formulated summarization as a sequence labeling problem and present several features to exploit the sequence aspect of sentences in a document. To train the model, the authors employed Conditional Random Fields (CRF) (Lafferty et al., 2001).

Conditional Random Fields

CRF is a learning model for sequence tagging. Given an observation sequence, i.e., sentences $X = (x_1, ..., x_T)$ and their corresponding state sequence $Y = (y_1, ..., y_T)$ (the relationship among states with a constraint that sequences are one-to-one), CRF learns to maximize conditional likelihood from training data. Eq. (5.15) shows the probability of Y given X:

$$P(Y|X) = \frac{1}{Z_X} \exp\left(\sum_{i,k} \lambda_k f_k(y_i - 1, y_i, X) + \sum_{i,l} \mu_l g_l(y_i, X)\right) \tag{5.15}$$

where Z_X is normalization which ensures the probability of all sequence states sums to one; $f_k(y_i - 1, y_i, X)$ is an arbitrary feature function of state i^{th} and $(i-1)^{th}$ and $g_l(y_i, X)$ is a feature function of state i^{th} over the entire observation sequence; λ_k and μ_l are the confident weights of feature functions f_k and g_l. CRF is trained in two steps: parameter estimation and inference.

Estimation The estimation estimates weights of CRF by usually using a maximum likelihood procedure which learns from labeled sequences in training data. Formally, let $W = \{\lambda_k, \mu_l\}$ be its weights which have to be estimated from training data $S = \{(X_1, Y_1), ..., (X_N, Y_N)\}$, the estimation maximizes conditional log-likelihood:

$$L_W = \sum_{j=1}^{N} log(P_W(Y_j, X_j)) \tag{5.16}$$

In training we can use some regularization methods to avoid over-fitting or adding a Gaussian prior over parameters.

$$L_W = \sum_{j=1}^{N} log(P_W(Y_j, X_j)) - \sum_{k} \frac{\lambda_k^2}{2\sigma_k^2} - \sum_{l} \frac{\mu_l^2}{2\sigma_l^2} \tag{5.17}$$

where $2\sigma_k^2$ and $2\sigma_l^2$ are the variance of Gaussian priors. The authors used quasi-Newton method (L-BFGS) (Sha and Pereira, 2003) to optimize Eq. (5.17).

Inference The most probable labeling sequence Y of an input X can be obtained by the conditional probability of CRF in Eq. (5.15) and its weight W.

$$Y* = argmax_Y P_W(Y|X) \tag{5.18}$$

In decoding, the marginal probability of states at each position in the sequence can be computed by using a dynamic programming inference. Finally, the label of each state is decided based on its probability $P(y_i = 1|X)$.

In practice, we define a sequence as a set of sentences in a document (or user posts) and then employ this method to train two CRF-based summarizers, for sentences and user posts. Summaries are extracted by selecting those labeled by summary.

Basic features

To train CRF-based summarizers, we adopted basic features from Shen et al. (2007). They include sentence position, sentence length, log-likelihood, thematic words, indicator words (count the number of indicator words using a dictionary), uppercase words, Cosine similarity with previous and next N ($N = 1, 2, 3$) sentences, LSA and HIT score. Their detailed description can be seen in Shen et al. (2007). Remaining sections describe our new features used to improve the performance of the basic model. Note that new features for sentences are describe; however, they can also be applied to user posts.

New local features

We first define a set of features, which represent the inherent information of each sentence. We denote them as local features, which capture three aspects: the similarity of a sentence (or comment/tweet) with a title, informativeness, and topical score.

Sentence-title common words This feature counts the number of words in a sentence s_i appearing in the title.

Stop words This feature counts the number of stop words.

Local LDA score This feature was already shown in Section 5.1.2. Given a set of topical words $T_d = \{w_1, ..., w_k\}$ inferred from LDA, Eq. (5.19) computes this feature:

$$\textit{local-lda-score}(s_i) = \frac{\sum_{j=1}^{k} weight(w_j)}{n} \text{ if } w_j \in s_i \tag{5.19}$$

where $weight()$ returns the word weight of w_j, e.g. 0.45 in s_i (normalized in [0, 1]); n is the number of words in s_i.

Table 5.9: Features used to train a L2R model..

Feature group	Description
Local features	The position of the sentence The length of a sentence. Log-likelihood of a sentence. Thematic words in a sentence using a dictionary. Indicator words in a sentence using a dictionary. Uppercase words. Cosine similarity of a sentence with previous and next sentences (N = 1, 2, 3). LSA score of a sentence. HIT score of a sentence.
	The number of common words with a sentence and title. The number of stopwords in a sentence. Local LDA score of a sentence.
User-generated features	Maximum semantic similarity of a sentence with user posts. Auxiliary LDA score of a sentence with user posts. Maximum distance similarity of a sentence and user posts. Maximum lexical similarity of a sentence and user posts.
Third-party features	Cosine voting of a sentence with sentences in relevant documents. Cluster distance of a sentence with relevant documents. Sentence-thir-party term frequency. The average probability of frequent terms. The frequent term summing score. The relative frequent term summing score.

5.2.4 Novel model with social context integration

The basic model with basic features acquires competitive ROUGE-scores in summarizing documents (Shen et al., 2007), however, its performance can be improved in two directions. Firstly, it bases on CRF to exploits the sequence aspect in sentences, but this aspect does not implicit exist in user posts, in which readers usually write separate social messages or a small number of relevant sentences in the form of a paragraph. This limits the efficiency of CRF in summarizing user posts (see ROUGE-scores of CRF in previous sections). Based on promising results of Ranking SVM, we again adopted this method to tackle the limitation of CRF in dealing with the non-sequence aspect in user posts.[3] Secondly, the basic model only uses inherent information of a Web document in the form of local features. Here we argue that information from social context can be beneficial for the importance estimation. This observation motivates to propose new features, which cover characteristics of a summary candidate that may match with gold-standard references.

[3]We use labels on USAToday-CNN for training instead of using ROUGE-1 F-score as Section 5.1.2.

User-generated features

When modeling sentences in documents, a set of features was extracted from their user-generated content, which has an explicit relationship (readers post these messages after reading the document) with primary documents. These features cover semantic similarity, topic, and distance and lexical similarity aspects. They share the implementation with social features in Section 5.1.2. This paragraph, therefore, only shows a short description of these features. Please refer Section 5.1.2 for their motivation and implementation. We rename the term of social features in Section 5.1.2 to user-generated features because social context now includes both user posts and relevant articles.

Table 5.10: User-generated features.

Feature	Description
Max semantic sim	Maximum semantic similarity of a sentence and user posts.
Auxiliary LDA score	Auxiliary LDA score of a sentence with user posts.
Max distance sim	Maximum distance similarity of a sentence and user posts.
Max lexical sim	Maximum lexical similarity of a sentence and user posts.

Third-party features

As discussed in Section 5.2, an event in a primary document is also mentioned in different relevant news articles published by other providers. This relationship is implicit because all relevant documents are created independently without the involvement of social users. It is possible to apply third-party features to user-generated content; however, they are only used for relevant documents due to the implicit relation. To capture relationships between primary documents and relevant new articles, we present new features which cover social voting, social distance, and the appearance of frequent words.

Voting This feature bases on an observation that relevant documents should include salient terms in a summary sentence. Given a sentence s_i in a primary document d, this feature counts Cosine similarity (greater a threshold) with sentences (stopword removal) in relevant documents. Eq. (5.20) shows the counting.

$$n\_vote(s_i) = \frac{\sum_{j=1}^{m} cos(s_i, t_j)}{N_S} \tag{5.20}$$

where m is total sentences in relevant documents, and N_S is the number of sentences in d. In practice, the threshold is empirically set as 0.65 when modeling sentences and 0.35 when modeling comments or tweets.

Cluster distance The hypothesis of this feature is that a summary sentence should be close to clusters represented by relevant documents. In this view, each relevant document is considered as a cluster. Given a sentence s_i (no stop words) and a relevant document $rd_j \in D$ presented by a set of frequent terms, Eq. (5.21) defines this feature.

$$c\text{-}dist(s_i, D) = \frac{\sum_{j=1}^{N_D} eucDist(s_i, rd_j)}{N_D} \tag{5.21}$$

where $eucDist()$ returns the Euclidean distance of s_i and rd_j using the bag-of-words model, and N_D is the number of relevant documents. The frequency threshold is 10 for both sentences and comments (tweets).

Sentence-third-party term frequency This feature presents the relationship of a sentence in a primary document and other sentences in relevant articles in the form of term frequency. It bases on a hypothesis that a sentence containing frequent words appearing in both primary documents and their third-party sources is more important than other ones. Given a sentence s_i (stopword removal) in a primary document d and N_{SD} is total sentences in relevant articles, Eq. (5.22) defines this feature.

$$stp\text{-}TF(s_i) = \frac{\sum_{j=1}^{|s_i|} TF(w_j) * IDF(w_j)}{|s_i|} \tag{5.22}$$

$$TF(w_j) = \text{the frequency of } w_j \text{ in } d \tag{5.23}$$

$$IDF(w_j) = log(\frac{N_{SD}}{DF(w_j)}) \tag{5.24}$$

where $DF(w_j)$ is the number of sentences in D containing word w_j.

Frequent-terms probability Remaining features base on an assumption that if the most frequent words appear in relevant articles, a summary candidate should contain these terms because they are essential for both document and global level.

A set of frequent terms was first collected from the raw texts of relevant articles based on frequency. In practice, the frequency threshold was empirically set as 5 for both sentences and user posts. If a term frequency is greater than a certain threshold, it is considered to be frequent. Given a frequent term set $FT = \{w_1, ..., w_t\}$ and a main document d, we included three probability features from Svore et al. (2007): the average probability of frequent terms (aFrqScore), frequent term sum score (frqScore), and relative frequent term sum score (rFrqScore). They are calculated by Eqs. (5.25), (5.27) and (5.28).

$$aFrqScore(s_i) = \frac{\sum_{w \in s_i} p(w)}{|w \in s_i|} \tag{5.25}$$

where $w \in FT$.

$$p(w) = \frac{count(w)}{|w \in d|} \tag{5.26}$$

where $count(w)$ is the frequency of w in d, and $|w \in d|$ is total number of frequent words in d.

$$frqScore(s_i) = \sum_{w \in s_i} p(w) \tag{5.27}$$

$$rFrqScore(s_i) = \frac{\sum_{w \in s_i} p(w)}{|s_i|} \tag{5.28}$$

Note that our features were also used for user posts in the same mechanism to exploit the mutual reinforcement relationship.

5.2.5 Sentence selection

This section describes two methods for selecting summaries. The first one uses a greedy strategy and the second one combines several L2R methods.

Score-based method We first consider the extraction by directly using scores generated from our L2R model by using a simple greedy method. After scoring sentences and user posts in a decreased order, it loops on ranked sentences and user posts and picks up a top one. This process stops when the number of summaries reaches to m.

Majority voting We further investigated the selection step by using majority voting from different L2R models. The motivation comes from promising results of our ILP model in Section 3.2.3, in that we use ILP to re-rank sentences. We developed this idea in a different direction by combining different L2R methods. We expect that by combining strong summarization methods, we can improve the quality of selection. To do that, we ran two additional L2R methods: RankBoost (Freund et al., 2003) (iteration of 300, metric is ERR@10), and Coordinate Ascent (CA) (Metzler and Croft, 2007) (random restart of 5, iteration of 25, performance tolerance of 0.001 with no-regularization) implemented in RankLib[4] with the same feature set of Ranking SVM. Final summaries were extracted by using majority voting among these L2R models.

Table 5.11: An example of majority voting. R_k is a ranker $k = (1, 2, 3)$.

Sentences	R_1	R_2	R_3	Voted Rank	Sentence Length
a	1	1	1	3	20
b	2	3	4	9	21
c	4	2	3	9	22
d	3	4	2	9	23

Output ranked sentences: $a > d > c > d$

Table 5.11 shows an example that contains four sentences with different rank positions corresponding to three rankers (R_1, R_2, R_3). It is possible to use scores generated from

[4]http://people.cs.umass.edu/∼vdang/ranklib.html

each ranker to compute voted ranks; however, scores computed by different mechanisms in various rages are inappropriate to be used in majority voting. Therefore, the voted rank of a sentence is the sum of its ranked positions generated by rankers. Eq (5.29) shows this formulation.

$$voted\_rank(s) = \sum_{k} \alpha_k * rank_k(s) \tag{5.29}$$

where α_k is the weight of each ranker, which is set to 1 (it means that three rankers are equal); k is the number of rankers; $rank_k(s)$ returns the rank position of sentence s using ranker k. The $rank_k(s)$ is defined in Eq. (5.30).

$$rank_k(s) = rank\_pos(s, k) \tag{5.30}$$

where $rank\_pos(s, k)$ returns the ranked position of s in the ranker k.

Our voting algorithm first selects sentences, which have the highest voted rank. If two sentences have the same voted rank ($voted\_rank(s_i) = voted\_rank(s_j)$), the longer one is chosen. For example, in Table 5.11, the sentence a is first selected. Sentences b, c, and d own the same voted rank, i.e. 9; however, our algorithm picks up d because it has the longest length. As a result, the final order of four sentences is $a > d > c > d$.

5.2.6 Baselines

We compared our model to CRF because it is the basic one that our model bases on. However, CRF is also a non-social context method; therefore, its ROUGE-scores are reported in this category. We also used baselines and advanced methods which are already mentioned in Section 3.1.4 for our comparison.

5.2.7 Results and Discussion

This section first presents the comparison in term of ROUGE-scores, followed by analyses which help to understand several aspects of our model.

ROUGE-scores

We first report ROUGE-scores of our model and non-social context methods. We next show our comparison with advanced methods, followed by an observation of ROUGE-scores between CRF and SoCRF (the CRF model uses all features).

Our model vs. non-social context methods Table 5.12 shows ROUGE-scores of our model and non-social context methods. From this, we can observe that our methods are consistently better than baselines over three datasets except for ROUGE-2 and ROUGE-W of sentence selection on SoLSCSum. In several cases, they achieve significant improvements denoted by † with $p \leq 0.05$.

Table 5.12: Our model vs. basic methods. RG means ROUGE.

Dataset	**Method**	**Sentences**			**User posts**		
		RG-1	RG-2	RG-W	RG-1	RG-2	RG-W
SoLSCSum	Lead-m	0.345	0.322	0.170	—	—	—
	LexRank	0.327†	0.243†	0.138†	0.210	0.115	*0.085*
	SVM*	0.325†	0.263†	0.147†	0.152†	0.089†	0.062†
	CRF*	0.393	**0.379**	**0.187**	0.091†	0.075†	0.037†
	Score-based method*	*0.398*	0.329	0.179	**0.230**	**0.137**	**0.087**
	Voting	**0.401**	*0.332*	*0.183*	*0.223*	*0.132*	*0.085*
USAToday-CNN	Lead-m	0.249	0.106	0.172	—	—	—
	LexRank	0.251	0.092	0.163	0.193†	0.068†	0.128†
	SVM*	0.261	0.106	0.171	*0.221*	0.084	0.149
	CRF*	0.186†	0.088	0.114†	0.190†	0.065†	0.119†
	Score-based method*	**0.287**	*0.113*	**0.185**	**0.243**	**0.094**	*0.155*
	Voting	**0.287**	**0.114**	*0.184*	**0.243**	0.095	**0.156**
VLSCSum	Lead-m	0.495†	0.420†	0.214†	—	—	—
	LexRank	0.506†	0.432†	0.219†	0.348†	0.198†	0.127†
	SVM*	0.497†	0.440†	0.208†	0.374†	0.212†	0.140†
	CRF*	0.422†	0.357†	0.172†	0.111†	0.062†	0.041†
	Score-based method*	**0.577**	**0.524**	**0.246**	*0.462*	*0.309*	*0.176*
	Voting	*0.576*	*0.523*	**0.246**	**0.467**	**0.319**	**0.178**

The score-based method obtains the best ROUGE-scores in many cases because it employs Ranking SVM with many sophisticated features extracted from three channels: local information, user-generated content, and third-party sources. When modeling a sentence, additional information from social context enhances its representation, which benefits the estimation. Similarly, our new features support basic ones in presenting user posts. For example, Cosine similarity with next and previous user posts are inefficient for user posts (see Table 5.14). In this case, our features extracted from sentences and relevant documents help to enhance the representation of user posts.

