GENERAL UNSUPERVISED EXPLANATORY OPINION MINING FROM TEXT DATA

BY

HYUN DUK KIM

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2013

Urbana, Illinois

Abstract

Due to the abundance and rapid growth of opinionated data on the Web, research on opinion mining and summarization techniques has received a lot of attention from industry and academia. Most previous studies on opinion summarization have focused on predicting sentiments of entities and aspect-based rating for the entities. Although existing techniques can provide general overview of opinions, they do not provide detailed explanation of the underlying reasons of the opinions. Therefore, people still need to read through the classified opinionated comments to find out why people expressed those opinions.

To overcome this challenge, we propose a series of works in general unsupervised explanatory opinion mining from text data. We propose three new problems for further summarizing and understanding explanatory opinions and general unsupervised solutions for each problem. First, we propose (1) Explanatory Opinion Summarization (EOS) summarizing opinions that can explain a particular polarity of sentiment. EOS aims to extract explanatory text segments from input opinionated texts to help users better understand the detailed reasons of the sentiment. We propose several general methods to measure explanatoriness of text and identify explanatory text segment boundary. Second, we propose (2) Contrastive Opinion Summarization (COS) summarizing opinions that can explain mixed polarities. COS extracts representative and contrastive opinions from opposing opinions. By automatically pairing and ranking comparative opinions, COS can provide better understanding of contrastive aspects from mixed opinions. Third, we consider temporal factor of text analysis and propose (3) Causal Topic Mining summarizing opinions that can explain an external time series data. We first propose a new information retrieval problem using time series as a query whose goal is to find relevant documents in a text collection of the same time period, which contain topics that are correlated with the query time series. Second, beyond causal documents retrieval, we propose Iterative Topic Modeling with Time Series Feedback (ITMTF) framework that mines causal topics by jointly analyzing text and external time-series data. ITMTF naturally combines any given probabilistic topic model with causal analysis techniques for time series data such as Granger Test to discover topics that are both coherent semantically and correlated with time series data.

Proposed techniques have been shown to be effective and general enough to be applied for potentially many interesting applications in multiple domains, such as business intelligence and political science, with minimum human supervision.

To my parents.

Acknowledgments

First of all, I would like to express my deepest gratitude to my advisor, Prof. ChengXiang Zhai, for all his help for the entire doctoral study. He has always been a great mentor for my academic life, and he supported me and gave me a lot of inspiration of research. Without his considerate guidance, this dissertation could not be done.

I also want to acknowledge my doctoral committee members, Prof. Jiawei Han, Prof. Kevin Chen-Chuan Chang, and Dr. Meichun Hsu, for their insightful guidance and constructive suggestions for this dissertation.

I want to express my great thanks to HP Labs for internships and funding for our research collaboration. With many researchers in HP Labs, Dr. Meichun Hsu, Dr. Malu Castellanos, Carlos Alberto Ceja Limn, Riddhiman Ghosh,and Dr. Umeshwar Dayal, we performed many fruitful research studies.

I have received much help from many collaborators, colleagues, and friends. Prof. Daniel Diermeier, Prof. Thomas Rietz, Prof. Indranil Gupta, and Prof. Thomas Huang helped me broaden my research horizon. I would like to express my thanks to the members of the TIMan Group, DAIS Group, and other friends for the their valuable discussions and supports, especially, Dae Hoon Park, Yue Lu, Danila Nikitin, V.G.Vinod Vydiswaran, Kavita Ganesan, Duo Zhang, Hongning Wang, Yuanhua Lv, Parikshit Sondhi, Huizhong Duan, Yanen Li, Liangliang Cao, Brian Cho, Min-Hsuan Tsai, Zhen Li, Sangkyum Kim, Hyungsul Kim, Tim Weninger, Wooil Kim, and Inwook Hwang.

I am grateful to other funding supports for my doctoral study from the computer science department of University of Illinois at Urbana-Champaign (UIUC), National Science Foundation (NSF), Department of Homeland Security (DHS), and Korea Foundation for Advanced Studies (KFAS).

Finally, I would like to thank my parents, grand parents, sisters, and all other family members for their endless love and strong support for my study and career. Without their encouragement and help, I could not have reached this far.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

EOS	Explanatory Opinion Summarization.
ESE	Explanatory Sentence Extraction.
CEOS	Compact Explanatory Opinion Summarization.
COS	Contrastive Opinion Summarization.
ITMTF	Iterative Topic Modeling with Time Series Feedback.
PLSA	Probabilistic Latent Semantic Analysis.
LDA	Latent Dirichlet Allocation.
MAP	Mean Average Precision.
wMAP	Weighted Mean Average Precision.
NDCG	Normalized Discounted Cumulative Gain.
TF	Term Frequency.
IDF	Inverse Document Frequency.
SentLRPoisson	Sentence Likelihood Ratio with Poisson Length Modeling.
SentLR	Sentence Likelihood Ratio.
SumWordLR	Sum of Word Likelihood Ratio.
HMM	Hidden Markov Model.
EC@k	Explanatory Characters at k characters.
EP@k	Explanatory Phrases at k characters.
SLR	Segment Likelihood Ratio.
SVM	Support Vector Machine.
WO	Word Overlap.
SEM	Semantic Word Matching.
RF	Representativeness-First.
CF	Contrastiveness-First.
DTW	Dynamic Time Warping.
AC	Average Correlation.

Chapter 1

Introduction

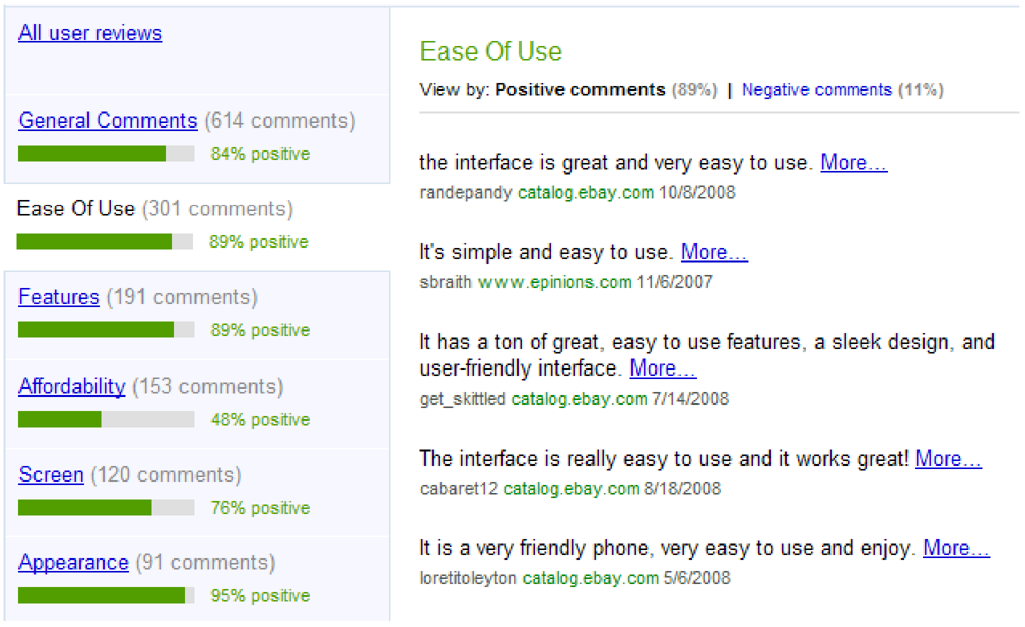
1.1 Background

The Web 2.0 environment results in vast amounts of text data published daily. People can now easily express opinions on various topics through platforms such as blog spaces, forums, and dedicated opinion websites. Since there is usually a large amount of opinionated text about a topic, users often find it challenging to efficiently digest all the opinions.

The abundance and rapid growth of opinionated data available on the Web has fueled a line of research on opinion mining and summarization techniques that has received a lot of attention from industry and academia. Most previous studies on opinion summarization have focused on predicting sentiments of entities and aspect-based rating for the entities, but they do not provide a detailed explanation of the underlying reasons of the opinions. For example, Figure 1.1 shows a part of a sample review summary generated using a state-of-the-art aspect-based (feature-based) opinion summarization technique [28, 56]. In such an opinion summary, a user can see the general sentiment distribution for each product aspect, and furthermore, as shown in the figure, a user can also see a list of positive comments about a specific aspect (i.e., "ease of use"). Negative sentences are also available via another tab on the top.

1.2 Challenges

Although these existing techniques can show the general opinion distribution, as shown for the aspect 'easy of use' (89% positive and 11% negative opinions), they cannot provide the underlying reasons why people have positive or negative opinions about the product. Therefore, even if such an opinion summarization technique is available, people would still need to read through the classified opinionated comments in both the positive and negative groups to find out why people expressed those opinions. This discovery task can be rather cumbersome and time consuming, and therefore it needs to be automated. We need techniques that can further summarize opinions and provide concise and detailed explanatory information from opinions. However, there are challenges in further analyzing and summarizing opinions.


All user reviews

General Comments (614 comments)
84% positive

Ease Of Use (301 comments)
89% positive

Features (191 comments)
89% positive

Affordability (153 comments)
48% positive

Screen (120 comments)
76% positive

Appearance (91 comments)
95% positive

Ease Of Use

View by: **Positive comments** (89%) | Negative comments (11%)

the interface is great and very easy to use. More...
randepandy catalog.ebay.com 10/8/2008

It's simple and easy to use. More...
sbraith www.epinions.com 11/6/2007

It has a ton of great, easy to use features, a sleek design, and user-friendly interface. More...
get_skittled catalog.ebay.com 7/14/2008

The interface is really easy to use and it works great! More...
cabaret12 catalog.ebay.com 8/18/2008

It is a very friendly phone, very easy to use and enjoy. More...
loretitoleyton catalog.ebay.com 5/6/2008

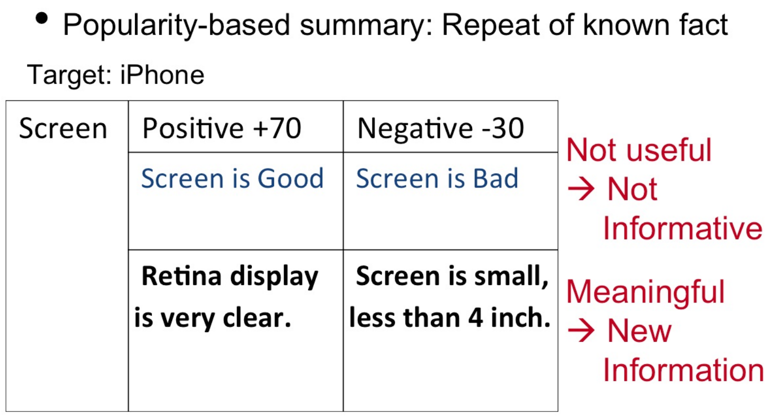
Figure 1.1: A sample state-of-the-art opinion summary (http://search.live.com/products/).

1.2.1 Not Informative Summary

Although general automatic summarization techniques may be used to shrink the size of text to read, they generally extract sentences based on 'popularity'. As a result, the output summary tends to cover already known information. For example, for the summary request for "positive opinions about iPhone screen", a pure popularity-based summary could be 'screen is good', as shown in the second row of Figure 1.2. Given that the sentences to be summarized are already known to be about "positive opinions about iPhone screen", such a summary is obviously **redundant and does not give any additional information to explain the reason** why a positive opinion about iPhone screen is held. In contrast, ideally, we would like a summary to contain sentences such as 'Retina display is very clear', which would be more explanatory and more useful for users understand the reason of the opinions. That is, useful explanatory sentences, such as those in the last row of Figure 1.2, should not only be relevant to the target topic we are interested in, but also include details explaining reasons of sentiments which are not redundant to the target topic itself.

1.2.2 Mixed and Contradictory Opinion

The fact that opinionated text often contains both positive and negative opinions about a topic makes it even harder to accurately digest mixed opinions. For example, some customers may say positive things about the battery life of iPhone, such as "the battery life [of iPhone] has been excellent," but others might say the opposite, such as "I can tell



- Popularity-based summary: Repeat of known fact

Target: iPhone

Screen	Positive +70	Negative -30
	Screen is Good	Screen is Bad
	Retina display is very clear.	**Screen is small, less than 4 inch.**

Not useful → Not Informative

Meaningful → New Information

Figure 1.2: Popularity-based vs. explanatory summary.

you that I was very disappointed with the 3G [iPhone] battery life."[1] Often such **contradictory opinions** are not caused by poor or wrong judgments of people, but due to the different context or perspective taken to make the judgments. For example, if a positive comment is 'the battery life is good when I rarely use button' and a negative comment is 'the battery life is bad when I use button a lot', the two comments are really made under different conditions. When there are many such contradictory opinions expressed about a topic, a user would need to understand what the major positive opinions are, what the major negative opinions are, why these people have different opinions, and how we should interpret these contradictory opinions.

1.2.3 Joint Analysis with External Temporal Factor

Most existing text and opinion analysis focus on text alone. However, text analysis often should be considered **in conjunction with other variables through time**. Stock price is one of the most representative variables reflecting people's opinion about a company. Opinion about a product would also affect sales revenue of the product. We may even want to find out the reason of changes in a numerical opinion curve such as review rating.

Such data calls for an integrated analysis of text and non-text time series data. The causal relationships between the two may be of particular interest. For example, news about companies can affect stock prices. Researchers may be interested in how particular topics lead to increasing or decreasing prices and use the relationships to forecast future price changes. Similar examples occur in many domains. Companies may want to understand how product sales rise and fall in response to text such as advertising or product reviews. Understanding the causal relationships can improve future sales strategies. In election campaigns, analysis of news may explain why a candidate's support has risen or dropped significantly in the polls. Understanding the causal relationships can improve future campaign strategies.

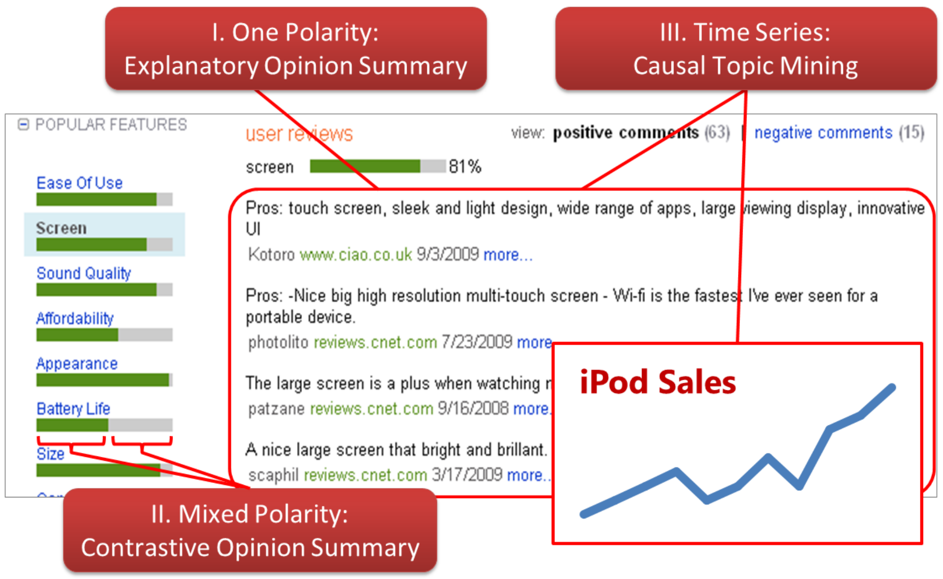
Finding explanatory and causal topics with consideration of time factor would give us much more powerful tool

[1]These sentences are real examples found by the Products Live Search portal at http://search.live.com/products/.

analysis text. While there are many variants of topic analysis models [8, 9, 87], no existing model incorporates jointly text and ‘external’ time series variables in search of causal topics.

1.3 General Unsupervised Explanatory Opinion Mining from Text Data

This dissertation focuses on studies of general unsupervised methods to mine explanatory opinion that shows more detailed reasons of opinion from text. Along this line, we propose three directions to extract explanatory information: (1) summarizing opinions that can explain a particular polarity of sentiment by measuring explanatoriness of text and extracting explanatory phrase, (2) summarizing opinions that can explain mixed polarities by extracting representative and contrastive opinions from opposing opinions. We further add a temporal factor and propose (3) summarizing opinions that can explain an external time series data by mining causal topics correlated with the external time series data. High-level overview of the dissertation is in Figure 1.3, and following are more details of each direction.



Pros: touch screen, sleek and light design, wide range of apps, large viewing display, innovative UI
Kotoro www.ciao.co.uk 9/3/2009 more...

Pros: -Nice big high resolution multi-touch screen - Wi-fi is the fastest I've ever seen for a portable device.
photolito reviews.cnet.com 7/23/2009 more

The large screen is a plus when watching r
patzane reviews.cnet.com 9/16/2008 more

A nice large screen that bright and brillant.
scaphil reviews.cnet.com 3/17/2009 more...

Figure 1.3: Dissertation overview.

Explanatory Opinion Summarization: In this work, we propose a novel opinion summarization problem called explanatory opinion summarization (EOS) which aims to extract explanatory text segments from input opinionated texts to help users better understand the detailed reasons of sentiments.

To solve the problem, we first present a sentence ranking problem called unsupervised explanatory sentence extraction (ESE) which aims to rank sentences in opinionated text based on their explanatoriness to help users better understand the detailed reasons of sentiments. We propose and study several general methods for scoring the explanatoriness of a sentence. We create new data sets and propose a new evaluation measure to evaluate this new task.

Experiment results show that the proposed methods are effective for ranking sentences by explanatoriness and also useful for generating an explanatory summary, outperforming a state of the art sentence ranking method for a standard text summarization method.

Second, beyond sentence level ranking, we propose a novel opinion summarization problem called compact explanatory opinion summarization (CEOS) which aims to extract 'within-sentence' explanatory text segments from input opinionated texts to help users better understand the detailed reasons of sentiments. We propose and study several general methods for identifying candidate boundaries and scoring the explanatoriness of text segments including parse tree search, probabilistic explanatoriness scoring model, and Hidden Markov Models. We create new data sets and use new evaluation measures to evaluate CEOS. Experimental results show that the proposed methods are effective for generating an explanatory opinion summary, outperforming a standard text summarization method in terms of our major measure of performance.

Generating Contrastive Summaries of Comparative Opinions in Text: This work presents a study of a novel summarization problem called *contrastive opinion summarization* (COS). Given two sets of positively and negatively opinionated sentences which are often the output of an existing opinion summarizer, COS aims to extract comparable sentences from each set of opinions and generate a comparative summary containing a set of contrastive sentence pairs. We formally formulate the problem as an optimization problem and propose two general methods for generating a comparative summary using the framework, both of which rely on measuring the content similarity and contrastive similarity of two sentences. We study several strategies to compute these two similarities. We also create a test data set for evaluating such a novel summarization problem. Experiment results on this test set show that the proposed methods are effective for generating comparative summaries of contradictory opinions.

Causal Topic Mining: In many applications, there is a need to analyze topics in text in consideration of external time series variables such as stock prices or national polls, where the goal is to discover causal topics from text, which are topics that might potentially explain or be caused by the changes of an external time series variable.

To solve this problem, we first propose a novel information retrieval problem, where the query is a time series for a given time period. The goal of such retrieval is to find relevant documents in a text collection of the same time period, which contain topics that are correlated with the query time series. We propose and study multiple retrieval algorithms that use the general idea of ranking text documents based on how well their terms are correlated with the query time series. Experiment results show that the proposed retrieval algorithm can effectively help users find documents that are relevant to the time series queries, which can help in the understanding of the changes in such time series.

Second, beyond just retrieving relevant documents, we propose a novel general text mining framework for discovering such causal 'topics' from text, i.e., Iterative Topic Modeling with Time Series Feedback (ITMTF). Topic modeling has recently been shown to be quite useful for discovering and analyzing topics in text data. The ITMTF framework naturally combines any given probabilistic topic model with causal analysis techniques for time series data such as Granger Test to discover topics that are both coherent semantically and correlated with time series data. The basic idea of ITMTF is to iteratively refine a topic model to gradually increase the correlation of discovered topics with the time series data by leveraging the time series data to provide feedback at each iteration to influence a topic model through imposing a prior distribution of parameters. Experiment results show that the proposed ITMTF framework can effectively discover causal topics from text data, and the iterative process improves the quality of the discovered causal topics.

To the best of my knowledge, this dissertation is the first systematic study on in-depth understanding of analyzing explanatory details of opinions. Especially, techniques in this dissertation focus on unsupervised approaches that do not require much human labeled data set.

The rest of the dissertation is organized as follows. We overview common related works in Chapter 2. We present studies on explanatory opinion summary (EOS) in Chapter 3 and contrastive opinion summary (COS) in Chapter 4, respectively. In Chapter 5, we present a technique to retrieve time-correlated documents and causal topics with time series query. And then, we conclude the dissertation with future work in Chapter 6.

Chapter 2

Related Work

2.1 General Automatic Text Summarization

Automatic text summarization has been studied for a long time due to the need of handling large amount of electronic text data. There are two representative types of automatic summarization methods. *Extractive Summary* is a summary made by selecting representative text segments, usually sentences, from the original documents. *Abstractive Summary* does not directly reuse the existing sentences from the input data; it analyzes documents and directly generates sentences. Because it is hard to generate readable, coherent, and complete sentences, studies on extractive summary are more popular than those on abstractive summary. Research in the area of summarizing documents focused on proposing paradigms for extracting salient sentences from text and coherently organizing them to build a summary of the entire text [27, 50, 66, 69]. While earlier works focused on summarizing a single document, later, researchers started to focus on summarizing multiple documents. Early extractive summary techniques were based on simple statistical analysis about sentence position or term frequency [18, 59], or basic information retrieval techniques such as inverse document frequency [81]. Machine learning and data mining techniques enabled summarizers to do work based on various training data [50, 53, 68]. More recent methods have been developed to find relationships between sentences based on a graph or tree structures. Among various kinds of recent researches, in particular, LexRank [20], which is a representative algorithm to measure the centrality of sentences, converts sentences into a graph structure, and finds central sentences based on their popularity (more mentioning) and coverage (cover various information). The graph-based approach showed good performance for both single and multi-document summarization. Moreover, because it does not require language-specific linguistic processing, it can also be applied to other languages [64]. General summarization techniques can shrink the size of text to read. However, general summarization focuses on centrality that does not guarantee explanatoriness. To show the difference between general and explanatory summarization, LexRank will be used as our main baseline. Term frequency-based and information retrieval-based methods will be also compared to our methods as TF-IDF baseline. Our problem setup is based on extractive summary generation and unsupervised learning. Therefore, abstractive summarization and supervised machine learning-based approaches are not comparable to our methods.

2.2 Opinion Summarization

Opinion mining and summarization techniques have attracted a lot of attention because of its usefulness in the Web 2.0 environment. There are several surveys that summarize the existing works [43, 54, 55, 70].

General opinion mining was focused on finding topics among articles and clustering positive and negative opinions on topics. Most of the results of opinion summarization focused on showing statistics of the number of positive and negative opinions. Usually people used table-shaped summary [28, 29, 62] or histogram [56]. Sometimes, each section had an extracted sentence from the article and had a link to the original one. It was not enough to show the details of the different opinions.

Compared with other summarization problems (e.g., news summarization which has been studied extensively), opinion summarization has some different characteristics. In an opinion summary, usually the polarities of input opinions are crucial. Sometimes, those opinions are provided with additional information such as rating scores. Also, the summary formats proposed by the majority of the opinion summarization literature are more structured in nature with segments organized by sub-topics and polarities.

For opinion summarization, mainly two approaches, probabilistic methods and heuristic rule-based methods, are used. Some opinion summarization work used probabilistic topic modeling methods such as probabilistic latent semantic analysis (PLSA) [25] and latent Dirichlet allocation (LDA) [10]. Topic sentiment mixture model [62] extended PLSA model with opinion priors to show positive and negative aspects of topics effectively. This model finds latent topics as well as its associated sentiment and also reveals how opinion sentiments evolve over the time line. In [85], multi-grain topic model was proposed as an extension of LDA. This work finds ratable aspects from reviews and generates summaries for each aspect. The proposed multi-grain LDA topic model can extract local topics which are ratable aspects written by an individual user as well as cluster local topics into global topics of objects such as the brand of a product type.

Heuristic rule-based methods have also been used in opinion summarization. Usually these methods have two steps: aspect extraction and opinion finding for each aspect. In [29, 56, 30], aspects of products are found using supervised association rule mining and rules such as opinion features are usually noun phrases. To connect extracted features with opinion words, WordNet is also used. [93] focused on movie review domain. Based on domain-specific heuristics such as many features tend to be around the cast of a movie, features can be found more efficiently. Machine learning techniques [48, 71] and relaxation labeling [74] are also used for features extraction and opinion summary.

Aspect-based summarization is one of the most popular types in opinion summarization and has been heavily explored over the last few years. It first finds subtopics (aspects) of the target and obtains statistics of positive and negative opinions for each aspect [28, 29, 30, 48, 56, 58, 62, 74, 85, 93]. By further segmenting the input texts into smaller units, aspect-based summarization can show more details in a structured way. Aspect segmentation can be even more useful when overall opinions are different from opinions of each aspect because an aspect-based summary can present opinion distribution of each aspect separately.

Although such technique can provide a general overview of opinions, users still must read all the actual text to understand the detailed reason of the opinion distribution. Thus, our work is a natural extension of these previous works to enable a user to understand why a particular opinion is expressed. In Chapter 3, we present a way to further summarize opinion to provide explanatory details. In Chapter 4, we focus on comparative opinions to help users further digest and understand contradictory opinions, which is none of the previous opinion summarization works focused on. Although some of previous opinion summarization works try to provide high probability words, phrases, or sentences as supplement, popularity-based selections may not yield explanatory information. We will compare our proposed techniques with TF-IDF baseline that will cover these techniques. Another work worth mentioning is Opinosis [22] which generates a short phrase summary of given opinions, but it also mainly focus on compressing frequently mentioned (popular) information. None of these existing opinion summarization methods is designed to solve the same problem as ours.

Chapter 3

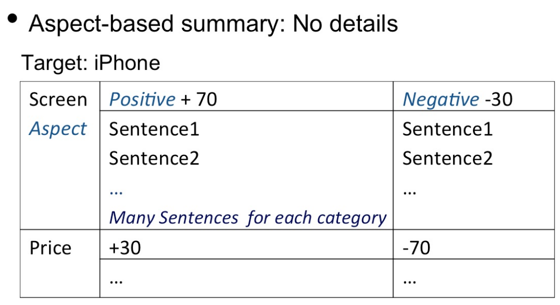
Explanatory Opinion Summarization [1]

3.1 Unsupervised Extraction of Explanatory Sentences for Opinion Summarization

3.1.1 Introduction

The increased user participation in generating contents in Web 2.0 environment has led to the quick growth of a lot of opinionated text data such as blogs, reviews, and forum articles on the Web because of increased user participation in generating contents. Due to the difficulty in digesting huge amount of opinionated text data, opinion mining and summarization techniques have become increasingly important, receiving a lot of attention from industry and academia.

Most previous studies on opinion mining and summarization have focused on predicting sentiments of entities and aspect-based rating for the entities, but they cannot provide a detailed explanation of the underlying reasons of the opinions. For example, to understand opinions about 'iPhone', people can use review articles from websites to find aspects such as 'screen', 'battery', 'design', and 'price', and then further predict the sentiment orientation (usually positive or negative) on each aspect as shown in Figure 3.1a.

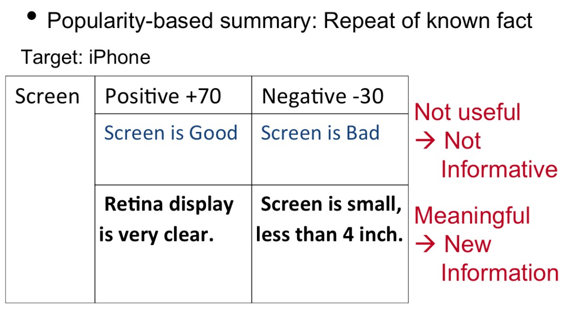


- Aspect-based summary: No details

Target: iPhone

Screen	*Positive* + 70	*Negative* -30
Aspect	Sentence1 Sentence2 ... *Many Sentences for each category*	Sentence1 Sentence2 ...
Price	+30	-70
	...	...

(a) Example of aspect-based opinion summary.



- Popularity-based summary: Repeat of known fact

Target: iPhone

Screen	Positive +70	Negative -30	
	Screen is Good	Screen is Bad	Not useful → Not Informative
	Retina display is very clear.	**Screen is small, less than 4 inch.**	Meaningful → New Information

(b) Popularity-based and explanatory summary.

Figure 3.1: Comparison of different types of summaries.

Although these existing techniques can show the general opinion distribution (e.g. 70% positive and 30% negative

[1]Part of this chapter has been published in [42].

opinions about battery life), they cannot provide the underlying reasons why people have positive or negative opinions about the product. Therefore, even if such an opinion summarization technique is available, people would still need to read through the classified opinionated text collection to find out why people expressed those opinions. This discovery task can be rather cumbersome and time consuming and therefore needs to be automated.

Although general automatic summarization techniques may be used to shrink the size of text to read, they generally extract sentences based on 'popularity'. As a result, the output summary tends to cover already known information. For example, for the summary request for "positive opinions about iPhone screen", a pure popularity-based summary could be 'Screen is good', as shown in the second row of Figure 3.1b. Given that the sentences to be summarized are already known to be about "positive opinions about iPhone screen", such a summary is obviously redundant and does not give any additional information to explain the reason why a positive opinion about iPhone screen is held. That is, useful explanatory sentences, such as those in the last row of Figure 3.1b, should not only be relevant to the target topic we are interested in, but also include details explaining reasons of sentiments which are not redundant to the target topic itself.

Unfortunately, none of the existing summarization techniques is capable of generating an explanatory summary that gives detailed reasons of the opinions for a given request, which can be more useful for users. To solve this problem, we propose a novel sentence ranking problem called unsupervised explanatory sentence extraction (ESE) which aims to rank sentences in opinionated text based on their explanatoriness to help users better understand the reasons of sentiments. As can be seen in the previous example, explanatory sentences should not only be relevant to the target topic we are interested in, but also include details *explaining* reasons of sentiments; generic positive or negative sentences are generally not explanatory. For example, the most explanatory sentence for 'positive opinion about iPhone screen' could be 'Retinal display is very clear'. In other words, we can regard this problem as to extract sentences to answer the question about why reviewers hold a certain kind of opinions. That is, useful explanatory sentences, such as those in the last row of Figure 3.1b, should not only be relevant to the target topic we are interested in, but also include details explaining reasons of sentiments which are not redundant to the target topic itself.

A main difference between ESE and a sentence ranking problem in a regular summarization is that in ESE we emphasize the selection of sentences that provide an explanation of the reason why an opinion holder has a particular polarity of sentiment about an entity, whereas in regular summarization, there is no guarantee of the explanatoriness of a selected sentence.

A main technical challenge in solving this problem is to assess the explanatoriness of a sentence in explaining sentiment. We focus on studying how to solve this problem in an unsupervised way as such a method would be generally applicable to many domains without requiring manual effort, and if we have labeled data available, we can always plug in an unsupervised approach into any supervised learning approach as a feature. We introduce three heuristics for scoring explanatoriness of a sentence (i.e., length, popularity, and discriminativeness). In addition to the representativeness of information which is a main criterion used in the existing summarization work, we also consider discriminativeness with respect to background information and lengths of sentences. We propose three general new methods for scoring explanatoriness of a sentence based on these heuristics, including a method adapted from TF-IDF weighting, and two probabilistic models based on sentence-level and word-level likelihood ratios, respectively.

To evaluate the proposed explanatoriness scoring methods, we use the modified version of standard ranking measure, weighted Mean Average Precision (wMAP). We propose a new method to assign weights to different test topics based on the expected gap between the performance of a random ranking and an ideal ranking when computing the average performance over a set of topics, which is more reasonable than the standard way of using uniform weights. Since the task of explanatory opinion summarization is new, there does not exist any data set that we can use for evaluation. We thus created two new data sets in two different domains, respectively, to evaluate this novel summarization task.

Experiment results show that all the proposed methods are effective in selecting explanatory sentences, outperforming a state of the art sentence ranking method for a regular text summarization method. Our results also show that adding length factor in sentence level modeling and using Dirichlet smoothing in probability estimation made our algorithm more effective in identifying explanatory sentences.

The main contributions of this work include:

1. We introduce a novel sentence ranking problem called explanatory sentence extraction (ESE), where the goal is to rank sentences by explanatoriness that can *explain* why a certain sentiment polarity of opinions are held by reviewers.

