

TOWSON UNIVERSITY

OFFICE OF GRADUATE STUDIES

**A STUDY ON TEXTUAL CONTENTS IN ONLINE COMMUNITIES AND  
SOCIAL MEDIA USING TEXT MINING APPROACHES**

by

Beomseok Hong

A Dissertation

Presented to the faculty of

Towson University

in partial fulfillment

of the requirements for the degree

Doctor of Science

Department of Computer and Information Sciences

Towson University

Towson, Maryland 21252

(December, 2017)

© 2017 By Beomseok Hong

All Rights Reserved

TOWSON UNIVERSITY OFFICE  
OF GRADUATE STUDIES

DISSERTATION APPROVAL PAGE

This is to certify that the thesis prepared by [INSERT Student's Name] \_\_\_\_\_

**Beomseok Hong** \_\_\_\_\_ entitled [INSERT Title of Thesis] \_\_\_\_\_


**A Study on Textual Contents in Online Communities  
and Social Media using Text Mining Approaches**

has been approved by the thesis committee as satisfactorily completing the dissertation  
requirements for the degree\_ [INSERT Type of Degree] Doctor of Science  
(for example, Doctor of Science)

 YANGGON KIM 11-27-2017  
Chairperson, Dissertation Committee Signature Type Name Date

 YANGGON KIM 11-27-2017  
Dissertation Advisor Signature, Type Name Date  
if other than Chairperson

 Sungchul Hong 11-27-2017  
Committee Member Signature Type Name Date

 Nam Nguyen 11-27-2017  
Committee Member Signature Type Name Date

 Ziyi Tang 11-27-2017  
Committee Member Signature Type Name Date

 Janet DeHany 12-11-17  
Dean of Graduate Studies Type Name Date

## Acknowledgments

This dissertation would not have been possible without the help, support, and guidance of many people.

I would like to express the deepest gratitude to my advisor Dr. Yanggon Kim. He has generously guided me through the research and writing of dissertation. His patience, training, and guidance are greatly appreciated and will not be forgotten. I also would like to thank my committee members, Dr. Sungchul Hong, Dr. Nam Nguyen, and Dr. Ziyang Tang for all the suggestions and constructive advice.

I would like to thank Dr. Kwangmi Kim for her assistance and guidance of the research. I have enjoyed doing research with her in that the techniques in my research could be applied in a different field of research. I would like to thank my colleagues, Dr. Youngsub Han and Seongik Park, for constructive discussions and the help during this dissertation.

Lastly, I would like to thank my parents, Gihong Hong and Yusun Park, and my sister, Miji Hong. They have always supported and thoughtfully encouraged me to continue to research during my graduate studies.

## Abstract

### A Study on Textual Contents in Online Communities and Social Media using Text Mining Approaches

Beomseok Hong

With the advent of Web 2.0, users have become more interactive, and the population of user-generated contents (UGC) has also increased drastically on the web. Among various Web 2.0 applications, we focus on textual contents in social media and online question answering communities.

Twitter has become one of the fastest growing social media sites, and is serving as an electronic word-of-mouth (eWOM) that affects customers' buying decisions by sharing opinions and information about brands. However, lexical ambiguity is an obstacle to analyzing the data in social media for online reputation management. The enormous amount of tweets makes it impossible for a human to manually disambiguate them. Therefore, we propose an automated company name discrimination using topic signatures. From the experiment, we found that news articles can be used to extract topic signatures, and these topic signatures improved the company name discrimination result as compared to the baseline.

Community Question Answering (CQA) sites are knowledge sharing platforms that allow users to post questions and answer questions asked by other users. There is a time lag between questions and answers. Askers need to wait for answers, and some of the questions are never answered. To solve this problem, we propose a weighted question retrieval method using the relationship between titles and descriptions. From the experiment, we found that exploiting the question descriptions increased the ranks of the relevant questions while reducing the recalls of them.

Software information sites such as Stack Overflow, Super User, and Ask Ubuntu are specific CQA sites that allow software-related questions and tagging systems. Tagging systems help to organize, search, and explore their questions for future use. However, the tag explosion and tag synonym are common problems in tagging systems, because tags are added and created by non-expert users. To mitigate these problems, we propose a tag recommendation method using the highest topic filtering. From the experiment, we observed that our tag recommendation method considerably improved rank-related results and that recommended tags can improve the quality of their questions.

## Table of Contents

<b>LIST OF TABLES .....</b>	<b>IX</b>
<b>LIST OF FIGURES .....</b>	<b>X</b>
<b>ACRONYMS .....</b>	<b>XI</b>
<b>CHAPTER 1 INTRODUCTION.....</b>	<b>1</b>
1.1 BACKGROUND .....	1
1.2 PROBLEM STATEMENT .....	5
<b>CHAPTER 2 LITERATURE REVIEW.....</b>	<b>11</b>
2.1 TEXT MINING METHODS .....	11
2.2 WORD SENSE DISAMBIGUATION IN SOCIAL MEDIA.....	14
2.3 QUESTION RETRIEVAL IN CQA .....	16
2.4 TAG RECOMMENDATION IN CQA.....	17
<b>CHAPTER 3 COMPANY NAME DISCRIMINATION IN SOCIAL MEDIA.....</b>	<b>20</b>
3.1 METHODOLOGY .....	21
3.1.1 <i>Collecting News Articles</i> .....	21
3.1.2 <i>Topic Signature Extraction</i> .....	23
3.1.3 <i>Classification</i> .....	25
3.2 EXPERIMENTAL SETUP AND RESULTS .....	26
3.2.1 <i>Data Sets and Preprocessing</i> .....	26
3.2.2 <i>Evaluation Metrics</i> .....	28
3.2.3 <i>Feature Selection</i> .....	29

3.2.4	<i>Threshold for Extracting Topic Signatures</i>	30
3.2.5	<i>Results</i>	34
<b>CHAPTER 4 RECOMMENDATIONS IN CQA</b>		<b>39</b>
4.1	QUESTION RETRIEVAL IN CQA	39
4.1.1	<i>Okapi BM25</i>	41
4.1.2	<i>Weighted Question Similarity</i>	41
4.2	EXPERIMENTAL SETUP AND RESULTS	43
4.2.1	<i>Data Sets and Preprocessing</i>	44
4.2.2	<i>Results</i>	45
4.3	TAG RECOMMENDATION IN CQA	50
4.3.1	<i>Topic Models</i>	50
4.3.2	<i>A Ranking Function using Highest Topic Filtering (HTF)</i>	52
4.4	EXPERIMENTAL SETUP AND RESULTS	55
4.4.1	<i>Data sets and Preprocessing</i>	56
4.4.2	<i>Evaluation Metrics</i>	59
4.4.3	<i>Training Topic Models</i>	61
4.4.4	<i>Results</i>	63
<b>CHAPTER 5 CONCLUSION</b>		<b>69</b>
<b>REFERENCES</b>		<b>72</b>
<b>CURRICULUM VITAE</b>		<b>85</b>



## List of Tables

Table 3-1. Relatedness of collecting news article using different search keywords .....	23
Table 3-2. A list of 27 companies used in the evaluation .....	28
Table 3-3. A topic signature for 'Lexus' .....	32
Table 3-4. The average number of words in topic signatures for each threshold .....	34
Table 3-5. The top 5 companies using the body feature by f-measure .....	36
Table 3-6. The top 5 companies using the snippet feature by f-measure .....	37
Table 3-7. A topic signature for 'Apple Inc'. .....	38
Table 4-1. The details of data sets .....	44
Table 4-2. The question retrieval result in Yahoo! Answers .....	46
Table 4-3. The question retrieval result in Stack Overflow .....	47
Table 4-4. Examples of the rank rises of the relevant questions by the weighted model in the Yahoo! Answers collection .....	49
Table 4-5. An example of document-topic distribution in LDA .....	52
Table 4-6. Document similarity between the target document and existing documents ..	54
Table 4-7. The details of the data sets .....	58
Table 4-8. Top 10 words for different topics discovered by LDA (K = 160) .....	63
Table 4-9. Tag recommendation result for the validation set with regard to top 300 tags	65
Table 4-10. Tag recommendation result for the test set with regard to the top 300 tags..	65
Table 4-11. Tag recommendation result for the test set with regard to all tags .....	66
Table 4-12. An example of the test set with the recommendation result of $TagScore_{HT} = 2$ .....	68

## **List of Figures**

Figure 1-1. Tag popularity curve in Stack Overflow .....	9
Figure 1-2. Tag synonym project in Stack Overflow .....	10
Figure 2-1. Latent Dirichlet Allocation (LDA) Plate Notation.....	14
Figure 3-1. A system architecture of the company name classifier .....	21
Figure 3-2. The retrieval rate of news article features .....	30
Figure 3-3. The accuracy of the classification result on various thresholds .....	33
Figure 3-4. The f-measure of the classification result on various thresholds .....	33
Figure 3-5. A comparison of the evaluation result .....	36
Figure 4-1. An architecture of a weighted question retrieval model .....	40
Figure 4-2. A weighted question similarity model .....	43
Figure 4-3. Document coverage by the most frequent tags .....	58
Figure 4-4. A log likelihood of LDA for a different topic K.....	61