The voting method acquires competitive results with score-based ranking because it exploits the efficiency from three powerful L2R models. In some cases, our score-based ranking achieves better ROUGE-scores than the voting, e.g. 0.230 vs. 0.223 because two incorrect sentence positions can dominate the correct one, leading to a wrong selection of the voting. However, compared to baselines in Table 5.12, it still obtains sufficient improvements. The comparison trend is consistent on three datasets validating the efficiency of our model and features. This indicates that our model and features are not only efficient in English but also effective in a non-English language, i.e. Vietnamese.

ROUGE-scores of CRF are competitive for document summarization, e.g. 0.379 of ROUGE-2 in Table 5.12 on SoLSCSum because, as discussed, it exploits the sequence aspect of sentences by using several features, e.g. Cosine with the previous and next sentences, leading to improvements over other baselines (Shen et al., 2007). However, it is inefficient for USAToday-CNN and VSoLSCSum. The reason is that the learning algorithm may not be appropriate with features on these datasets. On the other hand, CRF is limited in extracting comments or tweets e.g. 0.091 vs. 0.230 in Table 5.12 because firstly, the sequence aspect does not explicitly exist in user posts (see Section 5.2.7); therefore, it negatively affects the sequence labeling process. Secondly, similar to SVM, some basic features, e.g. sentence position or indicator words are inefficient in modeling comments or tweets. ROUGE-scores from Table 5.12 also validate that Lead-m and LexRank are competitive even they are unsupervised learning methods.

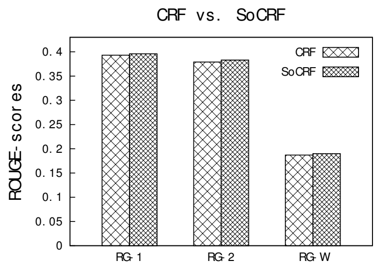
Comparison with social context methods We also compared our model with state-of-the-art methods and show their ROUGE-scores in Table 5.13.

Table 5.13: Our model vs. advanced methods.

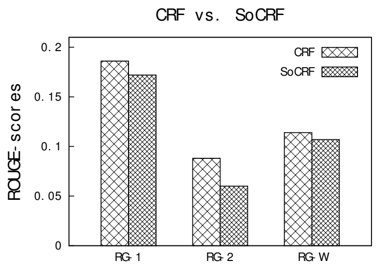
Dataset	**Method**	**Sentences**			**User posts**		
		RG-1	RG-2	RG-W	RG-1	RG-2	RG-W
SoLSCSum	cc-TAM	0.306†	0.238†	0.136†	0.054†	0.022†	0.024†
	HGRW	0.379	0.204	0.167	0.209	0.115	0.084
	RB* CCF	0.360†	0.283†	0.158	0.190†	0.098†	0.077
	Score-based method*	*0.398*	*0.329*	*0.179*	**0.230**	**0.137**	**0.087**
	Voting	**0.401**	**0.332**	**0.183**	*0.223*	*0.132*	*0.085*
USAToday-CNN	cc-TAM	0.229†	0.077†	0.145†	**0.249**	0.089	0.152
	HGRW	*0.279*	0.098	0.177	0.242	0.088	**0.157**
	RB* CCF	0.221†	0.070†	0.140†	0.233	0.091	0.132
	Score-based method*	**0.287**	*0.113*	**0.185**	*0.243*	*0.094*	0.155
	Voting	**0.287**	**0.114**	*0.184*	*0.243*	**0.095**	*0.156*
VLSCSum	cc-TAM	0.488†	0.377†	0.201†	0.301†	0.167†	0.111†
	HGRW	0.570	0.479	0.233	0.454	0.298	0.173
	RB* CCF	0.561	0.494	*0.235*	**0.471**	0.308	0.168
	Score-based method*	**0.577**	**0.524**	**0.246**	0.462	*0.309*	*0.176*
	Voting	*0.576*	*0.523*	**0.246**	*0.467*	**0.319**	**0.178**

The trend in this table is similar to Table 5.12, in which our score-based and voting methods are the best in almost all cases. For example, they are sufficiently better than HGRW of ROUGE-1 for sentence selection on SoLSCSum, i.e. 0.401 vs. 0.379. This is because: (i) our model integrates human knowledge in the form of sophisticated features extracted from three information channels to improve the importance estimation and (ii) it is a supervised learning method instead of ranking based on random walk graphs as HGRW. ROUGE-scores also show that HGRW is a strong method with very competitive ROUGE-scores on USAToday-CNN even it is unsupervised. RankBoost (CCF) based on RankBoost also comparably performs other methods showing the efficiency of features in Wei and Gao (2014). cc-TAM achieves quite poor results on our datasets because, as discussed, it is designed for multi-document summarization while our datasets are for single document summarization.

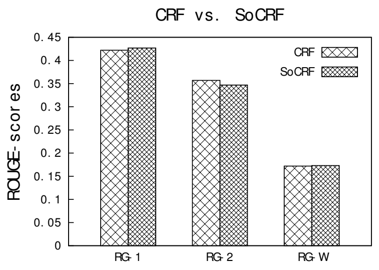
CRF vs. SoCRF We zoomed in to reveal ROUGE-scores of CRF using basic features and CRF using all features (called as SoCRF). This is because we would like to observe the effectiveness of our features on the same learning algorithm. We plot their ROUGE-scores on Figures 5.6 and 5.7.



(a) SoLSCSum

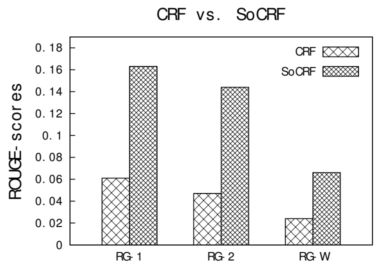


(b) USAToday-CNN

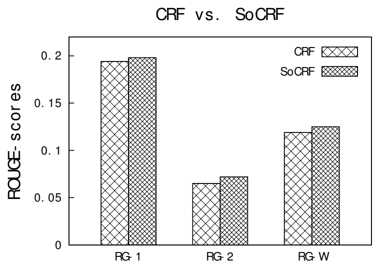


(c) VSoLSCSum

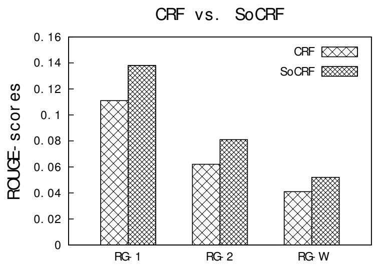
Figure 5.6: CRF vs. SoCRF for sentence selection.



(a) SoLSCSum



(b) USAToday-CNN



(c) VSoLSCSum

Figure 5.7: CRF vs. SoCRF for user post extraction.

For sentence selection from Figure 5.6, we can see that adding new features slightly improves the performance of CRF. For example, SoCRF is better than CRF only 0.03% of ROUGE-2 on SoLSCSum. The trend is similar to Figure 5.6c. This is because basic features of Shen et al. (2007) are efficient enough to capture the importance of each sentence. On USAToday-CNN, adding new features even reduces ROUGE-scores. The reason may come from a conflict when combining many features.

By contrast, for user post extraction, new features significantly benefit CRF. For example, they boost ROUGE-scores of SoCRF on SoLSCSum compared to CRF. This is also similar to Figure 5.7c, in that SoCRF is also sufficiently better than CRF. This is because new features cover several aspects, which may help to match a user post to references. For example, the sentence position is obviously inefficient for user posts; then new features can support the basic ones to improve the estimation of CRF. However, even SoCRF obtains better results than CRF; its performance is still far from ROUGE-scores of our model. A more detailed analysis is shown in Section 5.2.7.

Feature contribution analysis

This section investigates feature contribution in our model. It first presents feature weighs to reveal the influence of each feature. It also observes the contribution of each feature group. It finally investigates our features with various L2R methods.

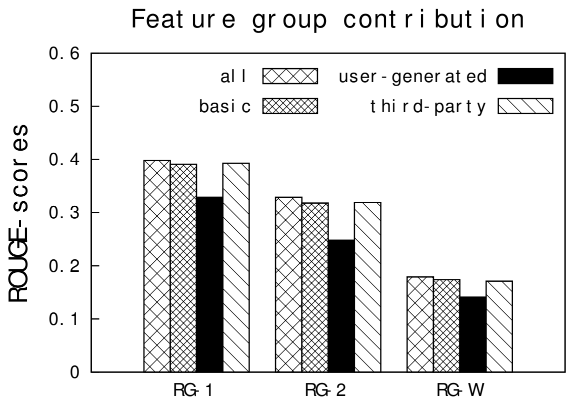
Feature weight observation The contribution of all features was investigated by averaging their weights generated from the training process in each fold on SoLSCSum. Table 5.14 presents top 15 effective features.

Table 5.14: Top 15 effective features on SoLSCSum dataset; **bold** denotes new features.

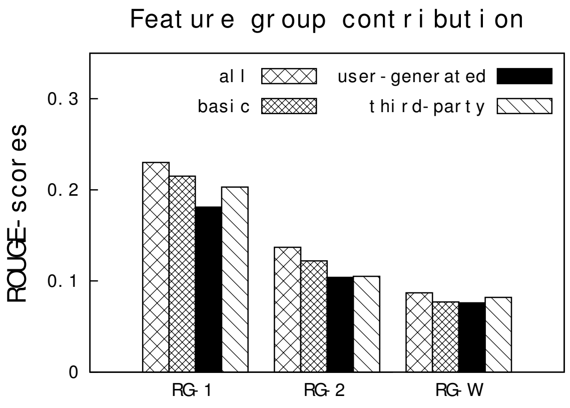
Document		Comment	
Feature	**Weight**	**Feature**	**Weight**
HIT-score	1.013	HIT-score	9.404
Cosine voting	0.703	**max-Cosine**	1.982
Cosine (N+1)	0.643	**max-W2V**	0.330
R-P-keyword	0.608	length	0.286
Cosine (N+3)	0.466	**P-keyword**	0.142
length	0.340	**# com-word-title**	0.132
Cosine (N-1)	0.322	**# stopword**	0.112
indicator word	0.311	thematic word	0.065
local-LDA	0.260	**local-LDA**	0.054
aux-LDA	0.260	**aux-LDA**	0.054
max-Cosine	0.256	indicator word	0.039
# com-word-title	0.081	log-likelihood	0.027
max-W2V	0.050	**STF-IDF**	0.006
# stopword	0.032	**R-P-keyword**	0.004
log-likelihood	0.032	**Cosine voting**	0.001

For sentence selection, HIT-score has the biggest feature weight. Our proposed features also play an important role by holding large weights, e.g. Cosine voting or R-P-keyword. Most basic features are in top eight explaining results in Table 5.12, in which CRF without new features obtains very competitive performance. Our new features appear in the rest. This validates results in Table 5.12, in which adding new features slightly improves ROUGE-scores. By contrast, for comment extraction, our new features are located in top eight, e.g. max-Cosine or max-W2V. This explains that integrating the support from sentences and relevant documents improves the quality of the estimation. Similar to sentence selection, HIT-score also obtains the highest weight. The contribution of each feature for sentence and comment extraction is different suggesting that different features should be used for sentences and user posts.

Feature group observation We investigated the contribution of each group by running Ranking SVM with four settings, using: basic, user-generated, third-party, and all features. We plot their scores on Figure 5.8.



(a) Sentence selection



(b) Comment extraction

Figure 5.8: The contribution of feature groups.

For sentence selection in Figure 5.8a, ROUGE-scores indicate that using only user-generated features outputs poor results. This is because they model a sentence by only using cross relationships with comments, e.g. max-Cosine. They do not consider inherent characteristics of a sentence such as its length. Using basic (local) features generates competitive results compared to using all features. As mentioned in Table 5.14, several summary aspects can be covered by basic features. Interestingly, the model using third-party features is also comparable to the model using all features because relevant documents include salient phrases which can be captured by using statistical features.

For comment extraction in Figure 5.8b, ROUGE-scores are quite similar to sentence selection. The model using third-party features can be compared to the model using all features. However, for ROUGE-2, the model with third-party features generates a poor result because contents mentioned in comments cover a wider range than sentences. For example, readers usually show their opinions along with the content of an event. As a result, third-party features many be inefficient to capture comments' summary aspects.

Summarization with L2R methods As mentioned, we employed Ranking SVM to train our summarization model. However, it is possible that using other L2R methods may obtain better results. We, therefore, investigated this aspect by running two additional L2R methods: RankBoost and Coordinate Ascent using all features with parameters are the same in Section 5.2.5. This section is an extension of Section 5.1.4.

Table 5.15 shows that Ranking SVM obtains competitive results in almost all cases. Margins among L2R methods are small, confirming that modeling the importance estimation in form of L2R benefits our summarization task.

We further investigated features over these L2R methods to answer a question that whether our features are efficient with other L2R approaches. To do that, we run three L2R methods (RankBoost, Coordinate Ascent, and Ranking SVM) in two settings: (i)

Table 5.15: L2R methods on the three datasets.

Dataset	Method	Sentences			User posts		
		RG-1	RG-2	RG-W	RG-1	RG-2	RG-W
SoLSCSum	RankBoost	**0.399**	**0.329**	0.177	0.145	0.089	0.059
	Coordinate Ascent	0.386	0.318	0.174	0.187	0.112	0.070
	SVMRank	0.398	**0.329**	**0.179**	**0.230**	**0.137**	**0.087**
USAToday	RankBoost	0.289	0.112	0.183	0.235	0.088	0.152
	Coordinate Ascent	**0.290**	**0.116**	**0.189**	0.240	0.093	**0.155**
	SVMRank	0.287	0.113	0.185	**0.243**	**0.094**	**0.155**
VSoLSCSum	RankBoost	0.570	0.519	0.243	0.460	0.301	0.166
	Coordinate Ascent	0.574	0.519	0.245	**0.467**	**0.311**	**0.177**
	SVMRank	**0.577**	**0.524**	**0.246**	0.462	0.309	0.176

using all features, including our new features and (ii) using basic features on SoLSCSum and summarize their ROUGE-scores in Table 5.16.

Table 5.16: The ROUGE-scores of L2R methods. NF means using all features.