2. We propose multiple general methods based on TF-IDF weighting and probabilistic modeling to solve the ESE problem in an unsupervised way.

3. We define a new measure and create two new data sets for evaluating this new task.

4. We evaluate all the proposed methods to understand their relative strengths and show that they are all more effective than a state of the art sentence ranking method for a regular summarization method for solving the ESE problem.

The rest of sections are organized as follows. In Section 3.1.2, we motivate a problem formulation and formally describe the problem. In Section 3.1.3, we explain how we measure explanatoriness of text. In Section 3.1.4, we show experiment results, and then we make a conclusion.

3.1.2 Problem Formulation

Our problem formulation is based on the assumption that existing techniques can be used to (1) classify review sentences into different aspects (i.e., subtopics); and (2) identify the sentiment polarity of an opinionated sentence (i.e., either positive or negative), and we hope to further help users digest the opinions expressed in a set of sentences with a certain polarity (e.g., positive) of sentiment on a particular aspect of the target entity commented on by extracting a set of *explanatory* sentences that can provide specific reasons why positive (or negative) opinions are held.

Thus, as a computational problem, the assumed input is (1) a topic T as described by a phrase (e.g., a camera model), (2) an aspect A as described by a phrase (e.g., "picture quality" for a camera), (3) a polarity of sentiment P (on the specified aspect A of topic T), which is either "positive" or "negative", and (4) a set of opinionated sentences $O = \{S_1, ..., S_n\}$ of the sentiment polarity P. For example, if we want to summarize positive opinions about iPhone screen, our input would be T="iPhone", A="screen", P="positive", and a set of sentences with positive opinions about the iPhone screen, O.

Given T, A, P, and O as input, the desired output is a ranked list of sentences by explanatoriness, L, which is ordered list of input sentences of O, i.e., $L = (S'_1, ..., S'_n)$ such that explanatory sentences would be ranked on top of non-explanatory ones (so as to enable a user to easily digest opinions). An ideal ranking is thus one where all the explanatory sentences would be ranked on top all the non-explanatory ones. To the best of our knowledge, such a ranking problem has not been studied in any previous work.

Such a ranked sentence list can be used to generate an explanatory opinion summary by feeding it into an existing summarization algorithm. Indeed, an explanatory summary can also be generated simply by taking a maximum number of most explanatory sentences to fill in a summary constrained by the specified summary length (e.g. 500 character) and removing redundancy using Maximal Marginal Relevance or clustering.

3.1.3 Explanatoriness Scoring Functions

In this section, we study the question of how to assess the likelihood that a sentence is explanatory for providing a reason why a particular sentiment polarity of opinions was expressed. We also propose several heuristics for designing the explanatoriness scoring function $ES(S)$. Scoring explanatoriness is especially challenging because we would like to design a scoring function that does not require (much) training data.

Basic Heuristics

We first propose three heuristics that may be potentially helpful for designing an explanatoriness scoring function.

1. Sentence length: A longer sentence is more likely explanatory than a shorter one since a longer sentence in general conveys more information.

2. Popularity and representativeness: A sentence is more likely explanatory if it contains more terms that occur frequently in all the sentences in O. This intuition is essentially the main idea used in the current standard extractive summarization techniques. We thus can reuse an existing summarization scoring function such as LexRank for scoring explanatoriness. However, as we will show later, there are more effective ways to capture popularity than an existing standard summarization method; probabilistic models are especially effective.

3. Discriminativeness relative to background: A sentence with more discriminative terms that can distinguish O from background information is more likely explanatory. As we observed an example in Section 3.1.1, too much emphasis on representativeness would give us redundant information. Explanatory sentences should provide us unique information about the given topic. Therefore, intuitively, an explanatory sentence would more likely contain terms that can help distinguish the set of sentences to be summarized O from more general background sets which contain opinions that are not as specific as those in O. That is, we can reward a sentence that has more discriminative terms, i.e., terms that are frequent in O, but not well covered in the background information.

There can be various background data sets that we can compare. Multiple background data sets can be obtained by topic relaxation. Because our input topic definition has 3 dimensions, (T, A, P), we can relax the condition on one of them. For example, for (iPhone, screen, Positive), relaxed topics are (iPhone, screen) (the P condition relaxed), (screen, Positive) (the T condition relaxed), (iPhone, Positive) (the A condition relaxed). Furthermore, for each dimension, we can even further relax to higher-level concepts. For example, we can relax the product condition, (iPhone), to 'smart phone' topic. For product entities, we can find product hierarchies in many review websites. If it is hard to relax topics, we can generalize very broadly. For example, we can observe all the 'product reviews' as background. In the usage scenario of the proposed algorithm, we would have opinionated sentences about one topic, T, as an input, and aspects and sentiments will be classified by the existing opinion mining techniques. That is, we would always have background at least within topic T.

The intuitions presented in this section can each be used individually to measure how likely a sentence is explanatory. However, a potentially more effective way to measure explanatoriness is to combine intuitions of these heuristics. Below, we propose several different ways to combine these three heuristics.

TF-IDF Explanatoriness Scoring

The first method is to adapt an existing ranking function of information retrieval such as BM25 [34], which is one of the most effective basic information retrieval functions. Indeed, our popularity heuristic can be captured through Term Frequency (TF) weighting, while the discriminativeness can be captured through Inverse Document Frequency (IDF) weighting. We thus propose the following modified BM25 for explanatoriness scoring (BM25$_E$):

From the original BM25 formula, given query $Q = q_1, q_2, ..., q_n$, BM25 score of document D is

$$BM25(D,Q) = \sum_{i=1}^{n} IDF(q_i) \frac{f(q_i,D)(k_1+1)}{f(q_i,D) + k_1(1-b+b\frac{|D|}{avgdl})}$$

$$IDF(q_i) = log\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5}$$

where $f(q_i, D)$ is term frequency of q_i in D, $|D|$ is document length, $avgdl$ is average document length, N is the total number of documents in the collection, and $n(q_i)$ is the number of documents having q_i in the collection.

For explanatoriness ranking, we can consider a sentence as a query and measure explanatoriness of each word of the sentence based on how frequent the word is in the input data set (O) and the background data set (B). Specifically, given a sentence $S = w_1, w_2, ..., w_n$, the modified BM25$_E$ would be defined as:

$$BM25_E(S,O,B) = \sum_{w \in S} IDF(w) \frac{c(w,O)(k_1+1)}{c(w,O) + k_1(1-b+b\frac{|B|}{avgdl})}$$

$$IDF(w) = log\frac{|B| - c(w,B) + 0.5}{c(w,B) + 0.5}$$

where $c(w, O)$ is frequency of w in data set O, $|B|$ is the total number of term occurrences in data set B, and $avgdl$ is the average number of total term occurrences of subclusters in T that O is extracted from. k_1 and b are parameters to be empirically set.

Probabilistic Models for Explanatoriness Scoring

To implement the proposed heuristics in a more principled way, we further propose new probabilistic models for explanatoriness scoring. Our basic idea here is to try to develop two generative models to model explanatory sentences (or words) and non-explanatory sentences (or words), respectively. We can then score a sentence based on the probability that it has been generated by the explanatory model as opposed to the non-explanatory model. In general, if we use the sentences to be summarized (O) to estimate the explanatory model, it would capture popularity, while using a background set to estimate the non-explanatory model would enable capturing of discriminativeness. We consider two different strategies to capture length heuristics, leading to the following two different ways to perform probabilistic scoring.

1. Sentence Modeling: In this approach, we develop a generative model for a whole sentence. Specifically, we assume that a sentence $S = w_1 w_2 ... w_n$ can be either explanatory or not, denoted by $E \in \{0, 1\}$. We further assume that both the content of S and the length of S affect how likely the sentence is explanatory.

We can thus score the explanatoriness of sentence S based on the conditional probability $p(E = 1|S) = p(E =$

$1|w_1...w_n, L = n)$, which can be interpreted as the probability that sentence S is explanatory.

According to Bayes rule, we have

$$\begin{aligned} p(E = 1|w_1...w_n, L = n) &= \frac{p(w_1...w_n, L = n|E = 1)p(E = 1)}{p(w_1...w_n, L = n)} \\ p(E = 0|w_1...w_n, L = n) &= \frac{p(w_1...w_n, L = n|E = 0)p(E = 0)}{p(w_1...w_n, L = n)} \end{aligned}$$

Since ranking sentences based on $p(E = 1|S)$ is equivalent to ranking them based on $\frac{p(E=1|S)}{p(E=0|S)}$, we will use the latter, which is

$$\begin{aligned} \frac{p(E = 1|S)}{p(E = 0|S)} &= \frac{p(w_1...w_n, L = n|E = 1)p(E = 1)}{p(w_1...w_n, L = n|E = 0)p(E = 0)} \\ &\propto \frac{p(w_1...w_n, L = n|E = 1)}{p(w_1...w_n, L = n|E = 0)} \\ &= \frac{p(w_1...w_n|L = n, E = 1)p(L = n|E = 1)}{p(w_1...w_n|L = n, E = 0)p(L = n|E = 0)} \end{aligned}$$

The second step is possible because $\frac{p(E=1)}{p(E=0)}$ is independent of S, thus not affecting ranking of sentences.

We now see that our model involves two components: one is a generative model for the content of the sentence (i.e., $w_1...w_n$) and the other is a generative model for the sentence length (i.e., $L = n$). So in order to further refine the model, we need to define these two generative models more concretely. In general, there are many possibilities of refining this general model. Here we look into the option of using a unigram language model for the content and a Poisson model for the length. With such a refinement, we have

$$\frac{p(E = 1|S)}{p(E = 0|S)} \propto \prod_{i=1}^{n} \frac{p(w_i|E = 1)}{p(w_i|E = 0)} \frac{\lambda_1^n e^{-\lambda_1}}{\lambda_0^n e^{-\lambda_0}}$$

Taking logarithm of both sides, we have the following explanatoriness scoring function:

$$ES(S) = \sum_{i=1}^{n} \log \frac{p(w_i|E = 1)}{p(w_i|E = 0)} + n \log \frac{\lambda_1}{\lambda_0} - \lambda_1 + \lambda_0$$

where λ_0 and λ_1 are the Poisson distribution parameters (mean) for the non-explanatory model and explanatory model, respectively. $p(w_i|E = 1)$ and $p(w_i|E = 0)$ are the unigram probability parameters for word w_i in the explanatory and non-explanatory content generative models, respectively.

The remaining question is how to estimate these parameters. The estimation of $p(w|E = 1)$ and $p(w|E = 0)$ will be discussed later as the same parameters will also occur in the second model (i.e., Word-Level Modeling). So we only discuss the estimate of λ_1 and λ_0 here. To estimate these two parameters, we assume that λ_1 can be approximated by the average sentence length in O since we can assume that sentences in O are explanatory. That is, $\lambda_1 = \bar{n}$, where $\bar{n}$ is the average length of sentences in O. To capture the intuition that shorter sentences tend to be more likely non-explanatory, we set $\lambda_0 = \lambda_1 - \delta$, where $\delta \in [0, \bar{n})$ is a parameter.

With these estimates, the scoring function would be:

$$ES(S) = \sum_{i=1}^{n} \log \frac{p(w_i|E=1)}{p(w_i|E=0)} + n \log \frac{\bar{n}}{\bar{n} - \delta} - \delta$$

Note that the last term δ can be ignored since it does not affect ranking sentences.

We refer to this scoring function as Sentence Likelihood Ratio with Poisson Length Modeling (**SentLRPoisson**). As a baseline, we also consider a special case of this function when we set $\delta = 0$, i.e., turning off the Poisson length model, which would give us the following scoring function:

$$ES(S) = \sum_{i=1}^{n} \log \frac{p(w_i|E=1)}{p(w_i|E=0)}$$

where we refer to as Sentence Likelihood Ratio (**SentLR**). We see that the main difference between SentLR and SentLRPoisson is that the latter attempts to reward long sentences, while the former only rewards a long sentence if the extra words have higher probabilities according to the explanatory model $p(w|E = 1)$ than the non-explanatory model $p(w|E = 0)$. As we will show later, SentLRPoisson is more effective than SentLR.

2. Word-level Modeling:

Instead of modeling a whole sentence, in this approach, we define the explanatoriness of a sentence as the sum of the explanatoriness of each word in the sentence, and model the explanatoriness of each word with a probabilistic model. Indeed, if we treat each word as a unit for modeling explanatoriness, it would be natural to define the explanatoriness of a sentence based on how explanatory each word in the sentence is. Thus if we use $ES(S)$ and $ES(w_i)$ to denote explanatoriness score of sentence S and that of word w_i respectively, we can define $ES(S)$ as the sum of $ES(w_i)$ over all the words in S, i.e.,

$$ES(S) = \sum_{i=1}^{n} ES(w_i) \tag{3.1}$$

Note that with this strategy, we explicitly encode the length heuristic and a longer sentence would be generally favored.

Now the question is how to model and estimate $ES(w_i)$. For this purpose, we assume that each word w can be either explanatory or not, denoted by $E \in \{0, 1\}$. We can thus score the explanatoriness of word w based on the conditional probability $p(E = 1|w)$, which can be interpreted as the posterior probability that word w is explanatory. That is, $ES(w) = p(E = 1|w)$.

According to Bayes rule, we have

$$p(E=1|w) = \frac{p(w|E=1)p(E=1)}{p(w)}\ ,\ p(E=0|w) = \frac{p(w|E=0)p(E=0)}{p(w)} \tag{3.2}$$

Since ranking sentences based on $p(E=1|w)$ is equivalent to ranking them based on $\frac{p(E=1|w)}{p(E=0|w)}$, we will use the latter, which is

$$\frac{p(E=1|w)}{p(E=0|w)} = \frac{p(w|E=1)p(E=1)}{p(w|E=0)p(E=0)} \propto \frac{p(w|E=1)}{p(w|E=0)} \tag{3.3}$$

The second step is possible because $\frac{p(E=1)}{p(E=0)}$ is independent of w, thus not affecting ranking of sentences.

With these estimates, the scoring function would be:

$$ES(S) \quad = \quad \sum_{i=1}^{n} \frac{p(w_i|E=1)}{p(w_i|E=0)} \tag{3.4}$$

We call this function Sum of Word Likelihood Ratio (**SumWordLR**). The explanatoriness score of a word is now seen as the ratio of the probability that word w is observed from an *explanatory source* (i.e., $E=1$) to that it is observed from a *non-explanatory source* (i.e., $E=0$). As in the case of SentLR and SentLRPoisson, SumWordLR also involves two sets of parameters that have to be estimated, i.e., $p(w|E=1)$, and $p(w|E=0)$, which we further discuss next.

3. Estimation of $p(w|E=1)$ and $p(w|E=0)$:

In general, our estimate of $p(w|E=1)$ and $p(w|E=0)$ would depend on what kind of words we would regard as explanatory. There may be multiple ways of doing this. In this section, we will explore the following options.

For estimating $p(w|E=1)$, without additional knowledge, a reasonable assumption is that the set of sentences to be summarized O can be used as an approximate of sample of words that are explanatory, which is to say that we will use O to approximate the explanatory source. With maximum likelihood estimate, we have

$$p(w|E=1) = p(w|O) = \frac{c(w,O)}{|O|} \tag{3.5}$$

We refer to this estimate Maximum Likelihood estimate of $p(w|E=1)$ (**ML1**).

With this estimate, a word that is popular in O would tend to have a higher explanatoriness score, capturing the popularity heuristic. Though not explored in this section, we may also use pseudo feedback to improve this estimate by using only the top-ranked sentences in O to estimate $p(w|E=1)$. Another way to improve this estimate is to use the text of opinions about the *same* aspect, but with different polarity of sentiment. This is because people tend to refer to similar features and use similar words even if they may be talking about a different polarity of sentiment. Yet another way to improve this estimate is to use related opinions on the *same* aspect of a *similar* topic.

In this section, we use O with Dirichlet prior smoothing using text of the "sibling" polarity of O. Let O' be the set of sentences about the same aspect as O but with opposite polarity. We have

$$p(w|E=1) = \frac{c(w,O) + \mu_1 p(w|O')}{|O| + \mu_1}$$

where μ_1 is a smoothing parameter. Parameter μ_1 can be interpreted as the equivalent number of words in O we can regard O' as and can be set empirically. We refer to this estimate Dirichlet prior smoothing of $p(w|E=1)$ (**Dir1**).

How do we model words from non-explanatory source? Such words may be of several kinds: (1) They may be general background English words. (2) They may be general words related to the topic. (3) They may be general sentiment words without explaining the reason for a sentiment polarity. (4) They may be commenting about other aspects of the topic (out of focus). (5) They may be words about the right aspect, but for a different sentiment polarity. Among all these different kinds, (1), (2), and (3) may be more frequent than (4) and (5). (5) is probably rare. Thus a good estimate of $p(w|E=0)$ should ideally be based on text samples corresponding to all these different types. We explore the following options in this section:

Our first idea is to use all the text available to us about this topic T (denoted by B_{topic} for background of all text about the topic). This background is always available to us in our assumed opinion summarization setup. In this case, the Maximum Likelihood Estimator would give

$$p(w|E=0) \quad = \quad \frac{c(w, B_{topic})}{|B_{topic}|} \tag{3.6}$$

We refer to this estimate of $p(w|E=0)$ Topic Background Maximum Likelihood Estimate (**TopicML0**).

Since the topic background cannot reflect general English words well, we further consider incorporating a general background model which can be estimated based on a much larger text database (e.g., a large sample of Web data), denoted by $p(w|B)$. In our experiments, we use Microsoft N-gram service [2] to obtain a unigram language model as our general background model. We can thus smooth the maximum likelihood estimate with this larger background model using Dirichlet prior smoothing:

$$p(w|E=0) = \frac{c(w, B_{topic}) + \mu_0 p(w|B)}{|B_{topic}| + \mu_0}$$

where μ_0 is a smoothing parameter to be empirically set. It can be interpreted as how many equivalent words in B_{topic} we would like to regard our general background model as being worth. We refer to this smoothed estimate of

[2] `http://web-ngram.research.microsoft.com/info/`

$p(w|E=0)$ as **TopicDir0**.

3.1.4 Experiments

Data Set

	PRODUCT	HOTEL
# of topic	66	23
Avg # of sentences of topic	21.9	102.3
# of sentences	1447	2352
# of unique word	4377	2970
Average explanatoriness	0.5412	0.5976
Label agreement	0.62	0.75

Table 3.1: Data set for explanatory summarization evaluation.

Because there is no data set for evaluation of explanatory opinion summarization, we created new data sets. We can use the output of opinion mining results as our input; thus, we assume that the sentences are already tagged with sentiment polarities and classified into different aspects. We used a publicly available product review data set [17, 28], which is one of the most popular data sets for opinion mining. And for generality, we also created another data set in the different domain, hotel. We obtained reviews from trip advisor [3] and asked human labelers to make aspect and sentiment labels. We generated an aspect and sentiment label for each sentence in the data set by three human labelers, and filtered out the ones with disagreement.

For each data set, we clustered sentences based on the aspect and sentiment labels. If there were less than 10 sentences in one cluster (that is, for each (T, A, P)), we discarded it because it did not need summarization.

Our key evaluation question is whether the proposed methods can really find explanatory sentences. Therefore, for gold standard, we asked two human labelers to mark if each sentence is explanatory or not. We provided the topic of each cluster, (T, A, P), and asked if 'each sentence is explanatory to understand detailed reasons of opinions in the given topic'. Human labelers generated a binary label for each sentence. They are asked to mark sentences with explanatory information that is relevant to the given topic and is not directly inferable (redundant) from the topic label. Data set with more detailed labeling instruction, examples provided to labelers, and generated labels are publicly available [4]. Basic statistics about the two data sets are in Table 3.1. Label agreement is the ratio of number of sentences having the same label to the total number of sentences for each data set. For the clearer evaluation, for our experiment, we discarded sentences that have not-agreed labels, and if there is any topic has no explanatory sentence, we also discarded the topic because it does not affect the result. As a preprocessing step, the input data set is processed by a basic stemmer to relieve a sparseness issue.

[3]http://www.tripadvisor.com
[4]http://sifaka.cs.uiuc.edu/ir/data/expSum/

Measures

Our task is ranking sentences based on the predicted explanatoriness of a sentence. Thus we can use the standard information retrieval measure, Mean Average Precision (MAP), to quantitatively evaluate the ranking accuracy. Formally, for given topics $\{Q_1, ..., Q_m\}$, suppose k_i is the number of explanatory sentences for each topic.

$$AvgPrec(Q_i) = \frac{\sum_{j=1}^{k} p(j)}{k_i}$$

$$MAP = \frac{\sum_{i=1}^{m} AvgPrec(Q_i)}{m}$$

where $p(j)$ is precision at rank where each new explanatory sentence is retrieved. For example, if the first explanatory sentence is at the 2nd rank in explanatory summary, then $p(1) = 1/2$.

Since it is more interesting to see the performance of a method on a topic where the percentage of explanatory sentences is low (i.e., "hard" topics) than one where most sentences in O are explanatory (i.e., "easy" topics), when we take the average of performance on individual topics, we should intuitively place more weight on a hard topic than an easy topic since in the case of an easy topic, there is little room for improvement. The challenge, though, is how to quantify the difficulty of a topic. Here we propose a general way to quantify the "room for improvement" of a topic based on the gap between the expected performance of a random ranking and the best performance achieved by an ideal ranking.

Formally, suppose p is a performance measure. Given a topic Q, let $R(Q)$ be the ranking results for Q. Let $p_{random}(Q)$ be the expected performance of a random ranking for Q, and $p_{ideal}(Q)$ be the performance of an ideal ranking for Q. We can measure the 'room for improvement' by:

$$gap(Q) = p_{ideal}(Q) - p_{random}(Q)$$

Intuitively, $gap(Q)$ would be large if the performance of a random ranking is much lower than that of the performance of the ideal ranking, while if the random ranking has perfect performance (as, e.g., in the case that all the sentences in O are explanatory), $gap(Q)$ would be zero.

We can then use $gap(Q)$ as a weight on each topic to combine the performance on different topics. Formally, the aggregated performance on a set of queries $\{Q_1, ..., Q_m\}$ is given by:

$$Average(p) = \sum_{i=1}^{m} gap(Q_i) p(R(Q_i)) \tag{3.7}$$

where $R(Q_i)$ is the ranking results on topic Q_i and $p(R(Q_i))$ is the performance value of $R(Q_i)$ as measured using measure p. It is easy to see that we put more weight on a query with more room for improvement, and if the random ranking gives performance, we would essentially eliminate the topic from the evaluation. We apply this weighting strategy to MAP and denote it by $wMAP$ (for weighted MAP).

Baselines

To get the minimum performance, let us check $wMAP$ scores with random ranking for each data set. For MAP score, ideal ranking score would be always 1.0. For the random ranking score, the probability that the explanatory sentences would be retrieved in the summary for each rank would actually depend on the ratio of explanatory sentences in the input sentence set. Formally, for given topics $\{Q_1, ..., Q_m\}$, suppose n_i is the number of sentences, and k_i is the number of explanatory sentences of topic Q_i, $MAP(Q_i)_{random} = \frac{k_i}{m_i}$. Finally, if we apply this to Equation 3.7 to obtain $wMAP$ for random ranking,

$$wMAP_{random} = \sum_{i=1}^{m}(1.0 - MAP(Q_i)_{random})MAP(Q_i)_{random}$$

The wMAP score with random ranking for product data set is 0.4494 and for hotel data set is 0.4764. These are minimum scores to say that a method is effective in finding explanatory sentences.

We hypothesize that the proposed sentence ranking method would work better than that of standard text summarization method that mostly implements the popularity heuristic. We use LexRank sentence ranking method, which is popularly used in a general summarization algorithm and can generate ranking score of sentences, as a baseline to test this hypothesis. Note that most existing summarization algorithms do not rank sentences explicitly, thus they cannot be easily evaluated for scoring explanatoriness of sentences. Indeed, the proposed explanatoriness scoring function is meant to be a new factor that can be incorporated to an existing summarization algorithm to generate explanatory opinion summaries. None of the other existing methods discussed in Chapter 2 is designed to solve the same problem as ours with unsupervised manner, thus they cannot be used as baselines.

Results

In Section 3.1.3, we proposed various ways to measure explanatoriness of sentences. In this section, we compare performance of each method on both data sets. Table 3.2 shows the list of proposed methods and their labels. In case of probabilistic models, we need to choose one strategy among choices at each sub-step. For probabilistic model performance, the basic combination is SumWordLR-ML1-TopicML0 that does not require parameter tuning, then for each variation, we change one of strategies for the target sub-step, and we fixed the other steps with strategies of basic combination. For methods with parameters, we measure both optimal parameter tuned performance as a reference of

TYPE	SUB-STEP	CHOICE	LABEL	PARAMETER
Individual Heuristic		Random performance	Random	
		LexRank	LexRank	
TF-IDF			BM25$_E$	k_1, b
Probabilistic Model	1. Modeling sentence explanatoriness with word explanatoriness	Sentence Likelihood Ratio	SentLR	δ
		Sentence Likelihood Ratio + Poisson Length Model	SentLRPoisson	
		Sum of Word Likelihood Ratio	SumWordLR	
	2. Estimation of $p(w\|E=1)$	Maximum Likelihood	ML1	
		Dirichlet smoothing	Dir1	μ_1
	3. Estimation of $p(w\|E=0)$	Topic Background Maximum Likelihood	TopicML0	
		Dirichlet smoothing with General Background	TopicDir0	μ_0

Table 3.2: Proposed method summary - The list of proposed methods and their labels.

the best possible performance and averaged 2-fold cross validation performance on test data sets.

1. Comparison of different methods:

	SCORING METHOD	PRODUCT	HOTEL
	LexRank	0.4612	0.5869
Optimal	BM25$_E$	0.7567	0.6100
Cross	BM25$_E$	0.7498**	0.6060
	SentLR	0.7247**	0.6241
Optimal	SentLRPoisson	0.7634	0.6660
Cross	SentLRPoisson	0.7586**	0.6494
	SumWordLR	0.7730**	0.6143

Table 3.3: Comparison of various methods for scoring explanatoriness (wMap). 'Optimal' is the best performance when parameter is tuned, and 'Cross' is the cross-validation performance. Compared to LexRank, significantly different values with 95% confidence level are marked as **. 'Optimal' results are not tested for significance test.

Table 3.3 shows the performance of LexRank baseline and proposed methods with significance test results by t-test. All of our methods show better performance than LexRank baseline, which represents a state of the art sentence ranking used in a standard text summarization method. As we expected, our proposed methods that consider both representativeness and discriminativeness perform better than standard summarization baseline, LexRank, which mainly focuses on centrality. For all cases of product data set, our methods show significantly better performance than LexRank with 95% confidence level. For hotel data set, although performance values that we obtained from our methods are higher than those from LexRank, they do not show significant differences. Because hotel data set has only 23 topics, it may not be big enough to show significant difference. Another possible explanation is that centrality-based method performs pretty well for hotel data set that has many sentences per topic. That is, there are more sentences in one input sentence set, and there are more sentences to compare, then centrality measure can work comparably.

We additionally perform significance test among our proposed methods. Although probabilistic models perform better than TF-IDF method in general, they do not show significant improvement. Among different probabilistic models, SentLRPoisson is consistently better than SentLR, and also improvement is significant for the product data set (95% confidence level). This means that the use of Poisson model for length is beneficial, also validating the proposed sentence length heuristic. SumWordLR also shows significantly better performance than SentLR for the product data set (95% confidence level). The sum of word likelihood ratio model (SumWordLR) performs better than sentence likelihood ratio with length modeling (SentLRPoisson) on the product data set, but the latter appears to be better on the hotel data set. Actually, their performance differences do not show significance.

In summary, all proposed method show better performance than the baseline method. Especially, TF-IDF weighting, sentence likelihood with Poisson length modeling, and sum of word likelihood ratio performed better than simple sentence likelihood ratio. Among superior methods, our general recommendation is sum of word likelihood ratio because it does not have any parameter. With given enough initial data for parameter tuning, the other two methods also have potential to show even better performance.

2. Comparison of smoothing variations of probabilistic model:

	ESTIMATE	PRODCUT	HOTEL
	ML1	0.7730	0.6143
Optimal	Dir1	0.7822	0.6454
Cross	Dir1	0.7501*	0.6428

	ESTIMATE	PRODUCT	HOTEL
	TopicML0	0.7730	0.6143
Optimal	TopicDir0	0.7863	0.6461
Cross	ToipcDir0	0.7825	0.6453

Table 3.4: Comparison of various methods for estimating $p(w|E = 1)$ (left) and $p(w|E = 0)$ (right). Unit: wMap. 'Optimal' is the best performance when parameter is tuned, and 'Cross' is the cross-validation performance. Compared to ML1, significantly different values with 90% confidence level are marked as *. 'Optimal' results are not tested for significance test.

We further compare different estimation methods for $p(w|E = 1)$ in the left table of Table 3.4 where we use the SumWordLR method. For estimating $p(w|E = 0)$, we used TopicML0. Among different smoothing methods for $p(w|E = 1)$, the optimal performance of Dirichlet smoothing (Dir1) performs better than maximum likelihood (ML1). However, compared to the cross validation performance of Dirichlet smoothing, Dir1 is worse than ML1. For Dir1, we smoothed original popularity to O' which is the data set with opposite sentiment orientation. Usually this is not a very big data set, so it is hard to obtain well-tuned parameter from training, which may explain the non-promising cross validation performance.

The right table of Table 3.4 shows the effectiveness of different estimation methods for $p(w|E = 0)$ where we control the estimate of $p(w|E = 1)$ as ML1 and score with the SumWordLR method. Among different smoothing methods for $p(w|E = 0)$, Dirichlet smoothing with general background (TopicDir0) performs the best. TopicDir0 uses both in-topic background data set and general English background. By measuring discriminativeness to both background data sets, TopicDir0 can penalize words which are common in a general English corpus as well as the input topic B_{topic}.

3. Example Result:

To show the usefulness of the proposed algorithm in summarization context, we generate a sample explanatory opinion summary using ESE. We generate a summary by picking sentences one by one from the top of the list ranked

EXPLANATORY SUMMARY	BASELINE SUMMARY
when i played back a symphony orchestra you would swear that **you were seated dead center in front of the orchestra** - it 's that good ! 1) great sound (> **98db signal-to-noise ratio beats ipod** 's " unspecified " ratio) and **good power output allow** the zen xtra to drive **large head-phones as well as external speakers** . once again , the sound is awesome , the batterly life is only 6-8 hours , and that is because all my music is 320 kbps which does affect the battery life . the sound is awesome ,	the sound from the player is **ok** . - best in class sound the sound is **excellent** as one would suspect from a creative product . the sound is **great** even with the supplied earbuds – but i find earbuds uncomfortable so i use different headphones . plusses are the easy to remove battery and the **terrific** sound produced by the nomad . on the positive side , the sound of the player is pretty **good** , once you have everything configured . the sound is **great** , and the volume is more than satisfactory for

Table 3.5: Example summary output comparison between explanatory summary (SumWordLR) and baseline summary (LexRank) about *positive opinion about MP3player1 sound*.

by the explanatory ranking algorithm until the length limit (500 character) is reached. We also apply redundancy removal technique. That is, when we pick sentences from the top of the explanatoriness ordered list, if the content of the next candidate sentence is covered enough by already selected sentences, we skip and do not include the candidate sentence to the summary. Here, we use threshold 0.5, which means the candidate sentence is not selected if more than 50% of the content is already covered by the chosen sentences. Baseline summary is also generated by the same procedure using LexRank scoring function.

Table 3.5 shows an example of explanatory summary generated by a proposed method and a baseline summary generated by LexRank about 'positive opinion about MP3 player sound'. This example shows the benefits of the explanatory summary over the general summary clearly. Explanatory summary shows good details about 'sound' of the target product such as 'seated center in front of the orchestra', '> 98db signal-to-noise ratio', 'beats ipod', 'allow large head-phones as well as external speakers'. On the other hand, in the baseline summary, despite of redundancy removal, we can see many repetitions of 'good sound' because it is popular contents in the input data. The large portion of space in the normal summary is consumed by the fact that we already know from the given topic.

3.1.5 Conclusions

We introduced a novel sentence ranking problem called unsupervised explanatory sentence extraction (ESE), which can extend the capabilities of the current opinion summarization methods. A key challenge in solving this problem is to automatically rank sentences by explanatoriness that shows how much a sentence can provide an explanation of reasons why a particular polarity of sentiment on an aspect of a topic is expressed. We proposed several general methods, including modified TF-IDF weighting and probabilistic models for scoring explanatoriness with variations such as Poisson length modeling and Dirichlet smoothing. We created two new test sets for evaluating this problem and proposed a new weighting method for averaging over multiple topics. Experiment results showed that proposed methods outperform a state of the art sentence ranking method for a standard text summarization method, LexRank. Our results also showed that adding length factor in sentence level modeling and using Dirichlet smoothing in probability estimation made our algorithm more effective in identifying explanatory sentences.