## **Acronyms**

<b>API</b>	Application Programming Interface
<b>CQA</b>	Community Question Answering
<b>eWOM</b>	Electronic Word-of-Mouth
<b>HTML</b>	Hypertext Markup Language
<b>JSD</b>	Jensen Shannon Divergence
<b>LDA</b>	Latent Dirichlet Allocation
<b>MAP</b>	Mean Average Precision
<b>MRR</b>	Mean Reciprocal Rank
<b>nDCG</b>	Normalized Discounted Cumulative Gain
<b>NLP</b>	Natural Language Processing
<b>ORM</b>	Online Reputation Management
<b>QA</b>	Question Answering
<b>TF-IDF</b>	Term Frequency-Inverse Document Frequency
<b>UGC</b>	User-Generated Content
<b>WSD</b>	Word Sense Disambiguation
<b>XML</b>	Extensible Markup Language

## **Chapter 1**

### **Introduction**

#### **1.1 Background**

Web 2.0 has quickly become popularized and has changed the way people behave online [1]. As the change of user behavior is a vital factor of Web 2.0, many applications provide users with the place to generate, disseminate, share, edit, and refine informational content. Users have become more interactive accordingly, and the population of user-generated content (UGC) has also increased drastically in social media [2]. Web 2.0 applications allow users to post content without requirements and qualifications, and encourage them to interact with each other. Examples of Web 2.0 applications include Facebook, YouTube, Twitter, Instagram, Stack Exchange, CiteULike, Flickr, and so on. Among various Web 2.0 applications, we focus on social media and online communities for question answering.

Social media has significantly impacted our lives and has changed the way people communicate with one another, largely due to the explosion of social media and technologies. With the rapid growth of smartphone ownership, the increasing popularity of social media has accelerated. Diverse social media platforms, such as Facebook, YouTube, and Twitter, have rapidly grown in size and influence. Marketers therefore pay attention to social media for advertising their products, and social media has become an influential viral marketing tool [3]. They also want to delve into immense amount of

social media data to examine the popularity, reputation, and people's opinions of their products and brands [4]. Twitter has become one of the fastest growing social media sites, and is serving as an electronic word-of-mouth (eWOM) that affects customer's buying decisions by sharing opinions and information about brands [5]. Since its inception in 2006, the average number of tweets that people exchanged per day has increased from 300,000 in 2008 to 200 million in 2011 and to 500 million by August 2013 [6]. Due to the enormous amount of tweets, it is impossible for a human to manually analyze them. An automated analysis system is necessary to deal with the huge amount of tweets. In addition, the shortness and informality of tweets, including grammatical errors, misspellings, and unreliable capitalizations increase the difficulty of understanding tweets.

Community Question Answering (CQA) sites are knowledge sharing platforms that allow users to post questions and answer questions asked by other users. CQA sites such as Yahoo! Answers<sup>1</sup>, Stack Overflow<sup>2</sup>, and Quora<sup>3</sup> have evolved as huge knowledge sharing platforms by their community users [7]. CQA services are derived from Question Answering (QA) systems that automatically retrieve succinct answers to factoid questions posed by human in a natural language (e.g. "Who is the US President now?") [8]. However, there are striking differences between QA systems and CQA services, which

---

<sup>1</sup> <https://answers.yahoo.com/>

<sup>2</sup> <https://stackoverflow.com/>

<sup>3</sup> <https://www.quora.com/>

cannot be dealt with the techniques used in QA systems. Blooma & Kurian [8] stated the five differences between QA systems and CQA services: The first difference is a type of a question. CQA questions tend to be longer, more specific and more complex than QA questions because users usually ask questions with additional information. For example, “Is Mac or PC better? I'm looking to get a new computer, which do you think is better for my purposes? Schoolwork, and a bit of gaming as well”. Conventional QA systems are not applicable to multiple-sentence questions [9]. The second difference is a source of answers. While QA answers are facts from reliable sources such as news articles or encyclopedia articles, CQA answers may be personal experiences, advices, or recommendations by expert users. A diversity of CQA users can result in the inconsistent quality of the answers. The third difference is the quality of answers. For evaluating quality of answers, QA systems determine only if answers are right or wrong. In CQA sites, however, any users are allowed to answer a question and each question can have many different answers depending on who is answering. Since every answer differs in the quality, there has been some research on investigating the factors that have an impact on the quality of answers [7, 10, 11, 12]. The fourth difference is the availability of metadata. CQA services have rich metadata such as comments, voting scores from users, best answers selected by askers, user's profile and interests, etc. The final difference is a time lag between questions and answers. While QA systems immediately respond to factoid questions, CQA services need to wait for users to answer the questions. Various approaches such as question retrieval, question routing, and expert recommendation have

been suggested to reduce the time lag between questions and answers [13, 14, 15, 16].

These differences imply that an approach to CQA should be different from that of QA.

CQA services heavily rely on participation of users and expect users to obey community rules so that they can develop and maintain the usefulness of the community [17]. There are three main roles of users in CQA services: askers, answerers, and evaluators. Askers post questions and those questions are answered by answerers. Askers can express their satisfactions of the answers by choosing them as the best answers. There are no best answers if the askers are not satisfied with any of the answers or if they forget to choose. In this case, the answers which get the most votes by evaluators are selected as the best answers. Yahoo! Answers and Stack Overflow are typical Community Question Answering (CQA) sites. They have common characteristics such as a reward system and best answers chosen by askers. A reward system encourages users to ask and answer questions by rewarding them with reputations. On the other hand, there are individual characteristics in Yahoo! Answers and Stack Overflow. Yahoo! Answers is an open-domain CQA site. There is no restriction on the question topic, question quality, and answer quality. Unlike Yahoo! Answers, Stack Overflow is limited to topics in computer programming and has rules and regulations users should follow. Duplicate and off-topic questions are restricted and will be closed. Users can upvote and downvote questions and answers depending on their quality. These make and keep Stack Overflow useful and of high quality.

## 1.2 Problem Statement

Firstly, as social media have become more popular, the demand for online reputation management (ORM) has increased. ORM involves monitoring and analyzing user opinions on social media to evaluate a reputation of a company and its products. The first step of ORM is to collect what people mention about the organization on social media. However, lexical ambiguity is a pervasive problem inhibiting researchers' abilities to retrieve desired data [18]. Many words can be interpreted in multiple ways. Unlike humans, machines need a process to understand the underlying meaning of the words. Word Sense Disambiguation (WSD) is a way for machines to understand the meaning of words, and different possible solutions have been suggested over decades [19]. Most of the solutions rely on knowledge-based approaches such as thesauri, ontologies, or sense-annotated corpora. Unfortunately, it is expensive and time-consuming to create the knowledge base system manually, and this problem is called the knowledge acquisition bottleneck [20]. A topic signature is a family of words related to a given topic, and can be used to solve the knowledge acquisition bottleneck though it is used for summarizing a document in the early stages. Manually-annotated knowledge base systems such as WordNet have insufficient lexical and semantic information with regard to the knowledge acquisition. Topic signatures from large-scale resources can perform better than manually-annotated knowledge base systems in WSD [21]. Newspapers are used in the field of WSD as external knowledge resources. Since collections of newspapers are unstructured resources, they need to be annotated with senses [22], or used as raw



corpora. Recently, The New York Times, an American daily newspaper, has provided public access to their newspaper repository using application programming interfaces (APIs). Newspapers can be retrieved by sending a query term, and the search engine retrieves news articles relevant to the query term. Consequently, only relevant newspapers can be retrieved from a large-scale repository. Together with other information retrieval techniques, we can collect the corpus related to a target sense without manual annotations. Moreover, new words that reflect certain topics will arise over time. The new words can be discovered as topic signatures by accessing up-to-date newspapers in the repository using The New York Times APIs. It is crucial to apply WSD prior to the analysis of user opinions in social media so that the reputations of companies or products can be analyzed accurately and reliably. Furthermore, their reputations could be properly managed.

Secondly, one of the characteristics of CQA is a time lag between questions and answers. Askers need to wait for answers to be posted after posting questions, and even some of the questions are never answered. In Stack Overflow, the question response time is different for each question and is affected by various factors such as tags and subscribers [23]. For the time lag between questions and answers, various solutions have been suggested such as question retrieval, and question routing. Question retrieval aims at finding similar questions from large CQA archives [13]. Question routing is a technique that identifies potential answerers and experts, and routes relevant questions to them [24].