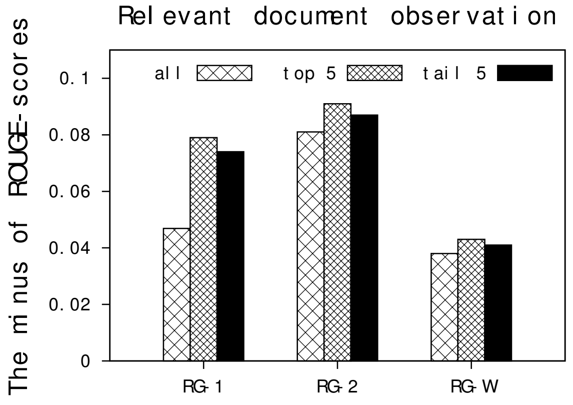
Method	Document			Comment		
	RG-1	RG-2	RG-W	RG-1	RG-2	RG-W
RankBoost	0.399	0.329	0.177	0.197	0.113	0.075
RankBoost+NF	**0.402**	**0.333**	**0.179**	**0.209**	**0.126**	**0.079**
Coordinate Ascent	**0.391**	0.318	0.174	0.187	0.112	0.070
Coordinate Ascent+NF	0.386	**0.324**	**0.175**	**0.196**	**0.114**	**0.075**
Ranking SVM	0.391	0.318	0.174	0.215	0.122	0.077
Ranking SVM+NF	**0.398**	**0.329**	**0.179**	**0.230**	**0.137**	**0.087**

Table 5.16 indicates that our features contribute to improve summarization performance in almost all cases. The improvement of comment extraction is larger than that of sentence selection because some basic features, e.g. sentence position are inefficient in modeling social messages. However, gaps among L2R methods are small compared to margins of L2R models and baselines in Tables 5.12.

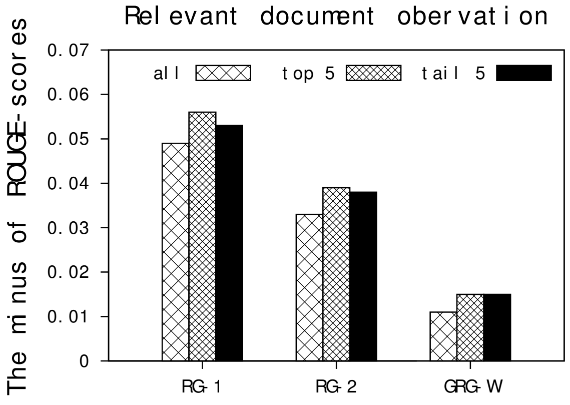
The results of our score-based ranking (Ranking SVM + NF) and Ranking SVM (basic features) show that integrating user-generated content and third-party information improves the quality of the importance estimation, especially for user posts. For sentence selection, adding new features slightly increases summarization performance, e.g. 0.398 vs. 0.391 of ROUGE-1 because basic features are efficient enough to capture summary aspects of each sentence. By contrast, for comment extraction, new features from social context help to improve ROUGE-scores of our model compared to the basic Ranking SVM, e.g. 0.230 vs. 0.215. This is because some basic features, e.g. sentence position are inefficient in modeling a comment or tweet.

Relevant document analysis

Relevant documents are a new component of our model compared to the previous one in Section 5.1. We, therefore, analyzed them to answer a question that how do relevant documents affect our model. To do that, relevant documents of SoLSCSum were divided into two sets: top and tail five documents. They were combined with primary ones and their comments to create three settings: (i) using all relevant documents, (ii) using top five relevant documents, and (iii) using tail five relevant documents. To measure their influence, we ran three Ranking SVM models corresponding to three settings with all features. ROUGE-scores of each model were normalized by the minus with the model only using user-generate features. We report these scores on Figures 5.8a and 5.8b.



(a) Sentence selection



(b) Comment extraction

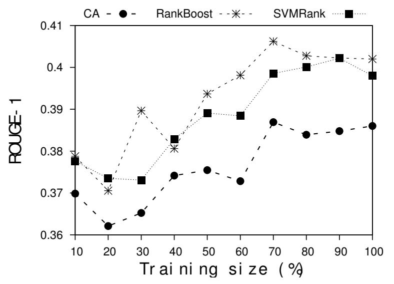
Figure 5.9: The observation of relevant documents.

Figures 5.9a and 5.9b indicate that using top or tail five related documents outputs better results than using all ones. It is perhaps using all relevant documents leads to the noise of data, which makes confusion in computing statistical features. The performance of top five is better than that of tail five because the former usually includes documents which are more relevant than the latter. This is because a search engine usually returns irrelevant documents which include noise in the tail of result page. However, margins among these models are small (0.015 of ROUGE-1 for sentence selection). In practice, we use all relevant documents to form third-party sources.

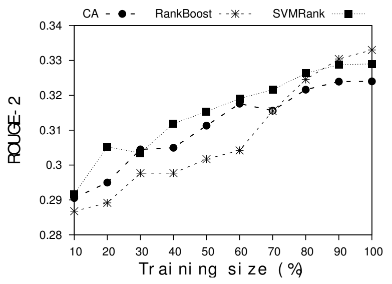
Training data observation

This section presents the investigation of training data to answers two questions: (i) how does training data-size affect our model and (ii) what is the relationship between training data and training time.

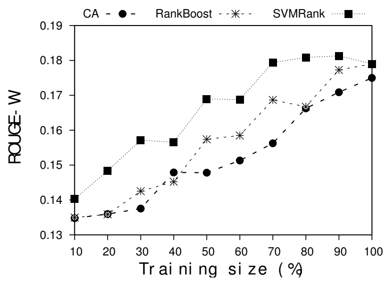
Data size and performance We observed the influence of data-size in training three L2R methods to reveal the relationship between training data and summarization performance on SoLSCSum. The observation used data in each fold as a testing set and the rest as a training set. For training, we randomly pick up a subset, which ranges from 10% to 100% from the training set for learning. The performance at each data size is the average ROUGE-scores over 10-folds.



(a) ROUGE-1

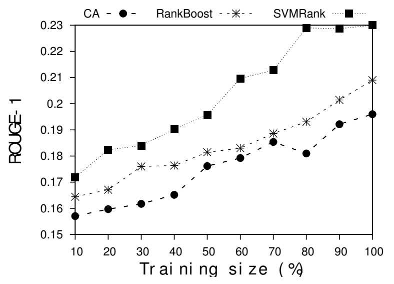


(b) ROUGE-2

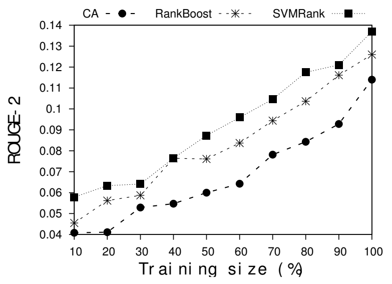


(c) ROUGE-W

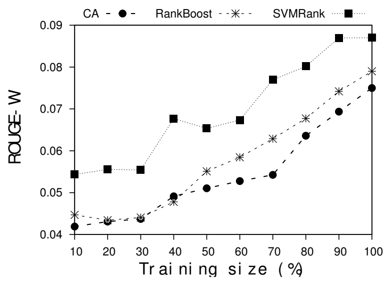
Figure 5.10: Training data-size for sentence selection.



(a) ROUGE-1



(b) ROUGE-2



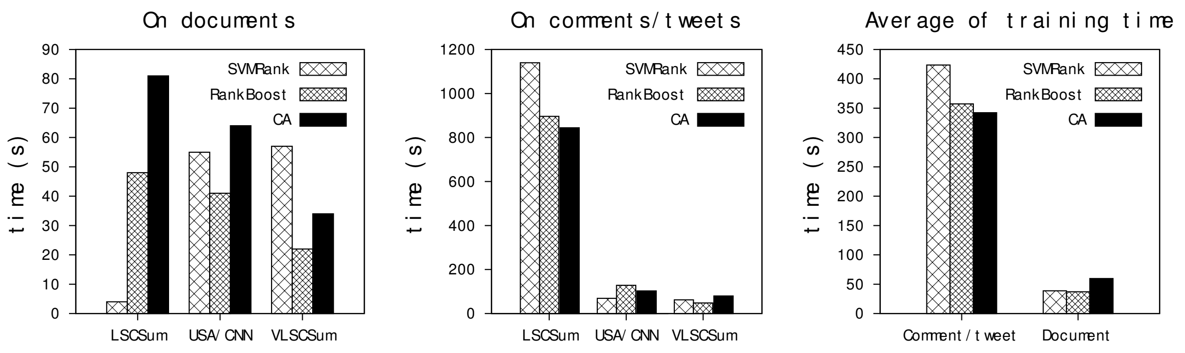
(c) ROUGE-W

Figure 5.11: Training data-size for comment extraction.

The general trend in Figures 5.10 and 5.11 indicates that adding training data improves summarization performance. It is understandable that putting more training data allows L2R methods to find out correct hypotheses. As a result, they can efficiently predict unseen data in the testing set. However, in some cases, adding more training data reduces summarization performance due to the noise of data, e.g. from 30% to 40% of RankBoost in Figure 5.10a. The trend also indicates that there is a mall gap between Ranking SVM and other methods during the training process.

Training time analysis We also analyzed the time aspect in training L2R methods on three datasets. To do that, we monitored the training process, recorded total training time in seconds, and plotted it on Figure 5.12.

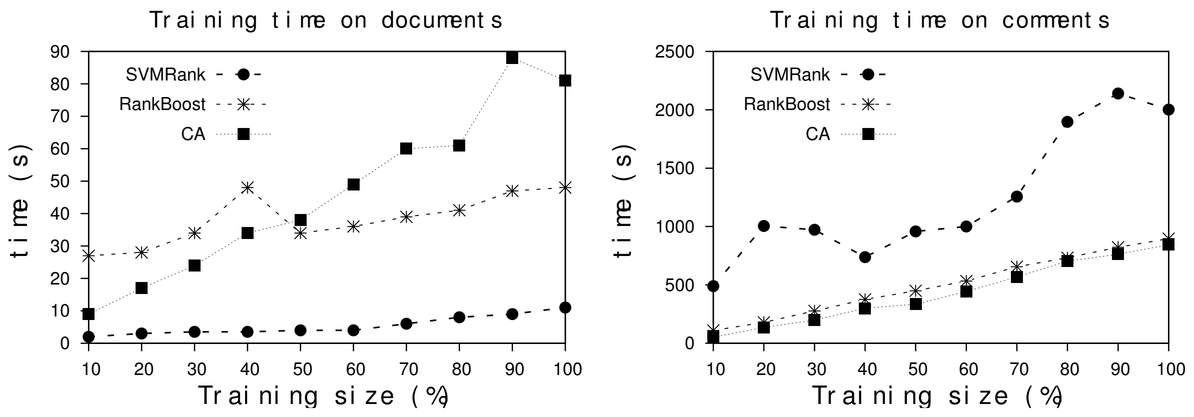
The trend in Figures 5.12a and 5.12b summarizes that training a model for sentences takes a shorter time than training with user posts. For example, Coordinate Ascent spends 80 seconds for training on sentences (Figure 5.12a) while it needs more than 800 seconds to complete when training with comments (Figure 5.12b). The average training time in Figure 5.12c supports this observation. In Figure 5.12c, there is a small difference



(a) Sentence selection (b) Comment extraction (c) Average

Figure 5.12: Training time on three datasets. CA is Coordinate Ascent.

when training L2R models on sentences because the number of sentences is small. For user posts, the time dramatically increases, especially with Ranking SVM.



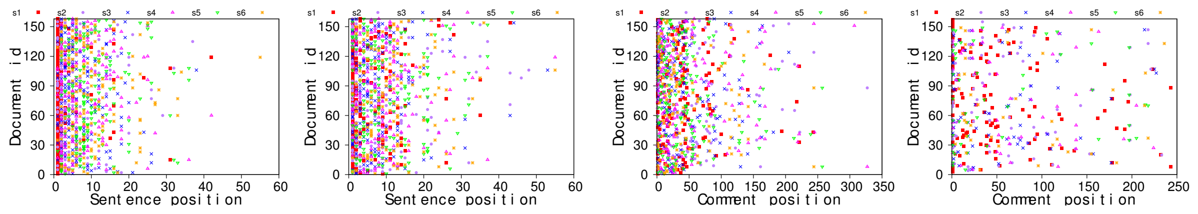
(a) Sentence selection (b) Comment extraction

Figure 5.13: Training time with different data size.

We zoomed in the training time of three L2R methods on SoLSCSum by observing on each training data size. The general trend in Figures 5.13a and 5.13b is consistent with results in Figures 5.12a and 5.12b, in which Ranking SVM is the fastest in training on sentences; however, on comments, it takes more time compared to other ones. It is because, in practice, Ranking SVM based on SVM spends a lot of time to find out optimal hyper-planes to separate training examples.

Sentence position observation

Sentence position is an important feature used in many studies of document summarization (Shen et al., 2007; Wei and Gao, 2014; Yang et al., 2011; Yeh et al., 2005). This section examines the position of extracted summaries generated from CRF and our score-based method on SoLSCSum. To do that, we matched extracted sentences and comments from these methods again original documents to extract their positions. Figures 5.14c and 5.14f visualize the position observation.



(a) Ranking SVM (b) CRF (d) Ranking SVM (e) CRF

(c) Sentence selection (f) Comment extraction

Figure 5.14: The position of extracted sentences and comments.

For sentence selection in Figures 5.14a and 5.14b, we observe that CRF and our method output a similar position distribution, in which most summaries are located within first 10 sentences. This explains that our score-based method slightly outperforms CRF (see Tables 5.12). By contrast, for comment extraction in Figures 5.14d and 5.14e, the trend is inconsistent. The distribution of comments extracted from our method is dense whereas it is so sparse in Figure 5.14e. This is because the lack of sequence aspect in comments limits CRF and explains that its performance is very poor on comments or tweets. Data observation from Figures 5.14c and 5.14f alo shows that Lead-m is efficient for sentence but not for user post extraction. There are some outlier points, e.g. 55th in Figures 5.14a or 5.14b because some documents contain a larger number of sentences and comments.

Error analysis

We analyzed outputs from our model on SoLSCSum. In Table 5.17, the score-based ranking selects four correct sentences and comments (denoted by [+]) which mention the event of Boston man shot by police. This is because summary sentences and comments contain important words *"Boston"*, *"police"* and *"arrest"*, which frequently appear in comments and relevant documents; therefore, our features, e.g. max-lexical similarity, social voting, or sentence-third-party term frequency can efficiently capture these sentences. Also, the max-W2V score feature can represent a sentence-comment pair containing words *"police"* and *"arrest"* by using semantic similarities, which supports the estimation. As our expectation, summary sentences and comments share salient words, e.g. *"Boston"*, *"arrest"* because of the formulation of our features.

Non-summary sentences (denoted by [-]), e.g. S3 and C4 or C5 also consist of relevant information in the form of salient words, which challenge our features. For example, S3 contains *"Evans"*, *"officers"*, and *"weapons"*; therefore, our features such as local-LDA, max-Cosine, max-W2V or Cosine-voting are inefficient in this case. In many cases, we can consider S3 as an important sentence. However, annotators decide it as a non-summary one because its content is included in S1 and S2. Also, non-summary sentences are as long as summary ones, e.g. S3 and C4 or C5 also challenging our model. However, even these sentences are unimportant; they are still relevant to the event.

Table 5.17: An example of 121th document from score-based ranking and voting.