Our work is the first step toward studying the new problem of measuring explanatoriness. With the two data sets and appropriate measures, we have paved the way for further studying this problem. As a future work, we can further explore different ways of estimating the proposed probabilistic models, especially through exploiting pseudo feedback to refine the estimate of the models. Using more complex features such as n-grams and frequent patterns can be useful to better capture explanatoriness. It would also be interesting to test the proposed algorithm on more diverse topics to further verify the generality of our algorithm. Moreover, we can also use different units rather than sentence-level extractive summary; indeed, many of our features can be applied to smaller than sentence units. Therefore, by extracting explanatory phrases or discourses, we may be able to generate an even more concise summary. Furthermore, we can explore abstractive explanatory summarization instead of extractive summarization.

3.2 Compact Explanatory Opinion Summarization

3.2.1 Introduction

In Section 3.1, we explored methods for measuring explanatoriness of sentences. While sentence-level explanatory sentence extraction showed its usefulness, treating a sentence as a unit has limitations. It is often the case that not all words in a sentence are useful for explaining an opinion, so it is also desirable to extract smaller segments within a sentence that can explain opinions. For example, a sentence may be verbose or include functional phrases such as ‘I love the fact that ...’ and ‘my friends said ...’. Some part of the sentence even may not be relevant to the given topic. For instance, a sentence, ‘The pictures are beautiful, and the camera is so small you can carry it anywhere’, could be classified as an opinion sentence of ‘camera size’ topic. However, the first half of the sentence is not related to the size of camera. It is always desirable to use minimum display space to show maximum amount of useful information, which is especially necessary when displaying summaries on the small space such as a smart phone.

In this work, we introduce a novel summarization problem called ‘compact’ explanatory opinion summarization (CEOS) which extracts explanatory text segments from sentences. We can regard the CEOS problem as extracting text segments to answer the question about why reviewers hold a certain opinions. Thus, a main difference between CEOS and a regular summarization problem is that in CEOS we emphasize on selecting texts that provide an explanation of why an opinion holder has a particular polarity of sentiment about an entity, whereas in regular summarization, there is no guarantee of the explanatoriness of a selected segment.

Two main technical challenges in solving the CEOS problem are: 1) to find the boundary of an explanatory text segment and 2) to assess the explanatoriness of the segment in giving a reason for the given sentiment. We focus on studying how to solve these problems in a systematic and unsupervised way as such a method would be generally applicable to many domains without requiring manual effort. In addition to the representativeness of information which is a main criterion used in the existing summarization work, we also consider discriminativeness with respect to background information, which can help avoid selecting segments about general superficial discussion of an entity without explanation. We propose several general new methods for identifying explanatory text segment boundary and scoring the explanatoriness of the text segment using on a Hidden Markov Model (HMM) approach and other likelihood-based probabilistic models used in a generate-and-test approach. Our focus is on statistical methods that are applicable to large-scale data without the need of training sets and that may even be generalized to multiple languages. Thus, in this work we do not consider linguistic nor semantic features that would add complexity and make the approach language and domain specific.

To evaluate the proposed explanatoriness scoring methods, we use two new measures, explanatory characters at k characters (EC@k) and explanatory phrases at k characters (EP@k), to measure the perceived utility of a summary from a user's perspective. Since the task of compact explanatory opinion summarization is new, there does not exist any data set that we can use as reference for evaluation. We thus created two new data sets in two different domains, respectively, to evaluate this novel summarization task. Experimental results show that the proposed methods are effective in selecting explanatory text segments, outperforming a representative regular text summarization method, and that HMM-based explanatory text extraction is the most effective method for generating an explanatory opinion summary.

The main contributions of this work include:

1. We introduce a novel summarization problem called compact explanatory opinion summarization (CEOS), where the goal is to select informative within-sentence text segments that can *explain* why a certain sentiment polarity of opinions is held by reviewers.

2. We propose two general strategies for solving the CEOS problem, including a generate-and-test approach based on parse-tree search and probabilistic scoring, and a unified Hidden Markov Model approach.

3. We define new measures and create two new data sets for evaluating this new task, which will be made publicly available.

4. We evaluate all the proposed methods to understand their relative strengths and show that they are more effective than a regular summarization method in terms of our major measure of the performance for solving the CEOS problem.

The rest of sections are organized as follows. In Section 3.2.2, we discuss previous works. In Section 3.2.3, we motivate a problem formulation and formally describe the problem. In Section 3.2.4, we overview a proposed method.

In Section 3.2.5, we explain the two-step strategy for explanatory summarization and propose detailed techniques for each step. In Section 3.2.6, we discuss a unified one-step strategy for explanatory summarization using HMM. In Section 3.2.7, we show experimental setup and results, and we make a conclusion in Section 3.2.8.

3.2.2 Related Work

In the information retrieval area, there is work to retrieve relevant passages instead of documents, called *passage retrieval*. Passage retrieval techniques can provide better output to users by filtering out nonrelevant parts of a document, and it can also be useful to improve general retrieval performance. One of the sophisticated methods uses Hidden Markov Model (HMM) [32]. By modeling words in documents with 'relevant' and 'background' state of HMM, they can naturally capture boundaries between relevant and nonrelevant passages to the query. Although passage retrieval has similar challenges to explanatory summarization such as finding relevant passage boundary and measuring relevancy of phrases, its focus is not on finding an 'explanatory' phrase, but a 'relevant' passage to query. In this dissertation, we show how we adapt the HMM-based passage retrieval technique to address explanatory opinion summarization.

3.2.3 Problem Formulation

Our problem formulation is based on the assumption that existing opinion mining techniques can be used to find subtopics (sentiment polarity on aspect). We hope to further help users digest the opinions expressed in a set of sentences with a certain sub topic by extracting a set of *explanatory* within-sentence segments that can provide specific reasons of opinions.

Thus, as a computational problem, the assumed input consists of two sets of sentences; a sentence set O to be summarized and a background sentence set T which is a superset of O; for example, the opinionated sentence set O about 'picture quality of camera' and the background sentence set T about 'camera'. In real application scenarios, when we have a data set T, existing opinion mining techniques would be able to classify sentences in T into subtopic clusters by aspects or opinion orientation. We can use one of the clusters as O.

Given T and O as input, the desired output is an explanatory opinion summary S, which is a subset of sentences segments of O, i.e., $S = \{s_1, ..., s_m\}$ such that every $s_i \subset o_i$ is an explanatory text segment that gives a specific reason why the reviewer has the particular polarity of sentiment. s_i is any consecutive sequence of words in any sentences in O, that is, it can be an entire sentence or a part of one sentence in O. The summary, S, would fit a display with k characters, i.e., $\sum_{i=1}^{m} |s_i| \leq k$, where $|s_i|$ is the length of sentence s_i.

3.2.4 General Approach

Given a boundary detection method and an explanatoriness scoring function $E(s)$, we can use the following algorithm to generate an explanatory opinion summary.

1. Find the best explanatory segment from each input sentence in O, i.e., $s_1, s_2, ..., s_n$.
2. Compute the explanatoriness scores of all the segments, i.e., $Score_E(s_1), Score_E(s_2), ..., Score_E(s_n)$.
3. Rank all the segments $s_1, s_2, ..., s_n$ in descending order based on $Score_E(s_i)$.
4. Let $S = \{\}$. Add the top ranked segment s_k to S
5. Find the next best explanatory segment from the remaining part of the sentence that generated the selected explanatory segment.
6. Repeatedly rank, extract, and add the top ranked segment to S until no additional text segment can be added to S without violating the constraint $\sum_{s \in S} |s| \leq k$. That is, $S = (s_1, ..., s_m)$ such that $\sum_{s \in S} |s| \leq k$, while $\sum_{s \in S} |s| + |s'_{m+1}| > k$.

The final segment, s'_{m+1} may not fit within character limit. In this case, we can use the part of the sentence that will fill up to k character limit because even a partial text segment may deliver explanatory details. By doing this, we do not give preference to methods that tend to extract shorter segments. In contrast, if we did not allow filling up the remaining space with a partial segment, methods which prefer longer segments would have more empty space and would tend to have lower score than methods that prefer short segments. It is also good for fair comparison because this strategy would not prefer specific methods such as long-segment-preferable method. That is, if we do not allow fill up the remaining space, methods which prefer long segments would have more empty space and tend to have lower score than methods which prefer short segments.

Clearly, the main challenge in making this algorithm work is to design a method for explanatory text boundary detection finding good candidate segments and a good explanatoriness scoring function $Score_E(s)$, which we discuss in the next section. Note that the algorithm above can be modified in a straightforward way to accommodate redundancy removal by using existing techniques such as Maximal Marginal Relevance [13] or clustering. However, we intentionally do not explore these variations in this dissertation in order to focus on studying the novel research challenge in extracting explanatory text segments, which has not been studied before. Specifically, we propose two general strategies for solving this problem, which are discussed in the following two sections, respectively.

3.2.5 Generate-and-Test Approaches

An obvious way to solve the CEOS problem is to first generate candidate text segments, and then rank them by explanatoriness. In this section, we propose several methods for each step.

Probabilistic Models for Explanatoriness Scoring

We first propose two heuristics that may be potentially helpful for designing an explanatoriness boundary detection technique and a scoring function.

1. Popularity and representativeness: A text segment is more likely explanatory if it contains more terms that occur frequently in all the texts in O. This intuition is essentially the main idea used in the current standard extractive summarization techniques.

2. Discriminativeness relative to background: A text segment with more discriminative terms that can distinguish O from background information is more likely explanatory. Intuitively, an explanatory segment would more likely contain terms that can help distinguish the set O of text segments to be summarized from more general background texts which contain opinions that are not as specific to the given topic as those in O. That is, we can reward a text segment that has more discriminative terms, i.e., terms that are frequent in O, but not well covered in T.

To implement the proposed heuristics in a more principled way, we further propose a new probabilistic model for the explanatoriness scoring function, $Score_E(s)$. Our basic idea here is to try to develop a generative model to model explanatory text segments and nonexplanatory text segments, respectively. We can then score a text segment based on the probability that it has been generated by the explanatory model as opposed to the nonexplanatory model. In general, if we use the text segments to be summarized (O) to estimate the explanatory model, it would capture popularity, while using a background set (T) to estimate the nonexplanatory model would enable capturing discriminativeness. We assume that a segment $s = w_1 w_2 ... w_n$ can be either explanatory or not, denoted by $E \in \{0, 1\}$. We can thus score the explanatoriness of the text segment S based on the conditional probability $p(E = 1|s) = p(E = 1|w_1 ... w_n)$, which can be interpreted as the probability that the text segment S is explanatory. According to Bayes rule, we have

$$p(E=1|w_1...w_n) = \frac{p(w_1...w_n|E=1)p(E=1)}{p(w_1...w_n)}$$
$$p(E=0|w_1...w_n) = \frac{p(w_1...w_n|E=0)p(E=0)}{p(w_1...w_n)}$$

Since ranking text segments based on $p(E=1|s)$ is equivalent to ranking them based on $\frac{p(E=1|s)}{p(E=0|s)}$,

$$\frac{p(E=1|s)}{p(E=0|s)} = \frac{p(w_1...w_n|E=1)p(E=1)}{p(w_1...w_n|E=0)p(E=0)}$$
$$\propto \frac{p(w_1...w_n|E=1)}{p(w_1...w_n|E=0))} = \prod_{i=1}^{n} \frac{p(w_i|E=1)}{p(w_i|E=0)}$$

Next, we refine the general model of content of the segment (i.e., $w_1...w_n$) using a unigram language model and taking logarithm of both sides to obtain the following explanatoriness scoring function:

$$Score_E(s) = \sum_{i=1}^{n} \log \frac{p(w_i|E=1)}{p(w_i|E=0)}$$

$p(w_i|E=1)$ and $p(w_i|E=0)$ are the unigram probability parameters for word w_i in the explanatory and nonexplanatory content generative models, respectively. We refer to this scoring function as Segment Likelihood Ratio (**SLR**).

For estimating $p(w|E=1)$ and $p(w|E=0)$, without additional knowledge, a reasonable assumption is that the set of text segments to be summarized O can be used as an approximate of sample of words that are explanatory, which is to say that we will use O to approximate the explanatory source. Likewise, with a background data set T, we can approximate the nonexplanatory source. This assumption also corresponds with our basic heuristics about 'explanatoriness'. That is, discriminative content is popular in O, but not in T. With maximum likelihood estimate, we have

$$p(w|E=1) = \frac{c(w,O)}{|O|}, p(w|E=0) = \frac{c(w,T)}{|T|}$$

where $c(w,O)$ is the count of word w in the set O.

Explanatory Segment Candidate Generation

Another important component of the generate-and-test approach is to find good explanatory text segment candidates. Finding meaningful boundaries is the main goal of this step.

One simple way of defining boundary is using a maximum boundary, the entire sentence (**SenOrder**). In this case, we just follow boundaries of the given segmentation by the input data set, and the problem would be the same as a sentence ordering problem by our scoring functions. As we discussed before, sentence may have unnecessary information for explanatoriness.

The opposite extreme of sentence boundary would be 'searching all the possible candidates'. We may use any subsequence of words as candidate boundaries, so called exhaustive search (**Exhaustive**). That is, we evaluate all the possible lengths of extraction by our scoring function, and choose the best scored text segment as our extraction.

Ideally, if we had the perfect scoring function that could make fair and comparable measurement on explanatoriness, this method should have the best performance. In reality, however, because the explanatoriness scoring function is not perfect, it might choose some meaningless boundaries suggested by the exhaustive search. Also, this method is inefficient because there would be $n!$ number of possible candidates for the n-word text segment, and the arbitrary boundary may not be able to extract meaningful phrases.

Another way to find a good explanatory text segment boundary is using the syntax structure given by a parse tree (**ParseTree**). It is likely that explanatory phrase boundary matches with syntax boundary. So, we can use subtrees of parse tree as our text segment candidate. We start from leaf nodes of a parse tree and search all the subtrees up to the entire sentence.

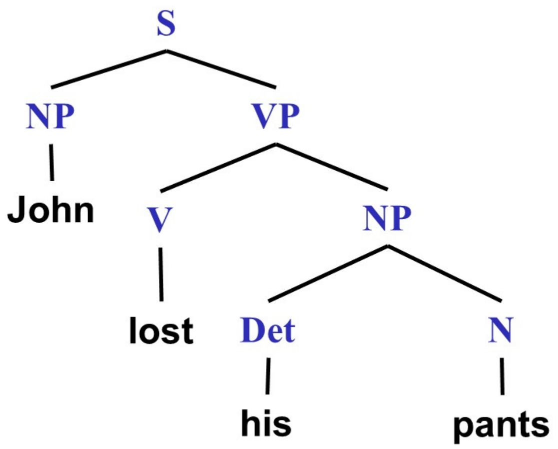


Figure 3.2: Example parse tree for 'John lost his pants'.

For example, if we have a parse tree like Figure 3.2, we would have seven total candidate segments; four one-word candidate segments from leaf nodes (*John*, *lost*, *his*, *pants*), two segments from sub trees (*his pants*, *lost his pants*), and one entire sentence candidate (*John lost his pants*).

By using parse tree, we can shrink the number of candidate segments to evaluate compared to the exhaustive search, and they are also more likely to have meaningful boundaries.

3.2.6 HMM-based Explanatory Text Segment Extraction

While previous methods have two separate steps including candidate segment generation and explanatoriness scoring, in this section, we propose a unified explanatory text segment extraction framework using Hidden Markov Model (HMM).

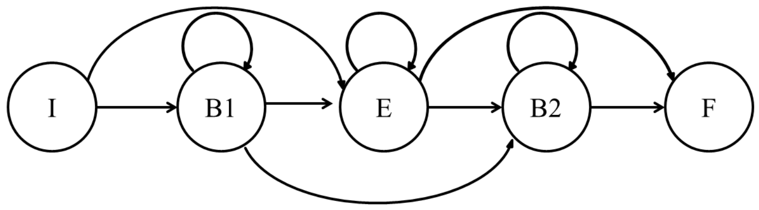


Figure 3.3: HMM structure for explanatory text extraction.

Basic idea of HMM-based explanatory text segment extraction

We propose a HMM-based explanatory text segment extraction method in similar way to the passage retrieval technique in information retrieval [32]. Figure 3.3 shows an HMM structure for modeling explanatory texts. State $B1$ and $B2$ are background (nonexplanatory) states, E is an explanatory state, I is an initial state, and F is a final state, respectively. I and F state are for start and end, other states' output is a word, and the set of output symbols is the word vocabulary of the text collection. Each word has a certain probability (including zero) of being emitted from each state. Arrows between states indicate nonzero transitions probabilities from one state to the other. The HMM structure in Figure 3.3 models the situation where an explanatory phrase (E) is surrounded by nonexplanatory (background) phrases ($B1$, $B2$). Although both $B1$ and $B2$ have basically the same functionality generating nonexplanatory words, we need both of them because there can exist nonexplanatory words before as well as after an explanatory phrase in a sentence. $B1$ would capture a nonexplanatory phrase before the explanatory phrase, and $B2$ would capture a nonexplanatory phrase after the explanatory phrase. Because an entire input sentence can be explanatory, transition probability from I to E and E to F are nonzero. Likewise, an entire input sentence can be nonexplanatory; thus, transition probability from the $B1$ to $B2$ is nonzero. Also, transition probabilities from each state (except I) into itself are nonzero because they can generate phrases of more than one word.

Let $p(w|X)$ be the output probability of word w in state X, $p(X_j|X_i)$ be the transition probability from state X_i to X_j, and $p(X_1)$ be the initial probability of state X_1. For each sentence, $s = w_1 w_2 ... w_n$, from the input data set O, we want to find the state sequence X^* which has the highest likelihood, $p(s|HMM)$.

$$X^* = argmax_{X=X_1...X_n} \ p(X_1)p(w_1|X_1)\prod_{i=1}^{n-1} p(X_{i+1}|X_i)p(w_{i+1}|X_{i+1})$$

where $X_i \in \{B1, B2, E, I, F\}$. The state sequence of S^* would be like $IB1...B1E...EB2...B2F$. Then, output sequence generated by the state E would be the candidate explanatory text part, s, within sentence. For example, with a sentence "The retina display is very clear", if we suppose "retina display" is an explanatory phrase, and "The" and "is very clear" is not. We would have state sequence like $IB1EEB2B2B2F$. The explanatory segments from the sentences of O are ranked by explanatoriness, and top ones are used to generate the summary as described by the algorithm in Section 3.2.4.

In this approach, we can handle explanatoriness scoring of candidate text segments using the background HMM. With the same structure of HMM, we can force it not to enter the explanatory state by setting all the incoming transition probabilities to explanatory state to zero; we can call this background HMM. That is, we can set the initial probability of E state and the transition probability from $B1$ to E to zero. With this setup, we can estimate parameters of HMM in the same way and find the likelihood of the input sentence s. Because the background HMM always stays in background states, the output value is the likelihood that the given text segment is generated by the background, $p(s|BackgroundHMM)$. Finally, by comparing the likelihood of the original HMM and the background HMM, we can measure how much the text segment is explanatory to the background.

For a given text segment t, the explanatoriness score by HMM would be defined as follows

$$Score_E(t) = \frac{p(t|HMM)}{p(t|BackgroundHMM)}$$

We refer to this basic HMM-based explanatory extraction method as **HMM**$_E$.

Estimating output and transition probabilities

The output distribution of state E is the explanatory language model, and the output distribution of state $B1$ and $B2$ is the background language model. As discussed before, we can say that information that is popular in the input data set but not in the background data set is discriminative. In our case, we can use the entire input sentence set, O, to estimate the explanatory language model, and the entire texts in topic T as the background data set to estimate background language model.

$$p(w_i|B) = \frac{c(w_i, T)}{|T|}, p(w_i|E) = \frac{c(w_i, O)}{|O|}$$

where B corresponds to the background states $B1$ and $B2$, and E is explanatory state, and $c(w, C)$ is the count of word w in word collection C.

Once the output probabilities are set, we can learn the transition probabilities of the HMM from an observed sequence such as each sentence of the input data set. We could learn the probabilities from training data. However, since our focus is on unsupervised methods, we can learn the probabilities in unsupervised learning using Baum-Welch algorithm on the input sentence as the only observed sequence. In this way, we can use this algorithm without training data.

We have seen that the explanatory model parameters for both segment likelihood ratio and HMM probabilities are estimated from input sentences. However, input opinion sentences may contain segments that are not useful for explanations such as “I love the fact that...”. Someone may doubt the proposed method would work well because parameters are estimated based on sentences containing noisy elements. In spite of this, the proposed method works on sentences containing such noisy elements because we use a background model for discrimination, as explained in Section 3.2.5. A non-explanatory element like “I love the fact that” would be penalized because its words are more common in the background set than words in explanatory phrases. Our methods are designed to favor a phrase with terms frequent in the input opinion set, but not common in the background set. In the same way, in HMM, those noisy elements are more likely to be captured by background states.

Smoothing on explanatory language model

The basic model above uses maximum likelihood estimator to model an explanatory language state. Because the vocabulary of input data set is not big enough compared to the one in the background data set, the estimated output probabilities may be too big compared to those in the ideal explanatory language model (which would include all possible explanatory sentences). In addition, because the current observing sentence used for transition probability estimation is a part of O, there is a concern that the trained HMM may overfit to the explanatory state. That is, it may stay at the explanatory state too long.

One easy way to avoid this problem is excluding the observing sentence in modeling explanatory language model (**HMM**$_E$**-Rmv**) . That is, when we estimate explanatory language model for s_i, we can use other sentences in O.

A more formal method for avoiding overfitting is to smooth the explanatory language model. One way to smooth is using Laplacian [91] smoothing that adds uniform weighting to each word (**HMM**$_E$**-LP**). The smoothed model can be defined like the following.

$$p'(w_i|E) = \frac{c(w_i, O) + \delta}{|O| + |V_O|\delta}$$

where V_O is vocabulary in O, and δ is a parameter controlling the strength of Laplacian smoothing.

Another way to smooth explanatory state is using Dirichlet [91] smoothing to background language model (**HMM**$_E$**-Dir**). While Laplacian smoothing just decreases word probability in explanatory language model, Dirichlet smoothing decreases the gap between explanatory state and background states by mixing them. Although this may be closer to the reality, one possible disadvantage of this approach is that our explanatory state becomes similar to background states, then HMM's power to find explanatory segments may be weakened.

$$p'(w_i|E) = \frac{c(w_i, O) + \mu_0 p(w_i|T)}{|O| + \mu_0}$$

where μ_0 is a parameter controlling strength of Dirichlet smoothing.

Pseudo Feedback on explanatory language model

We can enhance the explanatory language model by adding words from the initial extraction. This is similar to pseudo feedback in information retrieval. We assume the extracted text segments are explanatory (pseudo-explanatory), and use them to modify the initial explanatory language model (**HMM**$_E$**-FB**). From the first run of HMM over all the input sentences in O, we would have initial text segment extraction results, and using the extracted words we can also produce a feedback language model E^*. We can smooth the current explanatory language model with this pseudo explanatory model.

$$p'(w_i|E) = \frac{c(w_i, O) + \mu_1 p(w_i|E^*)}{|O| + \mu_1}$$

where μ_1 is the parameter controlling strength of feedback.

3.2.7 Experiments

Data Set

For experiments, we use the data set that we generated in Section 3.1.4. As a preprocessing step, the input data set is first processed by a basic stemmer to relieve sparseness issues.

Because our evaluation should be 'with-in' sentence level, we additionally label explanatory text segments on the data set. Our key evaluation question is whether the proposed methods can really find explanatory text segments. Therefore, for gold standard, we asked two human labelers to mark explanatory parts of each sentences. We provided the topic description of each cluster (product type, aspect, and sentiment orientation), and asked them to mark any part of each sentence that explains a reason of the opinions in the given topic. They are asked to mark segments with explanatory information that is relevant to the given topic and is not directly inferable from the topic label. For example, one of actual labels by an annotator about 'negative opinion about MP3player case' was "the case is strong and stylish , but unfortunately <exp>lacks a window (now a big deal) .</exp>". The labeler did not include functional words such as 'unfortunately' and non-topic-relevant phrase 'strong and stylish' which has opposite sentiment orientation to our target topic. They can mark as many parts as they want for each sentence. If two explanations were connected by 'and', and each explanation could be independently useful for readers, then the labelers would mark two separate text segments. The data sets with detailed labeling instructions, examples provided to the labelers, and the generated labels will become publicly available [5].

Basic statistics with new labeling about the two data sets are in Table 3.6. Average explanatoriness is the average ratio of the number of characters marked as explanatory by labelers to the entire number of characters for each data set. Label agreement is the ratio of the number of characters having consistent label to the total number of characters

[5] `http://sifaka.cs.uiuc.edu/ir/data/expSum/`

for each data set. In the evaluation results below, we measure performance for each label and report average value.

	PRODUCT	HOTEL
# of topic	66	23
Avg size of topic	21.9	102.3
# of sentences	1447	2352
# of unique word	4377	2970
Average explanatoriness	0.32	0.35
Label agreement	0.74	0.71

Table 3.6: Data set for evaluation.

Measures

In our problem setup, we use the following new measures, called explanatory characters at k characters (denoted as EC@k) and explanatory phrase at K characters (denoted as EP@k).

The idea of EC@k measure is to capture the perceived utility of CEOS when displayed on a screen with room for only k characters. The perceived utility is defined as the percentage of characters displayed that are parts of explanatory phrases. Formally, suppose $S = (s_1, ..., s_n)$ is an explanatory opinion summary with n text segments from original sentence set O, and $E = (e_1, ..., e_m)$ is a gold standard explanatory text segments of input sentences O. We have

$$EC@k(S,E) = (\frac{\sum_{s \in S} \sum_{e \in E} |s \cap e|}{k}) \frac{1}{idealEC@k(E)}$$

$$idealEC@k(E) = \frac{min(\sum_{e \in E} |e|, k)}{k}$$

Here, idealEC@k(S) is the maximum possible score. This normalization makes EC@k(S) score more interpretable. That is, EC@k(S) = 1.0 means achieving the best possible performance. Also, the normalization makes this measure comparable between different data sets; thus, we can report mean EC@K which is averaged performance over all input topics, and the averaged score would not be dominated by too difficult (there is no explanatory text) or easy (almost all parts of input sentences are explanatory) topics.

One concern of EC@k measure is that partial extraction of an explanatory segment can be over rewarded. EC@k measures overlapped length between extracted segments and gold standard explanatory segments. Therefore, when the extracted segment is part of the original explanatory segment, it would still get partial score in proportion to the overlapped character length. However, sometimes, partial extraction of an explanatory segment may lose original information it had. For example, assume that the input sentence is ‘the camera is so small you can carry it anywhere’, and ‘so small you can carry it anywhere’ is identified as an explanatory segment. If an algorithm extracted ‘small you can carry it anywhere’, it still can deliver meaningful explanation. However, if another algorithm extracted only ‘you can', it cannot deliver any good explanation, but it would still have partial score.

To relieve this problem, we propose EP@k as a supplementary measure, which only reward full extraction of an explanatory segment. That is, only when we extract the full explanatory phrase labeled by humans, it will be counted as good extraction, and we do not give any score for partial extraction.

$$EP@k(S,E) = (\frac{\sum_{i\in S}\sum_{e\in E}\delta(s\supset e)|s\cap e|}{k})\frac{1}{idealEP@k(E)}$$

$$idealEP@k(E) = \frac{min(\sum_{e\in E}|e|, k)}{k}$$

where δ is an indicator function, and we use the same normalization technique using the ideal score.

EC@k that shows the general utility of explanatory summarization will be our main measure, and EP@k will be also reported as a supplementary measure which is stricter and can show the different aspect of utility. In our experiments, for parameter k, we chose 150 to obtain short summaries similar in length to twitter, and 500 that is the length of about 100 words.

Although ROUGE is a popular measure for summarization, our task is more similar to a ranking problem than a standard summarization problem since our goal is to rank candidate segments based on how likely they are explanatory. Such a ranking task is more naturally evaluated using our proposed measures instead of ROUGE. The main technical challenge in the new summarization problem is to extract segments and score their explanatoriness; we focused on evaluating only this component in order to answer the research question about what is the best way of modeling explanatoriness without being affected by other factors such as redundancy.

ROUGE also has problem in matching contents with different expressions. For ROUGE evaluation, we would compare our outputs with the same k length of gold standard human summaries. Even though our gold standard summaries only use existing phrases (i.e. no paraphrasing), there can still exist different expressions for the same information, and probably only one of them will be included into the gold standard summary. Thus, two system summaries with the same information but using different expressions may be assigned different performance measurements by ROUGE depending on which expression was included into the gold standard summary. In this case, using ROUGE may not be able to distinguish which method is better for finding explanatory phrases. We could use concatenation of all the explanatory sentences as a gold standard summary. However, the ROUGE score would not be within $[0.0, 1.0]$, thus scores will be neither easily interpretable nor comparable to others with different length of model summaries.

Actually, we can think EC@k is similar to ROUGE-1 using entire explanatory summary as gold standard and score normalization. However, still ROUGE does not have any variation such as our EP@k measure, which is useful for our problem set up. Therefore, instead of using ROUGE measure, we propose EC@k and EP@k which fits better for measuring explanatoriness extraction and ranking.

Baselines

We hypothesize that the proposed summarization method would work better than a standard text summarization method that mostly implements the popularity heuristic. We use **LexRank** summarization algorithm as a baseline to test this hypothesis.

Another baseline for explanatory scoring function is TF-IDF-based scoring function from information retrieval such as BM25 [34], which is one of the most effective basic information retrieval functions. Indeed, our popularity heuristic can be captured through Term Frequency (TF) weighting, while the discriminativeness can be captured through Inverse Document Frequency (IDF) weighting. We can modify BM25 for our explanatoriness scoring (**BM25**$_E$). For explanatoriness ranking, we can consider a sentence as a query and measure explanatoriness of each word of the sentence based on how frequent the word is in the input data set (O) and the background data set (T). Specifically, given a text segment $s = w_1, w_2, ..., w_n$, $BM25_E$ would be defined as:

$$BM25_E(s, O, T) = \sum_{w \in s} IDF(w, T) \frac{c(w, O)(k_1 + 1)}{c(w, O) + k_1(1 - b + b\frac{|T|}{avgdl})}$$

$$IDF(w, T) = log \frac{|T| - c(w, T) + 0.5}{c(w, T) + 0.5}$$

where $c(w, O)$ is frequency of w in data set O, $|T|$ is the total number of term occurrences in data set T, and $avgdl$ is the average number of total term occurrences of subclusters in T that O is extracted from. In our experiment, k_1 and b are set as 1.5 and 0.75 respectively, which are recommended values in literature [80].

Note that none of the other existing methods discussed in Chapter 2 is designed to solve the same problem as ours, thus they cannot be used as baseline. Although they cannot be directly comparable to our method, some related previous techniques are partially relevant to our problem. Basic linguistic analysis such as parse tree search and HMM-based phrase extraction are adapted and incorporated into our methods. Therefore, instead of showing them as separate baselines, we will have a chance to compare them with other methods we propose.

Moreover, our emphasis on solely unsupervised approaches is very well justified since such an approach can in principle be applicable to any domain without requiring any manual effort. In contrast, a supervised learning approach would always require manual effort. Any previous work with supervised learning is not comparable to our methods. Moreover, the prediction results of our method can be easily incorporated into a supervised learning method as an additional feature to improve performance in case there is training data.

Table 3.7 shows the list of proposed methods, baseline, and their labels.

TYPE	SUB-STEP	TECHNIQUE	LABEL
Standard summarization		LexRank	LexRank
Generate-and-Test	Candidate generation	Sentence boundary	SenOrder
		Exhaustive search	Exhaustive
		Parse tree search	ParseTree
	Explanatoriness scoring	TF-IDF-based method	BM25$_E$
		Segment likelihood ratio	SLR
Unified HMM-based method		Basic HMM-based explanatory extraction	HMM$_E$
		Remove observing sentence from explanatory language model	HMM$_E$-Rmv
		Laplacian smoothing on explanatory language model	HMM$_E$-LP
		Dirichlet smoothing on explanatory language model	HMM$_E$-Dir
		Positive pseudo-feedback	HMM$_E$-FB

Table 3.7: Proposed method summary for compact explanatory summarization. The list of proposed methods and their labels.