In the early stage of CQA, each of the questions and answers is unique due to the fact that questions are specific and require informative and detailed answers. Currently, hundreds of millions of questions have been posted in Yahoo! Answers, and Stack Overflow hits 10 million questions. As questions are piling up, a new question may be no longer unique and other users have probably asked the same questions. It is expected that the time lag problem can be mitigated by finding the most similar question and its best answer if there is an accumulative large-scale question and best answer collection. Most of the research takes advantage of only question titles or combines titles and descriptions as questions.

Lastly, software information sites such as Stack Overflow, Super User, and Ask Ubuntu are knowledge sharing platforms that allow users to post software-related questions, answer the questions asked by other users, and add tags to their questions. Software information sites are subordinate to CQA in terms of categorization. Tags are referred to as freely determined keywords by non-expert users, which help to organize, search, and explore their posts for future use [25]. Tagging is a popular system across web communities because allowing users to classify their contents is less costly than employing an expert to categorize them [25, 26]. In software information sites (e.g. Stack Overflow), when posting a question, tagging is mandatory from a minimum of one tag to a maximum of five tags per question. Users may choose existing tags or create a new tag if a particular tag does not exist in the list of tags. Although uncontrolled vocabularies (i.e. tags) have potential advantages with regard to cost and specificity [27], there are

some known issues. As shown in Figure 1-1, a small subset of popular tags is used to annotate a majority of items in a collection and the rest of the tags are not frequently used, which is called a tag explosion [28, 29, 30]. Another well-known issue in software information sites is a tag synonym. Since a choice of tags depends on user preferences and some users are not educated to create new tags in a proper way, the words used for tags can be arbitrary [31]. Even for the tags that represent the same meaning, the words can be expressed differently depending on who is writing. For example, ‘javascript’ can be expressed as an acronym ‘js’, ‘algorithm’ can be written as a plural form ‘algorithms’, and ‘httprequest’ can be written with a hyphen ‘http-request’ depending on the user preferences. As the number of different tags grows, the list of tags becomes filled with synonyms and this makes it difficult for users to search for existing tags. It also negatively affects the speed and accuracy of queries [31]. Stack Overflow, one of the software information sites, is aware of the problem of tag synonym and is currently trying to resolve it by manually finding and merging the tag synonyms. Figure 1-2 is a tag synonym project in Stack Overflow. Only high reputation users can suggest the tag synonyms. The suggested tag synonyms is merged when taking a certain number of upvotes by other users. In this dissertation, in order to overcome three problems mentioned above, we propose three methods: company name discrimination using topic signatures in social media, answer recommendation based on question retrieval in CQA, and tag recommendation in CQA.

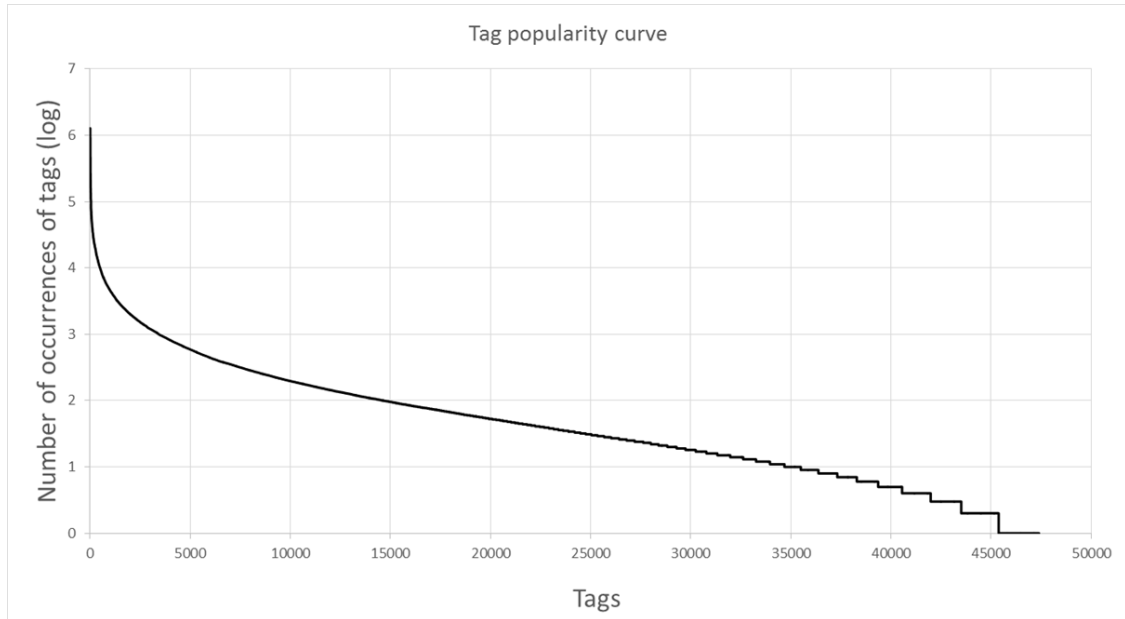




Figure 1-1. Tag popularity curve in Stack Overflow


[Questions](#)
[Developer Jobs](#)
[Documentation](#) BETA
[Tags](#)
[Users](#)

Tag Synonyms

all

Filter:

newest master

Master	←	Synonym	Creator	Renames	Last
<a href="#">ajax</a> × 175365		<a href="#">.ajax</a>	<a href="#">falstro</a> Aug 6 '10 at 12:13	124	Mar 20 at 21:32
<a href="#">avi</a> × 432		<a href="#">.avi</a>	<a href="#">falstro</a> Aug 6 '10 at 12:14	15	Nov 23 at 14:42
<a href="#">bash</a> × 88533		<a href="#">.bash-profile</a> × 538	<a href="#">Grisha Levit</a> Feb 14 at 8:03	7	13h ago
<a href="#">bash</a> × 88533		<a href="#">.bashrc</a>	<a href="#">Jefromi</a> Feb 23 '11 at 4:51	62	Aug 6 at 21:11
<a href="#">batch-file</a> × 37998		<a href="#">.bat</a>	<a href="#">McDowell</a> Mar 26 '11 at 9:06	482	38m ago
<a href="#">c#</a> × 1130131		<a href="#">.cs-file</a>	<a href="#">un-lucky</a> Mar 7 '16 at 7:58	1	Apr 11 '16 at 19:05
<a href="#">razor</a> × 24709		<a href="#">.cshtml</a>	<a href="#">Marc Gravell</a> ♦ Aug 23 '11 at 20:17	19	May 11 at 20:35
<a href="#">csv</a> × 47638		<a href="#">.csv</a>	<a href="#">Michael Myers</a> ♦ Aug 6 '10 at 15:10	214	Aug 14 at 17:35

Figure 1-2. Tag synonym project in Stack Overflow

## Chapter 2

### Literature Review

#### 2.1 Text Mining Methods

In this section, we introduce three text mining methods: term frequency-inverse document frequency (TF-IDF), Okapi BM25, and latent Dirichlet allocation (LDA). First of all, the term frequency-inverse document frequency (TF-IDF) is a term weighting method to discover the importance of a word in a text document from corpus using statistical methods [32]. TF-IDF is a combination of term frequency (TF) and inverse document frequency (IDF). Term frequency (TF) is the number of times that term  $t$  occurs in document  $d$ , and can be normalized by dividing the total frequency of terms in the document, which is defined as

$$tf(t, d)_{\text{normalized}} = \frac{tf(t, d)}{\sum_{t' \in d} tf(t', d)}$$

Inverse document frequency (IDF) is used to examine whether the term is common or rare across all documents, and is defined as

$$idf(t) = \log \frac{N}{1 + df(t)}$$

where  $N$  is the total number of documents, and  $df(t)$  is the number of documents with term  $t$  in it.

The term frequency-inverse document frequency (TF-IDF) is a multiplication of term frequency (TF) by inverse document frequency (IDF) so that a term that are frequent in a document earns a high weight but is offset by the IDF if the term is also frequently used in other documents. A TF-IDF weight of a term  $t$  in a document  $d$  is calculated as

$$\text{TF-IDF}(t,d) = \text{tf}(t,d) \cdot \text{idf}(t)$$

The Okapi BM25 (BM25) is a ranking model to calculate the similarity between a document and a query [33]. With the bag-of-words representation of a query  $Q = \{q_1, q_2, \dots, q_n\}$  and a document  $D = \{w_1, w_2, \dots, w_n\}$ , the Okapi BM25 is defined as

$$\text{BM25}(Q, D) = \sum_{i=1}^n \log \left( \frac{N - df(q_i) + 0.5}{df(q_i) + 0.5} \right) \cdot \frac{tf(q_i) \cdot (k_1 + 1)}{tf(q_i) + k_1 \cdot \left( 1 - b + b \cdot \frac{|D|}{avgdl} \right)}$$

where  $N$  is the total number of documents in the corpus,  $df(q_i)$  is the number of documents containing a query term  $q_i$  in the corpus,  $tf(q_i)$  is the term frequency of  $q_i$  in the document  $D$ ,  $|D|$  is the total number of words in the document  $D$ , and  $avgdl$  is the average document length in the corpus.  $k_1$  and  $b$  are free parameters.