SoSVMRank
Sentence selection
[+]S1: The 26-year-old man, identified as Usaamah Rahim, brandished a knife and advanced on officers working with the Joint Terrorism Task Force who initially tried to retreat before opening fire, Boston Police Superintendent William Evans told reporters.
[+]S2: BOSTON (Reuters) - Law enforcement officers in Boston shot dead a man on Tuesday who came at them with a large knife when they tried to question him as part of a terrorism-related investigation, authorities said, describing him as a “threat.”
[-]S3: Boston Police said in a statement on their website that “as part of this ongoing investigation, Boston Police and State Police made an arrest this evening in Everett.
[+]S4: Evans said officers had approached the man in a strip-mall parking lot without weapons drawn and opened fire only after he repeatedly advanced on them, leaving them in fear for their lives.
[+]S5: Suffolk County District Attorney Dan Conley said his office would investigate whether the shooting was justified.
[-]S6: (Reporting by Scott Malone; Editing by Alan Crosby) Boston PoliceJoint Terrorism Task ForceBoston-Law enforcement officers.
Comment extraction
[+]C1: “Boston Police and State Police made an arrest this evening in Everett”.
[+] C2: Either those cops weren’t switched on enough to grasp the scope of the threat or Boston PD needs to review their procedures for addressing these types of threats.
[+]C3: Boston man shot by police was target of terrorism probe.
[-]C4: That’s like “Good old fashioned police work”, which answers nothing means nothing.
[-]C5: War on BS is what the American people need to adopt when they hear those escape paths being uttered by anyone.
[+]C6: If I had been one of the police officers I would have whispered 3 times “drop the knife” then quickly fired several shots at his sternum.
Voting
Sentence selection
[+]S1: BOSTON (Reuters) - Law enforcement officers in Boston shot dead a man on Tuesday who came at them with a large knife when they tried to question him as part of a terrorism-related investigation, authorities said, describing him as a “threat.”
[+]S2: The 26-year-old man, identified as Usaamah Rahim, brandished a knife and advanced on officers working with the Joint Terrorism Task Force who initially tried to retreat before opening fire, Boston Police Superintendent William Evans told reporters.
[-]S3: Boston Police said in a statement on their website that ”as part of this ongoing investigation, Boston Police and State Police made an arrest this evening in Everett.
[+]S4: Evans said officers had approached the man in a strip-mall parking lot without weapons drawn and opened fire only after he repeatedly advanced on them, leaving them in fear for their lives.
[-]S5: (Reporting by Scott Malone; Editing by Alan Crosby) Boston PoliceJoint Terrorism Task ForceBoston-Law enforcement officers.
[+]S6: Suffolk County District Attorney Dan Conley said his office would investigate whether the shooting was justified.
Comment extraction
[+]C1: “Boston Police and State Police made an arrest this evening in Everett.
[-]C2: Once again, the police did what they’re supposed to do when threatened by an armed Muslim wacko who wasn’t doing what he was supposed to be doing.
[-]C3: This suspect is in the process of being booked, fingerprinted and interviewed What is that All about?
[-]C4: That’s like “Good old fashioned police work”, which answers nothing means nothing.
[+]C5: “Fear for your life” is exactly like a“ sincerely held belief”, there’s absolutely nothing to weigh and no measurement possible to make such a determination.
[+]C6: If I had been one of the police officers I would have whispered 3 times “drop the knife” then quickly fired several shots at his sternum.

Our voting correctly shares extracted sentences with the score-based method but in a different order. This is because it bases on two other L2R methods, which have similar behavior selection with Ranking SVM. The different order comes from majority voting used to re-rank sentences. This explains their ROUGE-scores are very similar in Tables 5.12. Interestingly, our methods extract the final sentence in the document (S5 and S6). It is irrelevant to the event but includes several important words, e.g. *"Boston Police"*, which confuse our features. For comment extraction, our voting outputs quite different summaries compared to the score-based method because L2R methods have different selection behavior on comments. We can also observe that extracted comments of voting seem to be longer than those of score-based ranking. It may come from the setting of voting, which biases long sentences if they have similar voted order.

Two selection methods extract C1 and C6 which perfectly reflect the event in the original document. While S1 and S2 show a complete story, C1 and C2 provide a shorter summary, which helps people to rapidly understand the story of this event. In this case, extracted comments can be represented as the first layer of summarization, which provides a snapshot and selected sentences can be seen as the second layer, from that people can zoom in the story of an event. C6 shows the opinion of readers after reading the document. In many cases, it can support sentences to provide a perspective viewpoint of readers regarding the event.

5.3 Conclusion

In this chapter, we describe our third effort to integrate the social context of a Web document to improve the importance estimation of sentences and user posts. We introduce two L2R models which exploit two different aspects of social context. The first aspect formulates mutual relationships between sentences and user posts as described in Chapter 3. We present new features extracted from two channels: local and social information to model these relationships. The second aspect is that we consider relevant news articles of a primary document as a factor of social context, which allows to extract new features from three channels: local, user-generated, and third-party information, instead of using two channels in the first model. Combining features in a mutual support fashion profits the importance estimation of sentences and user posts. We also investigate voting for re-ranking summary candidates. It utilizes the efficiency of three L2R methods to improve the quality of selection. We validate the efficiency of our models on three datasets in two languages. Experimental results indicate that our models obtain competitive ROUGE-scores compared to strong methods. We put our models together in Table 5.18.

We can see that adding third-party sources and using different basic features improve the performance of Ranking SVM on SoLSCSum and USAToday-CNN. On USAToday-CNN, the model using user posts and third-party sources obtains the best results. Note that it is a challenging dataset. On VLSCSum, using only using user posts outputs better results. Our voting also shows its efficiency in several cases.

We also observe several aspects to provide the better understanding of our models. For both models, we conclude that: (i) the importance estimation can be improved by using

Table 5.18: Results of our models. UP and TP are user posts and third-party sources.

Dataset	**Method**	**Sentences**			**User posts**		
		RG-1	RG-2	RG-W	RG-1	RG-2	RG-W
SoLSCSum	SVMRank (UP)	0.381	*0.304*	0.169	0.209	0.122	*0.085*
	SVMRank (UP+TP)	*0.398*	0.329	*0.179*	**0.230**	**0.137**	**0.087**
	Voting	**0.401**	**0.332**	**0.183**	*0.223*	*0.132*	*0.085*
USAToday-CNN	SVMRank (UP)	*0.253*	0.084	0.153	*0.213*	0.080	0.140
	SVMRank (UP+TP)	**0.287**	*0.113*	**0.185**	**0.243**	*0.094*	*0.155*
	Voting	**0.287**	**0.114**	*0.184*	**0.243**	**0.095**	**0.156**
VLSCSum	SVMRank (UP)	**0.582**	**0.527**	**0.249**	**0.482**	**0.319**	**0.183**
	SVMRank (UP+TP)	*0.577*	*0.524*	*0.246*	0.462	*0.309*	0.176
	Voting	0.576	0.523	*0.246*	*0.467*	**0.319**	*0.178*

additional information from social context denoted in the form of supporting features and (ii) formulating the scoring in the form of L2R benefits the selection. The analyses also indicate that relevant articles influence the ranking. Using top relevant articles is more beneficial than tail ones because those in the low order of result pages from a search engine usually are irrelevant to primary documents. Promising results of our second model which integrates third-party sources suggest that we can integrate any related sources of main documents or even global knowledge, e.g. Wikipedia into the summarization process.

Chapter 6

Summarization with Convolutional Neural Networks

Previous chapters present various models for exploiting social context to improve the importance estimation and selection. To make better understanding of the exploration, in this chapter, we adapt deep learning (DL) for our summarization task. Our motivation is that DL has been successful in several tasks of NLP such as neural machine translation (Bahdanau et al., 2014; Luong et al., 2015; Sundermeyer et al., 2014), text generation (Takase et al., 2016), or sentiment analysis (Kim, 2014). Recently, it has adapted for text summarization with promising results (Cheng and Lapata, 2016; Chopra et al., 2016; Ren et al., 2017). In addition, it can automatically learn representation from data, which addresses the obstacle of traditional methods in defining features. Although there is a considerable number of studies in applying DL to traditional text summarization, adapting this technique for our summarization task is still at an early stage.

In this chapter, we present a model based on Convolutional Neural Networks for estimating the importance of sentences and user posts. Instead of doing classification, we formulate the learning as regression. In training, besides using hidden features learned from training data, our model also exploits sophisticated features extracted from three channels: local information, user-generated content, and third-party sources to integrate social context. By doing this, we tackle the common issue of DL, that requires the large amount of data for learning representation. The integration enriches vector representation of sentences and user posts in a mutual fashion, which benefits their importance estimation. After ranking, summaries are extracted by using a greedy algorithm.

We report experimental results of our model on three datasets in two languages. ROUGE-scores indicate that our model obtains competitive results over advanced methods. In next sections, we first provide the background of Convolutional Neural Networks. We next describe a basic model and then introduce our model, followed by implementation. Finally, we show the comparison of our model with baselines, including discussion and analyses.

6.1 Convolutional Neural Networks

LeCun et al. (1998) originally designed Convolutional Neural Networks (CNN) for image processing. It includes a feed-forward network which uses a convolution operation to capture hidden features. Later, with its success, CNN has adapted to several NLP tasks (Cao et al., 2015c; Kim, 2014). Although input data in NLP such as sentences or documents is different from images, we can transform them into the representation of images. For example, Kim (2014) and Cao et al. (2015c) considered a sentence as a 2-dimensional matrix, in which each row i^{th} presents a word and each column j^{th} is a word embedding vector. Generally, given n sentences and k is a word embedding size, an embedding matrix $M \in \mathbb{R}^{n \times k}$ presents a map of sentences in a vector space. The standard CNN includes a convolution layer followed by a pooling operation.

Convolution layer Given an embedding matrix $M \in \mathbb{R}^{n \times k}$, CNN applies a *matrix kernel* (also called by a *filter*) for mapping each sub-region of M into values. The filter slides over every position of the matrix with the same weight to cover whole input data. Outputs from this filter form a *feature map*. The size of a filter (each sub-region) is denoted as a *kernel size*. Formally, let $v_i \in \mathbb{R}^k$ presents the k dimensional word embedding of a word i^{th} in a sentence; and $v_{i:i+j}$ is the concatenation of word embeddings $v_i, ..., v_j$. A convolution uses a filter $\boldsymbol{W}_t^h \in \mathbb{R}^{l \times hk}$, which operates on a window size of h words (a kernel size) to generate a new feature with l dimensions:

$$c_i^h = f(\mathbf{W}_t^h \times v_{i:i+h-1} + \boldsymbol{b}) \tag{6.1}$$

where $f()$ is a non-linear function, which usually uses *tanh()* in common practice and $\boldsymbol{b}$ is the bias vector. CNN applies $\boldsymbol{W}_t^h$ to each possible window of h words in the sentence (length N) to generate a feature map: $\boldsymbol{C}^h = [c_1^h, ..., c_{N-h+1}^h]$. In practice, we can use multiple kernel sizes corresponding to different size of h to obtain different feature maps for enriching the representation of input data. For example, Cao et al. (2015c) used kernel size = 1, 2, 3 corresponding to tri-grams for text summarization.

Pooling layer The pooling operation reduces the dimension of a feature map to retain the most important values. The intuition behind this operation is that it removes unimportant values and keeps critical ones which are used to represent input data. Also, by doing pooling, CNN can avoid over-fitting in the training process. Suppose we have a feature map $\boldsymbol{C}^h = [c_1^h, ..., c_{N-h+1}^h]$ after using convolution, the pooling selects important values to create new representation $\widehat{\boldsymbol{c}}^h$. Let's take Figure 6.1 as an example.

The max pooling applies a filter size of 2×2 on each sub-region to obtain a maximal value. Maximal values of all sub-region form new representation. There are different types of pooling such as max-over-time pooling operation (Collobert et al., 2011) or average pooling (Scherer et al., 2010).

In practice, there is another layer in a CNN-based model, that is fully-connected (also known as multi-layer perceptron - MLP). After applying several convolutional and pooling operations, MLP receives outputs from low-level layers to do classification or regression.

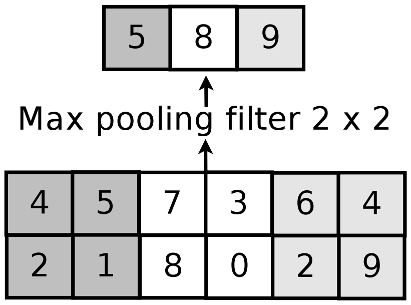


Figure 6.1: A max pooling operation.

As its name, this layer connects all inputs to outputs in a regular neural network. MLP facilitates classification by allowing to define the number of outputs corresponding to the number of classes.

In next sections, we first describe a basic model based on CNN and then introduce our model, which exploits social context to improve the estimation.

6.2 Basic Model

We start with a basic model named PriorSum introduced by Cao et al. (2015c) for multiple-document summarization. The idea of this model is that it captures the $n-$grams representation of sentences. The authors train their model in a regression fashion to rank sentences. More precisely, it adapts the standard CNN by using two new explorations. It first applies multiple filters with different window sizes for enriching the representation of a sentence. It second uses two max-pooling operations to capture the most important features for doing regression. Figure 6.2 illustrates the process of PriorSum.

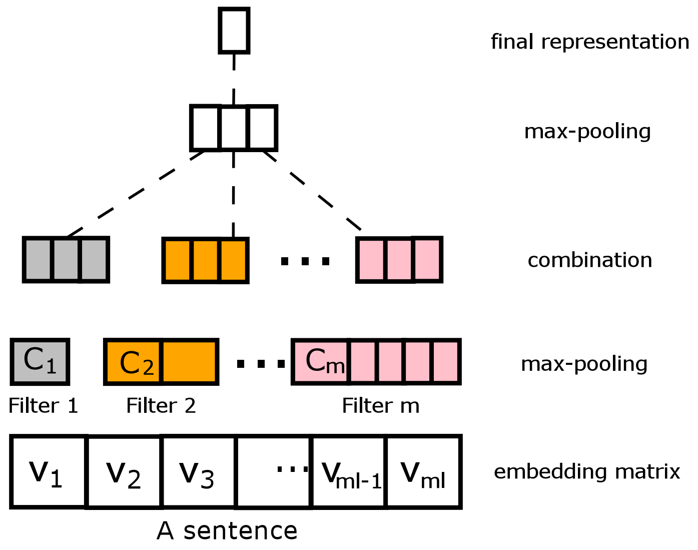


Figure 6.2: The overview of PriorSum.

In Figure 6.2, given a sentence s_i, a first step is to obtain the representation of each word in this sentence. This representation retains all information of this sentence, which could serve the learning process. Suppose s_i is expressed as $s_i = \{w_1, w_2, ..., w_n\}$ where w_j denotes the j^{th} word and n is the number of words, it first looks up a work embedding matrix $E_w \in \mathbb{R}^{d \times |V|}$ to get the word embedding of a word. Here, d is the dimension of embeddings and $|V|$ is the vocabulary size. Then the embeddings are fed into CNN which applies m filters to create m feature maps $\boldsymbol{C}^1, ..., \boldsymbol{C}^m$ by using the convolutional operation. For each feature map, PriorSum applies the first max-pooling operation on each $\boldsymbol{C}^i$ to capture the most important features of each window size i ($i = 1, 2, ..., m$). It next concatenates outputs of the first pooling operation to create new representation and then applies the second max-pooling to achieve the most salient features of an input sentence. The final vector representation of a sentence is the output of the second pooling layer. Eq. (6.2) summarizes the process of PriorSum.

$$x_p = max\{max\{\boldsymbol{C}^1\}, ..., max\{\boldsymbol{C}^m\}\} \tag{6.2}$$

The final vector x_p is fed into a MLP layer to do regression. In practice, we separately trained two PriorSum models, for sentences and user posts. After training, we applied these models to testing data and obtained scores (probabilities) of sentences and user posts. Based on their scores, we ranked to extract top m sentences (or user posts) to form a summary.

6.3 Our Model with Social Context Integration

We adapted PriorSum for our summarization task. There are two reasons behind our adaptation. Firstly, PriorSum bases on CNN, which has shown its ability to effectively learn n-grams representation and tackles sentences with variable lengths naturally. Secondly, it obtains promising results on DUC datasets in simple but effective architecture. However, for our task, we argue that its performance can still be improved. As presented, the social context of a Web document potentially benefits the estimation. However, the basic model ignores mutual relationships between documents and their relevant social information (user posts and related articles). We, therefore, exploit this information by adding sophisticated features into the representation of each sentence and user post in the final layer.