	PRODUCT				HOTEL			
SCORING METHOD	EC@150	EC@500	EP@150	EP@500	EC@150	EC@500	EP@150	EP@500
HMM$_E$	0.334	0.383	**0.136**	**0.255**	0.423	0.360	0.125	0.167
HMM$_E$-Rmv	**0.381**	0.378	0.114	0.068**	**0.519**	**0.496****	0.103	0.156
HMM$_E$-LP	0.347	**0.409****	0.124	0.209**	0.334	0.388	0.037**	0.085**
HMM$_E$-Dir	0.294	0.160**	0.003**	0.002**	0.356	0.327	0.011**	0.004**
HMM$_E$-FB	0.337	0.386	0.132	0.254	0.405	0.365	**0.144**	**0.169**

Table 3.8: Comparison of variations of HMM-based methods. Compared to the HMM$_E$, significantly different values with 95% confidence level is marked as **, and those with 90% confidence level is marked as *.

	PRODUCT				HOTEL			
SCORING METHOD	EC@150	EC@500	EP@150	EP@500	EC@150	EC@500	EP@150	EP@500
SenOrder + BM25$_E$	0.292	0.349	0.138	0.263	0.271	0.290	0.097	0.189
Exhaustive + BM25$_E$	0.292	0.349	0.138	0.263	0.271	0.290	0.097	0.189
ParseTree + BM25$_E$	0.296	0.345	0.143	0.262	0.271	0.290	0.097	0.189
LexRank	0.329	0.388	**0.181**	**0.288**	0.390	0.350	**0.199**	**0.238**
SenOrder + SLR	0.334	0.386	0.153	0.282	0.425	0.354	0.129	0.198
Exhaustive + SLR	0.329	0.388	0.144	0.256	0.434	0.362*	0.125**	0.156**
ParseTree + SLR	0.321	0.389	0.151	0.271	0.430	0.371	0.129	0.200
HMM$_E$-Rmv	**0.381****	0.378	0.114**	0.068**	**0.519****	**0.496****	0.103**	0.156**
HMM$_E$-LP	0.347	**0.409**	0.124**	0.209**	0.334	0.388	0.037**	0.085**

Table 3.9: Comparison of various methods for scoring explanatoriness. For the bottom 5 rows, compared to strong baseline, LexRank, significantly different values with 95% confidence level is marked as **, and those with 90% confidence level is marked as *.

Comparison of different HMM-based methods

We first examine different variations of the proposed HMM-based method to understand which variation works the best. We proposed a basic HMM-based method with variations such as smoothing and pseudo-feedback. Table 3.8 shows comparison of these variations with significance test by t-test. The proposed basic HMM method has no parameter to set. The parameters for smoothing and feedback with HMM were set to reasonable values based on their interpretations ($\delta = 10000$, $\mu_0 = 10000$, and $\mu_1 = 5$). Although we use fixed values with unsupervised learning setup, further tuning them will possibly lead to further improvement of performance.

In general, HMM$_E$-Rmv and HMM$_E$-LP show better performance in EC and worse performance in EP than HMM$_E$. We performed significance test between HMM$_E$ and HMM$_E$-Rmv/HMM$_E$-LP, and some EC values of HMM$_E$-Rmv/HMM$_E$-LP showed significant improvement from HMM$_E$ with 95% confidence level. Observing sentence removal or Laplacian smoothing avoid overfitting, so they would increase performance in EC. However, avoiding over fitting would generate shorter text segments; thus, it is more likely to have lower EP score.

HMM$_E$-Dir did not show any improvement neither in EC nor EP. To further understand the effectiveness of Dirichlet smoothing, we tried to change the strength of the smoothing parameter. Actually, increasing the smoothing parameter gave us worse performance. Dirichlet smoothing mixes background language model to explanatory language model. Although it may have some effect on avoiding overfitting, it dilutes the difference between the explanatory state and a background state and make the HMM less powerful.

HMM$_E$-FB shows similar or slightly worse performance in EC, but better performance in EP than basic HMM$_E$. Feedback increases probability of initially retrieved words in explanatory language model. Therefore, the modified HMM would tend to attract similar words around the initially retrieved words; thus, it is more likely to retrieve the longer phrase as an explanatory candidate. In this way, with a little sacrifice on EC, it can achieve higher EP.

Comparison of different methods

We now compare the two strategies we proposed (i.e., generate-and-test and HMM) with baseline methods. Table 3.9 shows the performance of baselines and proposed methods.

In most cases, our proposed methods show higher EC values than baselines. We can see that our methods clearly outperform the direct adaptation of TF-IDF (BM25$_E$). Even compared to the strong baseline LexRank, HMM$_E$-Rmv shows significant improvement in EC values.

Different candidate generation methods (SenOrder, Exhaustive, ParseTree) show similar performance. Exhaustive search shows significant performance difference from LexRank.

In general, we can see that there is a clear trade-off between EC and EP values. Usually, shorter text segment extraction increases EC and decreases EP. In general, our HMM-based methods show relatively lower EP values. As we discussed before, because EP measure is very strict, low EP values does not always mean it could not retrieve meaningful text. We can confirm this fact from example results in the following section. For more accurate evaluation, we may need manual observation of 'partial' extraction to see if they still convey the explanations which the original labeled segment had.

EXPLANATORY SUMMARY	BASELINE SUMMARY
Its **convenient location to both [Attraction A] and [Attraction B]** makes it worth the. as you can come back to the hotel to take a nap. The [Hotel Name] s location can not be beat. location with its own **private entrance to the [Attraction B] theme park and within easy walking**. to pay for this unbeatable location. location of the hotel was perfect. location is great especially if you have. You ca n t beat the. Great location. worth being so close to the parks. Downtown. As. Awesome.	The location was **great**. The location for this hotel is **wonderful**. The rooms were fine and the location was **great**. The location of the hotel was **perfect**. There is **no better Hotel location** than this one. The location can not be beat. The hotel was a little pricey but worth it considering the location. The location is beyond **wonderful**. The location to [Attraction A] was great they should just upgrade some things. The location could n t be beat and every other level of service was great. If you dont mind

Table 3.10: Example summary output comparison between explanatory summary (HMM$_E$-RMV) and baseline summary (LexRank) about *positive opinion about the location of Hotel2*.

EXPLANATORY SUMMARY	BASELINE SUMMARY
looked **moldy** and the lounge chairs looked to be. only complaint was the the **swimming pool was cold**. My **parking situation is a bit confusing**. The **pool is small** for the number of people in the. hotel **elevators get crowded** between. lounge chairs looked like they were once. **Elevators are small and slow**. group. there. pool in July during a heat. it. In. very difficult to walk around the pool area very. Very. **too crowded not enough lounge chairs**. d. I. Finally. children running up and down the hall.	The **pool** is small for the number of people in the hotel. Our **pool** experience was unsatisfactory they have 3 **pools** all of which were full of kids in the afternoon all lounge chairs were taken so we did n t get to go in. The [One of pool name] Pool was closed for refurbishment during our stay but it was n t warm enough to be in the **pool** anyway. The main **pool** is very busy. Also we spent a lot of time at the pool and were disappointed with the fact that the **pool** was almost warm. The one time over 3

Table 3.11: Example summary output comparison between explanatory summary (HMM$_E$-RMV) and normal summary (LexRank) about *negative opinion about the facility of Hotel1.*

Example Result

To make example summaries closer to actual 'summary', we applied the redundancy removal technique, Maximal Marginal Relevance. We pick text segments from the top of the ranked list one by one. However, if the content of the next candidate text segment is covered enough by already selected texts, we skip and do not include the candidate text to the summary. Here, we used threshold 0.5, which means the candidate text is not selected if more than 50% of the content is covered by the already chosen texts. We add texts to the output summary until the summary reaches 500 characters.

Table 3.10 shows example summaries, an explanatory summary generated by the HMM-based proposed method and a normal summary generated by the baseline LexRank, about 'positive opinion about the location of Hotel2'. (We anonymized some names that may reveal the identity of the target hotel.) This example clearly shows the benefits of the explanatory summary over the normal summary. Explanatory summary shows good details about 'location' of the target hotel such as 'convenient location to both [Attraction A] and [Attraction B]', and 'private entrance to the [Attraction B] theme park and within easy walking'. On the other hand, in the normal summary, we can see many repetitions of 'good location' because it is popular contents in the input data. We already know that the topic is about 'positive' opinion of the hotel location. Thus, a large portion of space in the normal summary is consumed by the already known fact from the given topic.

Moreover, some sentences in the normal summary include nonrelevant information to the given topic (location) such as 'The rooms were fine' and 'every other level of service was great'. These contents are actually filtered out in explanatory summary because they would have high frequency in background data that includes all other aspects (e.g. room, service). This also shows why we need within sentence segment extraction.

Table 3.11 shows another example summary comparison about 'negative opinion about the facility of Hotel1'.

In this case, the baseline summary shows some details, but they are only about a pool. For this topic, there were many people complaining about the pool; thus, all the top ranked sentences of LexRank results were dominated by the opinions about a pool even after applying redundancy removal. We additionally examined the entire ranking by LexRank and found out that texts about elevator and parking were ranked too low to be selected as an output summary.

On the other hand, explanatory summary shows details about negative opinions about various facilities such as pool, parking, and elevator. Explanatory summary not only includes details opinions about a pool which the baseline summary contains (pool is small and not warm enough), but it also could catch other complaints about elevator/parking and rank them high, which were discriminative information in the facility topic.

In spite of benefits, explanatory summary still has limitations. As we discussed before, a mistake in boundary detection may result in losing the entire information. For example, in the second example, ‘lounge chairs looked like they were once’ cannot give us information because the segmentation did not include the last part of the sentence. Also, extracting within-sentence segments has risk to lose context. For instance, although we can guess a text segment, ‘Downtown’ in the first example summary, means that the hotel is close to ‘Downtown’, it is not always clear. We can also see some meaningless text segments such as ‘As’, ‘there’, and ‘In’. They are stop words that usually do not deliver meaningful information by themselves. To overcome this limitation, we could introduce simple heuristics to filter out phrases containing only stop words. A more elegant way of handling this problem would be introducing an additional layer of background data set. Those stop words were selected as ‘explanatory’ because they were discriminative between our input data sets, O and T. In addition to the current discriminativeness measurement, if we also check discriminativeness of information to general word corpus, stop words would less likely to be selected as explanatory information.

Overcoming these limitations is remained as our future work. However, most of extracted phrases are still quite readable and informative, and the benefit of getting explanatory information is fairly clear.

3.2.8 Conclusions

In this section, we introduced a novel opinion summarization problem called compact explanatory opinion summarization (CEOS), which extends the capabilities of the current opinion summarization methods to further enable a user to more easily understand why opinion holders have a particular polarity of sentiment on an aspect of a topic. A key challenge in solving this problem is to automatically identify explanatory text segments that can provide an explanation of reasons why a particular polarity of sentiment is expressed. The main subcomponents of this problem are identifying explanatory text segment boundary and scoring explanatoriness of text. We proposed several general methods for each subcomponent including parse tree search and exhaustive search for explanatory boundary detection and segment likelihood ratio probabilistic models for scoring explanatoriness. Also, we proposed a unified HMM-based framework for explanatory text segment extraction.

We created two new test sets for evaluating this problem and used new measures, explanatory characters (EC) at k characters and explanatory phrases (EP) at k characters. Experimental results show that the proposed methods outperform general text summarization methods such as LexRank in EC, and especially HMM-based methods with modified explanatory language model is the most effective in selecting explanatory text segments.

Our work is the first step toward studying the new problem of explanatory opinion summarization. With the two data sets and appropriate measures, we have paved the way for further studying this problem. As future work, we can further explore different ways of estimating the proposed probabilistic models. Using more complex features such as n-grams and frequent patterns can be useful to better capture explanatoriness. It would also be interesting to test the proposed algorithm on more diverse topics to further verify the generality of our algorithm. Furthermore, we can explore abstractive explanatory summarization instead of extractive summarization.

Chapter 4

Generating Contrastive Summaries of Comparative Opinions in Text [1]

4.1 Introduction

The fact that opinionated text often contains both positive and negative opinions about a topic makes it even harder to accurately digest mixed opinions.

For example, some customers may say positive things about the battery life of iPhone, such as “the battery life [of iPhone] has been excellent,” but others might say the opposite, such as “I can tell you that I was very disappointed with the 3G [iPhone] battery life.”[2] Often such contradictory opinions are not caused by poor or wrong judgments of people, but due to the different context or perspective taken to make the judgments. For example, if a positive comment is ‘the battery life is good when I rarely use button’ and a negative comment is ‘the battery life is bad when I use button a lot’, the two comments are really made under different conditions. When there are many such contradictory opinions expressed about a topic, a user would need to understand what the major positive opinions are, what the major negative opinions are, why these people have different opinions, and how we should interpret these contradictory opinions.

Unfortunately although there has been much work on opinion summarization (see, e.g., [54, 70]), most existing work has gone only as far as separating positive and negative opinions about a topic. This summary cannot help a user to further digest the mixed opinions in each aspect. The user still has to read all the individual comments in both the positive and negative groups.

To help people digest such mixed opinions more efficiently, we propose to automatically generate a comparative summary of contradictory opinions. Specifically, given a set of positively opinionated sentences and a set of negatively opinionated sentences (which can be generated using existing techniques of opinion summarization), we would like to extract comparable sentences from each set of opinions and generate a comparative summary containing a set of contrastive sentence pairs. Each contrastive sentence pair consists of a sentence with positive opinions and a comparable sentence with negative opinions, thus enabling a user to understand contradictory opinions effectively. For example, if we can pair up two representative sentences with opposite opinions about the battery life of iPhone, it

[1]Part of this chapter has been published in [46].

[2]These sentences are real examples found by the Products Live Search portal at http://search.live.com/products/.

would help a user to understand possibly different conditions under which the specific polarity of opinions is expressed, and thus better understand why there are both positive and negative opinions about the battery life.

To the best of our knowledge, this summarization problem has not been addressed in the existing work, and we call it contrastive opinion summarization (COS). We formally formulate the COS problem as an optimization problem in which we attempt to find a list of contrastive sentence pairs that can both represent the two sets of opposite opinions well and offer interesting comparisons between positive and negative opinions about the same topical aspect (e.g., battery life). The objective function of the optimization framework encodes two criteria to be applied to choose sentence pairs. One is that a chosen sentence from the set of positive (negative) sentences should represent a major positive (negative) opinion, i.e., there should be many sentences similar to the chosen sentence. We call this criterion *representativeness*. The other is that the paired sentences should be comparable. That is, they should have opposite opinions about a *common* topical aspect. We call this criterion *contrastiveness*.

Intuitively we need different similarity functions to measure representativeness and contrastiveness. While we can generally use an existing sentence similarity function to measure representativeness, we need a new similarity function to measure contrastiveness. We solve this problem by excluding sentimental words from both sentences and then applying a regular similarity function. We also explore how to leverage resources such as WordNet to accommodate matching of words that are semantically related but have different forms.

Exact solution to the optimization problem is generally intractable for realistic applications. We propose two general approximation methods to solve the problem. Both methods are greedy algorithms, corresponding roughly to first maximizing representativeness and then maximizing contrastiveness, or the opposite, i.e., first maximizing contrastiveness and then maximizing representativeness.

Because no existing data can be used directly to evaluate this new summarization task, we opted to create our own test set based on some publically available resources from the previous work [28, 29]. To test the generality of our methods, we further extended the test set by adding an additional case from a different domain.

Experiment results on this test set show that the proposed methods are effective for generating comparative summaries of contradictory opinions.

The contributions of this work are:

1. We propose and define a novel summarization problem (i.e., contrastive opinion summarization).

2. We propose an optimization framework to model and solve this problem.

3. We propose specific methods to solve the optimization problem and generate contrastive opinion summaries.

4. We create the first test set and propose measures for evaluating this novel summarization problem.

5. We run experiments to test the proposed methods and show that the proposed methods are effective.

The rest of this chapter is organized as follows. We discuss related work in Section 4.2. In Section 4.3, we define the novel problem of contrastive opinion summarization. In Section 4.4, we formally model the problem with an optimization framework. In Section 4.5 and Section 4.6, we then present specific methods to refine and solve the optimization problem. We present our experiment design in Section 4.7 and results in Section 4.8. At last, we make a conclusion in Section 4.9.

4.2 Related Work

There were some works on extracting comparative sentences or detecting contradiction in text. In [33], methods are proposed for detecting comparative sentences by checking signal words like 'than'. In [15], the authors structurally analyzed the characteristics of contradiction and suggested heuristic methods for detecting contradiction in text. Some work considered finding contradiction as a binary classification problem. In [83], support vector machine(SVM) is applied on classification of support and oppositions in text, while in [6, 60], methods based on graph representing relationship between texts or authors are used for classifying texts. Although these works proposed methods to find contradictions, they did not directly address the problem of summarizing contradictory opinions, for which we need to model both contrastiveness and representativeness.

Different sentence similarity measures are explored in [1]. The authors compared the performance of word overlap measures, TF-IDF measures, linguistic measures and combination of them over different data corpus, and found that linguistic measures perform the best in finding similar sentences, and TF-IDF measures perform well for deciding whether input sentences are dissimilar or not. We also compared different similarity measures for the COS problem, and our results show that semantic matching appears to be not useful for this task.

4.3 Problem Definition

As discussed in the previous section, the current opinion summarization techniques can separate positive sentences from negative sentences about a topic (e.g., a product feature). We set up our problem as to take these sentences with different polarities as input and further generate a contrastive opinion summary to help users to digest the mixed opinions about the topic.

We thus will assume that we are given two sets of opinionated sentences about a topic, corresponding to positive and negative opinions about the topic, respectively. Our goal is to generate a list of sentence pairs with each pair containing a position sentence and a comparable negative sentence. Such a pair would allow a user to compare comparable positive and negative opinion and thus facilitate digestion of mixed opinions.

To formally define our problem, we first introduce a few basic concepts.

Definition 1 (Opinionated Sentence) *A sentence is an opinionated sentence if it expresses either a positive or a negative opinion. For convenience, we will simply call a positively (negatively) opinionated sentence a positive (negative) sentence.*

Definition 2 (Contrastive Sentence Pair) *A pair of opinionated sentences (x, y) is called a contrastive sentence pair if sentence x and sentence y are about the same topic aspect, but have opposite sentiment polarities.*

For example, x and y may both discuss the battery life of a laptop, but x says that that the battery life is long, while y says that it is short.

We may now define the novel problem of *contrastive opinion summarization*.

Definition 3 (Contrastive Opinion Summarization) *Let $X = \{x_1, ..., x_n\}$ be a set of positive sentences and $Y = \{y_1, ..., y_m\}$ be a set of negative sentences about a common topic Q, where x_i is a positive sentence and y_i is a negative sentence. The task of contrastive opinion summarization (COS) is to generate k* contrastive sentence pairs*: $\{(u_i, v_i)\}$, $i = 1, ...k$, $u_i \in X$, $v_i \in Y$, such that $U = \{u_i\}_{i=1}^k \subset X$ can represent the opinions in X well, and $V = \{v_i\}_{i=1}^k \subset Y$ can represent the opinions in Y well.*

Contradictory Aspect	**Positive**	**Negative**
Contradictory 1	u_1	v_1
Contradictory 2	u_2	v_2
...	...	...
Contradictory k	u_k	v_k

Table 4.1: Illustration of a contrastive opinion summary.

Table 4.1 illustrates how we may display a contrastive opinion summary in a tabular format to facilitate digestion of contradictory opinions. Each pair (u_i, v_i) summarizes a contradictory aspect. A user can use u_i and/or v_i as "entry points" to navigate into relevant discussion about each side of the opinions of the corresponding contradictory aspect.

Intuitively, to generate a good contrastive summary, we would need to match sentences in X with those in Y to discover potential candidate contrastive sentence pairs. At the same time, we also would like to assess which sentences can represent each polarity of opinions well. In the end, we would like to choose sentences from both X and Y that can not only form good contrastive pairs but also represent the corresponding complete set of opinions well. The problem is thus in nature an optimization problem involving multiple criteria. Below we will propose a formal optimization framework for solving COS, which would then use as a roadmap to derive several specific summarization algorithms.

4.4 Optimization Framework

In this section, we formally frame the contrastive opinion summarization problem as an optimization problem. Our optimization framework is based on two basic similarity measures defined on a pair of sentences. The first is to measure the content similarity of two sentences in the same group of opinions (i.e., either both are positive or both are negative). This similarity function allows us to assess which sentences are good representatives of each group. The second is to measure the contrastiveness of a positive sentence and a negative sentence. Since a good pair of contrastive sentences is generally also similar in content (but opposite in sentiment polarity), we also call this measure a *cross* group similarity measure. We formally define these two functions as follows.

Definition 4 (Content Similarity Function) *Given two opinionated sentences s_1 and s_2 with the same polarity, the content similarity function $\phi(s_1, s_2) \in [0, 1]$ measures the overall content similarity of s_1 and s_2.*

Definition 5 (Contrastive Similarity Function) *Given two opinionated sentences u and v with opposite polarities, the contrastive similarity function $\psi(u, v) \in [0, 1]$ measures the similarity of u and v excluding their difference in sentiment.*

Both ϕ and ψ are assumed to be symmetric, i.e., $\phi(s_1, s_2) = \phi(s_2, s_1)$, and $\psi(u, v) = \psi(v, u)$.

With these two functions, we can now define two additional functions that can measure the representativeness and contrastiveness of a contrastive opinion summary $S = \{(u_i, v_i)\}$ ($i = 1, ...k$) of two sets of opinionated sentences X and Y.

Definition 6 (Representativeness) *The representativeness of a contrastive opinion summary S, denoted as $r(S)$, measures how well the summary S represents the opinions expressed by the sentences in both X and Y. It is defined as*

$$r(S) = \frac{1}{|X|} \sum_{x \in X} \max_{i \in [1,k]} \phi(x, u_i) + \frac{1}{|Y|} \sum_{y \in Y} \max_{i \in [1,k]} \phi(y, v_i).$$

Intuitively, if for every sentence $x \in X$, we have at least one u_i with high similarity to x, our summary would represent X well. Similar reasoning can be applied to the set Y. $r(S)$ is simply an aggregation over all the sentences in both X and Y.

Definition 7 (Contrastiveness) *The contrastiveness of a contrastive opinion summary S, denoted as $c(S)$, measures how well each u_i matches up with v_i in the summary. It is defined as the average contrastive similarity of the sentence pairs in S:*

$$c(S) = \frac{1}{k} \sum_{i=1}^{k} \psi(u_i, v_i)$$

A good contrastive opinion summary should intuitively have both high representativeness and high contrastiveness, thus we may cast the problem of contrastive opinion summarization as the following optimization problem:

$$\begin{aligned} S^* &= \arg\max_S \ (\lambda\, r(S) + (1-\lambda)\, c(S)) \\ &= \arg\max_S \left(\frac{\lambda}{|X|} \sum_{x \in X} \max_{i \in [1,k]} \phi(x, u_i) + \frac{\lambda}{|Y|} \sum_{y \in Y} \max_{i \in [1,k]} \phi(y, v_i) + \frac{1-\lambda}{k} \sum_{i=1}^{k} \psi(u_i, v_i) \right) \end{aligned}$$

where $\lambda \in (0, 1)$ is a parameter to control the relative importance of representativeness and contrastiveness with a larger λ indicating more emphasis on the representativeness.

With such an optimization framework, we see that in order to find an optimal contrastive opinion summary, we will need to solve three problems:

1. Define an appropriate content similarity function ϕ.
2. Define an appropriate contrastive similarity function ψ.
3. Solve the optimization problem efficiently.

In the next two sections, we will discuss how we solve these problems.

4.5 Similarity Functions

In this section, we discuss how we implement the two similarity functions ϕ and ψ. Our optimization framework allows us to flexibly implement them in any reasonable way as long as the two similarity functions can be normalized into the same range. This normalization is needed to ensure that the terms of these two functions in the objective function be comparable.

The content similarity function ϕ is meant to be a normal sentence similarity measure applied to two sentences in the same opinion group. In order to consider semantic matching of terms, we define $\phi(s_1, s_2)$ generally as:

$$\frac{\sum_{u \in s_1} \max_{v' \in s_2} \omega(u, v') + \sum_{v \in s_2} \max_{u' \in s_1} \omega(u', v)}{|s_1| + |s_2|}$$

where $\omega(u, v) \in [0, 1]$ is a term similarity function and $|s_1|$ and $|s_2|$ are the total counts of words in sentences s_1 and s_2, respectively.

The idea of this formula is that we would first match every word in each sentence against words in the other sentence to find its best matching score, then take a sum of all the matching scores, and finally normalize the sum by the total number of words in the two sentences. Since $\omega(u, v) \in [0, 1]$, clearly $\phi(s_1, s_2)$ is also in the range of $[0, 1]$.

Depending on how we define ω, we can obtain several different variations of this general similarity function. In this work, we will experiment with the following two natural variants:

1. **Word Overlap (WO):** $\omega_{WO}(u,v) = 1$ iff $u = v$, and $\omega_{WO}(u,v) = 0$ otherwise. In this case, ϕ would be essentially the Jaccard similarity function that only considers word overlap.

2. **Semantic Word Matching (SEM):**: $\omega_{SEM}(u,v) = 1$ if $u = v$, and $\omega_{SEM}(u,v) = \gamma sim(u,v)$ otherwise, where γ is a parameter, and $sim(u,v)$ can be any semantic term similarity such as the value given by the WordNet:Similarity tool[3] [72], which we also use in our experiments. Clearly, if $\gamma = 0$, ω_{SEM} degenerates to ω_{WO}.

The contrastive similarity function ψ is meant to measure how well two sentences with opposite opinions match up with each other. Intuitively, we would like the two sentences to overlap on all words except for those sentiment-related words, where they are expected to differ. Thus, we define ψ in the same way as we define ϕ except that we now calculate the similarity *after* removing negation words and adjectives from both sentences. The rationale is that opinions are mainly expressed by adjectives and negation words. We also have two variations for ψ: WO and SEM.

Note that although in this work we only experiment with these simple similarity measures, our optimization framework would allow us to potentially use more sophisticated measures (e.g., [15, 83, 6, 60, 92]).

4.6 Optimization Algorithms

A brute-force solution to the optimization problem defined in Section 4.4 would be to enumerate all the possible candidate summaries (i.e., k sentence pairs) and compute the objective function for each candidate summary. Since $|X| = n$ and $|Y| = m$, there are altogether $\binom{n}{k}\binom{m}{k}$ possible candidate summaries. Thus enumerating all of them is generally not feasible especially because computation of the value of the objective function for a candidate summary also involves additional iterations.

We now propose two heuristic strategies to find an approximate solution to the optimization problem. Our objective function contains two parts, corresponding to the representativeness ($r(S)$) and contrastiveness ($c(S)$) of a summary, respectively. Thus a greedy way to optimize the objective function can be to first find a subset of summaries that can score well with one of them, and then try to further select a good summary that can also score well for the other component. Depending on whether we would first optimize $r(S)$ or $c(S)$, we naturally have two heuristic strategies to generate a contrastive opinion summary, called Representativeness-First and contrastiveness-first, respectively.

[3]http://www.d.umn.edu/ tpederse/similarity.html

4.6.1 Representativeness-First Approximation

To optimize $r(S)$ means to select k sentences from each of X and Y that can best represent all the sentences in X and Y. Intuitively, we may achieve this goal by clustering the sentences in X and Y independently to generate k clusters for each, and then take the most representative sentence from each cluster. Specifically, let $\{U_1, ..., U_k\}$ be k clusters of sentences in X, and $\{V_1, ..., V_k\}$ be k clusters of sentences in Y. We may assume that $S = \{(u_i, v_i)\}$ where $u_i \in U_i$ and $v_i \in V_i$, for $i = 1, ..., k$. In general, given an implementation of the content similarity function ϕ, any clustering algorithm can be used. In our experiments, we used the hierarchical agglomerative clustering algorithm and stopped it when we have obtained precisely k clusters.

Now, we may reasonably assume that the similarity of a sentence to another sentence in a different cluster is always lower than its similarity to a sentence in its own cluster. It is not hard to prove that the summary S to maximize $r(S)$ is given by the centroid sentences in each cluster. That is, $S = \{\bar{u}_i, \bar{v}_i\}$, and

$$\begin{aligned}\bar{u}_i &= \arg\max_u \frac{1}{|X|}\sum_{x\in X}\phi(u,x)\\ \bar{v}_i &= \arg\max_v \frac{1}{|Y|}\sum_{y\in Y}\phi(v,y)\end{aligned}$$

Next we would like to keep $r(S)$ constant and optimize $c(S)$. Clearly $c(S)$ depends on how we index the clusters of X and Y, that is, how we order $\{U_1, ..., U_k\}$ and $\{V_1, ..., V_k\}$. In other words, it depends on how we align a cluster U_i with a cluster V_i. Intuitively we would like to align them so that the corresponding u_i and v_i would have the highest contrastiveness similarity, i.e., to maximize $\psi(u_i, v_i)$. Since k is generally small, we can find the exact optimal alignment without approximation.

One may notice that the weighting parameter λ did not matter in this strategy. Indeed, we have implicitly set $\lambda = \infty$ by first attempting to optimize $r(S)$ and then fix it to further optimize $c(S)$. To further improve over this, we may search in each aligned cluster pair to find a potentially better pair of sentences that can lead to a higher objective function value than the centroid pair defined above. If we do this, we will see that λ would affect our solution.

Specifically, let $\{U_1, ..., U_k\}$ and $\{V_1, ..., V_k\}$ be our optimal alignment of clusters. We may rewrite our objective function as $g(S) = \sum_{i=1}^{k} g_i(u_i, v_i)$

where $g_i(u_i, v_i)$ is given by

$$\lambda[\frac{1}{|X|}\sum_{x\in U_i}\phi(x,u_i) + \frac{1}{|Y|}\sum_{y\in V_i}\phi(y,v_i)] + \frac{1-\lambda}{k}\,\psi(u_i,v_i).$$

This objective function is now a sum over all the aligned cluster pairs, and we can now find the solution by optimizing each (u_i, v_i) pair independently. Formally,

$$(u_i^*, v_i^*) = \arg \max_{u_i \in U_i, v_i \in V_i} g_i(u_i, v_i).$$

Clearly $g_i(\bar{u}_i, \bar{v}_i)$ is not necessarily optimal, so we would like to search in U_i and V_i for a truly optimal (u_i^*, v_i^*). The brute force search has a complexity of $O(|U_i||V_i|)$, but we do not have to try every pair of sentences. Instead, we only need to try those pairs with a higher contrastiveness score than our centroid pair $(\bar{u}_i, \bar{v}_i)$ because if a pair has a lower contrastiveness score than the centroid pair, it would be impossible for it to have a higher g_i value. Thus computationally, we can simply sort all the pairs in each pair of clusters in the descending order of contrastiveness scores and then go down the list to compute its g_i value, until we hit the centroid pair. The pair that gives the highest g_i would be taken as (u_i^*, v_i^*). We do this for each cluster pair to obtain the optimal summary $S^* = \{(u_i^*, v_i^*)\}$.

4.6.2 Contrastiveness-First Approximation

In this strategy, we first compute $\psi(u, v)$ for all $u \in X$ and $v \in Y$. We then sort these pairs and gradually add a sentence pair to our summary starting with the pair with the highest contrastive similarity score. If we just take the top k pairs, we would find a solution corresponding to setting $\lambda = 0$, i.e., solely based on contrastiveness and ignoring representativeness completely. Thus to improve our approximation, we would like to sacrifice some amount of contrastiveness score and gain more on the representativeness score.

A greedy algorithm to achieve this is as follows. First, we would take the pair with the highest value of ψ as a pair in our summary. Given that we have already chosen $i-1$ pairs $S_{i-1} = \{(u_j, v_j)\}_{j=1}^{i-1}$, we would like to choose the next pair (u_i, v_i) to add most to our objective function, which further means to maximize the increase of both the contrastiveness and the representativeness. Given a candidate pair (u_i, v_i), and let S_i be the augmented summary of S_{i-1} by adding this new pair. We want to choose (u_i, v_i) to maximize the following objective function :

$$\begin{aligned}
(u_i^*, v_i^*) &= \arg\max_{u_i, v_i} \lambda r(S_i) + (1-\lambda) c(S_i) \\
&= \arg\max_{u_i, v_i} \lambda \Big(\frac{1}{|X|} \sum_{x \in X_{u_i}} \phi(x, u_i) + \frac{1}{|Y|} \sum_{y \in Y_{v_i}} \phi(y, v_i) \Big) \\
&\quad + \frac{1-\lambda}{k} \psi(u_i, v_i)
\end{aligned}$$

where X_{u_i} (Y_{v_i}) is the set of sentences in X (Y) that are closer to u_i (v_i) than to any previously chosen u_j (v_j), $j = 1, ..., i-1$. That is,

$$X_{u_i} = \{x \in X | \phi(x, u_i) > \phi(x, u_j) \forall j = 1, ..., i-1\}$$

$$Y_{v_i} = \{y \in Y | \phi(y, v_i) > \phi(y, v_j) \forall j = 1, ..., i-1\}.$$

Thus in our greedy algorithm, after choosing the first pair (u_1, v_1), we would iteratively choose (u_i, v_i) to maximize the "gain function", $g(u_i, v_i)$ given by

$$\lambda(\frac{1}{|X|} \sum_{x \in X_{u_i}} \phi(x, u_i) + \frac{1}{|Y|} \sum_{y \in Y_{v_i}} \phi(y, v_i)) + \frac{1-\lambda}{k} \psi(u_i, v_i).$$

The algorithm stops after having chosen k pairs.