The concept of BM25 is similar to TF-IDF but BM25 is less sensitive to term frequency because free parameters make it reach a saturation point. This has fascinated search engine systems such as Lucene, Solr, and Elasticsearch.

Latent Dirichlet allocation (LDA) is a generative probabilistic model which finds topic probabilities over documents given that documents are represented as random mixtures over latent topics where each topic is characterized by a distribution over words [34]. Given a corpus  $D$  containing  $M$  documents where each document contains  $N$  words  $\{w_1, w_2, \dots, w_n\}$  over  $K$  topics, the generative process of LDA is as follows:

1. Choose  $\theta_i \sim \text{Dirichlet}(\alpha)$
2. Choose  $\varphi_k \sim \text{Dirichlet}(\beta)$
3. For each of the  $N$  words  $w_n$ :
  - a. Choose a topic  $z_{i,j} \sim \text{Multinomial}(\theta_i)$
  - b. Choose a word  $w_{i,j} \sim \text{Multinomial}(\varphi_{z_{i,j}})$

where  $i \in \{1, \dots, M\}, j \in \{1, \dots, N\}, k \in \{1, \dots, K\}$ , and  $\alpha$  and  $\beta$  are hyperparameters for the topic distribution per document  $\theta$  and the word distribution per topic  $\varphi$  respectively.

With the plate notation as shown in Figure 2-1, LDA can be described as follows:  $\alpha$  is the parameter of the Dirichlet prior on the document-topic distributions,  $\beta$  is the parameter of the Dirichlet prior on the topic-word distribution,  $\theta_m$  is the topic distribution for the document  $m$ ,  $\varphi_k$  is the word distribution for the topic  $k$ ,  $z_{mn}$  is the topic for the  $n$ -th word in the document  $m$ , and  $w_{mn}$  is the specific word that is the only observable variable.



In order to learn LDA with various distributions, many approximation methods have been suggested such as variational inference, Gibbs sampling, and online inference [34, 35, 36].

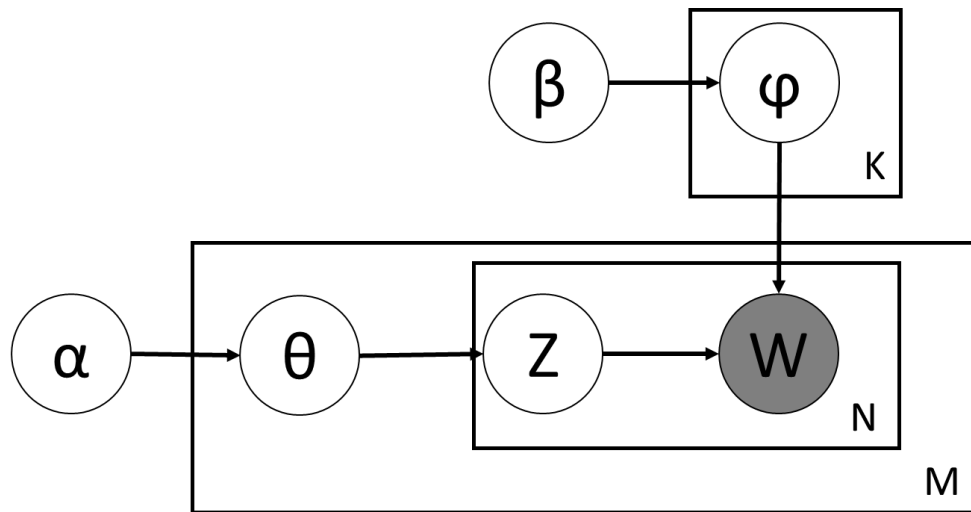


Figure 2-1. Latent Dirichlet Allocation (LDA) Plate Notation

## 2.2 Word Sense Disambiguation in Social Media

Word Sense Disambiguation (WSD) is the activity of identifying the senses of a word and has been one of the research areas in Natural Language Processing (NLP) for several decades. For example, the word ‘apple’ has multiple meanings such as a fruit and a company, and the meaning of the word depends on context. WSD is generally divided into two approaches: supervised approaches and unsupervised approaches. Supervised approaches use various machine learning methods with manually annotated resources for

identifying word senses. Various supervised methods have been adopted such as Naive Bayes, Neural Network, an instance-based learning, Support Vector Machine, and Ensemble methods [37, 38]. Building manually annotated resources is an expensive and time-consuming work as documents and contents on the Web grow continuously. In an effort to resolve the knowledge acquisition bottleneck, a bootstrapping method and a topic signature have been adopted to the word sense disambiguation [39, 40]. In SemEval-2007, an international word sense disambiguation competition, the best system achieved an 88.70% accuracy whereas the first sense baseline achieved 78% [41]. A gold standard data constructed by the manually tagged Wall Street Journal corpus was used for the evaluation. The accuracy of the disambiguation is relatively high because a newspaper corpus is a long document and contains enough clue words to be used for the disambiguation.

Recently, many studies have tried to discriminate word senses in Twitter. Due to the fact that tweets are usually short and informal, it is much more difficult for machines to understand the word senses of tweets. The third Web People Search (WePS-3) task-2 evaluation campaign was held to address the ambiguity of named entities in Twitter and to encourage research groups to resolve the problem by providing the information of companies and collections of tweets for each company. Several groups participated in the competition [42]. The best system was LSIR-EPFL which built six profiles of each company from external sources such as the home page, the metadata of the website, the

category profile using WordNet, GoogleSet, and the manually user-defined terms for both positive and negative aspects [43]. The second best system ITC-UT made use of six rules to categorize a company bias on tweets into 3 or 4 classes. For each bias, a procedure of the tweet classification was differently specified [44]. The middle-ranked system SINAI directly recognized the named entity using Wikipedia, DBpedia, and the company's home page [45]. The UVA system tried to build a general organization classifier by examining the characteristics of Twitter despite the low accuracy [46]. The KALMAR system used an initial bootstrapping model from the company's home page [47]. After the third WePS campaign, many studies have been done to disambiguate a named entity on tweets. Most of the researches tried to resolve the named entity ambiguity using the external sources such as Wikipedia, Google search, DBpedia, the company's home page, etc. [48, 49, 50, 51, 52]. Another approach was to build the named entity recognizer for the Twitter stream [53]. However, none of the research has exploited the news corpus for the tweet discrimination.

### **2.3 Question Retrieval in CQA**

Question retrieval is one of the challenging tasks in community question answering (CQA). Question retrieval aims at finding similar questions from large CQA archives. There have been many studies on question retrieval with various approaches. A word-based translation model bridges the lexical gap between question titles [13] and the translation model outperforms the traditional retrieval methods such as the cosine

similarity and the language model. As an extension of the word-based translation model, a phrase-based translation model is proposed [54] and outperforms the word-based translation model. A knowledge-based approach is also able to reduce the lexical gap for question retrieval [55]. On the other hand, a syntactic tree matching approach shows that syntactic information is applicable to address the question retrieval problem [56]. Topic models are applied to discover latent topics and similar questions are retrieved based on the topic distribution [57, 58] and topic models also improve the retrieval performance. However, most of the research takes advantage of only question titles. Recently, some studies have utilized question descriptions for the query expansion by adding descriptions [59].

## **2.4 Tag Recommendation in CQA**

Tag recommendation is another challenging task in community question answering (CQA). Tagging is a popular means of annotation for online users to freely attach additional information to their contents. With the advent of Web 2.0, many online social services such as Flickr, Delicious, and CiteULike have introduced tagging systems, and they have attracted great attention of researchers to tag recommendation [60, 61, 62]. In online social services, the goal of the tag recommendation is to recommend additional tags to user-defined tags using various approaches; Based on tag co-occurrence between two tags, candidate tags are selected, aggregated, and promoted to recommend tags for images [60], Latent Dirichlet Allocation (LDA) is applied for a resource-tag matrix

instead of a document-term matrix to find most relevant latent topics and to recommend the tags in the relevant topics [61], or documents are treated as triplets of (words, documents, tags) to estimate the document distribution using a two-way Poisson Mixture Model [62].