We incorporated human knowledge denoted in the form of features from Section 5.2 to enrich the representation of $\boldsymbol{x}_p$ in Eq. (6.2). There are two reasons behind our integration. The first reason comes from the fact that deep learning needs a lot of data to learn representation (Cao et al., 2015b; Chopra et al., 2016), but all datasets in this thesis are small (see Section 2.3.2). To support the capability of our model, we integrated many sophisticated features into our model. Second, as shown in previous chapters, social information enriches informative information of sentences. Therefore, such information may be beneficial to support hidden features learned from our model. By incorporating human knowledge, we expect that we can improve the representation of sentences (and user posts) which benefits the estimation. Table 6.1 shows our features.

Table 6.1: Local features (LF), social features (SF), and third-party features (TPF).

Group	Feature	Description
LF	Position	The sentence position of x_i in a document (pos = 1, 2, or 3).
	Length	The number of terms appearing in s_i after removing stop words.
	Title-words	#common words in s_i and the title after removing stop words.
	#Stopwords	Number of stopwords in a sentence s_i.
	HIT-score	The HIT score of a sentence s_i.
	LSA-score	The LSA score of a sentence s_i.
	Cosine	Cosine similarity with N next and previous sentences (N=1,2,3).
	Them-words	Count #frequent words calculated by TF appearing in x_i.
	In-words	Whether x_i contains indicator words appearing in a dictionary.
	Up-words	Whether x_i contains upper case words e.g., proper names.
SF	Max-cosine	Maximal Cosine of a sentence s_i with social information.
	Max-dist	Maximum distance of a sentence s_i with social information.
	Max-lexical	Maximal lexical score of s_i with social information.
	Max-W2V	Maximal W2V score of a sentence s_i with social information.
TPF	Voting	#sentences in relevant documents having Cosine $\geq$ a threshold regarding to s_i.
	Cluster-dist	The Euclidean distance from s_i to relevant documents.
	stp-TF	The score of s_i in relevant documents calculated by TF-IDF.
	aFrqScore	The average probability of frequent words in relevant documents regarding to s_i.
	frqScore	The probability of frequent words in supporting documents regarding to s_i.
	rFrqScore	The relative probability of frequent words in supporting documents regarding to s_i.

Features in Table 6.1 include three categories: local features (LF), social features (SF), and third-party features (TPF). Local features individually measure the importance of a sentence (or a comment/tweet), without considering social information or third-party sources. We eliminated topic modeling features because they are somehow similar to LSA score. Social features present relationships between sentences in a primary document and its user posts such. Third-party features estimate the importance of a sentence (or a user post) by using relevant documents. A detail description can be seen in Section 5.2.

Suppose $\boldsymbol{x}_e$ is the concatenation of three vectors $\boldsymbol{x}_l$, $\boldsymbol{x}_s$, and $\boldsymbol{x}_t$ corresponding to three groups, we concatenate x_p and x_e to create the final representation of a sentence.

$$\boldsymbol{x}_e = [\boldsymbol{x}_l, \boldsymbol{x}_s, \boldsymbol{x}_t] \tag{6.3}$$

$$\boldsymbol{\Theta} = [\boldsymbol{x}_p, \boldsymbol{x}_e] \tag{6.4}$$

By integrating indicators extracted from three channels: local features, user-generated content, and third-party features, our model not only enriches vector representation but also incorporates social information into the summarization process.

6.4 Learning

The final vector representation $\boldsymbol{\Theta}$ of a sentence s_i in both PriorSum and our model was fed to a MLP layer for learning. We used one neural as the output of MLP to estimate the importance of a sentence. In training, our model learns the weight $\boldsymbol{w}_r^T$ from training data to predict a score $\widehat{y}_i$ for a test input sentence.

$$\widehat{y}_i = \boldsymbol{w}_r^T \times \boldsymbol{\Theta} \tag{6.5}$$

where $\boldsymbol{w}_r \in \mathbb{R}^{l+|x_e|}$ is weights,[1] l is the hidden size of the MLP layer, $|x_e|$ is the size of vector $\boldsymbol{x}_e$, and $\widehat{y}_i$ is the importance of a sentence s_i predicted by the network. The training process uses the label of each sentence (or a comment/tweet) to minimize the loss function computed by using binary cross-entropy.

$$Loss = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\widehat{y}_i) + (1 - y_i)\log(1 - \widehat{y}_i)] \tag{6.6}$$

where N is the number of training data, y_i is a gold label, and $\widehat{y}_i$ is a predicted value.

In practice, we separately trained two our models, for sentences and user posts. After training, we applied the trained models on testing data to predict scores of sentences and user posts. To extract summaries, we employed the simple greedy method in Section 3.1.3. After ranking, we sorted sentences and user posts in a decreased order based on their scores. Summaries were extract by selecting top m rank sentences and user posts.

6.5 Implementation

For implementation, we followed Cao et al. (2015c) to set the hidden size $l = 20$ and the number of filters $m = 3$ corresponding to tri-grams. We trained two Word2Vec models for English and Vietnamese with embedding dimension of 25 by using SkipGram model[2] (Mikolov et al., 2013). Data is 1 million words for English and around 4 million news articles for Vietnamese. Table 6.2 summarizes parameters.

We trained our model on a GTX-1080 GPU card. In training, we used $k-$folds cross-validation with one fold for testing and other $k-1$ folds for training. We randomly selected 15% training data to form a validation set which helps our model to avoid over-fitting.

6.6 Results and Discussion

In this section, we first report the comparison of our model and baselines. We next show observation and analyses to reveal several aspects of our model.

[1]Note that $\boldsymbol{w}_r$ of PriorSum is in $\mathbb{R}^l$

[2]https://code.google.com/p/word2vec/

Table 6.2: Settings of our model.

Parameter	**Value**
Batch size	50
Number of epoch	25
Word dimension	25
Hidden size	20
Pooling size	2 x 2
Learning rate	1.0
Dropout rate	0.5
CNN kernels	[1, 2, 3]
Max sentence length	50 (for sentences and comments)
Max tweet length	20
Optimizer	Adadelta

6.6.1 ROUGE-scores

We first show the ROUGE-scores of CNN-based models, followed by the comparison with non-social context, and social context methods.

CNN-based models We report the performance of three CNN-based summarizers in Table 6.3. Their ROUGE-scores show three interesting points. Firstly, as our expectation, our model consistently outperforms PriorSum. Improvements come from the integration of social information in the form of features to enrich vector representation, which benefits for estimating the importance of each sentence and user post. For example, our model surpasses PriorSum 1% of ROUGE-1 for sentence selection on SoLSCSum. As mentioned, the small number of training examples on our datasets (see Table 2.5) challenges PriorSum and our model. Hence, adding features benefits the estimation. For user post extraction, the trend is consistent. Our model is also better than the basic one in all cases (significance is denoted by † with $p-$values ≤ 0.05). It validates our idea stated in Section 6.3.

Table 6.3: The performance of CNN-based summarizers; **bold** is the best value; *italic* is the second best. RG stands for ROUGE.

Dataset	**Method**	**Sentences**			**User posts**		
		RG-1	RG-2	RG-W	RG-1	RG-2	RG-W
SoLSCSum	PriorSum	0.413	0.367	0.187	0.211	*0.157*	0.082
	Our model	**0.423**	**0.380**	**0.190**	**0.222**	**0.164**	**0.087**
USAToday-CNN	PriorSum	0.214	0.071	0.136	0.242	0.082	0.156
	Our model	**0.216**	**0.074**	**0.140**	**0.247**	**0.089**	**0.158**
VLSCSum	PriorSum	0.543	0.447	0.225	0.318†	0.161†	0.118†
	Our model	**0.547**	**0.449**	**0.228**	**0.352**	**0.193**	**0.153**

Our model vs. non-social context methods We next report the comparison between our model and non-social context methods in Table 6.4.

Table 6.4: Our model vs. basic methods; **bold** is the best value; *italic* is the second best. RG stands for ROUGE.

Dataset	Method	Sentences			User posts		
		RG-1	RG-2	RG-W	RG-1	RG-2	RG-W
SoLSCSum	Lead-m	0.345†	0.322†	0.170	—	—	—
	LexRank	0.327†	0.243†	0.138†	0.210†	0.115†	*0.085*
	SVM*	0.325†	0.263†	0.147†	0.152†	0.089†	0.062†
	CRF*	0.393	0.379	0.187	0.091†	0.075†	0.037†
	Our model	**0.423**	**0.380**	**0.190**	**0.222**	**0.164**	**0.087**
USAToday-CNN	Lead-m	0.249	**0.106**	**0.172**	—	—	—
	LexRank	*0.251*	*0.092*	0.163	0.193†	0.068†	0.128†
	SVM*	**0.261**	**0.106**	*0.171*	0.221	*0.084*	0.149
	CRF*	0.186	0.088	0.114	0.190†	0.065†	0.119†
	Our model	0.216	0.074	0.140	**0.247**	**0.089**	**0.158**
VLSCSum	Lead-m	0.495†	0.420	0.214	—	—	—
	LexRank	0.506†	0.432	0.219	0.348†	0.198†	0.127†
	SVM*	0.497†	0.440	0.208	*0.374*†	0.212†	*0.140*
	CRF*	0.422†	0.357†	0.172†	0.111†	0.062†	0.041†
	Our model	**0.547**	**0.449**	**0.228**	**0.352**	**0.193**	**0.153**

ROUGE-scores from Table 6.4 indicate a consistent trend in which our model is the best in almost all cases, except for sentence selection on USAToday-CNN. For example, our model is bettern than CRF, which is a very strong baseline on SoLSCSum, e.g. 0.423 vs. 0.393. It is similar to VSoLSCSum, in that our model significantly surpasses baselines (values with † are significant with $p \leq 0.05$). This again confirms the efficiency of our model, which employs CNN in capturing local features and takes advantage of social context for providing additional useful information.

On USAToday-CNN, on the one hand, SVM is the best, followed by LexRank for sentence selection. It is possible to explain that features used by SVM are appropriate for this dataset. However, on other datasets, SVM does not prove to be efficient. On USAToday-CNN, our methods are not the best even it uses several sophisticated features. This is because all our features are for extraction but this dataset is for abstraction, which also challenges other advanced methods. For example, ROUGE-scores of RankBoost CCF (Wei and Gao, 2014) are lower than Lead-m (see Table 6.5). This also shows the limitation of CNN in capturing hidden features. On the other hand, for tweet extraction, our method is the best. This is because we enrich vector representation by integrating social information denoted in the form of features for improving the importance estimation.

Our model vs. social context methods We challenged our model by comparing it to advanced methods. Table 6.5 details their ROUGE-scores.

Table 6.5: Our model vs. advanced methods; **bold** is the best value; *italic* is the second best. RG stands for ROUGE.

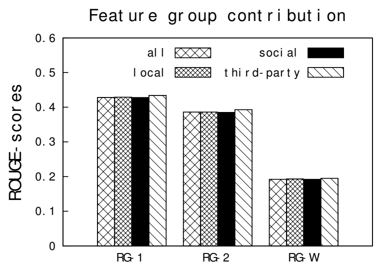
Dataset	Method	Document			Comment		
		RG-1	RG-2	RG-W	RG-1	RG-2	RG-W
SoLSCSum	cc-TAM	0.306†	0.238†	0.136†	0.054†	0.022†	0.024†
	HGRW	0.379†	0.204†	0.167	0.209†	0.115	*0.084*
	RB* CCF	0.360†	0.283†	0.158†	0.190†	0.098†	0.077
	Our model*	**0.423**	**0.380**	**0.190**	**0.222**	**0.164**	**0.087**
USAToday-CNN	cc-TAM	0.229	0.077	0.145	*0.249*	*0.089*	0.152
	HGRW	**0.279**	**0.098**	**0.177**	0.242	0.088	0.157
	RB* CCF	0.221	0.070	0.140	0.233	**0.091**	0.132
	Our model*	0.216	0.074	0.140	**0.247**	**0.089**	**0.158**
VLSCSum	cc-TAM	0.488†	0.377†	0.201	0.301†	0.167	0.111†
	HGRW	**0.570**	*0.479*	*0.233*	*0.454*	*0.298*	**0.173**
	RB* CCF	*0.561*	**0.494**	**0.235**	**0.471**	**0.308**	*0.168*
	Our model*	0.547	0.449	0.228	0.352	0.193	0.153

ROUGE-scores from Table 6.5 are similar to those in Table 6.4, in which our model produces promising results. For example, it is competitive on SoLSCSum and for tweet extraction on USAToday-CNN. Among social context methods, HRGW shows its efficiency. For example, HRGW obtains competitive on these datasets because it extends LexRank to exploit the social information of a Web document in an appropriate form. cc-TAM acquires the second best ROUGE-1 for tweet selection on USAToday-CNN. Results from unsupervised methods motivate that we can improve their quality to reach the performance of supervised learning ones by exploiting social information in a suitable form. RankBoost CCF also achieves consistent results. As discussed, it is trained with sophisticated features leading to improvements compared to unsupervised learning models, e.g. cc-TAM. Our model is not the best on VSoLSCSum because of a possible reason that we can not extract some features from this dataset, e.g. indicator words, due to the limitation of tools in Vietnamese.

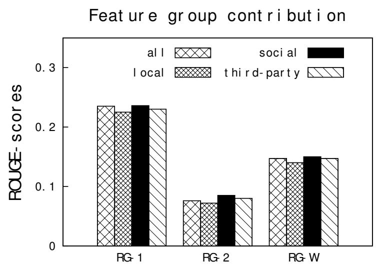
6.6.2 Feature contribution analysis

We observed the contribution of feature groups in our model. To do that, we ran this method with four settings: (i) using all features, (ii) using local features, (iii) using social features, and (iv) using third-party features. It is possible to observe the influence of each feature; however, due to training time, we leave this observation as a future task.

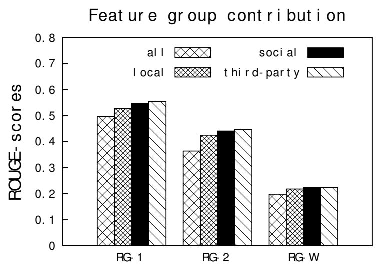
Figures 6.3 and 6.4 show ROUGE-scores of each feature group. The general trend on SoLSCSum and USAToday-CNN indicates that: (i) there are small margins among feature groups and (ii) using all and social features seem to be more efficient than other groups. This is because they include some good indicators, e.g. sentence length, which can support local features extracted by CNN. By contrast, ROUGE-scores on VSoLSCSum



(a) SoLSCSum

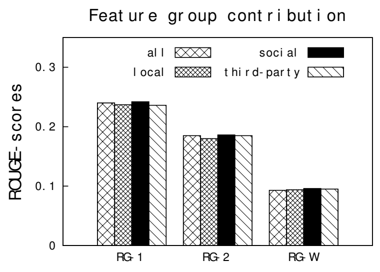


(b) USAToday-CNN

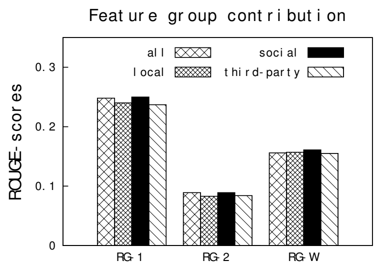


(c) VSoLSCSum

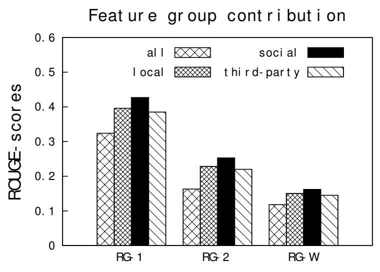
Figure 6.3: Feature group contribution of sentence selection.