Note that X_u and Y_v are relatively easy to compute if we remember the best ϕ values of all sentences given by the $i-1$ already chosen sentence pairs at each step because we only need to compare the remembered value with the new value of ϕ given by u_i or v_i.

In general, we would need to consider all the remaining sentence pairs in each step. However, we can further improve efficiency by only considering the sentence pairs whose contrastiveness scores are sufficiently high (e.g., above a threshold).

4.7 Experiment Design

4.7.1 Data Set

There is no existing data set for evaluating our new summarization task. We thus opt to create our own. Since a main assumption made in our problem definition is that we may separate positive and negative opinions about a topic using existing opinion summarization methods, a natural strategy to create a test set for evaluating COS would be to leverage such separated opinion data set generated by previous work. We thus obtained 14 tagged product review data sets from the previous work [28, 29][4], and have two human assessors to identify representative contrastive sentence pairs from these data. The 14 tagged review data sets contain reviews from Amazon[5]. All the sentences in these data sets have already been manually tagged with product features as well as sentiment polarities (i.e., positive or negative).

To show our methods can help users further understand opinions at a finer granularity level than the feature-level, we divided a product review into product-feature reviews based on the feature tags. Also, to make contrastive opinion summarization interesting, we chose reviews that are not extremely dominated by only positive (or negative) opinions using a threshold. Our assumption is that in those cases where reviews are dominantly of one polarity of opinions (e.g., dominantly positive), a regular summary would suffice, and we do not need to apply contrastive opinion summarization techniques. Because our data set is for evaluating the effectiveness of COS, we discarded extremely dominated sets.

[4]http://www.cs.uic.edu/ liub/FBS/sentiment-analysis.html
[5]http://www.amazon.com

Based on these criteria, we obtained 12 review sets on several products and features.

To test the generality of our methods, we also prepared another non-product-review data set. We used the Yahoo! search engine to retrieve sentences about aspartame, which is an artificial sweetener, and there are disputes about its safety. The constructed data set contains 50 positive and 50 negative matching sentences selected from the search results of the queries '`aspartame is safe`' and '`aspartame is dangerous`', respectively.

Table 4.2 shows the data set list. For each data set, ID is assigned for convenience.

ID	**Product Name**	**Feature**	**# of positive sen**	**# of negative sen**
1	Apex AD2600 Progressive-scan DVD player	player	44	56
2	MicroMP3	battery-life	9	7
3	MicroMP3	design	8	6
4	MicroMP3	headphones	7	6
5	MicroMP3	software	7	9
6	Nokia 6600	battery-life	7	8
7	Creative Labs Nomad Jukebox Zen Xtra 40GB	navigation	9	8
8	Creative Labs Nomad Jukebox Zen Xtra 40GB	software	37	41
9	Creative Labs Nomad Jukebox Zen Xtra 40GB	size	15	11
10	Creative Labs Nomad Jukebox Zen Xtra 40GB	weight	7	7
11	Creative Labs Nomad Jukebox Zen Xtra 40GB	transfer	9	7
12	Hitachi router	adjustment	7	6
13	aspartame	safety	50	50

Table 4.2: Data set.

For each test case, the two assessors were asked to cluster given sentences of each polarity of sentiment and align the contrastive clusters. Among the judgments made by human evaluators, clusters that cannot be aligned are discarded.

To assess the agreement of the two assessors, we compute their clustering agreement and pairing agreement, which are 0.76 and 0.47, respectively. The clustering agreement is the percentage of agreed decisions on putting a pair of sentences of the same polarity into the same cluster or not by the two annotators. In the original sentence sets, we can make same-polarity-sentence pairs, (u_i, u_j) where both u_i and u_j are positive (negative). For all the possible pairs, check if they are in the same cluster or not based on the two evaluators' judgments. If the judgments are same, it is an agreement. Then, clustering agreement is

$$\frac{\#\ of\ clustering\ agreement}{\#\ of\ all\ possible\ pairs\ of\ same\ polarity\ sentences}$$

The pairing agreement is computed using the Jaccard Index. First, we generate all the ideal contrastive sentence pairs from the evaluators' judgements on clustering and pairing. For example, if two clusters, $\{u_i, u_j\}$ and $\{v_l, v_m\}$, are paired, (u_i, v_l), (u_i, v_m), (u_j, v_l), and (u_j, v_m) would be generated as ideal pairs. Let A and B be the two sets of ideal pairs from two assessors, respectively. The Jaccard Index was calculated by the following formula

$$JaccardIndex(A,B) = \frac{A\ and\ B}{A\ or\ B}$$

In our experiments, we use each assessor's judgments separately for evaluation and then take the average of the two performance numbers. The constructed data sets are publically available[6].

No	Positive	Negative
1	oh ... and file transfers are fast & easy .	you need the software to actually transfer files
2	i noticed that the micro adjustment knob and collet are well made and work well too.	the adjustment knob seemed ok, but when lowering the router, i have to practically pull it down while turning the knob.
3	the navigation is nice enough , but scrolling and searching through thousands of tracks , hundreds of albums or artists , or even dozens of genres is not conducive to save driving .	difficult navigation - i wo n't necessarily say " difficult ," but i do n't enjoy the scrollwheel to navigate .
4	i imagine if i left my player untouched (no backlight) it could play for considerably more than 12 hours at a low volume level.	there are 2 things that need fixing first is the battery life. it will run for 6 hrs without problems with medium usage of the buttons.

Table 4.3: Sample contrastive sentence pairs.

4.7.2 Measures

We evaluate our contrastive opinion summary with the following two measures based on the aligned cluster data set:

Precision: The precision of a summary with k contrastive sentence pairs is the percentage of the k pairs that are agreed by a human annotator. If a retrieved pair exists in an evaluator's paired-cluster set, we assume that the pair is agreed by the annotator (i.e., "relevant"). Thus precision is basically the number of such agreed pairs divided by k. Intuitively, precision tells us how contrastive the sentence pairs of our summary are.

Aspect coverage: The aspect coverage of a summary is the percentage of human-aligned clusters covered in the summary. If a pair of sentences appears in a human-aligned pair of clusters, we would assume that the aligned cluster is covered. Intuitively, aspect coverage measures the representativeness of a summary.

The number k of a target summary was set heuristically by the following formula; $k = 1 + \log_2(|X| + |Y|)$. The intuition is that we will have a larger k if we have more sentences to summarize, but the growth will "saturate" as the number of sentences becomes very large.

[6]`http://timan.cs.uiuc.edu/data/cos/`

4.7.3 Questions to Answer

We design our experiments to answer the following questions: First, between representative-first approximation(RF) and contrastive-first approximation(CF), which optimization algorithm performs better? We can answer this question by comparing the performance of these two different approximations for various values of λ. Second, both ϕ and ψ can use semantic matching of words. Can semantic matching of words help improve performance on top of simple exact matching of words? We can answer this question by comparing the performance of methods using different semantic coefficient γ. Third, we have hypothesized that it would be beneficial to exclude sentimental words when computing the contrastive similarity. Is this heuristic effective? We can answer this question by comparing performance of excluding such words with that of not excluding them (i.e., using all the words).

4.8 Experiment Results

4.8.1 Sample Results

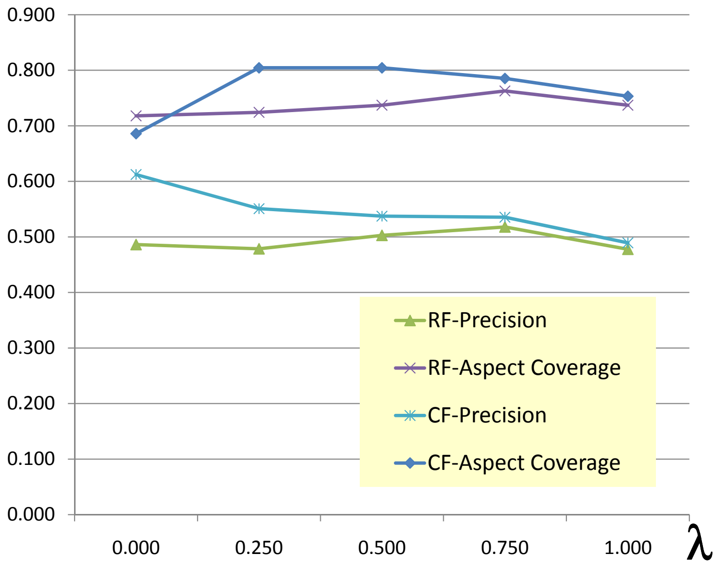


Figure 4.1: Comparison of RF and CF.

We first show some sample contrastive sentence pairs generated in our experiments in Table 4.3. Intuitively, these pairs are quite informative and can clearly help a user better understand the mixed opinions in different aspects. The first and second pairs show that different polarities of opinions are made from different perspectives. For example, from the first pair, a user would know that file transfer is fast, but you'll need transfer software. Similarly, the second pair shows that adjustment knob generally works well, but it is inconvenient when lowering the router. In the third pair, although the two sentences are classified as positive and negative, respectively, the difference is rather small, indicating that there is probably not that much disagreement here. In the fourth pair, we can learn even more details about the product. This example shows the battery life can vary depending on usage patterns even with specific number of hours it can last; users can decide whether to buy this product based on their own usage style.

4.8.2 Representativeness-First vs. Contrastiveness-First

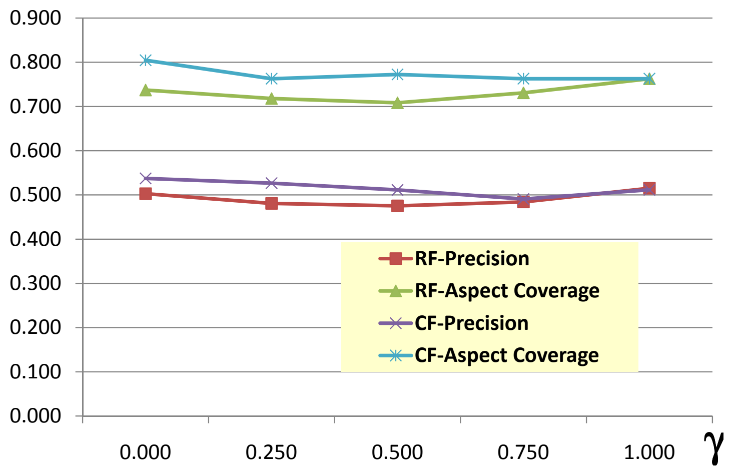
Next, we use the basic similarity measure (ω_{WO}) to compare the two approximation strategies, i.e., Representativeness-First (RF) and Contrastiveness-First (CF) in Figure 4.1. For both methods, aspect coverage is higher than precision, indicating that it is easier to achieve representativeness than contrastiveness. In general, we see that CF outperforms RF for almost all values of λ, indicating that it is more important to optimize contrastiveness first to ensure that we obtain the best contrastive alignments of sentences. We also see that the performance is sensitive to the setting of λ, the relative emphasis on the representativeness and contrastiveness of the summary. In order to examine other variables in our methods, in the following experiments, we set λ to a reasonable value of 0.5, which intuitively means that we put equal weights on the two criteria.

4.8.3 Semantic Term Matching

We now look into the effectiveness of semantic term matching. Since ω_{WO} is a special case of ω_{SEM} when $\gamma = 0$, we can see whether semantic matching helps by varying the value of γ. We show the results of using semantic term matching for content similarity and contrastive similarity respectively with two separate plots in Figure 4.2. In general, semantic term matching does not seem to help. Indeed, as we increase the value of γ, the performance tends to drop. We also see that the content similarity function is more sensitive to semantic matching than the contrastive similarity function. This may be because in the latter case, sentimental words are removed, so the overall influence of semantic matching would be reduced.

4.8.4 Contrastive Similarity Heuristic

We hypothesized that by removing sentimental words in computing the contrastive similarity function we can improve matching accuracy. So finally, in order to test this hypothesis, we compare the results of using this heuristic with those of not using it (i.e., computing the contrastive similarity in the same way as computing the content similarity) in Table 4.4. We see that if we keep all these sentimental words, the performance is consistently worse, indicating that the heuristic of removing sentimental words is effective.



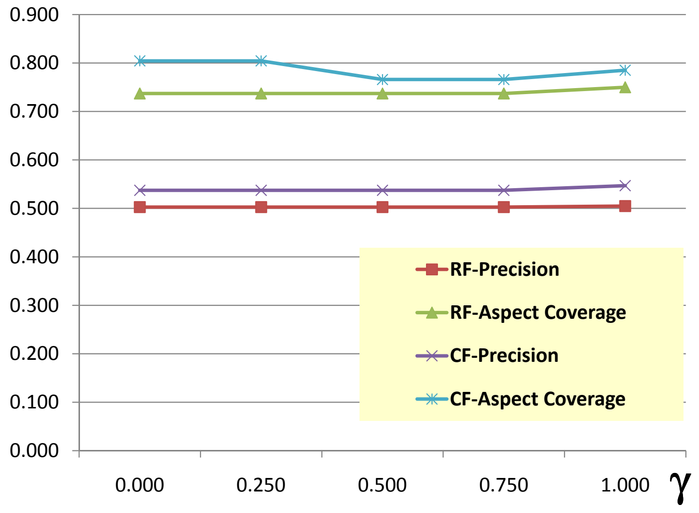


Figure 4.2: Effectiveness of semantic term matching for content similarity (top) and contrastive similarity (bottom).

4.9 Conclusion

In this chapter, we proposed a novel summarization problem, namely contrastive opinion summarization. It aims to summarize contradictory or mixed opinions about a topic and generate a list of contrastive pairs of sentences with different sentiment polarities to help users to digest contradictory opinions. We formally framed the problem as an optimization problem and proposed two approximation methods to solve the optimization problem. We also explored different similarity measures in our optimization framework. We leverage existing summarization resources to create a gold standard data set for evaluating the proposed new summarization task.

Experiment results using this data set show that the proposed methods are effective for generating contrastive opinion summaries. In particular, Contrastiveness-First approximation works better than Representativeness-First, and

	Precision		Aspect Coverage	
Opt. Method	RF	CF	RF	CF
WO	0.503	0.537	0.737	0.804
WO + all words	0.484	0.531	0.737	0.798
SEM	0.500	0.540	0.763	0.763
SEM + all words	0.470	0.507	0.718	0.686

Table 4.4: Effectiveness of removing sentimental words in computing contrastive similarity.

the heuristic of removing sentimental words in computing contrastive similarity is effective. However, semantic term matching based on WordNet is found to be not helpful. Sample summary results show that the generated summaries are informative and can help users digest contradictory opinions more effectively.

Our work can be extended in several directions. First, more experiments on additional larger data sets would be desirable. Second, we have only explored some basic semantic term matching method; it would be interesting to further explore more advanced similarity functions such as those based on sentence alignment or more in-depth semantic analysis. Finally, it should also be very interesting to further study how to develop algorithms to achieve better approximate solutions to the optimization problem using our framework.

Chapter 5

Causal Topic Mining [1]

5.1 Information Retrieval with Time Series Query

5.1.1 Introduction

Information retrieval technology helps people easily find relevant documents in any text collection and has found widespread applications, notably in Web search.

In most of the existing work on information retrieval, the problem involves a keyword text query entered by a user, and the goal is to retrieve documents from a collection of text documents, which are relevant to the query. In general, entering a keyword query is a natural way for a user to express an information need, and search engines supporting such queries have been very useful in finding relevant documents to satisfy any ad hoc information needs.

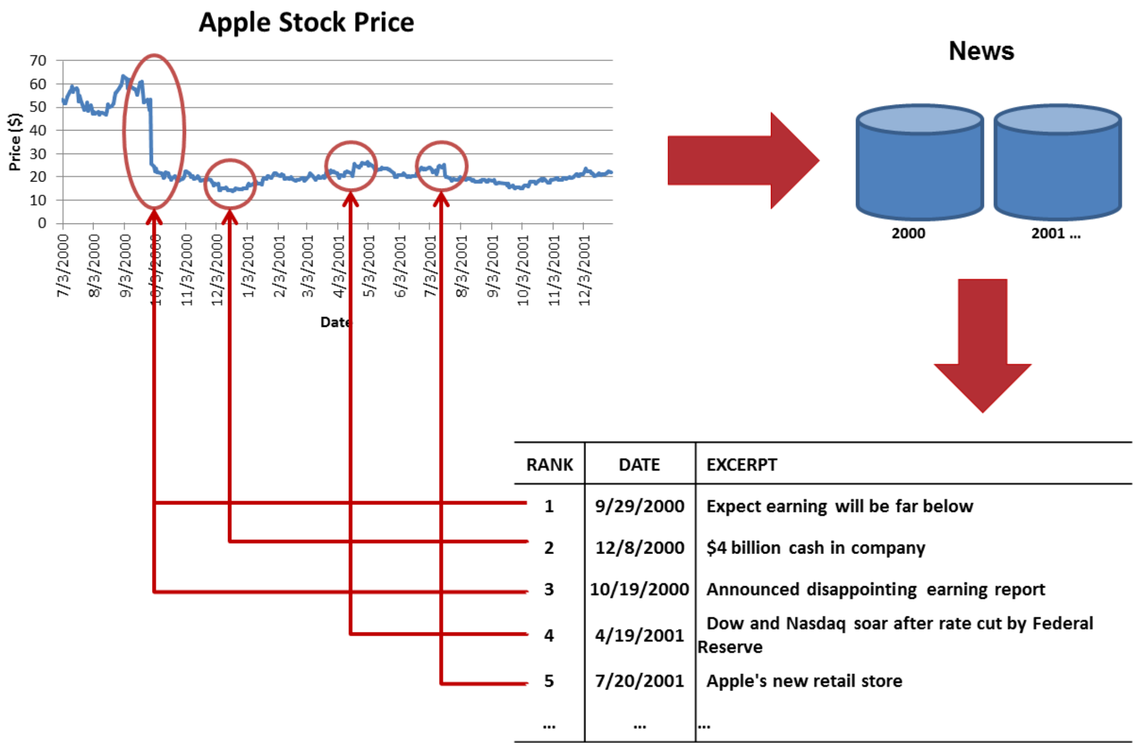
While satisfying a user's information needs via keyword queries has so far been the primary application of information retrieval, in this work we show that there is also another type of application, where the goal is to find the documents in a text collection that can explain the changes of an external time series variables. Indeed, topics discussed in the text documents often have relations with other non-textual variables. For example, the reporting of a negative event about a company in the news might affect its stock prices. Similarly, the dramatic changes of the stock price of a company might also trigger discussions about it in the news. Another example is a new election pledge in a campaign that may affect support rate to the candidates in a presidential election.

In such cases, there is often a need to understand which topics in the text data may be correlated with the time series data. For example, investors are probably interested in what leads to the increase or decrease of the stock prices and use such relationships to forecast future price changes. Companies may want to understand how product sales rise and fall in response to such text data as advertisements or product reviews. In election campaigns, analysis of news or social media may explain why a candidate's support has risen or dropped in the polls. Such understanding of the situation can help to make better future strategies for sales or polls.

One way to help users find such relevant topics is to use time series as a query. Ideally, the documents retrieved from a time-stamped text collection will include those topics that are correlated with the given time series. For

[1]Part of this chapter has been published in [44, 47].

example, using Apple stock price time series query, we may be able to retrieve relevant documents from general news stream (Figure 5.1). The highly ranked documents are not only articles ‘about’ Apple company, but also potentially explain major changes of the input time series.



RANK	DATE	EXCERPT
1	9/29/2000	Expect earning will be far below
2	12/8/2000	$4 billion cash in company
3	10/19/2000	Announced disappointing earning report
4	4/19/2001	Dow and Nasdaq soar after rate cut by Federal Reserve
5	7/20/2001	Apple's new retail store
...	...	...

Figure 5.1: Example results. Apple stock price and retrieved documents from news collection.

This is an interesting and novel retrieval problem, which has not been studied before. Although there were some previous works about text and time-series retrieval as we explain more in Sec 5.1.2, the main difference between this problem and other existing formulations is that instead of using text for the query we are using time series.

In this section, we study this novel retrieval problem. In this problem, our goal is to find relevant documents in a text collection of the same time period that contain topics correlated with the query time series. Such a retrieval task is especially useful for analyzing text data to discover potentially causal topics that could explain the changes of external time series variables. Information retrieval directly from time series query has several possible advantages over traditional keyword-based retrieval: an ability to find correlated documents over simple relevancy, no presence of user bias from keyword selection, and applicability to unknown or artificial time series input. We will discuss these examples in more detail in Section 5.1.3.

To solve this new retrieval problem, the main challenge is how to cross the barrier between a time series query and text documents. Our key idea for solving this challenge is to first find terms whose frequency distribution over the time period of a collection is correlated with the query time series, and then use these terms to further find possibly relevant documents. For our solution, we propose and study multiple retrieval algorithms that use the general idea of ranking text documents based on how well their terms are correlated with the query time series. We first compute the word frequency curves from the text stream and calculate their correlations with the input time series query. Then, the document ranking score is calculated by the weighted average of the words in the document, where the weight is a function of the value of the correlation between the word and the input query.

To validate our method, we test proposed algorithms on the news data with stock prices. In addition to quantitative evaluation with standard information retrieval measures such as mean average precision (MAP) and normalized discounted cumulative gain (NDCG), we also qualitatively evaluate top ranked documents to see if they are useful in the understanding of the changes in the input time series. Experimental results show that the proposed retrieval algorithms can effectively help users find documents that are relevant to the time series queries.

The main contributions of this work include:

1. We introduce a novel information retrieval problem setup using time series as a query, which has many applications in text data mining.
2. We propose multiple algorithms for the new information retrieval problem.
3. We evaluate the proposed algorithms on real application data sets and make findings to help understand which algorithms effectively retrieve potentially useful documents for understanding changes of the input time series.

The rest of the sections are organized as follows. In Section 5.1.2, we discuss previous related works. In Section 5.1.3, we formally describe the new proposed problem: information retrieval with time series query In Section 5.1.4, we describe the proposed methods for document retrieval with time series query. We show our experiment design in Section 5.1.5 and show quantitative and qualitative experimental results in Section 5.1.6. We discuss about related issues in Section 5.1.7 and conclude in Section 5.1.8 with future works.

5.1.2 Related Work

Information retrieval is a widely studied discipline [61, 90]. With evolution of the Web, it has received a lot of attention from the industry, and many of its techniques have been commercialized. The main goal of traditional information retrieval is finding relevant documents to the textual keyword query. However, so far there have been no attempts to retrieve relevant documents directly from time series queries.

While there were studies in the time series matching and retrieval, they mainly focused on the retrieval from a collection of time series, and not from text data [3, 4, 5, 14, 21, 24, 38, 39, 40, 67, 76, 77, 89]. The main technical challenges for this problem are about the accurate measurement of similarity between two time series, and their efficient search. There are various methods for measuring similarity between time series, from basic measures such as Euclidean distance and correlation to more advanced techniques such as shape definition language [5], pattern matching and analysis [4, 21], longest common sequence matching [38], and dynamic time warping [89]. While these techniques are solely focused on numerical time series data without the consideration of related text data, they are useful tools for an approach to measuring similarity between the time series of word frequency and the input time series.

For more efficient and fast retrieval, various indexing and dimension reduction techniques can also be used, such as discrete Fourier transformation [3, 76, 77], wavelet transformation [14], and a probabilistic approach [39]. Like the general information retrieval problems, feedback on time series is also studied[40]. In this work, we focus more on how accurately we can retrieve relevant and causal documents. These techniques can be possibly applied to speed up our method.

In addition to the previous methods, there were attempts to co-analyze text data with time series which showed interesting results. NTCIR proposed GeoTime [2] that is a geographical and time-dependent event retrieval task by question queries including location and time constraints. People have tried to summarize time series by extracting textual expressions [36, 86] and use their linguistic features to retrieve similar time series [82]. Temporal similarity analysis of word dynamics has helped to find semantically related terms [75]. Recent works showed that using different term weighting based on word dynamics over time, can help to improve document retrieval [19, 35, 49, 51]. Our retrieval problem shares a similar concept to these works in that we give different weights to terms, based on the time series analysis. However, we relate word dynamics to the input external time series that has not been considered in the previous works. Moreover, in problem setup, we use time series itself as a query, while previous works are still based on the keyword query retrieval task.

Research on stock prediction using financial news content is also relevant to our work [37, 65, 79]. One of its goals is to find the most predictive words and label the news according to their effect on the stock prices for that specific day. Most of these works are based on a regression or a classification problem setup, unlike our problem.

Causal topic mining [47] is a yet another closely related work. It automatically models topics that have causal relationships to the external time series input. However, in this work, instead of topic retrieval based on probabilistic topic modeling, we try to retrieve relevant documents with word-level-based analysis. This approach is more simple and efficient, because we do not need complex parameter estimation steps of a probabilistic topic model.

[2]http://metadata.berkeley.edu/NTCIR-GeoTime

5.1.3 Information Retrieval with Time Series Query

Motivation

Information retrieval with a time series query is a practical problem that arises in many text mining applications where there is a need to analyze text data and discover potentially causal topics in order to explain the changes of external time series variables.

Given an input time series and the knowledge about its topic, it would be natural to think that we could use the topic keywords with traditional IR techniques to obtain relevant documents that could explain the evolution of the external time series. However, there are three main advantages of our time series-based information retrieval approach over the traditional keyword-based information retrieval.

First, beyond simple relevancy, we want to find articles containing topics that are highly correlated to the changes in the input time series. Traditional information retrieval does not consider *dynamics of terms* and how they are related to the external time series. Instead, it focuses just on the occurrences of the terms. In contrast, our technique measures the correlation between the time series of each term and the input time series query, and uses correlation to weigh the terms. In this way, we can rank documents based on how correlated their content is.

Second, the usage of time series as a query has an advantage of avoiding user bias in keyword selection that may give us misleading results, and irrelevant documents. For example, let us assume that Apple's iPhone sales revenue is our time series query. An analyst may use the company and product name as a query to find relevant documents that explain the behavior of the revenue. However, a competitor's product, such as Samsung's Android phone, may also be a good keyword if it had a significant effect on the sales of iPhone. We could add "Samsung" and "Android" as additional keywords, but still there could be even more related terms, corresponding to related entities, facts, and aspects that could have serious impact on the sales. For example, there could have been a new trend in the smartphone market or even a global economic trend, which the analyst would likely have missed. Therefore, rushing to specify the subtopic and using its keywords can lead to the loss of important hidden signals, related to the time series. By using the time series query itself, we let the algorithm find the most related documents corresponding to different signals that affect the behavior of the external time series.

Third, in addition to the previous examples of understanding stock prices, sales revenues, and election polls, we can even apply the technique to an unknown or an artificial time series curve. Despite the bias, if we have clear information of what the input time series is about, we may still be able to get limited help from the traditional information retrieval techniques. However, when it is hard to specify related keywords, using time series directly as a query would be much more helpful. For example, we may have unknown traffic signals from the web and may be able to find the reason for the traffic by searching on social media sites such as Facebook and Twitter. For another example, we could analyze signals obtained from physical sensors. In that case, we might be able to find unknown reasons of the signal changes or glitches from the log data or other kinds of possibly related text articles (e.g. local news, and web articles of the locations where sensors are installed). Furthermore, in addition to the given time series as an input, users may even be able to draw an artificial time series curve based on their needs and retrieve correlated documents.

Problem Definition

The inputs of our retrieval problem are a text stream and a time series query. The output is a ranked list of input text documents. Formally, given a numeric time series query with time stamps $TS = \{(x_1, t_1), (x_2, t_2), ..., (x_n, t_n)\}$, and a text stream which is a collection of documents with time stamps within the same time period, D = $\{(d_1$,$t_{d_1})$, ..., $(d_m$, $t_{d_m})\}$, we have to compute a ranked list of documents, D' = $\{(d'_1$,$t_{d'_1})$, ..., $(d'_m$, $t_{d'_m})\}$. These are the documents that explain the behavior of the time series. To the best of our knowledge, no previous work has solved or described this novel problem.

The main challenge in solving our problem is how to cross the barrier between the query and the document. In regular text retrieval, both the query and the document are text objects, making it easy to match them, but in our case, the query is a non-textual time series. In the next section, we describe our methods to solve this challenge.

5.1.4 Method

General Method

The goal of our retrieval problem is to find documents having contents that could possibly help to explain the changes in the input time series. In order to cross the barrier between a non-textual query and a text document, our key idea is to first obtain words whose frequency distributions over time are correlated to the query time series, and then use such correlated words as a weighted text query to retrieve documents that match such a query well. This idea naturally leads to a general three-step retrieval procedure as shown in Figure 5.2. In the first step, we pre-process the text stream and obtain a frequency distribution over time for each term. Such a frequency distribution gives us a time series characteristics when the term is mentioned frequently, and when infrequently, over time. In the second step, we can then match such a term time series with the given query time series to compute a correlation or association score. If whenever the value of a query time series variable is high, a word would occur frequently, we would have a high correlation for such a word. For example, if mentioning of a particular political issue tends to be associated with the increase of votes for a presidential candidate in a national poll, words about such a political issue can be expected to have a relatively high correlation compared with other words. Finally, we would use words with relatively high correlations to compute the score of a document by essentially treating such words as forming a weighted text query.

Clearly, the two main technical challenges in such an approach are: 1) how to measure the correlation between a term and the input time series, and 2) how to aggregate the signals from each term for document ranking. In general, there are many different ways of solving each of these two challenges within the proposed general retrieval framework. In this work, as a first step in studying this new problem, we focus on studying a few natural ways to solve these challenges, leaving the exploration of more sophisticated methods as future work. Specifically, we will experiment with two methods for computing correlation of two time series variables (i.e., Pearson correlation and Dynamic Time Warping), and two different strategies for computing document scores based on term correlations (i.e., feeding an existing retrieval function with an artificial weighted text query constructed based on correlated terms, and direct aggregation of term correlations). We now describe all these methods in detail.

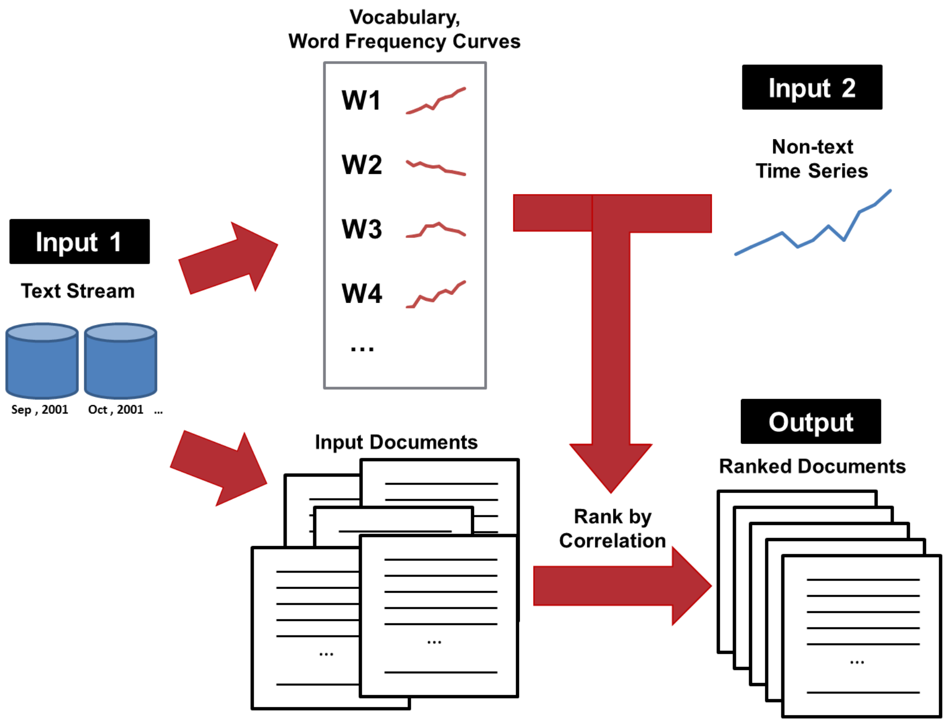


Figure 5.2: Overview of information retrieval with time series query.