Software information sites such as Stack Overflow, Super User, and Ask Ubuntu are community question answering (CQA) sites and have characteristics of online social services with regard to tagging. While most CQA sites use controlled vocabularies to categorize questions, software information sites make use of uncontrolled vocabularies (i.e. folksonomies), called tags. Many studies have been focusing on improving the quality of tags and solving the problem of the tag synonym and the tag explosion in software information sites by recommending appropriate tags. Wang et al. proposed a tag recommendation method called EnTagRec [63]. EnTagRec consists of two components called Bayesian inference component (BIC) and Frequentist inference component (FIC). The BIC uses Labeled LDA (LLDA) to learn a probability distribution of tags and the FIC employs the co-occurrence of a tag and a term to compute term-tag probabilities. Tags are recommended based on the score of both the BIC and the FIC. Wang et al. proposed TagCombine, a tag recommendation method that combines three approaches of tag recommendation: a multi-label classification ranking approach, a similarity-based ranking approach, and a tag-term affinity ranking approach [64]. Tags are recommended based on the score of these three ranking functions. Wu et al. proposed a content-based

tag recommendation model called Tag2Word [65]. Tag2Word calculates the tag-word distribution using LLDA and makes use of the tag appearance in the content to improve the accuracy. Zhou et al. proposed TagMulRec that is scalable for large-scale information sites [31]. TagMulRec finds candidate software objects by calculating the similarity between software objects using word co-occurrence and normalizing the similarity score, and ranks tags by the similarity scores of the candidate software objects and their tag frequencies. To speed up the computation, TagMulRec uses indexing techniques. Joorabchi et al. proposed a novel approach in order to resolve the problem of the tag explosion by mapping Stack Overflow tags into Wikipedia concepts using machine learning techniques [28]. Many researchers have applied LDA to the tag recommendation and showed the improvement in accuracy [61, 63, 65, 66, 67]. However, there is limited research on considering the relevance of topics in calculating the document similarity.

## **Chapter 3**

### **Company Name Discrimination in Social Media**

Company name discrimination is considered a binary classification task that checks whether the classification result would be related or non-related to a target company. In this chapter, we propose a company name discrimination method in social media using topic signatures from news articles. Figure 3-1 depicts the overall process of the proposed approach for the company name discrimination in tweets. Firstly, the New York Times (NYT) articles related to a target company are collected from the NYT repository using APIs. Various features are available in the articles such as abstracts, headlines, lead paragraphs, snippets and article bodies. Before extracting topic signatures, news articles are converted into the bag-of-words representation. In the bag-of-words model, a document is represented as a bag of words and the word order is ignored. Texts are segmented for the bag-of-words representation and tagged with their parts-of-speech. Only nouns are extracted for the company name discrimination because most nouns are concrete terms and they are used for the subject and object of the sentence. After NLP processing, topic signatures are extracted from the collected news articles by the document frequency. A straightforward classification method is used to classify the tweets based on the topic signatures. The classification method determines whether a tweet is relevant or irrelevant to a given company.

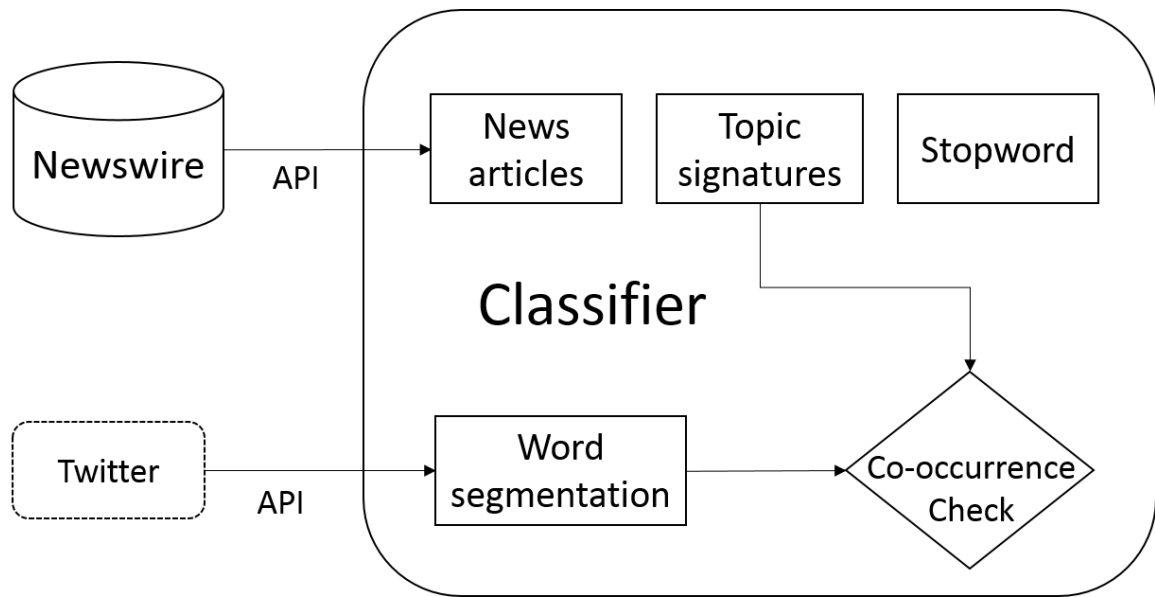


Figure 3-1. A system architecture of the company name classifier

### 3.1 Methodology

In this section we explain how to collect news articles from the NYT repository, the process of the topic signature extraction, the classification method in detail.

#### 3.1.1 Collecting News Articles

The New York Times provides a public access to their news articles using The New York Times APIs<sup>4</sup>. Thus, searching a company's name can automatically retrieve related articles from The New York Times repository. News articles can be collected by a search

<sup>4</sup> <http://developer.nytimes.com/>



keyword using the NYT APIs up to 1,010 articles per search keyword. The search keyword should be carefully decided to retrieve relevant articles. Searching for a full name of the company mitigates the ambiguity in the collected articles by retrieving more related articles so that the articles can be used as an external knowledge resource without manual annotations. Although we rely on the NYT APIs to collect relevant articles, the verification of the collected articles is necessary.

Table 3-1 shows the result of the relatedness of collecting news article using different search keywords. With the search keyword ‘apple’, we collected 1,002 news articles. Out of 1,002 articles, 621 articles were related to the company *Apple Inc.*, 184 articles were not related, and 197 articles did not have contents enough to recognize the relatedness. With the search keyword ‘apple inc’, we collected 1,010 news articles. Out of 1,010 articles, 925 articles were related to the company, no articles were unrelated, and 85 articles were not determined. After examining the collected articles, we observed that the full name of the company (e.g. Apple Inc.) retrieves related articles much more than the part of the company name (e.g. apple), and the collected articles can be used as an external knowledge resource.

Table 3-1. Relatedness of collecting news article using different search keywords

Search Keyword	Related	Unrelated	Undetermined	Total
apple	621	184	197	1002
apple inc	925	0	85	1010

### 3.1.2 Topic Signature Extraction

A topic signature is a family of terms that are highly correlated with a target concept and is defined as follows:

$$\begin{aligned} \text{Topic Signature} &= \{\text{topic, signature}\} \\ &= \{\text{topic}, < (t_1, w_1), \dots, (t_n, w_n) >\} \end{aligned}$$

where *topic* is the target concept and *signature* is a vector of related terms [68]. Each  $t_i$  is a term highly correlated to *topic* with a weight  $w_i$ . A topic signature is a statistical approach that exploits the natural tendency of the semantically related words which co-occur more often than by chance in the same context [69]. Topic signatures are typically extracted from a pre-classified corpus because the relatedness of topic signatures is generally measured by tf-idf, the chi-squared test, or mutual information.

Topic signatures are extracted from news articles. Various features are available in each collected article, and the four features associated with contents are considered to be

suitable for extracting topic signatures: bodies, abstracts, lead paragraphs, and snippets.

The body feature is a full text of the news article and the other features are the short summaries of the news article. However, detailed explanations of the features are not specified in the NYT API document. Although various features are available in the news articles, it is necessary to examine which feature is more useful for the company name discrimination. Some of data are empty and null data can be retrieved from the NYT repository. Retrieval rate is used to determine how much data in the feature is available in the collected articles, and it is defined as:

$$\text{Retrieval rate} = \frac{\text{Non-empty data in the feature}}{\text{Total collected articles}}$$

A higher retrieval rate implies that the feature has less missing data and more available data in the collected articles. Therefore, the feature with the higher retrieval rate is more likely to be useful for extracting topic signatures.

Topic signatures are extracted from the high retrieval rate features based on the document frequency. The document frequency of a term is defined as the number of articles that contain the term in the collected articles, and the term's specificity is related with the document frequency [70]. Since a news article usually covers a specific topic, the document frequency is used as a main criterion to discover the topic signatures. The document frequency is also useful to mitigate a bias towards longer documents. Since the topic signature is the key to discriminate word senses, it is important to extract

meaningful topic signatures from news articles. Typically, topic signatures are extracted by comparing the occurrence in related articles and unrelated articles to a target word sense. While related articles can be collected by the search engine, it is nearly impossible to construct unrelated articles without human annotations. For this reason topic signatures are extracted from only related articles.