(a) SoLSCSum



(b) USAToday-CNN



(c) VSoLSCSum

Figure 6.4: Feature group contribution of user post extraction.

are in a different trend, which indicates that using all features obtains the worst results. This is similar to SoCRF in Section 5.2.7, in that, even using all features, it can not achieve better results than other methods on this dataset. The reason may come from the conflict when combining many features. ROUGE-scores also show that using social and third-party features may be appropriate on this dataset.

6.6.3 ROUGE-scores and sentence selection behaviour

Since CNN-based methods acquire quite similar results on three datasets, except for comment extraction on VSoLSCSum, we, therefore, investigated their selection behavior. This investigation validates an assumption that despite similar ROUGE-scores, two summarizers select different summaries (Hong et al., 2014). On USAToday-CNN, we observed the score of extracted sentences and user posts in each document and its corresponding references and visualized the trend of ROUGE-1 on Figures 6.5 and 6.6. It is possible to report ROUGE-2 and ROUGE-W; however, they are small to plot.

From Figures 6.5 and 6.6 we see that there is variability in ROUGE-1 over different documents. The trend shows that PriorSum is effective for some documents while our model obtains better results for other documents. The reason is masked behind ROUGE-1, in which two methods produce quite different outputs. Note that ROUGE-1 is quite low because references are highlights written by humans whereas two methods are for extractive summarization.

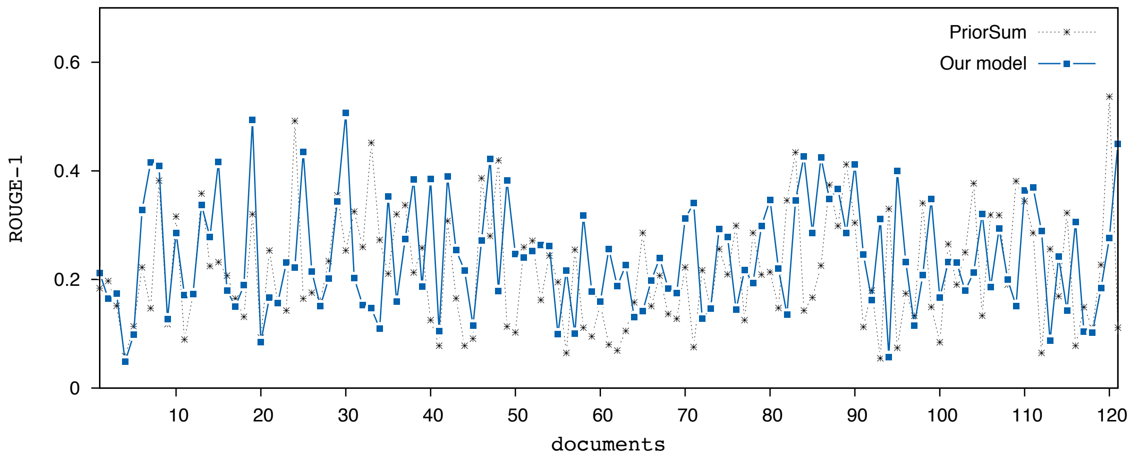


Figure 6.5: ROUGE-1 effectiveness profiles of sentence selection on 121 documents.

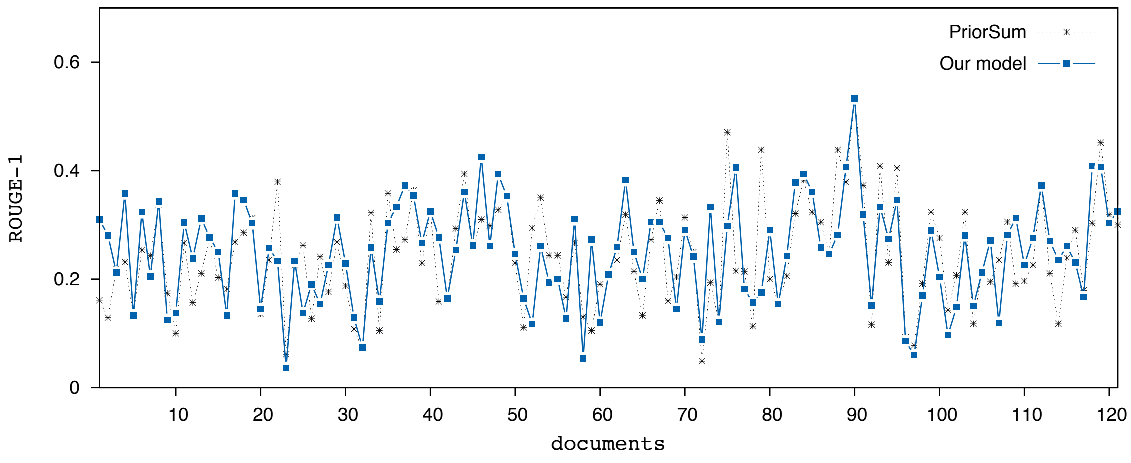
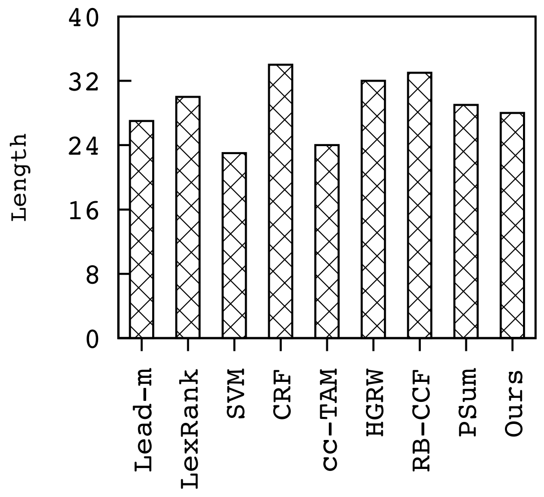


Figure 6.6: ROUGE-1 effectiveness profiles of tweet extraction on 121 documents.

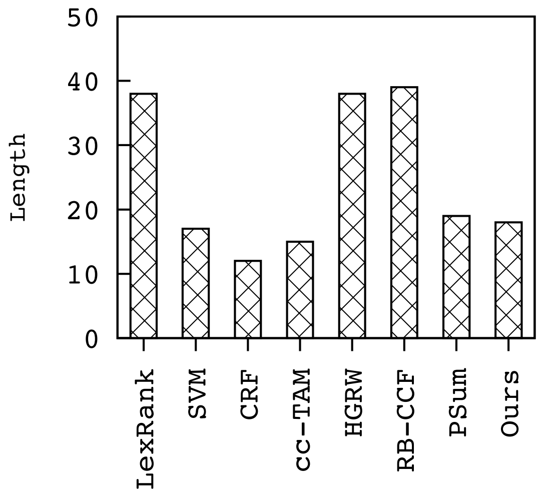
6.6.4 Sentence length observation

We investigated the average length of extracted sentences and comments on SoSLCSum to reveal the relationship between ROUGE-scores and sentence length.

From Figure 6.7 we observe that long sentences belong to sophisticated algorithms whereas weak models output shorter sentences. For sentence selection, CRF and RankBoost CCF select the longest sentences while SVM and cc-TAM select shorter ones. For comment extraction, SVM, CRF, and cc-TAM extract short comments while other methods output longer ones. This explains ROUGE-scores in Tables 6.4 and 6.5, in which SVM, CRF, and cc-TAM obtain poor results for comment extraction. Interestingly, two CNN-based summarizers output shorter sentence length than CRF and RabkBoost CCF but their ROUGE-scores are comparable (Tables 6.4 and 6.5). This reveals the compact aspect of CNN-based summarizers in selecting short sentences while still keeping competitive ROUGE-scores. The observation of CRF in Figures 5.14e and 6.7b also explain that



(a) Sentence selection



(b) Comment extraction

Figure 6.7: The average sentence length on SoLSCSum.

it is inefficient for extracting comments or tweets.

6.6.5 Output observation

We show outputs of CNN-based methods from the document *"Seconds before crash, passengers knew they were too low"* (*Asiana Airlines Flight 214 crash*) in Table 6.6.

We can observe that extracted sentences and tweets provide useful information for readers about this event. This shows the efficiency of two methods in extracting salient information. Their outputs also share common extracted sentences and tweets because they have a quite similar behavior of extraction (see Section 6.6.3). This comes from two reasons: (i) they employ CNN for learning and (ii) some sentences contain important information, e.g. S3 of Ring-CNN, or S1 and S4 of Ring-CNN-F; therefore they are selected by our model. They also extract different outputs, supporting the analysis in Section 6.6.3, in that, two models produce different summaries even their scores are similar.

For sentence selection, PriorSum extracts S2 and S3, which seem to be relevant to the highlights. By reading these sentences, we can partly guess the situation of the event. However, it selects S1, containing information in the past, which may support the main story but it does not directly relate to the event. This is similar to S4, which shows the opinion of Hayes-White. By contrast, outputs of our model may be closer to the highlights than those of PriorSum. For example, we know the total number of passengers (S3 of Ring-CNN) or what happened with flight recorders (S2 of the two models). It clearly provides richer information than PriorSum, which leads to our improvements.

For tweet extraction, two methods output valuable tweets, which include important information. For example, T2 of PriorSum or T3 of Ring-CNN completely match with the first highlight. Ring-CNN-F selects two very important tweets: T1 and T3. While T1 is similar to PriorSum and Ring-CNN, T3 provides valuable information, which completely

Table 6.6: A summary example generated from the document 14th on USAToday-CNN.

Highlights
Flight recorders have been found, the NTSB says.
Asiana identifies the two 16-year-old girls killed in the crash.
182 people were hospitalized, while 123 were uninjured.
Passengers say the plane's rear struck the edge of the runway.
PriorSum Summarization
Sentence selection
S1: In 1993, a crash near South Korea's Mokpo Airport killed 68 of the 116 people on board.
S2: The flight recorders from the plane have been recovered and are on the way to Washington, the NTSB said Sunday.
S3: The Boeing 737-500 went down in poor weather as the plane was attempting its third landing, the Aviation Safety Network said.
S4: "We're lucky there hasn't been a greater loss of life," San Francisco Fire Chief Joanne Hayes-White said.
Tweet extraction
T1: Seconds before crash, passengers knew they were too low - Asiana Airlines Flight 214 was seconds away from landing...
T2: NTSB says Asiana Airlines Flight 214 flight recorders have been found. - @CNN.
T3: That had to be scary! "@cnnbrk: Flight recorders recovered from San Francisco crash site, NTSB says.
T4: Seconds before crash, passengers knew they were too low Look these pics! A terrible plane accident in San Francisco.
Our model
Sentence selection
S1: Perhaps one of the reasons so many people survived Saturday's crash was because the Boeing 777 is built so that everybody can get off the plane within 90 seconds, even if half the doors are inoperable.
S2: The flight recorders from the plane have been recovered and are on the way to Washington, the NTSB said Sunday.
S3: Once the plane fell short of the runway, passengers found themselves on a roller coaster.
S4: Asiana Airlines Flight 214 was seconds away from landing when the passengers sensed something horribly amiss.
Tweet extraction
T1: NTSB says Asiana Airlines Flight 214 flight recorders have been found. - @CNN.
T2: That had to be scary! "@cnnbrk: Flight recorders recovered from San Francisco crash site, NTSB says.
T3: 2 teens killed in San Francisco plane crash; 182 hospitalized - CNN: ABC News2 teens killed in San Francisco.
T4: Seconds before crash, passengers knew they were too low - Asiana Airlines Flight 214 was seconds away from landing...

matches with the second and the third highlight. It includes the status of two teens (*"killed"*), the name of the event (*"San Francisco plane crash"*), and the number of victims (*"182 hospitalized"*). In general, extracted tweets from our model can cover almost important information from the highlights. Two methods also share some common tweets because they have the similar selection behavior.

Extracted tweets possibly include more important information than selected sentences. There are two possible reasons behind this. Firstly, readers usually directly use the title or one of the highlights to create their tweets. They use the title as primary information and add new one as T2 of our model. Readers also extract a sentence from the main document to create their tweets such as T4 of our method. This provides an opportunity that these methods can extract correct information. Secondly, the number of tweets is still large after removing near-duplicate tweets facilitating the training process.

6.7 Conclusion

This chapter introduces a CNN-based model to improve the importance estimation. Our model learns representation of an input sentence by combining tri-grams, which form three feature maps to effectively capture hidden local features. To integrate social context and improve the representation, we use many surface features extracted from three channels: local information, user-generated content, and third-party sources. By combining those with hidden local features learned from data, our model obtains competitive results on three datasets, in two languages, English and Vietnamese. From experimental results, we find that taking advantage of many features profits our model in capturing the representation of sentences and user posts. Our analyses indicate that using all our features is not efficient in all cases. By observing ROUGE-1 of CNN-based summarizers, we confirm that two models with similar ROUGE-scores yield quite different outputs. We also validate that sentence length is an important feature for our summarization task and text summarization.

Chapter 7

Conclusion and Future Work

In this thesis, we study a problem of how to take advantage of social context of a Web document to improve the content selection of single-document summarization. In this chapter, we summarize our study, point out main findings, and discuss future directions.

7.1 Conclusion and Main Findings

The objective of this thesis is to improve the summarization of single Web documents by exploiting and integrating their social context into the summarization process. Our motivation is inspired by the fact that there exist relationships between Web documents and their relevant social information. In order to achieve our objective, we first provide necessary knowledge in Chapters 1 and 2, including the definition of our summarization task, data preparation, and evaluation procedure. We next approach our objective in three directions: unsupervised, supervised, and deep learning.

Chapter 3 presents two unsupervised ranking models to integrate social information for improving the estimation and selection steps. The idea of these models is that we formulate mutual relationships between Web documents and their social context. Our models measure the importance of sentences and user posts by combining intra-relations and inter-relations, which are captured by a rich set of features. We also introduce a new ILP method for re-ranking summary candidates. Experimental results show that these models obtain competitive ROUGE-scores compared to strong methods. For the first model, we find that the number of user posts affects the ranking because it computes a score of a sentence or a user post in a mutual manner. If the number of user posts is small, it does not provide enough supporting information for sentences. By contrast, the scoring step can utilize plenty information if the number of user posts is large. For the second model, we point out that by using semantic similarity, a simple greedy algorithm can obtain high ROUGE-scores and the ILP method is not efficient in all cases. For example, in cases that the greedy strategy achieves high results, adding constraints is unnecessary because they force to eliminate some appropriate candidates. In other cases, adding constraints improves the performance of the selection step. The analysis of important words also indicates that they significantly affect the selection of summaries. This is because our model considers a salient sentence which has to include at least one important word. If the number of these words is small, they do not cover all sentences, leading they are eliminated in by our constraints.

Chapter 4 shows our model to encode common topics between sentences and user posts in a share hidden topic matrix, which is found and analyzed by our matrix co-factorization algorithm. Experimental results show that our matrix co-factorization achieves promising ROUGE-scores compared to several advanced methods. The main finding from our analyses is that our model depends on the number of topics, but changing this number does not significantly influence our ranking algorithm. We hypothesize that this number slight affects the decomposition when computing weights of sentences and user posts by using the share topics. In this aspect, the analysis of topic number in Section 3.2.4 may be beneficial, in which we already indicated that changing the topic number of LDA slightly influence the scoring step. Although the topic number of this model is quite different from LDA, we can still use this analysis to provide better understanding regarding the role of topics in our matrix co-factorization model.