Correlation Function

To compute the correlation of a term with a query time series, we first generate the term frequency curve for each term w in the collection, WF_w, over the given time period of the collection, t_1, ..., t_n.

$$WF_w = (wf_{w,t_1}, wf_{w,t_2}, ..., wf_{w,t_n})$$

$$wf_{w,t_i} = c(w, D_{t_i})$$

where wf_{w,t_i} is the frequency of the word w at the time t_i, and $c(w, D_{t_i})$ is the count of w over all documents d having t_i time tag in collection D.

We can now use any similarity metric for the time series to measure the correlation between a term and the query time series. Depending on the specific applications, we may also be interested in shifting the time points when computing correlations. For example, aligning term frequencies in earlier time points (e.g. 1 day before) to a stock time series can potentially discover terms that might "predict" stock changes, while aligning them from a later time period (e.g. a few days later) than the stock prices might reveal words that discuss the changes of stock that have already happened. In this work, we do not systematically explore all these options, but instead simply align the time points exactly in order to focus on understanding the relative effectiveness of different methods, though one of the two methods to be presented below can capture the time shift to some extent.

Pearson Correlation

Pearson correlation (**Pearson**) is the most representative metric for measuring how much two time series are related. Particularly, Pearson correlation checks if two time series move in the same direction (an increase or decrease), which fits our purpose very well.

For two time series X=(x_1, x_2, ..., x_N) and Y=(y_1, y_2, ..., y_M), correlation coefficient can be defined as following [3].

$$\rho_{X,Y} = corr(X,Y) = \frac{cov(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}$$

where μ_X and μ_Y are average of X and Y, σ_X and σ_Y are standard deviation of X and Y, E is expected value (average) operator, cov is covariance operator, and corr is correlation.

Dynamic Time Warping

Although Pearson correlation gives a good idea of how much the two time series are correlated, it has a limitation in capturing the similarity when one of the series is stretched or shifted. It is often the case that two time series have overall similar shapes, but are not exactly lined up on the timeline. In reality, when one factor may affect another, there can be a delay in the impact, or this impact may last longer even once the causal factor has disappeared. Therefore, it is possible for the two correlated time series to not be in sync.

To overcome this limitation, dynamic time warping (**DTW**) [41, 63] has been proposed. Dynamic time warping is a dynamic programming algorithm that aligns time series with a flexible timeline mapping. That is, depending on the shape of the series, one time period (e.g. a day) can be mapped to several time periods of the same kind (e.g. days) of

[3] `http://en.wikipedia.org/wiki/Correlation_and_dependence`

the other series dynamically. By following the mapping path from the beginning to the end of the time series, DTW finds the best alignment path with the minimal distance between the time series (e.g. Euclidean distance).

Algorithm starts by building the distance matrix having distances for all pair between X and Y. Each component of cost matrix C is $c_{i,j} = \|x_i - y_j\|$. Then, using the cost matrix, optimal alignment path, p^*, with minimal cost is searched.

$$\begin{aligned} DTW(X,Y) &= c_{p^*}(X,Y) \\ &= min\{c_p(X,Y), p \in P^{N*M}\} \\ &= D(N,M) \end{aligned}$$

where D is accumulated cost matrix. That is, D(i,j) is the minimal cost to align x_1, ..., x_i to y_1, .., y_j. D can be computed with the following rule with dynamic programming.

1. First row: $D(1,j) = \sum_{k=1}^{j} c(x_1, y_k)$, $j \in [1, M]$.
2. First column: $D(i,1) = \sum_{k=1}^{i} c(x_k, y_1)$, $i \in [1, N]$.
3. Other elements: $D(i,j) = \min\{D(i-1, j-1)$, $D(i-1,j)$, $D(i, j-1) + c(x_i, y_j)\}$, $i \in [1, N], j \in [1, M]$.

As we have mentioned in Section 5.1.2, dynamic time warping has also been used in previous works on time series retrieval [89] and word similarity by time dynamics [75].

Aggregation function

Once we obtain the word correlation scores, our next task is to score documents based on their coverage of highly correlated words. We propose two strategies for doing this. The first one is to formulate a weighted text query using the highly correlated words, and adapt an existing retrieval function to score and rank documents. The second one is to directly compute an aggregated correlation score for a document based on the correlation scores of the words matched in the document. Compared with the first strategy, the second strategy has the advantage of generating a more meaningful score (i.e., correlation score) for our task, but the first one may have the advantage of leveraging existing retrieval heuristics. We now present these two strategies in detail.

Weighted TF-IDF (BM25)

The first strategy is to leverage an existing retrieval model. In our original problem formulation, the query is non-textual, thus we cannot apply a regular retrieval model. However, after we compute word correlations, we can generate a weighted text query based on the highly correlated words. This would allow us to leverage an existing retrieval model for scoring documents. In our experiments, we will use BM25 [34], which is one of the most effective basic retrieval functions. Specifically, we can slightly modify BM25 formula to use a correlation coefficient for each term as a weight. With sorted terms w'_i by correlation coefficient, we can define "top-K term BM25" as

$$score_{TopK-BM25}(d) = \frac{\sum_{i=1}^{min(l,K)} |cor_{TS}(w'_i)| BM25(w'_i, d)}{\sum_{i=1}^{min(l,K)} |cor_{TS}(w'_i)|}$$

$$BM25(w,d) = IDF(w) \frac{f(w,d)(k_1+1)}{f(w,d) + k_1(1 - b + b\frac{|d|}{avgdl})}$$

$$IDF(w) = log \frac{N - n(w) + 0.5}{n(w) + 0.5}$$

where $f(w,d)$ is term frequency of w in d, $|d|$ is document length, $avgdl$ is average document length, N is the total number of documents in the collection, and $n(w)$ is the number of documents having w in the collection. In our experiment, k_1 and b are set as 1.5 and 0.75 respectively, which are recommended values in literature [80].

Average Correlation

One potential disadvantage of the Weighted TF-IDF approach is that the scores are not so meaningful. We thus propose a second strategy where we aggregate the correlation values of words matched in a document and obtain an aggregated correlation value, which can be interpreted as the correlation of the topic covered in the document with the query time series.

Average over all terms: A natural way to aggregate word correlations at the document level is to compute the average correlation of all the terms in the document (**AC**). Formally, each document consists of words that form a subset of the entire vocabulary of the input data set, $d = w_1, w_2, .., w_l$. The ranking score of document d can then be defined as

$$score_{AC}(d) = \frac{1}{|d|} \sum_{i=1}^{l} |cor_{TS}(w_i)|$$

where $|d|$ is the length of document and $cor_{TS}(w_i)$ is the correlation between input time series and the word frequency curve of w_i. Note that this aggregation function is independent of the correlation function for terms, thus we can apply it on top of any correlation function. We use the absolute value of the correlation as the weight because we want to focus on the strength of the correlation, regardless its sign. Moreover, we do not want to have two highly correlated terms with different direction that negate each other when being averaged. However, we could also consider the direction of the time series to obtain more detailed scores that can tell which documents had a positive impact or a negative impact on the query time series.

Since stopwords do not provide any meaningful signal about the time series changes, we can preprocess the documents to remove them.

Average over top-k terms: Obtaining a high score for the average correlation requires for all the words in the document to be highly correlated. However, in reality, even if every single word is not highly correlated, there can still exist some important information that could explain the input time series. Besides, a relevant document containing correlated topics may cover non-correlated topics. Based on this intuition, we propose another aggregation method that uses only the top-K highest correlated terms to compute average correlation (**TopK-AC**). That is, documents will compete with each other using their best K terms. Ranking score of the document d can be defined as follows.

For each document, the terms will be sorted in descending order of the absolute values of their correlations. Then, the top K correlations will be used for computing each document's ranking score. The ranking score of the document d can be defined as

$$score_{TopK-AC(d)} = \frac{1}{|K|} \sum_{i=1}^{min(l,K)} |cor_{TS}(w'_i)|$$

where w'_i are the sorted terms.

Unlike the average correlation, top K average correlation will penalize documents that are shorter than K words, because their score will still be divided by K. This is reasonable, since such short documents are less likely to contain enough information.

Average over top-k unique terms: When we observe only the top K correlated words, the list of their scores may be dominated by multiple occurrences of the same term. In large documents, multiple occurrences are more likely to happen. This may lead to documents that are more diverse to be ranked lower than the less diverse documents where a few highly correlated words are repeated a large number of times. To avoid this, we propose a variation of the top K average correlation where the list of top K correlated words contains only the unique terms (**TopK-AC-Uniq**).

To achieve this, input documents are reprocessed to have each word only once, $d'' = w''_1, w''_2, .., w''_m$. Then, the previous strategy of top K average correlation can be applied.

$$score_{TopK-AC-Uniq}(d) = \frac{1}{|K|} \sum_{i=1}^{min(m,K)} |cor_{TS}(w''_i)|$$

5.1.5 Experiment Design

Data Set

Considering the availability of many stock price time series data and many news articles of the same time period, we decided to use stock price time series data as queries and the companion news stream as document collection. Specifically, for the text stream, we used The New York Times annotated corpus [4] covering all the news published

[4]`http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2008T19`

for the period between July 2000 and December 2001, which includes the total of 144261 articles. For the input time series queries, we have used stock prices of multiple major companies. Daily stock price information was obtained from Yahoo! Finance [5].

Creating relevance judgments turned out to be a challenging task. To find if one document has contents associated with changes in time series, annotators would have to have profound knowledge for both time series and text data. We propose to solve this problem by assuming that a document mentioning the name of a company is relevant to the stock time series of the corresponding company. That is, given a query representing the stock prices of company entity A, we assume that any documents mentioning A are relevant. This gives us an objective way of evaluating our algorithms quantitatively. However, while it is reasonable to assume that if a document mentions entity A, it can be regarded as relevant, not all relevant documents necessarily mention entity A. We thus also examine the top-ranked results and evaluate them qualitatively.

To ensure each query has a reasonably large number of relevant documents in the collection, we selected companies that had more than 50 relevant articles in our document collection and whose daily values of stocks were available on Yahoo! Finance. We ended up obtaining 24 such companies, giving us 24 time series queries with relevance judgments.

Measures

To measure the performance of our algorithms, we used standard information retrieval measures, mean average precision (MAP) and normalized discounted cumulative gain (NDCG) [31]. In our problem setup, we only used the binary relevance, that is, the gain will be either 0 or 1.

Research Questions

First, we want to know if our proposed methods are capable of finding meaningful relevant documents based solely on a non-textual time series query. Second, we want to compare the different variations of our proposed algorithms and understand which method works the best. In particular, we would like to compare the Pearson correlation and the dynamic time warping correlation, and compare the weighted TF-IDF document scoring method with the correlation aggregation methods.

[5] `http://finance.yahoo.com/`

Implementation Detail

For the implementation of dynamic time warping, we have used the Machine Learning Python package (mlpy) [6]. We removed stop words using the stopword list provided by the Python Natural Language Toolkit (NLTK [7]).

5.1.6 Experiment Results

Qualitative evaluation

We first examine the quality of the top-ranked documents to see if it is feasible to retrieve meaningful relevant documents solely based on a non-textual time series query. In Table 5.1, we show the dates and excerpts of the top ranked documents obtained by using American Airlines stock price query.

Interestingly, all of the top ranked documents are distributed over the late 2001, and related to the September 11 terrorist attacks and the anti-terrorism activities in 2001. In Table 5.2, we further show the top-10 most correlated terms to the input time series. We again see that all the top terms are related to terror attack and also confirm that the highly correlated terms indeed contributed to the retrieval of the top-ranked relevant documents in Table 5.1.

RANK	DATE	EXCERPT
1	10/22/2001	FLEEING THE WAR
2	12/11/2001	US and anti-Taliban forces in Afghanistan
3	11/18/2001	Fate of Taliban Soldiers Under Discussion
4	11/12/2001	Tally of dead and missing in Sept 11 terrorist attacks (S)
5	11/12/2001	Western soldiers in Afghanistan with Northern Alliance troops near city of Taliqan
6	9/25/2001	Recovery operation at World Trade Center following Sept 11 terrorist attacks
7	11/19/2001	Officials estimate that as of Friday, 4,343 people had died, or were missing and presumed dead, as a result of the attacks on Sept. 11. AT THE WORLD TRADE CENTER 3,953 dead or missing 636 confirmed dead, with 594 identified 157 dead on two hijacked planes, including 10 hijackers.
8	11/3/2001	Dead and Missing report of Sep 11 attack
9	11/17/2001	Dead and Missing report of Sep 11 attack
10	11/18/2001	Dead and Missing report of Sep 11 attack

Table 5.1: Top ranked documents by American Airlines stock price query.

Since our relevance judgments are created automatically based on entity recognition, our algorithm can also be used to (automatically) rank stocks based on how strongly their prices are correlated with topics covered in our news stream. Results in Table 5.4 are ranked by the MAP of Pearson correlation. Results with the American Airline stock query that had a significant cause, namely the September 11 attacks, which was mentioned a lot in the news media, shows the highest precision compared to other companies. This matches our intuition. In general, our algorithm can be used to analyze comparable time series data based on their correlations with topics in text data, offering an interesting

[6] `http://mlpy.sourceforge.net/`
[7] `http://nltk.org/`

WORD	$\lvert\rho\rvert$
challenged	0.887031
afghanistan	0.861351
security	0.858745
sept	0.858309
terrorism	0.854865
pakistan	0.848829
afghans	0.844596
afghan	0.843481
islamic	0.842499
taliban	0.841455

Table 5.2: Top 10 highly correlated words to AA stock (Pearson).

novel way of mining text data for interesting topic patterns that might explain changes in time series data and obtaining potentially useful knowledge.

The results of American Airlines stock price query show that our algorithms can effectively cross the barrier of from time-series to text data to retrieval meaningful relevant articles based solely on a time series. We now show that the results can be significantly improved by further imposing a simple keyword constraint. In Table 5.3, we show the top-10 results of using the stock time series of Apple after imposing the constraint that the documents must contain the term "Apple". Although the Apple stock price has also dropped due to the September 11 attacks, there was an even more significant drop at the end of the September 2000, when it turned out that the Apple's earnings report was not as good as its expectation. At that time, the company's stock price made a historical drop (about 50% off). In Table 5.3, we can observe that all of the top 3 ranked documents directly explain this event. In addition to the earnings report, we can observe the various events that may have affected the company's stock such as the Federal Reserve and the opening of new retail stores. Notice that our algorithm ignored the September 11 related articles, which had a relatively low correlation to stocks in the computer industry.

These results also show that our algorithms can generate an additional ranking score orthogonal to the other ranking features used in a search engine to prefer documents that cover topics correlated with a time series, and thus can help improve the current search engines to generate more useful ranking of documents for the purpose of text analysis in association with a time series.

Quantitative Evaluation

Table 5.4 shows a comparison of the two correlation methods, i.e., Pearson and DTW, by fixing the aggregation function to average correlation (**AC**). We see that the number of relevant documents (from 52 to 575) is very small, when compared to the size of the whole collection (total of 144261 documents). Our algorithm with Pearson correlation shows a 0.0019 mean average precision that is higher than the random precision of 0.0013. It also shows a relatively

RANK	DATE	EXCERPT
1	9/29/2000	Apple Computer says its fourth-quarter earnings will fall far below analysts' estimates ...
2	12/8/2000	Apple Computer; company has about $4 billion in cash and short-term investments in reserve, not $11 billion
3	10/19/2000	Apple Computer announces earnings in fiscal fourth quarter ended Sept 30 that narrowly miss Wall Street's lowered estimates
4	4/19/2001	Dow and Nasdaq Soar After Rate Cut by Federal Reserve
5	7/20/2001	Apple Computer's new retail stores regarding history of company's relationship with several retailers
6	12/6/2000	Apple Warns It Will Record Quarterly Loss
7	3/24/2001	Stocks Perk Up, With Nasdaq Posting Gain in a Harsh Week
8	8/10/2000	Mixing Mac and Windows
9	1/18/2001	Apple Posts $247 Million Loss for Quarter
10	6/10/2001	... corporate alliances unravel and dissolve within decade of inception; prompts consideration of other business alliances, including those between .... Apple and Motorola;

Table 5.3: Top ranked relevant documents by Apple stock price query.

high NDCG, 0.3515. When we used the dynamic time warping for correlation measure, the results showed an even better performance, with a 0.0022 mean average precision and a 0.3609 NDCG. DTW shows a significant improvement over the Pearson correlation (**paired t-test, 95% confidence interval, t-value=2.25 for MAP and 3.43 for NDCG**). The evidence shows that the flexible timeline alignment of dynamic time warping helps to better model the real world stock prices.

Table 5.5 shows the retrieval performance with various aggregation methods when the correlation function is fixed to Pearson correlation. For major comparison, we also performed significant tests with paired t-test with 95% confidence level.

Compared to the average correlation, all variations of the top K average correlation variations and BM25 have shown a significant improvement in NDCG. The results have shown that using only the most correlated terms of the document, instead of all of its terms, results in a more useful ranking. Such technique allows us to avoid the noisy insignificant terms that most documents contain.

When we compared the topK to topK-uniq methods, there were no significant differences. For average correlation, when we use a higher K, it shows a better performance for topK-AC methods (significant improvement from top5-AC to top20-AC). We tried larger values of K, the performance may be further improved slightly, but insignificantly. Extremely large values of K hurt the performance. Overall, K=20 seems to be a reasonable value.

BM25-based method showed higher sensitivity to K. It showed the best performance when $K = 10$, which is comparable to the top-K average correlation methods, but its performance for $K = 5$ and $K = 20$ was significantly worse than the top-K average correlation methods. One possible explanation is that the correlation aggregation methods can more directly and likely more accurately quantify the correlation of the topic in a document than using a

QUERY	REL DOCS	Pearson MAP	Pearson NDCG	DTW MAP	DTW NDCG
American Airlines	393	0.0159	0.5143	0.0159	0.5143
Microsoft Corp	575	0.0027	0.449	0.0027	0.449
Ford Motor	440	0.0027	0.4319	0.0038	0.4515
Verizon	128	0.0026	0.3651	0.0013	0.3401
Time Warner	359	0.0026	0.419	0.0041	0.445
Boeing	137	0.002	0.3561	0.0014	0.3453
General Electric	245	0.0017	0.3809	0.0029	0.407
AT&T	273	0.0017	0.3874	0.0026	0.4072
Intel Corp	168	0.0016	0.3601	0.0016	0.3635
Citigroup	157	0.0014	0.3521	0.0009	0.3379
Cisco Systems	154	0.0012	0.3445	0.0016	0.3576
Amazon.com	159	0.0012	0.3455	0.0014	0.3559
Yahoo	126	0.0012	0.3322	0.0011	0.3313
Sony Corp	206	0.0011	0.3554	0.0015	0.3706
Disney	190	0.0011	0.3541	0.0019	0.3771
Hewlett	145	0.001	0.3346	0.0014	0.3495
Goldman Sachs	121	0.001	0.3248	0.0015	0.3429
Coca-Cola	146	0.0009	0.3341	0.0012	0.3437
Apple Computer	89	0.0006	0.2938	0.0006	0.2938
Toyota	107	0.0006	0.3043	0.0009	0.3157
eBay	99	0.0006	0.3008	0.0009	0.3117
Procter&Gamble	84	0.0005	0.2826	0.0009	0.3033
Wal-Mart	77	0.0004	0.271	0.0008	0.2938
Nissan	55	0.0003	0.2428	0.0004	0.2547
Average	193.04	0.0019	0.3515	0.0022	0.3609

Table 5.4: Comparison of Pearson and DTW.

standard retrieval function such as BM25.

In our qualitative evaluation, we found many top ranked documents are actually quite relevant, but they do not contain the company's name, thus the reported evaluation scores can be assumed to be the minimum potential scores; they would most likely be much higher if we used more complete relevance judgments.

Results show low MAP values, but relatively high NDCG values. The difference is due to the different discounting coefficients used in the two measures to penalize results with lowly ranked relevant documents. As we can tell the definition of MAP and NDCG, while MAP score penalize low position linearly, NDCG uses a logarithmic discounting scheme for lowly ranked relevant documents. Consider a case when we have just one relevant document, If the relevant document is ranked at the 10th rank, the MAP score would get $1/10$, but the NDCG would be $1/log(11)$, substantially higher than the MAP value. Indeed, it seems that we can write the two formulas in a similar form and make a more rigorous comparison as follows.

Suppose we want to compute $MAP@K$ and $NDCG@K$ for a query with a total of R relevant documents in the collection. Assume that there are n relevant documents returned in the top K results at ranks, r_1, .., r_n, respectively.

	MAP	NDCG
AC	0.0019	0.3515
Top5-AC	0.0021	0.361
Top10-AC	0.0023	0.3618
Top20-AC	0.0024	0.3629
Top5-AC-Uniq	0.0022	0.3613
Top10-AC-Uniq	0.0022	0.3616
Top20-AC-Uniq	0.0022	0.3619
Top5-BM25	0.0019	0.3584
Top10-BM25	0.0023	0.361
Top20-BM25	0.0019	0.3582

Table 5.5: Comparison of correlation aggregation methods.

Then, we have

$$MAP@K = \frac{1/r_1 + 2/r_2 + ...n/r_n}{R}$$

$$NDCG@K = \frac{\frac{1}{log(r_1+1)} + \frac{1}{log(r_2+1)} + ... + \frac{1}{log(r_n+1)}}{1 + \frac{1}{log3} + ... + \frac{1}{log(n+1)}}$$

where log is log with base 2. Thus, if we view each as a sum over all the ranks of a relevant document retrieved, then the weight at r_i (rank of the i-th relevant doc) would be i/r_i for MAP and $1/[log(r_i+1)*(1+1/log3+...1/log(n+1))]$ for NDCG. Therefore, the impact of adding 1 to r_i (i.e., moving a relevant down one position in the ranked list), measured by the ratio of the degraded weight to the original weight, would be: $r_i/(r_i+1)$ for MAP and $log(ri+1)/log(ri+2)$ for $NDCG$. We clearly see that $NDCG$ degrade much more slowly.

The low MAP values are due to two reasons. First, the main reason is because our retrieval task is much more challenging than a normal retrieval task as we face a huge gap between the query (a time series) and a relevant document. Thus the task is much harder than a normal retrieval task on average. Second, as we mentioned, our current gold standard does not contain complete relevance judgments, and many top-ranked ones are actually relevant. While we believe the results of relative comparison between different ranking methods are meaningful, the MAP values we obtained are likely a significant under-estimate of the actual MAP values. Using more complete judgments for evaluation is an important future research to be done. The difference between MAP and NDCG is further amplified because we computed them over the whole set of documents. Normally, in retrieval evaluation, we compute NDCG@10 because users generally only look at the top 10 documents, and any differences afterward probably do not matter to our users. However, in our evaluation, our goal is to compare two ranking methods for a very difficult ranking task, we did not use any cutoff and simply used the entire set of documents in order to see any small differences between different methods.

5.1.7 Discussions

In this work, we proposed a general framework to solve a non-traditional retrieval problem. The framework is general and can thus easily support further exploration of this novel problem with more sophisticated correlation functions and aggregation functions, opening up many interesting directions for future research in this direction.

Efficiency

Although we focused on studying the accuracy of the proposed algorithm for solving this new problem in this work, in practical applications, we would also need to consider the efficiency of the algorithm. Our envisioned application of the proposed novel retrieval problem is not for interactive search with time-series queries, but rather for joint analysis of text data and time-series data to discover topics that can potentially explain the changes of the time series data. Thus it does not have to respond in real time as in a regular Web search engine. For example, when analyzing stocks, we can easily afford to spend a few hours to find the most relevant topics. However, it is also possible to imagine other applications where faster responses are highly desirable. In such a case, there are several strategies to consider. First, when we save text collection, as a preprocessing step, we can generate word frequency time series for each term in the data set and index them by terms. Second, our proposed methods can be easily parallelized. When an input time series query arrives, we can calculate correlation coefficient with each term in parallel because the computation for each term is independent. The aggregation step can also be easily parallelized by aggregating a subset of data. The correlation coefficient computation may even be pipelined to the aggregation step for more speedup.

However, because the vocabulary size is huge, the correlation computation may still take a long time. To solve this problem, we can reduce dimensions by clustering temporally and semantically similar terms into a cluster. By using precomputed cluster time series as a unit, we would be able to decrease the amount of computation significantly in the correlation coefficient step as well as the aggregation step.

Relevancy

We assumed that ‘relevant documents to time series’ are highly correlated documents to the input time series that can potentially cause or be caused by the change of time series. Depending on what kind of correlation function to use, users may capture different notions of relevancy thus flexibly supporting different kinds of applications. For example, if a user is only interested in unidirectional causality from a document to input time series, it is possible to use a causality test with lagged value such as Granger test [23] as a correlation function.

In our evaluation, we assumed a document mentioning a company name to be relevant to the query representing the corresponding company’s stocks. While this assumption makes it easy to perform quantitative evaluation, it also has its limitation. In particular, it appears that many relevant documents with topics related to the query time series do not necessarily mention the company's name. In our qualitative evaluation, we found that many top-ranked documents are clearly relevant, but they did not mention the company's name. As a result, they were regarded as non-relevant. This may also be a factor that has artificially lowered the MAP values. Further evaluation with more complete relevance judgments is thus needed in order to draw more reliable conclusions. However, we would like to stress that our results clearly demonstrate the usefulness of the proposed new retrieval task in helping an analyst to go from an arbitrary time series query into a text collection to explore relevant and related topics in the text data, providing a novel way to support text analysis.

5.1.8 Conclusions

In this work, we have presented a novel problem, the information retrieval with time series query. The goal of this retrieval is to find relevant documents in a text collection of the same time period, which contain topics that are correlated with the query time series. The retrieved documents can possibly help the users to understand why the given time series have changed. To solve this new problem, we have proposed and studied multiple retrieval algorithms that use the general idea of ranking text documents based on how well their terms are correlated with the query time series. Experiment results have shown that the proposed retrieval algorithms can effectively help users in finding the relevant documents.

In this work, we have focused on introducing the novel problem setup and have shown a reasonably simple solution. Our work opens up a new line of novel retrieval problems, where the queries are not the usual text queries, but are non-textual variables. While we have only studied the inputs of time series queries in this work, we can imagine other possibilities as well. For example, in the future work, we want to extend the current algorithms with other advanced information retrieval techniques, such as query expansion and feedback process. For example, we may retrieve the top N highly correlated terms and use them as a supplementary measure of the input time series to find correlated documents.

Furthermore, we can think of a more topical analysis, based on the relationships among the terms. The proposed method assumes an independence of terms, and we check the correlation between each term to the input time series independently. We may be able to group related terms (e.g. via semantic similarity, often co-occurring terms, or a high temporal correlation between terms) and measure correlation between the groups of terms and the input time series.

Lastly, we can generate gold standard annotation by experts for better evaluation. Annotating actual correlated articles would give us clearer results in quantitative analysis.

5.2 Mining Causal Topics in Text Data: Iterative Topic Modeling with Time Series Feedback

5.2.1 Introduction

The Web 2.0 environment results in vast amounts of text data published daily. Analyzing and understanding the text data can provide useful information to individuals, industry and government. Finding latent topics is particularly helpful for understanding text data. Probabilistic topic models [10, 25] have proven very useful for mining text data. Topic models are widely used in various areas of text mining including opinion analysis [52, 84, 85], text information retrieval [88], image retrieval [26], natural language processing [12], and social network analysis [57].

Most existing topic modeling techniques focus on text alone. However, text topics often occur in conjunction with other variables through time. Such data calls for an integrated analysis of text and non-text time series data. The causal relationships between the two may be of particular interest. For example, news about companies can affect stock prices. Researchers may be interested in how particular topics lead to increasing or decreasing prices and use the relationships to forecast future price changes. Similar examples occur in many domains. Companies may want to understand how product sales rise and fall in response to text such as advertising or product reviews. Understanding the causal relationships can improve future sales strategies. In election campaigns, analysis of news may explain why a candidate's support has risen or dropped significantly in the polls. Understanding the causal relationships can improve future campaign strategies.

While there are many variants of topic models [8, 9, 87], no existing model incorporates jointly text and 'external' time series variables in search of causal topics. While there may be more rigorous ways to define a causal topic, in this work, we take a computational view and simply loosely define a causal topic as a semantically coherent topic covered in text data that have strong associations with a non-textual time series variable with a possible time lag. This definition includes both topics that potentially cause changes of the time series variable and topics that might have been caused by the changes of the latter. However, we intentionally left open the question whether a causal topic really has a causal relation with the time series variable as this would need a lot of difficult-to-obtain knowledge to assess. For the same reason, we may also use terms 'correlation' and 'causal' interchangeably for convenience.

A natural way to address the problem of discovering causal topics would be to combine existing topic modeling techniques and causality testing methods. One could first find topics with topic modeling techniques, then identify causal topics using correlations and causality tests such as Granger tests. However, this does not consider the time series data in the topic modeling process. Thus, the topic modeling itself does not focus on topics that are particularly closely related to the time series data. As a result, even if two time series variables are completely different (e.g., stock prices for an airline vs. national polls in a presidential candidate), the candidate set of causal topics to be considered would always be the same (a single set of topics is mined from the text corpus), making it impossible to adapt topics to specific time series data.

In this work, we propose a novel general text mining framework, i.e., Iterative Topic Modeling with Time Series Feedback (ITMTF), for discovering causal topics from text. Beyond just retrieving relevant documents as we discussed in Section 5.1, we want to find 'causal topics'. The ITMTF framework naturally combines probabilistic topic modeling with causal analysis techniques for time series data to uncover topics that are both coherent semantically and correlated with time series data. As a general framework, ITMTF can accommodate the use of any topic modeling technique and any causality measure. We believe that the generality is a significant advantage of the framework, which enables a user to easily plug in different topic models and causality measures as needed by a specific application, thus can potentially support many different applications using the same framework.

ITMTF iteratively refines a topic model, gradually increasing the correlation of discovered topics with the time series data through a feedback mechanism. In each iteration, the time series data is used to develop a prior distribution of parameters that feeds back into the topic model. As a result, the discovered topics are dynamically adapted to fit different patterns of different time series data variables.

More specifically, ITMTF works as follows. Given a stream of text articles with time stamps and a time series data, we first apply a topic model (e.g., PLSA or LDA) to discover a set of topics from the text collection. We then check the correlation between each topic and the time series data and identify those topics with high correlations. We further analyze the specific words characterizing each of these topics and pick a subset of words that have the highest correlations to the time series data to further refine a topic. These selected words together with non-selected ("distracting") words can then be fed into a topic model as feedback information to guide the topic model to discover topics that are potentially better correlated with the time series data than the topics discovered by our initial application of the topic model. Such a refinement process of topics based on feedback from time series data can be iteratively repeated to obtain increasingly more meaningful causal topics.

We evaluate ITMTF on a news data set (i.e., one and a half years of New York Times) with multiple stock price time series, including stock prices from the Iowa Electronic Markets and those of two large US companies (i.e., American Airlines and Apple). The experiment results show that the proposed ITMTF framework can effectively discover causal topics from text data, and the iterative process improves the quality of the discovered causal topics.

The main contributions of this work include:

1. We go beyond the existing work on topic modeling to introduce a novel problem setup of discovering causal topics from text data with supervision by time series data, which has many applications in multiple domains.
2. We propose a general framework for solving this new problem that can naturally combine any probabilistic topic model with any correlation or causal analysis method for time series data, and propose a novel iterative

feedback algorithm that uses time series data to influence topic discovery from text by iteratively imposing a time-series-based prior on parameters of a topic model.

3. We evaluate the proposed framework and algorithms on real application data sets and show that the proposed framework can effectively exploit time series data to supervise and improve the discovery of topics from text.

The rest of the sections are organized as follows. In Section 5.2.2, we discuss related previous works. In Section 5.2.3, we motivate a problem formulation and describe the general problem: causal topic mining. In Section 5.2.4, we describe basic idea of the proposed method: Iterative Topic Modeling. After we further explain more background knowledge in Section 5.2.5, we present the proposed framework and algorithm in detail in Section 5.2.6. In Section 5.2.7, we show quantitative and qualitative experimental setup and results. We conclude in Section 5.2.8.