Vocabulary is a list of unique words in the collected articles, and topic signatures are the terms, extracted from the vocabulary, whose document frequency is greater than the threshold which is heuristically determined. It is important to find the reasonable threshold to exclude insignificant terms from the topic signature. If the threshold is too low, superfluous terms would be included. On the contrary, the immoderately high threshold leads to a false bias and most of the results are labelled as ‘non-related’ since too few words would be included in the topic signature.

### **3.1.3 Classification**

The classification method exploits the occurrence of topic signatures in tweets because tweets are short messages within 140 characters and each word of a tweet has a strong meaning. Algorithm 1 explains the algorithm of the tweet classification. Topic signatures are extracted from news articles, and tweets and topic signatures are prepared in the form of the bag-of-words model. In the classification, the tweet is classified to a related tweet to a target company if any topic signatures occur in the tweet. If no topic

signatures occur in the tweet, the tweet is classified to a non-related tweet to a target company.

---

**Algorithm 1** Tweet Classification
 

---

```

tweet  $\leftarrow (t_1, t_2, t_3, \dots, t_n)$ 
topic signature  $\leftarrow (w_1, w_2, w_3, \dots, w_n)$ 

1: procedure RELATEDNESS (tweet, topic signature)
2:   if (tweet  $\cap$  topic signature)  $\neq \emptyset$  then
3:     set tweet related
4:   else
5:     set tweet non-related
6:   end if
7: end procedure

```

---

### 3.2 Experimental Setup and Results

#### 3.2.1 Data Sets and Preprocessing

In WePS-3 task 2 Online Reputation Management, 47 named entities and around 500 tweets corresponding to each named entity are provided as a test set. The test set is labelled by 5 human annotators using Amazon Mechanical Turk. We categorized the 47 named entities into subcategories, and decided to use 27 company-related entities for the experiment since companies are occasionally main topics of newspapers and company names are frequently mentioned in newspapers. The experiment was carried out with the 27 companies listed in Table 3-2. The total number of tweets provided by WePS-3 for 27 companies is 11,526 tweets. In this experiment the URL addresses and username tags (@) were ignored and the hash tags (#) were regarded as normal words.

The total number of news articles we collected for 27 companies is 20,492 articles. We collected the body, the abstract, the lead paragraph, and the snippet from each article. All the raw texts were converted to a bag-of-words representation using word segmentation and the part-of-speech tagger in the Stanford Natural Language Processing (NLP) module<sup>5</sup>. Only nouns are extracted for the company name discrimination because most nouns are concrete and used for the subject and object of the sentence. In the preliminary experiment with a narrow data set, the result showed that the noun case was most accurate in tweet classification compared to other cases such as noun & verb, noun & verb & adjective, and all parts of speeches.

Stop word removal was applied to the all the raw texts. Stop words are the words that should not be used as topic signatures. There are two types of stop words. The first type of the stop words is the tweet search keywords for collecting a test set. All tweets retrieved by the search keywords include the same keywords in the contents themselves. If the tweet search keywords are not excluded, the result will be under a bias that most of the tweets are related to a company. The second type of the stop words is the news words. A news article is commonly written in a fixed format. The words that are not related to a company may frequently appear in the news articles such as the date published, the location, the appellation, the name of a newspaper, etc. The news words should be

---

<sup>5</sup> <http://nlp.stanford.edu/index.shtml>

excluded from the classifier due to the fact that these words decrease the quality of the topic signature.

Table 3-2. A list of 27 companies used in the evaluation

27 companies (full names used in news search)
Amazon.com, Apache Software Foundation, Apple Inc., Blizzard Entertainment, Canon Inc., Cisco Systems, CVS/pharmacy, Ford Motor Company, T.G.I. Friday's, General Motors, Gibson Guitar, Jaguar Cars Ltd., Lexus, McDonald's, Metro Supermarket, Oracle Corporation, Orange S.A., Paramount Group, Seat S.A., Sharp Corporation, Sonic.net, Sony, Starbucks, Subway, Tesla Motors, US Airways, Virgin Media

### 3.2.2 Evaluation Metrics

For the performance evaluation, we estimated the accuracy, precision, recall and f-measure for each company. The measures are defined as:

$$\text{Accuracy} = \frac{TP + TN}{N}$$

$$\text{Precision (related)} = \frac{TP}{TP + FP}$$

$$\text{Recall (related)} = \frac{TP}{TP + FN}$$

$$\text{Precision (non-related)} = \frac{TN}{TN + FN}$$

$$\text{Recall (non-related)} = \frac{TN}{TN + FP}$$

$$\text{F-measure} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

where N is the number of tweets, TP is true positive, TN is true negative, FP is false positive, and FN is false negative.

### 3.2.3 Feature Selection

Figure 3-2 illustrates the retrieval rate of the four different features of news articles. The snippet feature obtained the 0.98 retrieval rate on average, which is the highest retrieval rate in the four features. A feature with a higher retrieval rate is more likely to be useful for extracting topic signatures. For example, when we collect 1,000 news articles, 980 news articles contain the snippets whereas 20 news articles do not have the snippets. The average retrieval rate of the lead paragraph feature is 0.91, and that of the body feature is 0.90. The abstract feature obtained a 0.54 retrieval rate, which is relatively low. Despite the high retrieval rate, the lead paragraph feature is excluded from the experiment. The contents of the lead paragraph feature are almost identical to those of the snippet feature but the retrieval rate of the lead paragraph is lower than that of the snippet feature. According to the average retrieval rate, we selected the snippet feature and the body feature as candidate features for extracting topic signatures. The main difference



between the snippet feature and the body feature is their respective lengths. The snippet consists of at most two sentences whereas the body contains several paragraphs.

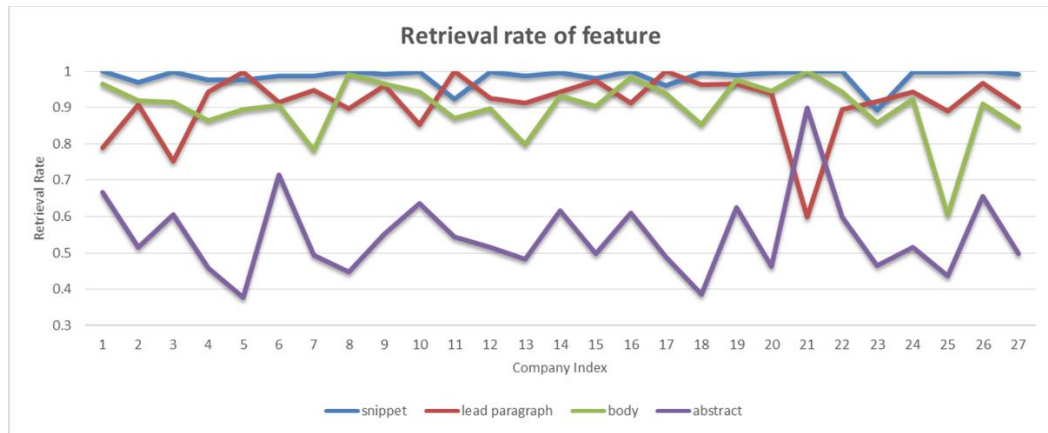


Figure 3-2. The retrieval rate of news article features

### 3.2.4 Threshold for Extracting Topic Signatures

Topic signatures are extracted based on a threshold of the document frequency. It is important to find the reasonable threshold to exclude insignificant words from the topic signature. If the threshold is too low, superfluous words would be included. On the other hand, if the threshold is too high, it leads to a false bias and most of the results are labelled as non-related since too few words would be included in the topic signature. Because the number of the collected news articles is different for each company, we set the threshold by the ratio of collected documents. For example, if the threshold is set to

15%, the words whose document frequency is greater than  $0.15N$  are chosen as a topic signature where  $N$  is the total number of the news articles for the given company.

The threshold for the body feature is determined at 15% of the total number of the news articles in the previous research [71]. As the threshold increases, the f-measure has fallen off steadily. The 2% threshold achieved the highest f-measure. We compared the number of extracted words in topic signatures to figure out what threshold value can be a reasonable threshold. As shown in Table 3-4, on average, the 2% threshold extracted 75.81 words as topic signatures; the 3% threshold extracted 44.11 words, and 12% threshold extracted 6.07 words. As a result, the 2% threshold has the most topic signatures without decreasing the performance in both accuracy and f-measure. Therefore, the threshold for the snippet feature is determined as 2% of the total number of news articles.

Table 3-3 is a topic signature for the company 'Lexus' when applying 15% as a threshold, and 44 terms were extracted as a topic signature. In the same manner, we compared various thresholds for the snippet feature to determine the best threshold. As shown in Figure 3-3 and Figure 3-4, various thresholds from 1% to 20% were tested. In Figure 3-3, the accuracy is fluctuating and unpredictable as the threshold changes. The 12% threshold achieved the highest accuracy. Figure 3-4 shows the sum of the f-measure of the related class and the non-related class. As the threshold increases, the f-measure has fallen off steadily. The 2% threshold achieved the highest f-measure. We compared

the number of extracted words in topic signatures to figure out what threshold value can be a reasonable threshold. As shown in Table 3-4, on average, the 2% threshold extracted 75.81 words as topic signatures; the 3% threshold extracted 44.11 words, and 12% threshold extracted 6.07 words. As a result, the 2% threshold has the most topic signatures without decreasing the performance in both accuracy and f-measure. Therefore, the threshold for the snippet feature is determined as 2% of the total number of news articles.