Chapter 5 introduces two L2R models which use many sophisticated indicators extracted from three channels: local, social, and third-party information to estimate the importance of sentences and user posts. Our models learn from these indicators to measure informative information in each sentence and user post in a mutual support fashion. Experimental results indicate that combining sophisticated features from three channels benefits the estimation. An interesting finding is that relevant Web articles of primary documents can also be considered as a type of social information. For selection, besides employing a greedy method, we also introduce our voting to re-rank summary candidates. We find that our voting improves ROUGE-scores of our models.

The analyses show four important points. Firstly, features differently affect sentences and user posts because they are distinct in term of data characteristics. It suggests that different features should be separately used for modeling sentences and user posts. Secondly, the number of relevant documents influences our models because the various amount of such data provides different salient words, which influence several statistical features. In addition, the analysis in Section 5.2.7 indicates that CRF is inefficient for user posts. A possible explanation is that CRF exploits the sequential dependencies of sentences which are unavailable in comments or tweets. Finally, promising ROUGE-scores of L2R models suggest that formulating summarization in the form of L2R benefits the importance estimation.

Chapter 6 investigates a well-known deep learning architecture named CNN and adapts it to our task. Our model improves PriorSum of Cao et al. (2015c) in enriching the representation by integrating social context denoted in form of many advanced features. Under a regression mechanism, our model achieves very competitive results. Results make a surprise that using all features does not help to improve ROUGE-scores because the performance of a summarizer depends on the way of combining its features. The analyses also confirm that two models with similar ROUGE-scores have different selection behavior.

Limitation All our models are inefficient in dealing with informal social messages, i.e. very short, abbreviated, or ungrammatical tweets or comments. This is possibly addressed by integrating a sophisticated pre-processing step. In addition, tweets or comments did not come from the same news sources challenge all our approaches due to content inconsistency between documents and social messages. This can be addressed by using a refined crawling method to capture relevant information from other sources. The performance of our models are also limited if the content of sentences and tweets or comments is highly abstractive, e.g. need an inference. In this case, a more novel approach, e.g. recognizing textual entailment should be considered.

Summarization systems We introduce six models for exploiting the social context of documents to improve importance estimation and selection steps. Here we discuss scenarios of using our models to build summarization systems. In practice, we can utilize our models to create two types of summarization systems. Figure 7.1 visualizes our systems.

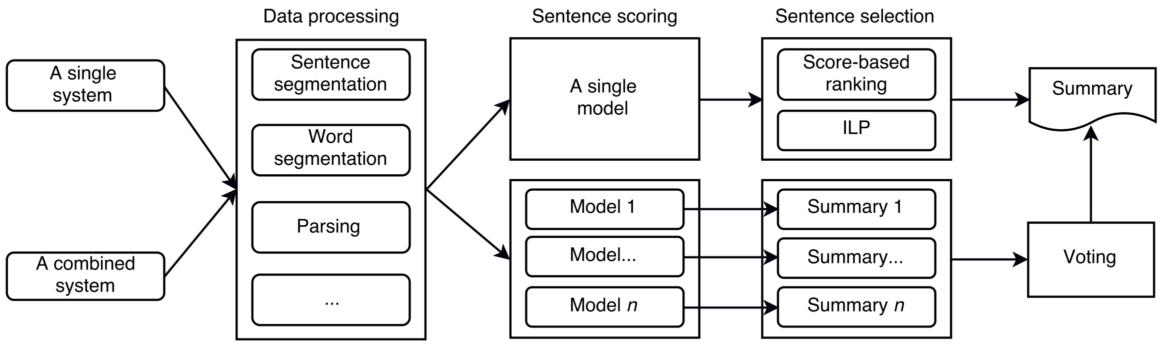


Figure 7.1: The overview of our systems.

The first system (a single system) uses one of our models to extract summaries of input Web documents. It is possible to use any our method because all of them achieve competitive results. However, due to their characteristics, we suggest that in cases if we are in the first stage of building a summarization system or our domain does not have annotated data, unsupervised methods such as SowsRank or NMCF are preferable. This is because they do not need training data. By contrast, if annotated data is available, we suggest that supervised methods such as L2R should be used. In addition, if the amount of annotated data is large, our model based on CNN may be appropriate because it utilizes the characteristics of deep learning, which operates well on big data. For the selection of the first system, score-based ranking or ILP could be used.

The combined system, as its name, combines several our models to extract summaries. Similar to the first scenario, if our domain has no annotated data, we can combine several unsupervised methods for voting. By contrast, if annotated data is available, we can use supervised models such as L2R or DL. Here we also discuss that, in this case, it would be better if we combine models from three directions: unsupervised learning, L2R, and DL. This is because those from these directions have different selection behaviors, which would be beneficial for voting.

To conclude we answer our two research questions stated in Section 1.2. For the first question, our experiments show that the social context of a Web document enriches the summarization of primary documents. Our models and analyses confirm that enrichment is in two levels. First, social information mutually helps to improve the importance estimation of sentences in main documents and user posts in their social context. Second, extracted user posts enhance information in summaries, in that, they provide new information which is not usually available in main documents. For the second question, we point out that there are several ways that we can integrate social information into the summarization process. Among them, formulating summarization in the form of L2R with the support of social information obtains stable and the best results on three datasets in two languages, English and Vietnamese. Promising results of our models also suggest that they can be viable alternative to extraction-based systems.

7.2 Future Work

Based on promising results of this thesis, we discuss four avenues for future research.

Improving the importance estimation of concepts

An obvious direction is to deeply investigate the importance estimation of concepts. In this thesis, we consider a concept as a sentence unit, which is similar to several prior studies in this task (Svore et al., 2007; Wei and Gao, 2014; Yang et al., 2011). The authors present many features, which exploit inherent and social information for estimating the importance of each sentence. Most our proposed features in Nguyen and Nguyen (2016, 2017); Nguyen et al. (2016d, 2017b,c) enrich the estimation task on the sentence level. A small number of our features considers the estimation of uni-grams, which benefit the estimation of sentences. In this aspect, we can adapt the method of Hong and Nenkova (2014), who presented many uni-gram indicators for measuring sentence importance. Also, since all features in this thesis are in the surface level, thus, several indicators can be developed in deeper levels such as dependency trees (Woodsend and Lapata, 2010, 2012), named entities, or production rules, e.g. $NP \rightarrow VB\ VP$.

As shown in Section 3.2.3 we regard a concept as a sentence and use several constraints to select summaries. By redefining concepts as words or phrases such as Woodsend and Lapata (2010, 2012), we can adapt the method of Hong and Nenkova (2014) to estimate the importance of uni-grams or Li et al. (2015, 2013, 2016) fo estimating bi-grams weights, which can be used to compute the importance of sentences. Estimating the importance of small concepts may assist the ranking step because they require deeper analyses than the sentence level. This could improve the importance estimation.

Re-ranking models for selecting summary candidates

As mentioned, we follow the literature in presenting our summarization models in two steps: sentence scoring and sentence selection. While the scoring explores several aspects such as semantic similarity or feature-driven methods, the selection is still in an early stage. In this thesis, we mainly employ a simple greedy method which selects summaries based on their scores. We also do preliminary investigations of re-ranking candidates by presenting an ILP model in Section 3.2.3 and voting in Section 5.2.5, which achieve promising results.

From these results, we argue that re-ranking may benefit the selection. There are three ways we can extend the re-ranking. Firstly, scores from the scoring can be used as features of other ranking models as in Section 5.2.5. This idea is already explored in Shen et al. (2007), in which the authors utilize LSA and HIT score as indicators. The second way is that ILP can be used as a re-ranking method, in that we can define new constraints for capturing concepts of summaries. We contribute our ILP model in Section 3.2.3 to this way. Other models relating to this direction can be investigated (Banerjee et al., 2015; Li et al., 2016; Woodsend and Lapata, 2010, 2012). A more recent paper describes a joint optimization by learning from user feedback (P.V.S and Meyer, 2017). The authors presented an iterative learning algorithm on user feedback to support the concept selection of an ILP model. It differs from prior studies by looping on the feedback to improve the extraction of summaries. Therefore, it is possible to adapt this method to the selection. The third way is that summaries can be extracted by combining several summarization models. This idea is similar to Hong et al. (2015), in which the authors also combine several well-known extractive summarization methods for extracting summaries. Tan et al. (2017b) introduced another variation of this method, in that the authors present a coarse-to-find model for document summarization. They first extract summary candidates by using extractive methods such as LexRank and put them into a sequence-to-sequence model for summarization. In this thesis, we do a preliminary step by using majority voting on three L2R models. Promising results support that combining summarizers potentially improves ROUGE-scores.

Improving representation with deep learning

Since DL achieves promising results in text summarization (Cao et al., 2015b; Cheng and Lapata, 2016; Chopra et al., 2016; Ren et al., 2017; Rush et al., 2015; Zhang et al., 2017), we encourage next studies to adapt this technique to the task of summarization with social context. In this thesis, we basically investigate representation by a CNN-based model. Our model employs CNN combined with rich features to present a sentence. Experimental results confirm the efficiency of our model and our features.

The extension of DL can be in two ways. The first is to exploit user posts in the form of features to enhance the representation of sentences. This idea is usually used for DL, especially on small datasets (Cao et al., 2015b,c; Ren et al., 2017). The authors showed that basic surface features help to improve the representation. Our investigation in Section 6.3 also follows this way. We use many sophisticated features extracted from both Web documents and their social context to support the estimation. Experimental results support this method. Therefore, we suppose that the more appropriate features are extracted, the more summarization performance can be gained. The second one is that we can adapt the attention mechanism for reader-aware aspect summarization. The idea comes from the fact that user posts include viewpoints of readers and summarization based on their viewpoints is a natural approach (Hu et al., 2008). However, there are very little studies which exploit attention for such problem. We guess that the attention mechanism can naturally model this problem and then may improve ROUGE-scores of attention-based summarization systems.

Abstractive summarization

Abstraction is the final venue that we can point out for future work. As mentioned, all our models and state-of-the-art methods in this task (Gao et al., 2012; Wei and Gao, 2014, 2015; Yang et al., 2011) are extraction, facilitating readers in catching main information of a Web document. However, the extraction, as discussed in Section 1.1, eliminates other important information. Therefore, it inspires to move to abstraction, which can deal with both single or multiple documents.

To explore this direction, there are many studies of text summarization that we can adapt (Banerjee et al., 2015; Tan et al., 2017b; Woodsend and Lapata, 2010, 2012). Li et al. (2016) presented a more recent paper which describes a way to exploit user posts. The authors introduce several weighting metrics, which integrate user posts to improve the estimation of bi-grams for extraction and abstraction of single Web documents. We, therefore, can extend this research to do abstraction for multiple Web documents. An interesting issue of abstraction is the combination of sentences and user posts, as shown in Li et al. (2016). In this work, the authors present simple rules to decide whether a summary should include sentences, user posts, or both. We argue that a more sophisticated method of combination should be considered, e.g. classification, which may improve the quality of summarization.

Bibliography

Amitay, E. and Paris, C. (2000). Automatically summarising web sites: is there a way around it? In *Proceedings of the Ninth International Conference on Information and Knowledge Management (CIKM), pp. 173-179. ACM.*

Ani Nenkova, K. M. (2011). Automatic summarization. *Foundations and Trends in Information Retrieval, 5(2-3), pp. 103-233.*

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. In *arXiv preprint arXiv:1409.0473.*

Banerjee, S., Mitra, P., and Sugiyama, K. (2015). Multi-document abstractive summarization using ilp based multi-sentence compression. In *Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI), pp. 1208-1214.*

Bing, L., Li, P., Liao, Y., Lam, W., Guo, W., and Passonneau, R. J. (2015). 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 (ACL-IJCNLP), pp. 1587-1597. Association for Computational Linguistics.*

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *The Journal of Machine Learning Research, 3, pp. 993-1022. MIT Press.*

Cao, Z., Chen, C., Li, W., Li, S., Wei, F., and Zhou, M. (2015a). Tgsum: Build tweet guided multi-document summarization dataset. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI), pp. 2906-2912.*

Cao, Z., Qin, T., Liu, T.-Y., Tsai, M.-F., and Li, H. (2007). Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th International Conference on Machine Learning (ICML), pp. 129-136. ACM.*

Cao, Z., Wei, F., Dong, L., Li, S., and Zhou, M. (2015b). Ranking with recursive neural networks and its application to multi-document summarization. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI), pp. 2153-2159.*

Cao, Z., Wei, F., Li, S., Li, W., Zhou, M., and Wang, H. (2015c). Learning summary prior representation for extractive summarization. In *Proceedings of the 53rd Annual Meeting* *of the Association for Computational Linguistics (2) (ACL), pp. 829-833. Association for Computational Linguistics.*

Cheng, J. and Lapata, M. (2016). Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL), pp 484-494, Association for Computational Linguistics.*

Chopra, S., Auli, M., and Rush, A. M. (2016). Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of The 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), pp. 93-98. Association for Computational Linguistics.*

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research 12, no. Aug, pp. 2493-2537. MIT Press.*

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning 20(3), pp. 273-297. Springer.*

Delort, J.-Y. (2006). Identifying commented passages of documents using implicit hyperlinks. In *Proceedings of the Seventeenth Conference on Hypertext and Hypermedia (Hypertext), pp. 89-98. ACM.*

Delort, J.-Y., Bouchon-Meunier, B., and Rifqi, M. (2003). Enhanced web document summarization using hyperlinks. In *Proceedings of the fourteenth ACM Conference on Hypertext and Hypermedia (Hypertext), pp. 208-215. ACM.*

Erkan, G. and Radev, D. R. (2004). Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research, 22, pp. 457-479. AAAI Press.*

Freund, Y., Lyeryer, R. D., Schapire, R. E., and Singer, Y. (2003). An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research 4, pp. 933-969. MIT Press.*

Freund, Y. and Schapire, R. E. (1995). A decision-theoretic generalization of online learning and an application to boosting. *Journal of Computer and System Sciences, 55(1), pp. 119-139.*

Gao, W., Li, P., and Darwish, K. (2012). Joint topic modeling for event summarization across news and social media streams. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM), pp. 1173-1182. ACM.*

Gong, Y. and Liu, X. (2001). Generic text summarization using relevant measure and latent semantic analysis. In *Proceedings of the 24th Annual International ACM SIGIR*

Conference on Research and Development in Information Retrieval (SIGIR), pp. 19-25. ACM.