5.2.2 Related Work

The two basic topic models, on which all other models have been built, are Probabilistic Latent Semantic Analysis (PLSA) [25] and Latent Dirichlet Analysis (LDA) [10]. Both focus on word co-occurrences. Recent advanced topic modeling techniques try to analyze the dynamics of topics on a time line [8, 87]. By adding time-related variables to the model, they could find topics on the time line better. By over-laying other time-related variables on the time line, these systems help to visualize the time series characteristics of the topic data. However, they do not conduct integrative analyses of topics and external variables; the topic analysis is separate from the external time series. As a result, the discovered topics would not vary according to different time series data fed into the mining process. In contrast, our approach iteratively guides a topic model to discover topics that are highly correlated with an external time series data.

Previous works also include some efforts to incorporate external knowledge to the modeling process. Supervised topic model [9] and Labeled LDA [78] are methods to model topics with external training labels. Supervised topic model proposed supervised LDA which incorporates a reference value (e.g. movie review article with movie rating) in the modeling process; the modeled topic showed better prediction power on the reference value than simple LDA. Labeled LDA extended the idea of supervised LDA to enable the prediction of categorical labels and even text labels for topics. As another way of incorporating external knowledge to the topic modeling, conjugate prior probability is used to incorporate background knowledge in the topic modeling process [62]. Topic Sentiment Mixture (TSM) modeled positive and negative sentiment topics using seed sentiment words such as ‘good’ or ‘bad’. Although these methods show that topic mining can be guided by external variables, none of these models attempt to capture the correlation between a topic in text and an external time series data variable, which is achieved in our proposed framework. Moreover, while these models are specialized for supervision with specific external data, our proposed approach is general and can thus flexibly combine any reasonable topic model with any causal analysis method.

There is another line of associated work in data stream mining [2]. The main techniques in text data stream mining are clustering and classification. However, they do not address discovering 'topics' in the text alone nor jointly with external time series..

Granger testing [23] is a popular method for testing causality in economics using lead/lag relationships in multiple time series. Recent evidence shows that Granger tests can be used in an opinion mining context: predicting stock price movements with a sentiment curve [11]. However, Granger testing has not been used directly in text mining and topic analysis.

5.2.3 Mining Causal Topics in Text with Supervision of Time Series Data

In many contexts, text data occurs in conjunction with related time series data. Finding causal inter-relationships between the two can be particularly enlightening. For instance, in planning sales strategies, companies would consider the causal relationships between advertising, production reviews, social networking articles (text data) and product sales (external time series data). Similarly, political candidates would be interested in knowing the relationships between campaign rhetoric and advertising (text data) with poll numbers (external time series data).

We can formulate this task as a new research problem, i.e., causal topic mining. Specifically, given a time series data x_1, ..., x_n, with time stamps t_1, ..., t_n, and a text stream which is a collection of documents with time stamps within the same time period, D = $\{(d_1$,$t_{d_1})$, ..., $(d_m$, $t_{d_m})\}$. The goal is to discover a set of causal topics T_1, ..., T_k with associated time lags L_1, ..., L_k. A causal topic T_i with time lag L_i is a topic that is semantically coherent and has a strong correlation with the time series data with time lag L_i. Note that L_i can be positive or negative, corresponding to topics that might cause, or might be caused by, respectively, the change of a time series data.

To the best of our knowledge, no previous work has attempted to solve this new computation problem. The main research question we are interested in is how we can develop *general* methods to solve this new problem. A natural solution to the problem would be to leverage existing work on topic modeling (e.g., PLSA [25] or LDA [10]) to find coherent topics. These topic models can be used to extract multiple topics from text data, where a topic is typically represented as a word distribution. The discovered topics can be loosely regarded as clusters of words based on their co-occurrences in documents. However, in addition to topic coherence which general topic modeling techniques aim at, we also want to make sure mined topics are well-correlated to the external time series variables. We thus would also want to rely on causality measures such as Granger Test to assess the causality of a topic. A straightforward way to solve our problem is thus the following. We first model topics based only on text data to maximize word coherence, and then rank topics based on topic correlation with the time series data. However, this approach has a drawback, i.e., the formation of topics is completely independent of the time series data, thus we may be "stuck" with whatever topics a topic model can discover. Indeed, even if we are given two different time series data, our candidate set of topics to choose would still be exactly the same. This is clearly non-optimal. To overcome this limitation, in the next section, we propose a general framework called Iterative Topic Modeling with Time Series Feedback (ITMTF), which attempts to allow the formation of a topic to be influenced by its potential correlation with the time series variable, thus jointly optimizing both the coherence and correlation of topics.

5.2.4 Iterative Topic Modeling with Time Series Feedback

By definition of our causal topic mining task, there are two criteria to optimize: topic coherence and topic correlation. That is, we would like to discover topics from text that are both semantically meaningful (if interpreted by a person), and are well correlated with the time series data by a time lag. Thus conceptually, we face an optimization problem where we would like to search for a set of topics to maximize both their coherence and correlations. The simple approach described above searches in a very limited space of topics determined by the topic model, and the time series data do not influence the topic formation at all. However, the approach does have the advantage that it is general and can thus accommodate any specific topic model and any causality measure. Our main idea is thus to retain the generality of the framework, but extend the approach to allow the time series variable to influence the formation of topics so that we can search in a much larger and more flexible topic space. That is, we would repeat the simple approach multiple times, and every time, we would "steer" the topic model to form and discover topics that are more likely to be correlated with the time series variable by imposing a prior to favor topics that have high correlations with the time series variables. This results in a general causal topic mining framework for Iterative Topic Modeling with Time Series Feedback (ITMTF).

More specifically, the framework works as follows. After the first execution of topic modeling and correlation measurement (straightforward combination), the topics are remodeled using **feedback** from the correlation measures. This allows a correlated topic to be further refined to improve both correlation and coherence. The main idea is to first select some of the most significantly correlated words (or terms) from the correlated topics to form a new candidate topics (which improves the correlation relation at the cost of potentially generating an incoherent topic), and then try to make the topic more semantically coherent by using it as a prior for another round of topic modeling.

Clearly the ITMTF framework is general and can take any topic model and any causality measure as components to perform causal topic mining; from application perspective, this is a desirable advantage as it enables a user to flexibly choose any topic model and causality measure according to the needs of a specific appication problem.

Before we present our framework in detail, we first give a brief introduction to topic models and time series data analysis as background.

5.2.5 Background

Probabilistic Topic Models

Topic modeling is a generative process for latent topic discovery from documents. It specifies a probabilistic process by which documents can be generated from a set of topics, where a topic is represented by a multinomial distribution of words, usually based on a unigram language model. In the document generation process, one usually first chooses a distribution over topics. Then for each word in the document, one chooses a topic at random according to this distribution and draws a word from that topic.

More formally, for the document collection $D = \{d_1, ..., d_m\}$, assuming that documents cover a finite set of k topics, each topic j is associated with a multinomial word distribution $\phi^{(j)}$ over the vocabulary V. A document d is represented as a bag of words. We use $\theta^{(d)}$ to denote the multinomial distribution over topics for document d. The parameter ϕ indicates which words are important for which topic, and the parameter θ indicates which topics are important for a particular document.

As an output of topic modeling, topics can be defined as a list of words with a probability for each word.

$$T_j = (w_1, \phi^{(j)}_{w_1}), (w_2, \phi^{(j)}_{w_2}), ..., (w_{|V|}, \phi^{(j)}_{w_{|V|}})$$

For example, we can say that 'attack 0.4, september 0.3, airline 0.3' is a topic of 'September 11', and 'good 0.4, excellent 0.4, awesome 0.2' is a topic of 'positive opinion'.

We now briefly introduce a basic topic model, i.e., Probabilistic Latent Semantic Analysis (PLSA) [25], which we use in our experiments. Note that our framework is completely general and any other topic model can be plugged into the framework for causal topic analysis.

Probabilistic latent semantic analysis (PLSA) [25] regards a document d as a sample of the following mixture model.

$$P(w|d) = \sum_{j=1}^{k} \phi^{(j)}_w \theta^{(d)}_j$$

The word distributions $\hat{\phi}$ and topic mixture distributions $\hat{\theta}$ can be estimated with an Expectation-Maximization (EM) algorithm [16] by maximizing the (log) likelihood that the collection D is generated by this model:

$$\log P(D|\phi, \theta) = \sum_{d \in D} \sum_{w \in V} \left\{ c(w, d) \log \sum_{j=1}^{k} \phi^{(j)}_w \theta^{(d)}_j \right\}$$

$$(\hat{\phi}, \hat{\theta}) = argmax_{\phi,\theta} P(D|\phi, \theta)$$

where $c(w,d)$ is the number of times that word w occurs in the document d.

Parameters can also be estimated in consideration of prior on parameters by using Bayesian estimation such as Maximum a Posteriori estimation:

$$(\hat{\phi}, \hat{\theta}) = argmax_{\phi,\theta} P(D|\phi,\theta)P(\phi,\theta)$$

To make prior affect the modeling process, when we estimate $\phi_w^{(j)}$, we can add pseudo counts of w from prior ϕ'_w:

$$\phi_w^{(j)} = \frac{\sum_{d\in D} c(w,d)P(z_{d,w}=j) + \mu\phi'_w}{\sum_{w'\in V}\sum_{d\in D} c(w',d)P(z_{d,w'}=j) + \mu}$$

where, μ is the sum of all pseudo counts which actually controls the strength of prior. In effect, when we add a prior, we would "force" the estimated topic word distribution to be close to the word distribution specified in the prior (i.e., ϕ'_w). How close it is to the prior is controlled by parameter μ. Thus, we can leverage the prior to influence and bias a topic model to discover a certain kind of desired topics; indeed, this is what we will do in the proposed iterative topic modeling approach.

Causal analysis with time series data

Between two time series, if one of the time series changes or is changed by the influence of other time series, we can say they have a causal relationship. As mentioned in the Introduction, we loosely use the term "causality" to broadly include when they have sufficiently strong correlations. In the same sense, we also use terms such as "correlated" and "causal" interchangeably.

We consider the concept of 'lag' that is the gap to deliver the effect of one series to another. The sign of the lag also suggests the direction of causality. That is, positive lag between A to B means, changes of A affects B after the lag time, and negative lag mean that B causes changes of A.

There are various ways to measure causality between two time series. The simplest measure is Pearson correlation. Pearson correlation is one of the most common methods to measure correlation between two series. It gives us a $[-1, +1]$ range score, and the sign of the output value indicates the orientation of the correlation, which can be used as 'impact' in our framework. Depending on the correlation value and the number of samples, we can even measure the significance of the output value. The basic Pearson correlation would have zero lags because it compares values on the same time stamp. Lagged correlation is also possible by shifting one of the input time series by the lag and measuring the Pearson correlation.

To focus more on 'causal' relationships, we can apply Granger Tests. Granger tests perform statistical significance tests with different time lags using auto regression to see if one time series has a causal relationship to another series.

Let y_t and x_t be two time series to be tested. To see if x_t has Granger causality to y_t with maximum p time lag, basic formula for Granger test is

$$y_t = a_0 + a_1 y_{t-1} + ... + a_p y_{t-p} + b_1 x_{t-1} + ... + b_p x_{t-p}$$

Then, Granger testing performs F-tests to evaluate if the lagged x terms should be retained or not. Because Granger tests are basically F-tests, they naturally give us significance values for causality. We can estimate the impact of the x to y from the coefficients of x terms; for example, we can average the x term coefficients, $\frac{\sum_{i=1}^{p} b_i}{|p|}$, as an impact value.

5.2.6 An Iterative Topic Modeling Framework with Time Series Feedback

In this section, we present the ITMTF framework and algorithm in detail. The input and output of the algorithm is as follows:

1. **Input:** time series data X = x_1, ..., x_n with time stamp t_1, ..., t_n, and a text stream which is a collection of documents with time stamp within the same time period, D = $\{(d_1,t_{d1})$, ..., $(d_m, t_{dm})\}$, Topic modeling method M, Causality measure C, a parameter tn (how many topics to model)
2. **Output:** k potentially causal topics (k$\leq tn$): $(T_1, L_1), ..., (T_k, L_k)$

Topic modeling method M would give us estimated topics, and we assume that causality measure C would give us output of significance (e.g. p-value) and orientation of impact. ITMTF is a general novel algorithm for iterative analysis of causal topics as illustrated in Figure 5.3. It works as follows:

1. Apply M to D to generate tn topics $T_1, .., T_{tn}$
2. Use C to find topics with significance values $sig(C, X, T) > \gamma$ (e.g. 95%). Let CT be the set of candidate causal topics with lags: $CT = \{(T_{c1}, L_1), ..., (T_{ck}, L_k)\}$.
3. For each candidate topic in CT, apply C to find most significant causal words among top words $w \in T$. Record the impact values of these significant words (e.g., word-level Pearson correlations with time series variable).
4. Define a prior on the topic model parameters using significant terms and their impact values.
   (a) Separate positive impact terms and negative impact terms. If one orientation is very weak ($<\delta\%$, e.g. 10%), ignore the minor group.
   (b) Assign prior probability proportion to the level of significance.
5. Apply M to D by using the prior obtained in the previous step (note that the imposing of prior injects feedback signals into the topic model and helps steering the topic model to form topics that are more likely correlated with the time series data)

6. Repeat 2-5 until satisfying stopping criteria (e.g. reach topic quality at some point, no more significant topic change). When the process stops, CT would be the output causal topic list.

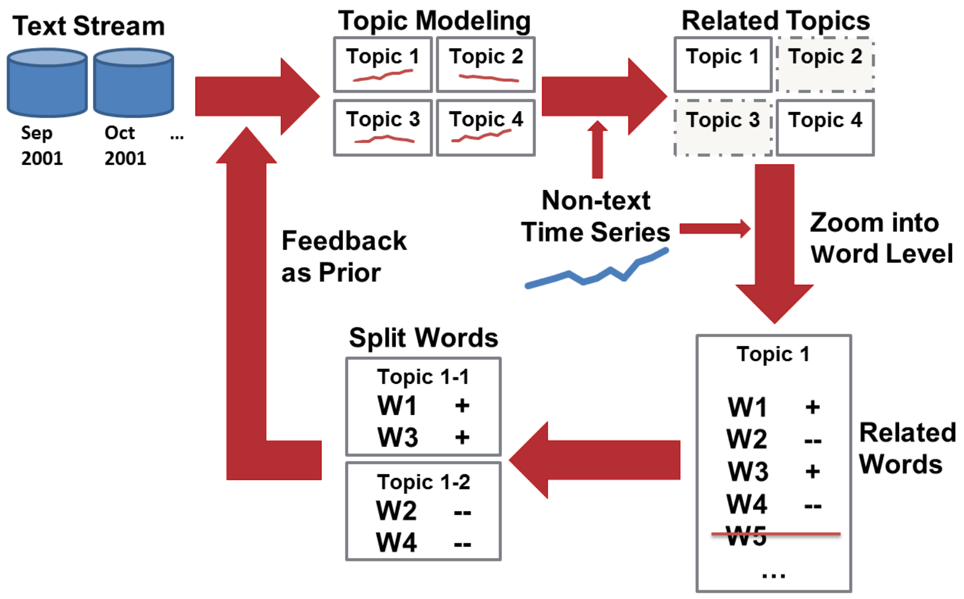


Figure 5.3: Overview of iterative topic modeling algorithm.

Unlike other general topic modeling systems, ITMTF considers the non-textual time series data in the text mining process and finds topics that are more highly correlated to non-textual data. Moreover, ITMTF iteratively improves the modeling results with consideration of interactions between the text and time series data at both the topic and word levels. ITMTF first finds causal topics and then zooms into the word level and checks correlations between word frequency time series and external non-textual data. It also improves the topic modeling process by splitting positively and negatively impacting terms into different topics. Because generating and testing all the word time series is inefficient, ITMTF focuses only on the words with the highest probabilities in the most highly correlated topics discovered in each iteration.

As we discussed before, conceptually, the ideal set of causal topics should have high causality with respect to external time series as well as high topic quality. Traditional topic modeling algorithms form topics based on word coherences in the text data, while causality tests can filter out non-causal topics. If we only focus on one of these criteria, we would sacrifice the other criterion. We can regard this as an optimization problem with two terms to maximize. Our iterative process is an greedy approximate solution to the problem. Our iterative process takes turns to optimize each criterion. Figure 5.4 illustrates the "alternating optimization" procedure adopted by the ITMTF algorithm, in which the prior formation based on causality attempts to optimize causality while an execution of topic model optimizes coherence of a prior topic.

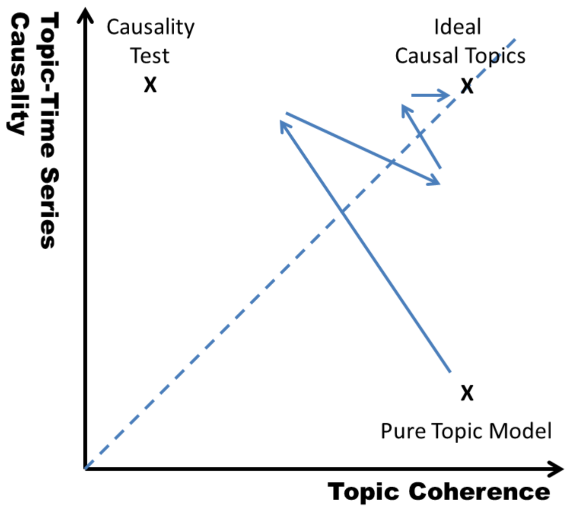


Figure 5.4: Conceptual idea of iterative topic modeling process.

Below we will discuss each step in more detail. Note that the instantiation discussed below is only one of the many ways to refine the framework; the framework itself is quite general and can certainly be potentially instantiated in other ways.

Topic-level Causality Analysis

From topic modeling results, we would be able to generate a topic curve over time by computing the coverage of each topic on each time unit (e.g., one day). The coverage can be based on the count of documents covering a topic. Specifically, we can compute the coverage of a topic in each document, $p(topic_j|Document_d)$. Since this is actually the parameter $\theta_j^{(d)}$ which we estimated in the modeling process, we naturally obtain it after estimation of the model. We can estimate the coverage of topic j at t_i, tc_i^j based on the sum of $\theta_j^{(d)}$ over all the documents with t_i time stamp. We will call the list of tc^j for all the time stamps *topic stream* TS_j:

$$tc_i^j = \sum_{\forall d \ with \ t_i} \theta_j^{(d)}$$

$$TS_j = tc_1^j, tc_2^j, ..., tc_n^j$$

Since a topic stream is also a numerical time series, we can apply any time series causality measure C on top of the topic stream and the provided non-textual time series data to generate a causality score for a topic.

When we measure causality, we also need to find the optimal lag value. One way to search optimal lag is using the most significant lag within the given maximum lag, p. For example, with a day as a time unit, if we want to find causal topics within 5 days, we can test causality with lag 1 to 5 and pick the lag that showed the highest significance value.

If we want to focus on yesterday's effect, we can just use the fixed lag 1. For contemporaneous Pearson correlations, we use a zero lag. There are many variants that a user can choose based on specific needs in an application.

Word-level Causality Analysis and Prior Generation

TOPIC	WORD	IMPACT	SIGNIFICANCE (%)
1	social	+	99
	security	+	96
	tax	+	97
	gun	-	98
	control	-	96
4	september	-	99
	airline	-	99
	terrorism	-	97
	... (5 more words)		
	attack	-	96
	good	+	96

Table 5.6: Example of topic and word correlation analysis result.

TOPIC	WORD	PROBABILITY
1	social	0.8
	security	0.2
2	gun	0.75
	control	0.25
3	september	0.1
	airline	0.1
	terrorism	0.075
	... (5 more words)	
	attack	0.05
	good	0.0

Table 5.7: Example prior generated.

Based on causality scores of topics, we can find a subset of promising topics with the highest causality scores and further analyze the words within each topic to provide feedback for the next iteration by generating topic priors. Specifically, for each significant topic, we check whether the top words in the topic are also significantly correlated with the external time series. Thus for each word, we can generate a word count stream WS_w by counting its frequencies in the input document collection for each day:

$$wc_i = \sum_{\forall d \ with \ t_i} c(w, d)$$

$$WS_w = wc_1, wc_2, ..., wc_n$$

where $c(w, d)$ is the count of word w in document d.

We can then measure correlations and significance between word streams and the non-textual external time series as we did between topic streams and the non-textual time series. This would allow us to identify words that are significantly correlated and their impact values.

Intuitively, we would like to keep these significant topics and words in our next topic modeling iteration so that we can further search in a more promising topic space. To do this, we generate topic priors for significant words of significant topics. A topic prior is a Dirichlet distribution that favors topics assigning high probabilities to the identified "significant words" in significant topics. We assign word probabilities in the prior in proportion to the significance value of the words. In effect, the prior helps "steer" the topic modeling process to form/discover topics similar to the prior topics [62], thus in the next iteration of topic modeling, the discovered topics would be relatively close to the prior we provided, which is further based on the feedback from the time series variable through causality scores.

In addition to keeping significant topics and words, we also want to improve topic quality. A 'good' correlated topic has a consistent impact relative to the correlated external time series. We thus identify relatively consistent or pure topic, which have the words that have mostly 'positive' or mostly 'negative' impact relative to the non-textual external time series. Therefore, if one topic has both positive and negative impacting words, we separate the positive and negative impact words into two topics in the topic prior for the next topic modeling iteration. If one of the groups is much smaller than the other (e.g. the number of positive impact words ¡ 0.1 * the number of negative impact words), we do not make the smaller group as a separate group, and we just make probability of words in smaller group zero, which "tells" the topic model not to include such words in the refinement of the topic.

For example, suppose among N total topics, we found topic 1 and topic 4 with sufficiently high correlations with the external time series (Table 5.6). We checked the correlations of the top words in these two topics, and found 5 words and 11 words were significant, for the two topics respectively. For topic 1, because we observe both positive and negative word groups with similar sizes, we would split them into two topics and assign word probabilities based on significance values (i.e., how much more significant compared to the cutoff (95%)). For topic 4, only one word has a different impact orientation from the others. Thus the negative group is much smaller than the positive group. Therefore, instead of making a separate negative word group topic, we just make sure it does not appear in the topic corresponding to the positive word group by assigning it a zero weight. Table 5.7 shows the example prior generated.

How to place a cutoff on 'top' words in the correlation list is an interesting challenge. The simplest solution is to set a fixed cutoff, say k, and use the top k words. However, rank alone does not determine the importance of words. Importance is determined by word probabilities as well. For example, suppose the top three words in Topic 1, A, B, and C have probabilities 0.3, 0.25 and 0.2, respectively, and the top three words in Topic 2, D, E, and F have probabilities 0.002, 0.001 and 0.0001. In this case, A, B and C are very important in Topic 1. However, D, E, and F combined only represent a small part of Topic 2. Hence, Topic 2 may require more words to be considered. We solve this problem by using a cumulative probability mass cutoff, $probM$. That is, we use all the words whose accumulated probabilities are within a probability mass cutoff. For example, with 0.5 cutoff, we would use A and B for Topic 1 and may use many more words for Topic 2. In both cases, the set of words tested for significance explains the top 50% of the topic.

Formally, for each topic $T_j = (w_1, \phi^{(j)}_{w_1})$, ..., $(w_{|V|}, \phi^{(j)}_{w_{|V|}})$ ($|V|$ is the number of words in the input data set vocabulary), when items are sorted by $\phi^{(j)}_{w_i}$, we can add the top ranked word to the top word list TW without violating the constraint $\sum_{w \in TW} \phi^{(j)}_w \leq probM$ where $TW = (w_1, ..., w_m)$. That is, $\sum_{w \in TW} \phi^{(j)}_w + \phi^{(j)}_{w_{m+1}} > probM$. With top word $TW = (w_1, ..., w_m)$ and significance value of each word $sig(C, X, w)$, topic prior ϕ'_w can be computed by the following formula:

$$\phi'_w = \frac{sig(C, X, w) - \gamma}{\sum_{j=1}^{m}(sig(C, X, w_j) - \gamma)}$$

where γ is a significance cutoff (e.g. 95%). This prior would be plugged into the topic model estimation algorithm as described in Section 5.2.5.

Iterative Modeling with Feedback

Using the generated prior, we remodel topics. Now, topics will be guided by priors, which depend on correlations with the external data. High probability words in the prior would have a greater impact in the modeling results and words with zero probability will not be included in the topic. By repeating the pipelined process of topic modeling, correlation analysis, and prior generation, the resulting topics can be expected to be more highly correlated to the external time-series data.

The strength of prior in the next iteration can be decided by a parameter μ in the modeling process. With $\mu = 0$, the topic modeling would not consider the prior information at all, and it will be just same as independent modeling. With a very high μ, words in the prior are very likely to appear in the topic modeling results. We will study the influence of this parameter in our experiments.

Note that, while we observe correlations between non-textual series and both word streams and topic streams, we do not compute correlations for all word streams. Word level analysis would give us finer grain signals. However, generating all the word frequency time series and testing correlations would be very inefficient. By narrowing down to significant topics first, we can prune the number of words to test. By doing this, we increase efficiency and effectiveness. Also, note that this iterative topic modeling algorithm can be generalized using any topic modeling techniques and causality/correlation measures desired.

5.2.7 Experiments

Experiment Design

We evaluated the proposed algorithms on a large news data set, i.e., the New York Times data set [8] with multiple stock time series data.

In one set of experiments, we examined the 2000 U.S. Presidential election campaign. The input text data comes from New York Times articles from May through October of 2000. We filtered them for key words 'Bush' and 'Gore,' and used paragraphs mentioning one or both of these key words. The idea will be to find specific topics that caused support for Bush or Gore to change and not simply election related topics.

As a non-textual time series input, we used prices from the Iowa Electronic Markets (IEM)[9] [7] 2000 Presidential Winner-Takes-All Market. In this on-line futures market, prices forecast the probabilities of candidates winning[10] the election. We follow the standard practice in the field and use the "normalized" price of one of the candidates as a sufficient statistic for the probability of the election outcome. We use Gore price/(Gore price + Bush price) that represents a forecast of Gore's probability of winning (and, by definition, 1 minus Bush's probability).

In another experiment, we used the stock prices of two large US companies (American Airlines and Apple) and the same New York Times text data set with longer time period without keyword filtering, where we could examine the influence of having different time series variables for supervision.

We use PLSA for topic modeling implemented based on the Lemur information retrieval toolkit.[11] For correlation measures, we use both simple Pearson correlation coefficients and Granger tests. For Granger tests, we use the R statistical package[12] implementation. Granger tests require stationary time series as inputs. To make the input series stationary, we first smooth series with moving average filter with window size 3 (average with adjacent values) and use first difference $(x_t) - (x_{t-1})$ of each series. We test causality up to 5 lags with a day as a time unit, and pick the lag that shows highest significance. For Pearson correlation, we use a zero lag.

Evaluation

1. Measure: To measure the quality of topics mined, we report two measures, causality confidence and topic purity. Causality confidence shows the level of assurance of causality/ correlation. Better-mined topics should have higher causality to the input external time series data. We checked significance value (e.g. p-value of the Granger test) between the external variable and the topic stream for confidence.

[8] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2008T19

[9] http://tippie.uiowa.edu/iem/

[10] "Winning" as defined by the IEM is taking the majority of the two-party popular vote

[11] http://www.lemurproject.org/

[12] http://www.r-project.org/

Topic purity is a measure of directional impact consistency across words in a topic. "Pure" topics are characterized by words that have the same impacts orientations. Topic purity is calculated based on the impact orientation distribution of significant words. That is, if all the significant words in one topic have the same orientation, it would have 100% purity, and if significant words are evenly split into positive and negative impact words, it would have 0% purity. We calculate the entropy of significant word distributions and normalize it to the $[0, 100]$ range. The Purity of topic T is defined as follows:

$$pProb = \frac{the\ number\ of\ positive\ impact\ words}{the\ number\ of\ significant\ words}$$

$$nProb = \frac{the\ number\ of\ negative\ impact\ words}{the\ number\ of\ significant\ words}$$

$$Entropy(T) = pProb * log(pProb) + nProb * log(nProb)$$

$$Purity(T) = 100 + 100 * Entropy(T)$$

When we report causality confidence and topic purity, we are going to report average confidence and purity for all the significant topics (more than 95% significance). That is, when there are more significant topics, this measure may be penalized. However, because measuring general utility of 'retrieved' topics is meaningful as a user perspective, we do not change the formula of this measure.

2. Baseline Our natural baseline is traditional topic modeling technique with causality test "without feedback." In our experiment reported in Section 5.2.7, the 'first iteration' results corresponds to this straight forward combination (no feedback). From the second iteration on, the framework uses feedback from the previous iteration. By comparing the performance of the first iteration and later iterations, we can assess if the feedback process has benefits in finding causal topics.

3. Parameter test We test two parameters that can affect performance of our algorithm. The first parameter is the number of topics (tn) for topic modeling. A large number of topics may help identify more specific topics. Topics are also likely to be coherent (have higher purity). However, topics that are too specific result in sparseness of data that reduces the power of any significance testing. A smaller number of topics would give us the opposite effects. Topics are likely to have higher statistical significance, but would have lower purity. Also, because many meaningful topics may be merged into a single topic, topics may be too coarse to interpret meanings easily.

The second parameter is the strength of prior (μ). A stronger topic prior would guarantee prior information is delivered to the next iteration of topic modeling. However, if the initial topic modeling (which uses random initiation without a prior) ends up at a poor local optimum, a strong prior may keep the process near the local optimum and result in poor topics. Strong priors may also exacerbate problems resulting from noise in the topic or word correlation analyses. In contrast, a weaker prior allows a less restricted iteration of topic modeling. This would reduce the negative effects of a poor initial modeling round and noise. However, positive signals would also be delivered weakly.

Because prior research gives no guidance for selecting these parameters, we examine how they affect the performance of our algorithm.

Experiment Result

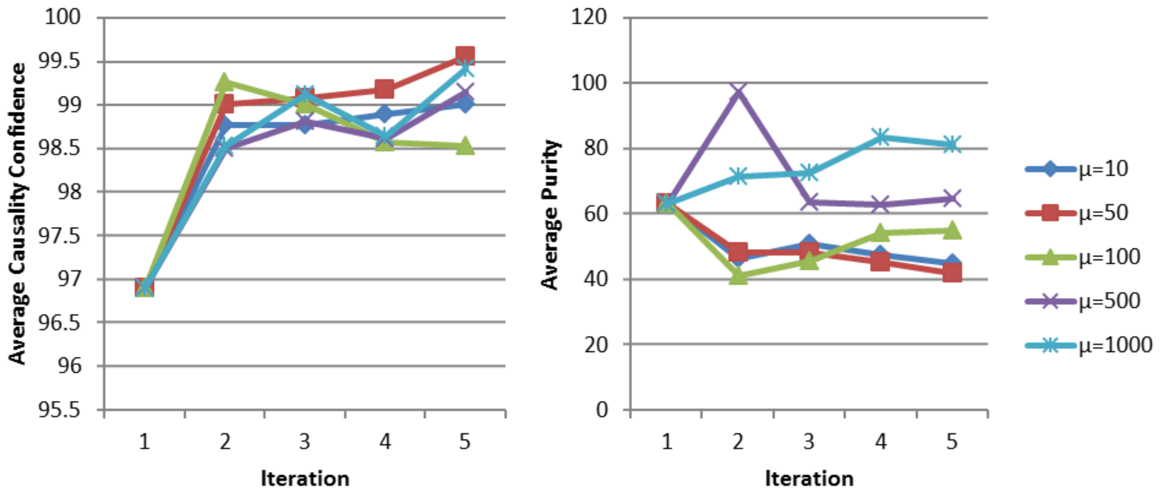


Figure 5.5: Performance with different μ over iteration. Left: causality confidence, right: purity. (Presidential election data, Granger Test.)

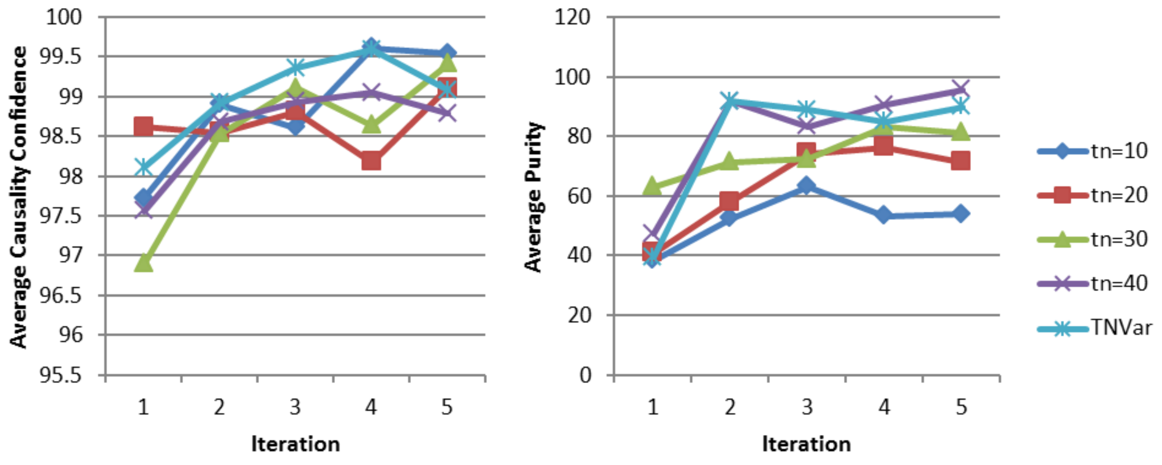


Figure 5.6: Performance with different topic number, tn, over iteration. Left: causality confidence, right: purity. (Presidential election data, Granger Test.)