Table 3-3. A topic signature for 'Lexus'

<b>Topic signature (44 terms)</b>
model, luxury, sedan, sale, auto, automaker, motor, brand, ford, sport, driver, bmw, engine, honda, consumer, detroit, highway, version, hybrid, safety, mercedes-benz, industry, buyer, motors, road, mile, general, customer, division, truck, cadillac, acura, design, test, dealer, report, traffic, nissan, product, feature, utility

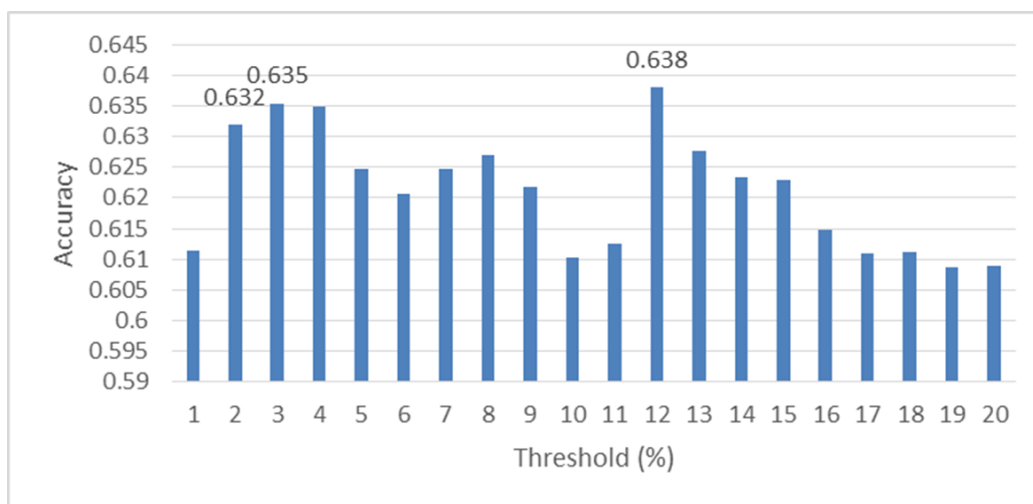


Figure 3-3. The accuracy of the classification result on various thresholds

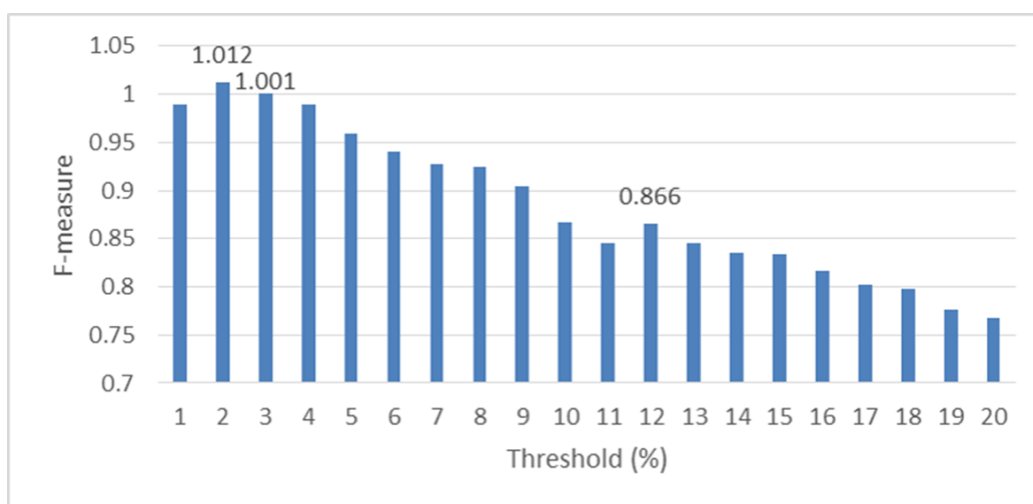


Figure 3-4. The f-measure of the classification result on various thresholds

Table 3-4. The average number of words in topic signatures for each threshold

Threshold (%)	Average topic signatures (words)
2	75.81
3	44.11
12	6.07

### 3.2.5 Results

The performance of the classification is measured by accuracy, precision, recall, and the f-measure. A random baseline is a result of randomly labeled tweets, and is used to evaluate the improvement of the company name classification in tweets. Figure 3-5 shows an evaluation of the result compared to the baseline. The result shows that the topic signatures extracted from the article body increased the accuracy by 10.4% and those of the snippet feature increased the accuracy by 12.5% as compared with the random baseline. The precisions, recalls, and f-measures for both the snippet feature and the body feature are also increased as compared with the random baseline. The company name discrimination result was more accurate when using the snippet feature compared to the body of the article. Whereas f-measure of the body feature in the related class is higher than that of the snippet feature, the f-measure of the body feature in the non-related class is lower than that of the snippet feature. We investigated the reason for the difference of the f-measures between the related class and non-related class, and the

reason is that the body feature has more topic signatures (140.44 words on average) than the snippet feature (75.81 words on average).

Table 3-5 and Table 3-6 illustrate the company name discrimination results of the top five companies out of the twenty-seven companies in order of the sum of the related f-measure and the non-related f-measure for the body feature and the snippet feature respectively. Companies with high scores on f-measures indicate that they are well discriminated in our approach for both related and non-related tweets. We observed that Apple Inc., General Motors, Tesla Motors, and Jaguar Cars Ltd. are highly ranked in terms of accuracy and f-measure for both using the body feature and the snippet feature because product-related words are extracted well from news articles as topic signatures. For example, topic signatures of automobile manufacturers include car-related words such as vehicle, car, and model. Topic signatures of Apple Inc. include names of their products such as iPhone, iPod, etc.

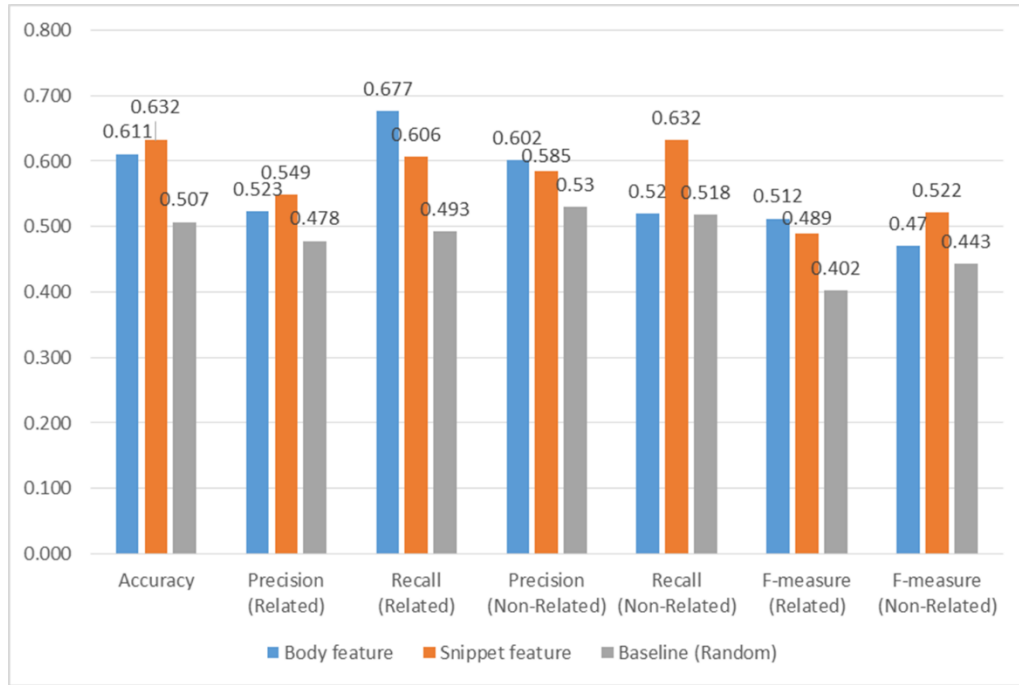


Figure 3-5. A comparison of the evaluation result

Table 3-5. The top 5 companies using the body feature by f-measure

Entity	Accuracy	Related			Non-related		
		Precision	Recall	F-measure	Precision	Recall	F-measure
Apple Inc.	0.835	0.942	0.856	0.897	0.495	0.729	0.590
General Motors	0.742	0.659	0.872	0.751	0.861	0.638	0.733
Apache Software	0.710	0.677	0.751	0.712	0.748	0.674	0.709
Tesla Motors	0.693	0.497	0.858	0.630	0.909	0.621	0.738
Jaguar Cars Ltd.	0.663	0.683	0.728	0.705	0.635	0.583	0.608