Graves, A., Mohamed, A.-R., and Hinton, G. E. (2013). Speech recognition with deep recurrent neural networks. In *Acoustics, Speech and Signal processing (ICASSP), 2013 IEEE International Conference, pp. 6645-6649. IEEE.*

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9(8), pp. 1735-1780. MIT Press.*

Hong, K., Conroy, J. M., Favre, B., Kulesza, A., Lin, H., and Nenkova, A. (2014). A repository of state of the art and competitive baseline summaries for generic news summarization. In *The Ninth International Conference on Language Resources and Evaluation (LREC), pp. 1608-1616.*

Hong, K., Marcus, M., and Nenkova, A. (2015). System combination for multi-document summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 107-117, Association for Computational Linguistics.*

Hong, K. and Nenkova, A. (2014). Improving the estimation of word importance for news multi-document summarization. In *The 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL), pp. 712-721. Association for Computational Linguistics.*

Hu, M., Sun, A., and Lim, E.-P. (2008). Comments-oriented document summarization: Understanding document with readers' feedback. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), pp. 291-298. ACM.*

Joachims, T. (2006). Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pp. 217-226. ACM.*

Kan, M.-Y., McKeown, K. R., and Klavans, J. L. (2001). Domain-specific informative and indicative summarization for information retrieval. In *Proceedings of the Document Understanding Conference (DUC), pp. 19-26.*

Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1746-1751. Association for Computational Linguistics.*

Lafferty, J. D., McCallum, A., and Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML), pp. 282-289.*

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE, 86(11), pp. 2278-2324. IEEE.*

Lee, D. D. and Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature 401, no. 6755, pp. 788-791.*

Lee, D. D. and Seung, H. S. (2001). Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems (NIPS), pp. 556-562.*

Lee, J.-H., Park, S., Ahn, C.-M., and Kim, D. (2009). Automatic generic document summarization based on non-negative matrix factorization. *Information Processing & Management, 45(1), pp. 20-34.*

Li, C., Liu, Y., and Zhao, L. (2015). Using external resources and joint learning for bigram weighting in ilp-based multi-document summarization. In *Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the ACL (NAACL-HLT), pp. 778-787. Association for Computational Linguistics.*

Li, C., Qian, X., and Liu, Y. (2013). Using supervised bigram-based ilp for extractive summarization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL), pp. 1004-1013. Association for Computational Linguistics.*

Li, C., Wei, Z., Liu, Y., Jin, Y., and Huang, F. (2016). Using relevant public posts to enhance news article summarization. In *Proceedings of the 26th International Conference on Computational Linguistics (COLING), pp. 557-566. Association for Computational Linguistics.*

Lin, C.-J. (2007). Projected gradient methods for nonnegative matrix factorization. *Neural computation, 19(10), pp. 2756-2779. MIT Press.*

Lin, C.-Y. and Hovy, E. H. (2003). Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1 (NAACL-HLT), pp. 71-78. Association for Computational Linguistics.*

Lin, H. and Bilmes, J. A. (2011). A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1, pp. 510-520. Association for Computational Linguistics.*

Liu, T.-Y. (2011). *Learning to Rank for Information Retrieval.* Springer, ISBN 978-3-642-14266-6, pp. I-XVII, 1-285.

Lu, Y., Zhai, C., and Sundaresan, N. (2009). Rated aspect summarization of short comments. In *Proceedings of the 18th International Conference on World Wide Web (WWW), pp. 131-140. ACM.*

Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM Journal of Research Development, 2(2), pp. 159-165.*

Luong, M.-T., Pham, H., and Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1412-1421. Association for Computational Linguistics.*

Metzler, D. and Croft, W. B. (2007). Linear feature-based models for information retrieval. *Information Retrieval, 10(3), pp. 257-274. Springer.*

Mihalcea, R. and Tarau, P. (2004). Textrank: Bringing order into texts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 404-411. Association for Computational Linguistics.*

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems (NIPS), pp. 3111-3119.*

Nakov, P., Popova, A., and Mateev, P. (2001). Weight functions impact on lsa performance. In *Proceedings of EuroConference Recent Advances in Natural Langueage Processing (RANLP), pp. 187-193.*

Nenkova, A. (2005). Automatic text summarization of newswire: lessons learned from the document understanding conference. In *Proceedings of The Twentieth National Conference on Artificial Intelligence (AAAI), vol. 5, pp. 1436-1441.*

Nenkova, A. and Passonneau, R. (2004). Evaluating content selection in summarization: The pyramid method. In *Proceedings of the 2004 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT), pp. 145-152. Association for Computational Linguistics.*

Nguyen, M.-T., Cuong, T. V., Hoai, N. X., and Nguyen, M.-L. (2017a). Utilizing user posts to enrich web document summarization with matrix co-factorization. In *Proceedings of the Eigth International Symposium on Information and Communication Technology (SoICT). ACM.*

Nguyen, M.-T., Ha, Q.-T., Nguyen, T.-D., Nguyen, T.-T., and Nguyen, L.-M. (2015). Recognizing textual entailment in vietnamese text: An experimental study. In *The Seventh International Conference on Knowledge and Systems Engineering (KSE), pp. 108-113. IEEE.*

Nguyen, M.-T., Lai, V. D., Do, P.-K., Tran, D.-V., and Nguyen, M.-L. (2016a). Vsolscsum: Building a vietnamese sentence-comment dataset for social context summarization. In *The 12th Workshop on Asian Language Resources, pp. 38-48. Association for Computational Linguistics.*

Nguyen, M.-T. and Nguyen, M.-L. (2016). Sortesum: A social context framework for single-document summarization. In *Proceedings of the 38th European Conference on Information Retrieval (ECIR), pp. 3-14. Springer.*

Nguyen, M.-T. and Nguyen, M.-L. (2017). Intra-relation or inter-relation?: Exploiting social information for web document summarization. *Expert Systems with Applications 76, pp. 71-84. Elsevier.*

Nguyen, M.-T., Phan, V.-A., Nguyen, T.-S., and Nguyen, M.-L. (2016b). Learning to rank questions for community question answering with ranking svm. In *CoRR abs/1608.04185.*

Nguyen, M.-T., Tran, C.-X., Tran, D.-V., and Nguyen, M.-L. (2016c). Solscsum: A linked sentence-comment dataset for social context summarization. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (CIKM), pp. 2409-2412. ACM.*

Nguyen, M.-T., Tran, D.-V., and Nguyen, M. L. (2018a). Exploiting user posts for web document summarization. *Transactions on Knowledge Discovery from Data (TKDD). ACM.*

Nguyen, M.-T., Tran, D.-V., and Nguyen, M. L. (2018b). Social context summarization using user-generated content and third-party sources. *Knowledge-Based Systems. Elsevier.*

Nguyen, M.-T., Tran, D.-V., Tran, C.-X., and Nguyen, M.-L. (2016d). Learning to summarize web documents using social information. In *Proceedings of the 28th International Conference on Tools with Artificial Intelligence (ICTAI), pp. 619-626. IEEE.*

Nguyen, M.-T., Tran, D.-V., Tran, C.-X., and Nguyen, M.-L. (2017b). Summarizing web documents using sequence labeling with user-generated content and third-party sources. In *Proceedings of 22nd International Conference on Applications of Natural Language to Information Systems (NLDB), pp. 454-467. Springer.*

Nguyen, M.-T., Tran, D.-V., Tran, X.-C., and Nguyen, M.-L. (2017c). Exploiting user-generated content to enrich web document summarization. *International Journal on Artificial Intelligence Tools (IJAIT), 26(5), pp. 1-26. World Scientific.*

Park, S., Lee, J.-H., Ahn, C.-M., Hong, J. S., and Chun, S.-J. (2006). Query based summarization using non-negative matrix factorization. In *Knowledge-Based Intelligent Information and Engineering Systems (KSE), pp. 84-89. Springer Berlin/Heidelberg.*

P.V.S, A. and Meyer, C. M. (2017). Joint optimization of user-desired content in multi-document summaries by learning from user feedback. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 1353-1363. Association for Computational Linguistics.*

Ren, P., Chen, Z., Ren, Z., Wei, F., Ma, J., and de Rijke, M. (2017). Leveraging contextual sentence relations for extractive summarization using a neural a ention model. In *Proceedings of the 40th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), pp. 95-104. ACM.*

Rush, A. M., Chopra, S., and Weston, J. (2015). A neural attention model for sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 379-389. Association for Computational Linguistics.*

Scherer, D., Muller, A., and Behnke, S. (2010). Evaluation of pooling operations in convolutional architectures for object recognition. In *Proceedings of the 20th International Conference on Artificial Neural Networks (ICANN), pp. 92-101. Springer.*

Sha, F. and Pereira, F. (2003). Shallow parsing with conditional random fields. In *Proceedings of the North American Chapter of the Association for Computational Linguistics-Human Language Technology (NAACL-HLT) pp. 134-141. Association for Computational Linguistics.*

Shen, D., Sun, J.-T., Li, H., Yang, Q., and Chen, Z. (2007). Document summarization using conditional random fields. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI), vol. 7, pp. 2862-2867.*

Sun, J.-T., Shen, D., Zeng, H.-J., Yang, Q., Lu, Y., and Chen, Z. (2005). Web-page summarization using clickthrough data. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), pp. 194-201. ACM.*

Sundermeyer, M., Alkhouli, T., Wuebker, J., and Ney, H. (2014). Translation modeling with bidirectional recurrent neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 14-25. Association for Computational Linguistics.*

Svore, K. M., Vanderwende, L., and Burges, C. J. C. (2007). Enhancing single-document summarization by combining ranknet and third-party sources. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pp. 448-457. Association for Computational Linguistics.*

Takase, S., Suzuki, J., Okazaki, N., Hirao, T., and Nagata., M. (2016). Neural headline generation on abstract meaning representation. In *In Proceedings of the 2016 Confer-*

ence on Empirical Methods in Natural Language Processing (EMNLP), pp. 1054-1059. Association for Computational Linguistics.

Tan, J., Wan, X., and Xiao, J. (2017a). Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 1171-1181. Association for Computational Linguistics.*

Tan, J., Wan, X., and Xiao, J. (2017b). From neural sentence summarization to headline generation: A coarse-to-fine approach. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI), pp. 4109-4115.*

Wang, D., Li, T., Zhu, S., and Ding, C. (2008). Multi-document summarization via sentence-level semantic analysis and symmetric matrix factorization. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), pp. 307-314. ACM.*

Wei, Z. and Gao, W. (2014). Utilizing microblogs for automatic news highlights extraction. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING), pp. 872-883. Association for Computational Linguistics.*

Wei, Z. and Gao, W. (2015). 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 (SIGIR), pp. 1003-1006. ACM.*

Woodsend, K. and Lapata, M. (2010). Automatic generation of story highlights. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 565-574. Association for Computational Linguistics.*

Woodsend, K. and Lapata, M. (2012). Multiple aspect summarization using integer linear programming. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pp. 233-243. Association for Computational Linguistics.*

Yang, Z., Cai, K., Tang, J., Zhang, L., Su, Z., and Li, J. (2011). Social context summarization. In *Proceedings of the 34th International SIGIR Conference on Research and Development in Information Retrieval (SIGIR), pp. 255-264. ACM.*

Yeh, J.-Y., Ke, H.-R., Yang, W.-P., and Meng, I.-H. (2005). Text summarization using a trainable summarizer and latent semantic analysis. *Information Processing & Management 41(1), pp. 75-95. Elsevier.*

Zhang, Y., Er, M. J., Zhao, R., and Pratama, M. (2017). Multiview convolutional neural networks for multidocument extractive summarization. *IEEE Transactions on Cybernetics, Vol 47(10), pp. 3230-3242. IEEE.*

Zitouni, I., editor (2014). *Natural Language Processing of Semitic Languages.* Springer.

Publications and Awards

Journals

[1] Minh-Tien Nguyen, Duc-Vu Tran, and Minh-Le Nguyen, **Exploiting User Posts for Web Document Summarization**, ACM Transactions of Knowledge Discovery from Data (TKDD) (accepted). ACM.

[2] Minh-Tien Nguyen, Duc-Vu Tran, and Minh-Le Nguyen, **Social Context Summarization using User-generated Content and Third-party Sources**, Knowledge-Based Systems (KNOSYS), 144(2018), pp. 51-64, December 2017. Elsevier.

[3] Minh-Tien Nguyen, Duc-Vu Tran, Xuan-Chien Tran, and Minh-Le Nguyen, **Exploiting User-generated Content to Enrich Web Document Summarization**, International Journal on Artificial Intelligence Tools (IJAIT), 26(5), pp. 1-26, October 2017. World Scientific.

[4] Minh-Tien Nguyen and Minh-Le Nguyen, **Intra-relation or Inter-relation?: Exploiting Social Information for Web Document Summarization**, Expert Systems with Applications (ESWA), 76, pp. 71-84, January 2017. Elsevier.

International Conferences

[5] Minh-Tien Nguyen, Duc-Vu Tran, Viet-Anh Phan, and Minh-Le Nguyen, **Towards Social Context Summarization with Convolutional Neural Networks**, 19th International Conference on Computational Linguistics and Intelligent Text Processing (CICLING) (accepted), March 2018. Springer.

[6] Minh-Tien Nguyen, Viet-Cuong Tran, Xuan-Hoai Nguyen, and Minh-Le Nguyen, **Utilizing User Posts to Enrich Web Document Summarization with Matrix Co-factorization**, The Eighth International Symposium on Information and Communication Technology (SoICT), pp. 70-77, December 2017. ACM (best paper award).

[7] Minh-Tien Nguyen, Duc-Vu Tran, Chien-Xuan Tran, and Minh-Le Nguyen, **Summarizing Web Documents using Sequence Labeling with User-generated Content and Third-party Sources**, 22nd International Conference on Natural Language & Information Systems (NLDB), pp. 454-467, June 2017. Springer.

[8] Minh-Tien Nguyen, Duc-Vu Tran, Chien-Xuan Tran, and Minh-Le Nguyen , **Learning to Summarize Web Documents using Social Information**, IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI), pp. 619-626, November 2016. IEEE.

[9] Minh-Tien Nguyen, Chien-Xuan Tran, Duc-Vu Tran, and Minh-Le Nguyen , **SoLSCSum: A Linked Sentence-Comment Dataset for Social Context Summarization**, 25th ACM International on Conference on Information and Knowledge Management (CIKM), pp. 2409-2412, October 2016. ACM.

[10] Minh-Tien Nguyen and Minh-Le Nguyen , **SoRTESum: A Social Context Framework for Single-Document Summarization**, 38th European Conference on Information Retrieval (ECIR), pp. 3-14, March 2016. Springer.

[11] Minh-Tien Nguyen, Quang-Thuy Ha, Thi-Dung Nguyen, Tri-Thanh Nguyen, and Minh-Le Nguyen, **Recognizing Textual Entailment in Vietnamese Text: An Experimental Study**, The Seventh International Conference on Knowledge and Systems Engineering (KSE), pp. 108-113, October 2015. IEEE.

Peer-reviewed International Workshops

[12] Minh-Tien Nguyen, Viet Dac Lai, Phong-Khac Do, Duc-Vu Tran, and Minh-Le Nguyen, **VSoLSCSum: Building a Vietnamese Sentence-Comment Dataset for Social Context Summarization**, In The 12th Workshop on Asian Language Resources associated with 26th International Conference on Computational Linguistics (COLING), pp. 38-48, December 2016. Association for Computational Linguistics.

[13] Khac-Phong Do, Huy-Tien Nguyen, Chien-Xuan Tran, Minh-Tien Nguyen, and Minh-Le Nguyen, **Legal Question Answering using Ranking SVM and Deep Convolutional Neural Network**, Tenth International Workshop on Juris-informatics (JURISIN 2016) associated with JSAI International Symposia on AI 2016 (IsAI-2016), October 2016.

[14] Minh-Tien Nguyen, Viet-Anh Phan, Truong-Son Nguyen, and Minh-Le Nguyen, **Learning to Rank Questions for Community Answering with Ranking SVM**, ECML/PKDD 2016 Discovery Challenge: Learning to Re-Rank Questions for Community Question Answering, August 2016.

Related papers which are not in the thesis

[15] Danilo Carvalho,Minh-Tien Nguyen, Chien-Xuan Tran, and Minh-Le Nguyen, **Lexical-Morphological Modeling for Legal Text Analysis**, JSAI International Symposium on Artificial Intelligence, pp. 295-311, March 2016. Springer.

[16] Minh-Tien Nguyen, Dac Viet Lai, Huy-Tien Nguyen, and Minh-Le Nguyen, **TSix: A Human-involved-creation Dataset for Tweet Summarization**, 11th Edition of the Language Resources and Evaluation Conference (LREC) (accepted), May 2018.

Awards

- The best paper award of The Eighth International Symposium on Information and Communication Technology (SoICT), December, 2017.
- SIGIR Student travel award, 25th ACM International on Conference on Information and Knowledge Management (CIKM), October 2016.
- The second best system at ECML/PKDD 2016 Discovery Challenge: Learning to Re-Rank Questions for Community Question Answering, August 2016.
- Student travel award, 38th European Conference on Information Retrieval (ECIR), March 2016.