Figure 5.5 shows performance evaluation results with different μs using Granger tests. Average causality confidence increases over iterations regardless of the strength of feedback, μ. The performance improvement is particularly notable between the first iteration (which is a straightforward combination without feedback) and the second iteration (which is the first modeling using feedback). This shows the clear benefit of feedback. After the second iteration, performance shows slower improvement. This occurs because, after the first feedback round, later feedback rounds fine tune topics resulting from the first feedback round.

The average purity graph shows mixed results. For small μ values (μ=10, 50, 100), iterations do not always improve purity. With higher μ values (μ=500, 1000), average purity improved from the first iteration to the second. Furthermore, for the highest μ value (μ=1000), it showed a steady increasing trend. Weak μs would allow topic modeling more room to explore variations regardless of prior. Therefore, the purity improvement may not be guaranteed in the next iteration. Thus, high μ values lead to higher increases in purity. The reported purity is the averaged purity values of all the significant topics found. The number of significant topics increased dramatically between the first and second iteration. As a result, the drop in average purity may not be a negative sign. Still, if we want to ensure purity improvement, it would be better to use high μ value.

Figure 5.6 shows performance evaluation results with different topic numbers (tn) using Granger tests. On average, small topic numbers (tn=10, 20) had higher confidence levels than large topic numbers (tn=30, 40) from the beginning. This corresponds with our intuition that the statistical signal is stronger with small topic numbers, while large topic numbers would suffer because of weak statistical signals from sparseness. However, as the iterative process goes on, relative order changes. We can interpret that the feedback process helps to overcome sparseness problem by delivering 'good' signals to the next iteration. That is, because significantly correlated topics and words will be kept by the priors, and topic modeling would add value to them, the number of topics has little effect on average confidence.

In general, modeled topics with larger tn shows higher purity than modeled topics with small tn. As we expected, with topics, each topic would be more specific and the chance to have multiple real topics within one modeled topic would be smaller. Therefore, it is more likely to have better purity.

We performed the same series of tests using Pearson correlation measure with various parameter settings. In general, we could observe similar trends as observed with Granger tests. Graphs are omitted because of the space limit.

To prove the improvement is not from randomness of topic modeling, we performed significance test on performance between the first iteration and the second iteration. We executed 20 separate trials with random initiation and applied ITMTF (tn=30, μ=1000). Paired t-test between the first and the second iteration showed 99% significance for each measure (t-value: 3.87 for average confidence, 14.34 for average purity). This confirms that our feedback process helps to improve average correlation and topic quality of significant topics over the simple topic modeling technique.

According to the experiment results, although it is rather clear that high μ values are helpful to improve topic quality, finding the appropriate tn value is still not clear. In the next section, we will describe a new approach and experiment results for finding the optimal tn.

Finding Optimal Number of Topics

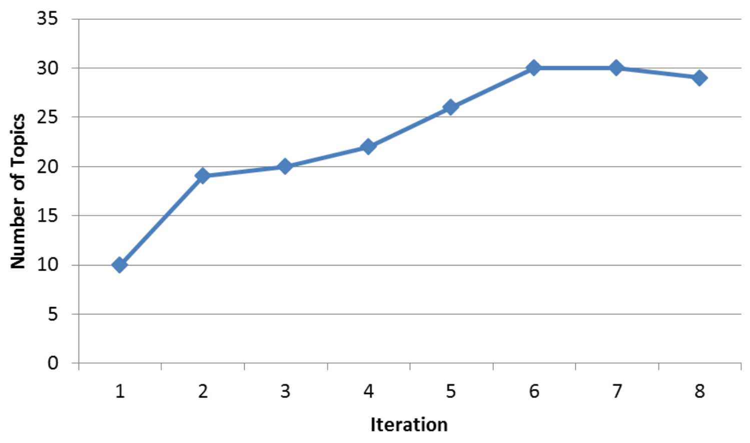


Figure 5.7: The number of topics used in each iteration with variable topic number approach. (Presidential election data, Granger Test.)

In our problem setup, we specified the number of topic, tn, as one of our inputs. In practice, deciding appropriate tn may not be easy. To help to find the optimal number of topics, tn, we propose a 'variable topic number' approach. Our algorithm naturally split topics over the iteration. Therefore, we can start with the small tn, let the algorithm split the topics in prior, and use the increased number of topics in the next iteration. Here, to give topic modeling room to explore more hidden topics, we can add some buffer topics (e.g. 5 more topics than increased number of topics). For example, if we start with $tn = 10$, 7 out of 10 topics are found as causal, and 5 out of 7 are split, then 12 topics will be generated in the topic prior. By adding 5 more buffer topics to 12, our tn value for the next iteration will be 17.

As we observed In Section 5.2.7, topics tend to have high causality with small tn. Therefore, it is likely most of topics are found to be causal from the beginning. As iteration goes on, topics are split, causal topic proportion will be decreased, and finally we will be able to find the optimal number of topics. We can stop when the number of topics starts to decrease compared to the previous iteration, which means topic split hurts to find more causal topics. Figure 5.7 shows how topic numbers changed with variable topic number approach with the presidential election data set. We can see the number of topics starts to decrease after 30, then 30 would be our tn to use.

We also tested the performance of the variable topic number method during topic number search. *TNVar* in Figure 5.6 shows average confidence and purity compared to the fixed tn methods. In both average confidence and purity, variable topic number approach shows best performance in most iteration. These results show the proposed approach is helpful to find the optimal number of topics.

TOP THREE WORDS IN SIGNIFICAN TOPICS
tax cut 1
screen pataki giuliani
enthusiasm door symbolic
oil energy prices
pres al vice
love tucker presented
partial **abortion** privatization
court supreme **abortion**
gun control nra
news w top

Table 5.8: Significant topic list of 2000 Presidential Election. (Each line is a topic. Top three probability words are displayed.)

Sample Result: 2000 Presidential Election

Table 5.8 shows sample results from the 2000 U.S. Presidential election campaign case. It shows top three words of significant causal topics mined (Pearson correlation, tn=30, μ=50, 5th iteration). The mining result reveals several important issues that form causal topics in the 2000 U.S. Presidential elections, e.g. tax cuts, abortion, gun control, and energy. Such topics are also cited in political science literature [73] and Wikipedia [13] as important issues of the election. This results shows that our iterative topic mining process is effective in mining causal topics.

Sample Result: One Text Two Time Series

AAMRQ	AAPL
russia russian putin	paid notice st
europe european germany	russia russian europe
bush gore presidential	olympic games olympics
police court judge	she her ms
airlines airport air	oil ford prices
united trade terrorism	black fashion blacks
food foods cheese	**computer technology software**
nets scott basketball	**internet com web**
tennis williams open	football giants jets
awards gay boy	japan japanese plane
moss minnesota chechnya	...

Table 5.9: Significant topic list of two different external time series: AAMRQ and AAPL. (Each line is a topic. Top three probability words are displayed.)

To show the effect of our algorithm intuitively, we show how one text data set is modeled into two different sets of topics based on different external time series. For this experiment, we used New York Times articles from July 2000 through December 2001 as the text input. We use American Airlines (AAMRQ) and Apple (APPL) stock prices as external time series. American Airlines' stock dropped significantly in September 2001 because of the 9/11 terrorist

[13]http://en.wikipedia.org/wiki/United_States_presidential_election,_2000#General_election_campaign

attack, while Apple stock in IT industry was affected less by the event. We start with the same modeled topic list at the first iteration. Thus, any differences in modeled topics result are from the impact of the external time series.

Table 5.9 shows the top three words of significant topics mined using the two different external time series after three feedback rounds (tn=30 and μ=1000). For topics guided by the American airlines stock price (AAMRQ), we see clearly relevant topics such as 'airlines airport air' and 'united trade terrorism.' One topic is clearly about the terrorist attack on the World Trade Center. Topics guided by Apple stock price (AAPL) differ dramatically from the American Airlines set. Topics that reference the IT industry such as 'computer technology software' and 'internet com web' are very relevant to Apple.

This example also shows some limitation of our current algorithm. In addition to clearly relevant topics, there appear many other general topics such as sports or the election (which occurred in 2000). This is clearly a more challenging task than the 2000 U.S. Presidential election example because of the diversity in text data and long time period. While we use text articles that are related to candidates for the Presidential election case, we use the entire newspaper within the time period for this example. Therefore, the topic diversity is much larger here and some topics happen to correlate statistically regardless of real causality. Moreover, for this experiment, our analysis is over 18 months. Our algorithm measures correlation over the entire input time period. Therefore, if an event is only locally correlated, it may not be selected in the final output topic list. How to measure and deliver local correlation feedback remains for future work.

Despite of these difficulties, our algorithm shows interesting output topics are selected by using different external time series on the same text data and initially modeled topics. This result shows our algorithm can effectively guide topic modeling. By prefiltering relevant articles and focusing on shorter time periods, our algorithm would give better results. Moreover, although some topics do not seem related to the stock price, they may be revealing relationships that were unexpected. Thus, we may be able to find interesting hidden relationships.

5.2.8 Conclusions

In this work, we presented a novel general text mining framework: Iterative Topic Modeling with Time Series Feedback (ITMTF) for causal topic mining. ITMTF takes a text data stream and a non-text external time series as inputs and iteratively models topics related to changes in the external time series. ITMTF analyzes topic correlations as well as word correlations, and then generates feedback for the next iteration of topic modeling to arrive at topics that are more correlated with the external time series. According to experimental results, ITMTF is helpful to find topics that are both more pure and more highly correlated with the external time series, especially with a strong feedback loop.

In future work, we hope to extend ITMTF's ability to identify locally correlated events. An initial approach will be to divide the time series into segments and apply the technique to individual segments or use individual segments for feedback information. Using moving windows may also allow the framework to identify local correlations.

Finally, the general problem of mining causal topics opens up new directions for future research, both theoretical and applied. Our framework has the limitation that it may end up in a local optimal solution. On the theoretical side, there may be ways for improving iterative algorithms for optimizing two criteria of causal time mining or non-iterative ways to find globally optimal solutions. We may be able to design an objective function reflecting causality and topic coherence, and then we can try to find global optimal or apply other approximation algorithms to solve the problem efficiently. On the applied side, there is a myriad practical applications where the text based "buzz" of the news, blogs, social media and other Internet sources drives or is driven by a non-text time series.

5.3 Summary

In this chapter, we described two lines of work, both aiming to help users analyze causal topics in text in association with a time series input. In the first line, we first proposed methods for retrieving relevant documents that contain topics correlated with the query time series. In the second line, we proposed a framework to mine causal topics by jointly analyzing text and non-text data using iterative feedback.

Because of the decoupling of the time series correlation measure from the topic analysis part, all these methods can be flexibly combined with many different ways of preprocessing the time series data and measuring correlations. Below we discuss some possible extensions of the work in this way.

5.3.1 Patterns to Replace Words

Using larger than word unit has advantages not only in computational efficiency as we explained in Sec 5.1.7 but also in richer information delivery to the framework. For example, we can use n-gram or word clusters instead of unigram as a unit. Furthermore, we can mine patterns from input text data [45] and use the patterns to find correlated documents or causal topics. Top correlated patterns would show highly correlated topics more intuitively, and they are likely to deliver more accurate information in finding relevant documents and causal topics.

5.3.2 Time Series Normalization

In our experiments, we used raw stock price data for correlation calculation. Although company's stock price is affected by how the company does in the market, it is also highly affected by a general economic trend. In that case, our algorithm may retrieve documents reflecting the general economic trend, not the company-specific one. To solve this problem, we can apply appropriate normalization techniques to the non-textual data as a preprocessing step. For example, by normalizing raw stock price by Dou Jones Industrial Average, we would be able to get the company specific stock price trend. Or, if we use artificial non-textual data such as sentiment curves, we can use the positive sentiment value normalized by negative sentiment one.

5.3.3 Local Correlation

In our experiments, correlation/causality measures we used are for global correlation/causality over the entire time period. Surely, for the specific time window we are interested in, we can simply shorten the gap between starting and ending date of observation. Otherwise, we can plug-in different correlation measures that capture local correlation to the current framework. For example, simply, we can think of a correlation measure that divides the input time frame evenly and returns the maximum correlation among all the divided time windows. For this strategy, any words/topics highly correlated 'in at least one time window' would have high correlation.

Another way to emphasize a part of the entire time period is using different weights depending on time. The basic approach presented in this work uses uniform weighting over the entire time period. By assigning higher weights in correlation/causality calculation, we can make the algorithm consider more in the interested time window.

Weighting strategy to the causal topic mining framework can have another advantage in making the causality measurement more intelligent, and it can possibly improve topic modeling process. For example, assigning higher weights on highly causal words may lead the modeled list of topics to be more causal.

Chapter 6

Conclusion and Future Work

This dissertation systematically studies to understand explanatory opinion that shows more detailed reasons of opinion from text. We presented series of techniques for providing detailed explanatory summary of opinion to users: (1) techniques to measure explanatoriness of text and extract explanatory phrase, and (2) techniques to effectively extract representative and contrastive opinions from different opinion orientations. We further added temporal factor and presented (3) techniques to retrieve documents and mine causal topics correlated with external time series data.

To generate an explanatory summary of opinions, we proposed and evaluated multiple algorithms for generating an explanatory summary at the level of sentences. Experiment results show that the proposed probabilistic models appear to be most effective. We further proposed a HMM-based method for compact explanatory summary extracting within-sentence explanatory segments. Experiment results show that HMM-based unified framework is the most effective for compact explanatory summary.

To summarize contrastive opinions, we formulated an optimization framework for finding contrastive subset pairs and proposed two approximation algorithms. According to experiment results, Contrastiveness-First approximation works better than Representativeness-First, and our heuristics of removing sentiment words in measuring contrastiveness is effective.

To analyze causal topics in text with supervision of time series data, we first proposed methods for retrieving relevant documents containing correlated topics with input time series query. Experiment results have shown that the proposed retrieval algorithm can effectively help users to find the relevant documents. Beyond retrieving correlated documents, we further proposed a general text mining framework for finding causal topics using iterative feedback between text and non-text external time series. Experiment results show that the proposed method is helpful to find topics that are both consistent and highly causal to the external time series with a feedback loop.

Although we evaluated these algorithms on only a few selected domains, they are generally and mostly language-independent, and thus can be applied to any domain. For many techniques proposed, they even can be applied to non-opinionated data. As a result of our study, we also created multiple test collections that can facilitate further study of these problems.

There are many possible future directions in this work. We already proposed various future works for each work at the end of each chapter. Below we further discuss some interesting general directions for future work.

By using more domain and language specific features, we can improve the performance of proposed algorithms. In this dissertation, we mainly focused on proposing new problems and providing general unsupervised solutions for the problems. These approaches require minimum training data. For better accuracy with specific application purposes, it is possible to use additional training data or domain-specific information such as linguistic features. It is also possible to use supervised learning for solving the proposed problems. In this case, the result of current methods always can be used as one of the features of the supervised learning approaches.

In addition to ideas for algorithm accuracy and performance, implementing proposed algorithms as a part of a system is one of the most effective ways to show the usefulness of algorithms intuitively. For example, we can think of unsupervised text analysis system. Users can search or upload their own data set to analyze. They can perform explanatory or contrastive summarization. With additional upload of external time series data, users may be able to find causal topics by jointly analyzing text data with external times series.

References

[1] P. Achananuparp, X. Hu, and X. Shen. The evaluation of sentence similarity measures. In *DaWaK '08: Proceedings of the 10th international conference on Data Warehousing and Knowledge Discovery*, pages 305–316, Berlin, Heidelberg, 2008. Springer-Verlag.

[2] C. C. Aggarwal. Mining text streams. In C. C. Aggarwal and C. Zhai, editors, *Mining Text Data*, pages 297–321. Springer, 2012.

[3] R. Agrawal, C. Faloutsos, and A. N. Swami. Efficient similarity search in sequence databases. In *Proceedings of the 4th International Conference on Foundations of Data Organization and Algorithms*, FODO '93, pages 69–84, London, UK, UK, 1993. Springer-Verlag.

[4] R. Agrawal, K.-I. Lin, H. S. Sawhney, and K. Shim. Fast similarity search in the presence of noise, scaling, and translation in time-series databases. In *Proceedings of the 21th International Conference on Very Large Data Bases*, VLDB '95, pages 490–501, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[5] R. Agrawal, G. Psaila, E. L. Wimmers, and M. Zaït. Querying shapes of histories. In *Proceedings of the 21th International Conference on Very Large Data Bases*, VLDB '95, pages 502–514, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[6] M. Bansal, C. Cardie, and L. Lee. The power of negative thinking: Exploiting label disagreement in the min-cut classification framework. In *Proceedings of COLING: Companion volume: Posters*, pages 13–16, 2008.

[7] J. Berg, R. Forsythe, F. Nelson, and T. Rietz. *Results from a Dozen Years of Election Futures Markets Research*, volume 1 of *Handbook of Experimental Economics Results*, chapter 80, pages 742–751. Elsevier, 2008.

[8] D. M. Blei and J. D. Lafferty. Dynamic topic models. In *Proceedings of the 23rd international conference on Machine learning*, pages 113–120, New York, NY, USA, 2006. ACM.

[9] D. M. Blei and J. D. Mcauliffe. Supervised topic models. 2007.

[10] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

[11] J. Bollen, H. Mao, and X.-J. Zeng. Twitter mood predicts the stock market. *CoRR*, abs/1010.3003, 2010.

[12] J. Boyd-Graber, D. M. Blei, and X. Zhu. A topic model for word sense disambiguation. In *EMNLP '07: Proceedings of the 2007 conference on Empirical Methods in Natural Language Processing*, 2007.

[13] J. Carbonell and J. Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '98, pages 335–336, New York, NY, USA, 1998. ACM.

[14] K.-P. Chan and A. W.-C. Fu. Efficient time series matching by wavelets. In *Data Engineering, 1999. Proceedings., 15th International Conference on*, pages 126 –133, mar 1999.

[15] M. C. de Marneffe, A. N. Rafferty, and C. D. Manning. Finding contradictions in text. In *Proceedings of ACL-08: HLT*, pages 1039–1047, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[16] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of Royal Statist. Soc. B*, 39:1–38, 1977.

[17] X. Ding, B. Liu, and P. S. Yu. A holistic lexicon-based approach to opinion mining. In *Proceedings of the international conference on Web search and web data mining*, WSDM '08, pages 231–240, New York, NY, USA, 2008. ACM.

[18] H. P. Edmundson. New methods in automatic extracting. *J. ACM*, 16(2):264–285, 1969.

[19] M. Efron. Linear time series models for term weighting in information retrieval. *J. Am. Soc. Inf. Sci. Technol.*, 61(7):1299–1312, July 2010.

[20] G. Erkan and D. R. Radev. Lexrank: graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.*, 22(1):457–479, 2004.

[21] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos. Fast subsequence matching in time-series databases. In *Proceedings of the 1994 ACM SIGMOD international conference on Management of data*, SIGMOD '94, pages 419–429, New York, NY, USA, 1994. ACM.

[22] K. Ganesan, C. Zhai, and J. Han. Opinosis: a graph-based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 340–348, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[23] C. W. J. Granger. Essays in econometrics. chapter Investigating causal relations by econometric models and cross-spectral methods, pages 31–47. Harvard University Press, Cambridge, MA, USA, 2001.

[24] D. Gunopulos and G. Das. Time series similarity measures. In *Tutorial notes of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '00, pages 243–307, New York, NY, USA, 2000. ACM.

[25] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57, New York, NY, USA, 1999. ACM.

[26] E. Hörster, R. Lienhart, and M. Slaney. Image retrieval on large-scale image databases. In *CIVR '07: Proceedings of the 2007 ACM international conference on Image and video retrieval*, pages 17–24, New York, NY, USA, 2007. ACM.

[27] E. Hovy and C.-Y. Lin. Automated text summarization in SUMMARIST. In I. Mani and M. T. Maybury, editors, *Advances in Automatic Text Summarization*. MIT Press, 1999.

[28] M. Hu and B. Liu. Mining and summarizing customer reviews. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177, New York, NY, USA, 2004. ACM.

[29] M. Hu and B. Liu. Mining opinion features in customer reviews. In *AAAI'04: Proceedings of the 19th national conference on Artifical intelligence*, pages 755–760. AAAI Press, 2004.

[30] M. Hu and B. Liu. Opinion extraction and summarization on the web. In *AAAI'06: proceedings of the 21st national conference on Artificial intelligence*, pages 1621–1624. AAAI Press, 2006.

[31] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, Oct. 2002.

[32] J. Jiang and C. Zhai. Extraction of coherent relevant passages using hidden markov models. *ACM Trans. Inf. Syst.*, 24(3):295–319, July 2006.

[33] N. Jindal and B. Liu. Identifying comparative sentences in text documents. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 244–251, New York, NY, USA, 2006. ACM.

[34] K. S. Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: development and comparative experiments. *Inf. Process. Manage.*, 36(6):779–808, Nov. 2000.

[35] R. Jones and F. Diaz. Temporal profiles of queries. *ACM Trans. Inf. Syst.*, 25(3), July 2007.

[36] J. Kacprzyk and A. Wilbik. Linguistic summarization of time series using linguistic quantifiers: Augmenting the analysis by a degree of fuzziness. In *Fuzzy Systems, 2008. FUZZ-IEEE 2008. (IEEE World Congress on Computational Intelligence). IEEE International Conference on*, pages 1146 –1153, june 2008.

[37] M. Kaya and M. Karsligil. Stock price prediction using financial news articles. In *Information and Financial Engineering (ICIFE), 2010 2nd IEEE International Conference on*, pages 478 –482, sept. 2010.

[38] E. Keogh. Fast similarity search in the presence of longitudinal scaling in time series databases. In *Proceedings of the 9th International Conference on Tools with Artificial Intelligence*, ICTAI '97, pages 578–, Washington, DC, USA, 1997. IEEE Computer Society.

[39] E. Keogh and P. Smyth. A probabilistic approach to fast pattern matching in time series databases. page 24. AAAI Press, 1997.

[40] E. J. Keogh and M. J. Pazzani. Relevance feedback retrieval of time series data. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '99, pages 183–190, New York, NY, USA, 1999. ACM.

[41] E. J. Keogh and M. J. Pazzani. Derivative dynamic time warping. In *In First SIAM International Conference on Data Mining (SDM'01*, 2001.

[42] H. D. Kim, M. G. Castellanos, M. Hsu, C. Zhai, U. Dayal, and R. Ghosh. Ranking explanatory sentences for opinion summarization. In *Proceedings of the 36th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '13, New York, NY, USA, 2013. ACM.

[43] H. D. Kim, K. Ganesan, P. Sondhi, and C. Zhai. Comprehensive review of opinion summarization. *Computer Science Research and Tech Reports*, 2011.

[44] H. D. Kim, D. Nikitin, C. Zhai, M. Castellanos, and M. Hsu. Information retrieval with time series query. In *Proceedings of the 4th International Conference on the Theory of Information Retrieval*, ICTIR '13. Springer, 2013.

[45] H. D. Kim, D. H. Park, Y. Lu, and C. Zhai. Enriching text representation with frequent pattern mining for probabilistic topic modeling. *Proceedings of the American Society for Information Science and Technology*, 49(1):1–10, 2012.

[46] H. D. Kim and C. Zhai. Generating comparative summaries of contradictory opinions in text. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 385–394, New York, NY, USA, 2009. ACM.

[47] H. D. Kim, C. Zhai, T. A. Rietz, D. Diermeier, M. Hsu, M. Castellanos, and C. A. Ceja Limon. Incatomi: integrative causal topic miner between textual and non-textual time series data. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, CIKM '12, pages 2689–2691, New York, NY, USA, 2012. ACM.

[48] L.-W. Ku, Y.-T. Liang, and H.-H. Chen. Opinion extraction, summarization and tracking in news and blog corpora. In *AAAI Symposium on Computational Approaches to Analysing Weblogs (AAAI-CAAW)*, pages 100–107, 2006.

[49] A. Kulkarni, J. Teevan, K. M. Svore, and S. T. Dumais. Understanding temporal query dynamics. In *Proceedings of the fourth ACM international conference on Web search and data mining*, WSDM '11, pages 167–176, New York, NY, USA, 2011. ACM.

[50] J. Kupiec, J. Pedersen, and F. Chen. A trainable document summarizer. In *SIGIR '95: Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 68–73, New York, NY, USA, 1995. ACM.

[51] R. Liebscher and R. K. Belew. Lexical dynamics and conceptual change: Analyses and implications for information retrieval. *Cognitive Science Online*, pages 46–57, 2003.

[52] C. Lin and Y. He. Joint sentiment/topic model for sentiment analysis. In *CIKM '09: Proceedings of the 2009 ACM international Conference on Information and Knowledge Management*, pages 375–384, New York, NY, USA, 2009. ACM.

[53] C.-Y. Lin. Training a selection function for extraction. In *CIKM '99: Proceedings of the eighth international conference on Information and knowledge management*, pages 55–62, New York, NY, USA, 1999. ACM.

[54] B. Liu. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data (Data-Centric Systems and Applications)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[55] B. Liu. Sentiment analysis and subjectivity. In N. Indurkhya and F. J. Damerau, editors, *Handbook of Natural Language Processing, Second Edition*. CRC Press, Taylor and Francis Group, Boca Raton, FL, 2010. ISBN 978-1420085921.

[56] B. Liu, M. Hu, and J. Cheng. Opinion observer: analyzing and comparing opinions on the web. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 342–351, New York, NY, USA, 2005. ACM.

[57] Y. Liu, A. Niculescu-Mizil, and W. Gryc. Topic-link lda: joint models of topic and author community. In *ICML '09: Proceedings of the 2009 annual International Conference on Machine Learning*, pages 665–672, New York, NY, USA, 2009. ACM.

[58] Y. Lu, C. Zhai, and N. Sundaresan. Rated aspect summarization of short comments. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 131–140, New York, NY, USA, 2009. ACM.

[59] H. P. Luhn. The automatic creation of literature abstracts. *IBM J. Res. Dev.*, 2(2):159–165, 1958.

[60] R. Malouf and T. Mullen. Graph-based user classification for informal online political discourse. In *Proceedings of the 1st Workshop on Information Credibility on the Web (WICOW)*, Miyazaki, Japan, 2007.

[61] C. D. Manning, P. Raghavan, and H. Schtze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[62] Q. Mei, X. Ling, M. Wondra, H. Su, and C. Zhai. Topic sentiment mixture: modeling facets and opinions in weblogs. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 171–180, New York, NY, USA, 2007. ACM.

[63] M. Meinard. *Introduction to Information Retrieval*. Springer, 2007.

[64] R. Mihalcea and P. Tarau. A language independent algorithm for single and multiple document summarization. In *In Proceedings of IJCNLP'2005*, 2005.

[65] G. Mitra and L. Mitra. *The handbook of news analytics in finance /*. Wiley ;, Hoboken, N.J. :, 2011.

[66] A. Nenkova and K. McKeown. *Automatic Summarization*. Now Publishers, 2011.

[67] M. Ng and Z. Huang. Temporal data mining with a case study as astronomical data analysis. *Lecture Notes in Computer Sciences*, pages 2–18, 1997.

[68] M. Osborne. Using maximum entropy for sentence extraction. In *Proceedings of the ACL-02 Workshop on Automatic Summarization*, pages 1–8, Morristown, NJ, USA, 2002. Association for Computational Linguistics.

[69] C. D. Paice. Constructing literature abstracts by computer: techniques and prospects. *Inf. Process. Manage.*, 26(1):171–186, 1990.

[70] B. Pang and L. Lee. *Opinion Mining and Sentiment Analysis*, volume 2(1–2) of *Foundations and Trends in Information Retrieval*. Now Publ., 2008.

[71] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *EMNLP '02: Proceedings of the ACL-02 conference on Empirical methods in natural language processing*, pages 79–86, Morristown, NJ, USA, 2002. Association for Computational Linguistics.

[72] T. Pedersen, S. Patwardhan, and J. Michelizzi. Wordnet: : Similarity - measuring the relatedness of concepts. In *AAAI*, pages 1024–1025, 2004.

[73] G. Pomper. *The election of 2000: reports and interpretations*. ELECTION OF. Chatham House Publishers, 2001.

[74] A.-M. Popescu and O. Etzioni. Extracting Product Features and Opinions from Reviews. In *Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, pages 339–346. Association for Computational Linguistics, October 2005.

[75] K. Radinsky, E. Agichtein, E. Gabrilovich, and S. Markovitch. A word at a time: computing word relatedness using temporal semantic analysis. In *Proceedings of the 20th international conference on World wide web*, WWW '11, pages 337–346, New York, NY, USA, 2011. ACM.

[76] D. Rafiei and A. Mendelzon. Similarity-based queries for time series data. *SIGMOD Rec.*, 26(2):13–25, June 1997.

[77] D. Rafiei and A. Mendelzon. Efficient retrieval of similar time series. In K. Tanaka, S. Ghandeharizadeh, and Y. Kambayashi, editors, *Information Organization and Databases*, volume 579 of *The Springer International Series in Engineering and Computer Science*, pages 75–89. Springer US, 2000.

[78] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning. Labeled lda: a supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 248–256, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[79] R. P. Schumaker and H. Chen. Textual analysis of stock market prediction using breaking financial news: The azfin text system. *ACM Trans. Inf. Syst.*, 27(2):12:1–12:19, Mar. 2009.

[80] A. Singhal. Modern information retrieval: A brief overview. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 24(4):35–42, 2001.

[81] K. Sparck Jones. Document retrieval systems. chapter A statistical interpretation of term specificity and its application in retrieval, pages 132–142. Taylor Graham Publishing, London, UK, UK, 1988.

[82] K. Takahashi and M. Umano. Retrieval of similar time series with similarity degree of linguistic expressions for global trend and local features. In *Fuzzy Systems (FUZZ-IEEE), 2012 IEEE International Conference on*, pages 1 –8, june 2012.

[83] M. Thomas, B. Pang, and L. Lee. Get out the vote: Determining support or opposition from Congressional floor-debate transcripts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 327–335, 2006.

[84] I. Titov and R. McDonald. A joint model of text and aspect ratings for sentiment summarization. In *ACL '08: Proceedings of the 2008 annual meeting on Association for Computational Linguistics*, pages 308–316, Columbus, Ohio, 2008. Association for Computational Linguistics.

[85] I. Titov and R. McDonald. Modeling online reviews with multi-grain topic models. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 111–120, New York, NY, USA, 2008. ACM.

[86] M. Umano, M. Okamura, and K. Seta. Improved method for linguistic expression of time series with global trend and local features. In *Fuzzy Systems, 2009. FUZZ-IEEE 2009. IEEE International Conference on*, pages 1169 –1174, aug. 2009.

[87] X. Wang and A. McCallum. Topics over time: a non-markov continuous-time model of topical trends. In *Proceedings of the 12th ACM SIGKDD international conference*, pages 424–433, New York, NY, USA, 2006. ACM.

[88] X. Wei and W. B. Croft. Lda-based document models for ad-hoc retrieval. In *SIGIR '06: Proceedings of the 2006 international ACM SIGIR conference on research and development in Information Retrieval*, pages 178–185, New York, NY, USA, 2006. ACM.

[89] B.-K. Yi, H. V. Jagadish, and C. Faloutsos. Efficient retrieval of similar time sequences under time warping. In *Proceedings of the Fourteenth International Conference on Data Engineering*, ICDE '98, pages 201–208, Washington, DC, USA, 1998. IEEE Computer Society.

[90] C. Zhai. *Statistical Language Models for Information Retrieval.* Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, 2008.

[91] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214, Apr. 2004.

[92] C. Zhai, A. Velivelli, and B. Yu. A cross-collection mixture model for comparative text mining. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 743–748, New York, NY, USA, 2004. ACM.

[93] L. Zhuang, F. Jing, and X.-Y. Zhu. Movie review mining and summarization. In *CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 43–50, New York, NY, USA, 2006. ACM.