Table 3-6. The top 5 companies using the snippet feature by f-measure

Entity	Accuracy	Related			Non-related		
		Precision	Recall	F-measure	Precision	Recall	F-measure
Apple Inc.	0.835	0.942	0.856	0.897	0.495	0.729	0.590
General Motors	0.742	0.659	0.872	0.751	0.861	0.638	0.733
Apache Software	0.710	0.677	0.751	0.712	0.748	0.674	0.709
Tesla Motors	0.693	0.497	0.858	0.630	0.909	0.621	0.738
Jaguar Cars Ltd.	0.663	0.683	0.728	0.705	0.635	0.583	0.608

Despite of the improvement of the results, there exist some limitations in this experiment. First, Natural Language Processing (NLP) is useful but not perfectly accurate. For natural language processing, we exploited the Stanford Natural Language Processing (NLP) module. The Stanford NLP performs well on the news article. However, the informality of tweets such as grammatical errors, misspellings, and unreliable capitalizations may depreciate the quality of the NLP module in tweets. For example, a tweet ‘This Is Apples Next iPhone(<http://bit.ly/cEJuUq>)’ is segmented into ‘be’, ‘apple’, ‘next’, ‘bit.ly’, and ‘cejuuq’ through the Stanford NLP because of the lack of the blank space in between the words. The word ‘iPhone’, a topic signature of Apple Inc., is not extracted from the tweet due to both the limitation of the NLP module and the informality of the tweet. Second, there is a time gap between the collected news articles and the tweets used in the experiment. When we collected news articles, the latest articles



were retrieved by the New York Times APIs. For example, out of 1,010 news articles about Apple Inc., 299 news articles were generated after January 2015. As privacy became a controversial issue for Apple Inc. in recent years, the word ‘privacy’ is selected as a topic signature as shown in Table 3-7. On the other hand, the evaluation data was released in 2010. This time gap may affect the evaluation result.

Table 3-7. A topic signature for ‘Apple Inc’.

<b>Topic signature (36 terms)</b>
watch, tech, computer, company, executive, tablet, mac, government, steve, operating, smartphone, ios, music, version, system, share, application, app, case, phone, market, iPad, user, customer, steven, jobs, iPod, device, privacy, technology, software, problem, iPhone, feature, cook, security

## **Chapter 4**

### **Recommendations in CQA**

Community Question Answering (CQA) mainly consists of questions and answers. Questions in CQA archives are divided into question titles, question descriptions, and tags. Question titles are usually short but contain the keywords describing the askers' interests while question descriptions are rather long and include detailed information. Tags are used to help to organize, search, and explore the questions for future use. In this chapter, we propose two recommendation methods in CQA: a question retrieval method for the answer recommendation and a tag recommendation method.

#### **4.1 Question Retrieval in CQA**

In CQA, answers are categorized into best answers and non-best answers. Best answers are the selected answers by askers to express their satisfaction of the answers. If we find similar questions of the new question, their best answers can also be the answers for the new question and satisfy the asker's curiosity. We present a weighted question retrieval model that finds similar questions and recommends their best answers in large-scale CQA archives. Figure 4-1 describes an overview of the weighted question retrieval model. When a new question comes, the proposed model calculates the question similarity between existing questions and the new question, and finds similar questions

based on the question similarity scores. Subsequently, their best answers are recommended as an answer of the new question.

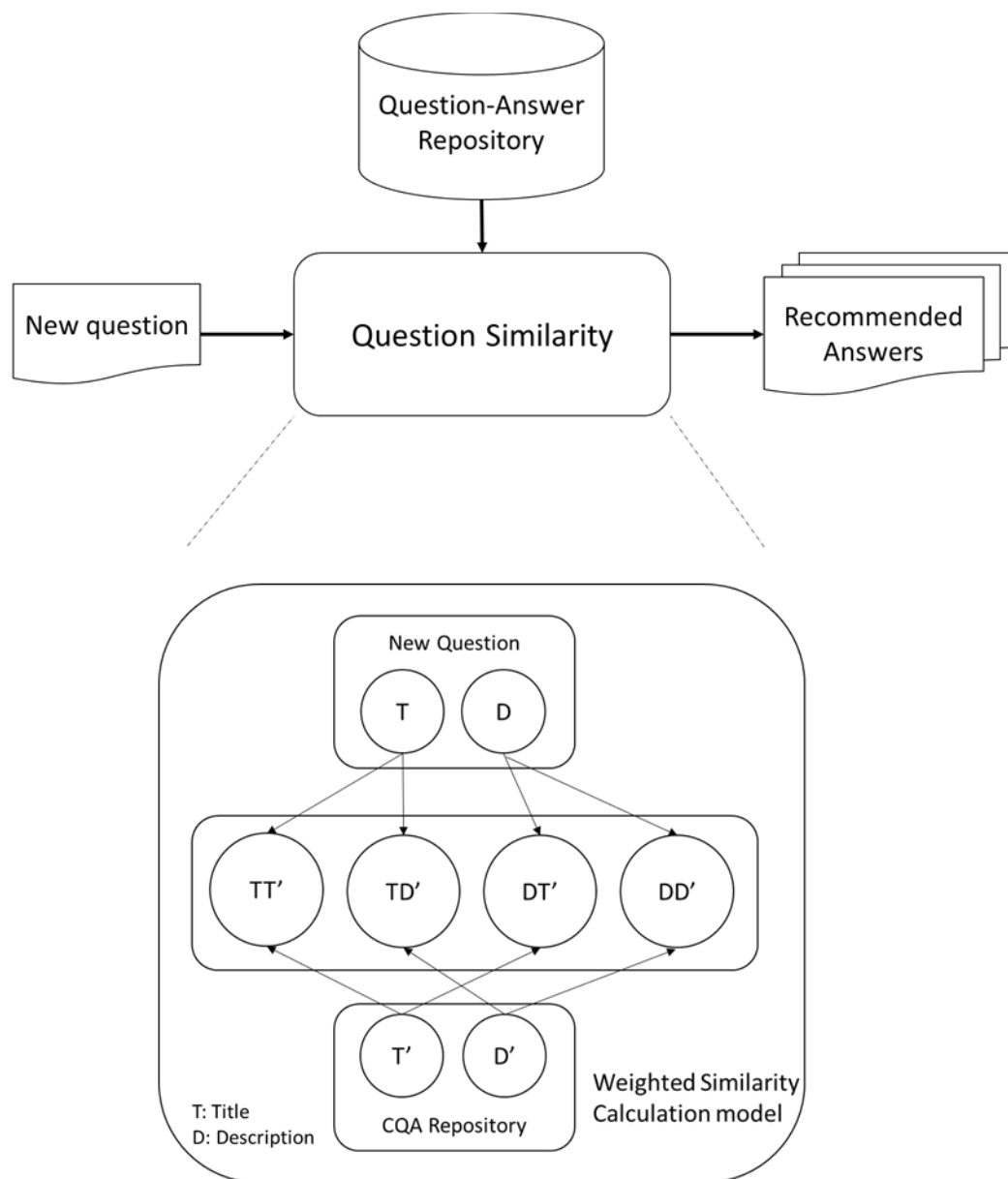


Figure 4-1. An architecture of a weighted question retrieval model

#### 4.1.1 Okapi BM25

The Okapi BM25 is a ranking model to calculate the similarity between a document and a query [34]. We used the Okapi BM25 to measure the question similarity. For the Okapi BM25, we consider a new question as a query and an existing question in the CQA archives as a document. With the bag-of-words representation of a query  $Q = \{q_1, q_2, \dots, q_n\}$  and a document  $D = \{w_1, w_2, \dots, w_n\}$ , the Okapi BM25 score is defined as follows:

$$\text{BM25}(Q, D) = \sum_{i=1}^n \log \left( \frac{N - df(q_i) + 0.5}{df(q_i) + 0.5} \right) \cdot \frac{tf(q_i) \cdot (k_1 + 1)}{tf(q_i) + k_1 \cdot \left( 1 - b + b \cdot \frac{|D|}{avgdl} \right)}$$

where  $N$  is the total number of documents in the CQA collection,  $df(q_i)$  is the number of documents containing  $q_i$  in the CQA collection,  $tf(q_i)$  is the term frequency of  $q_i$  in the document  $D$ ,  $|D|$  is the length of the document  $D$  in words, and  $avgdl$  is the average document length in the CQA collection.  $k_1$  and  $b$  are free parameters, and they are chosen as  $k_1 = 1.2$  and  $b = 0.75$ . Documents with higher BM25 scores are more likely to be related questions.

#### 4.1.2 Weighted Question Similarity

Figure 4-2 describes a weighted question similarity model. To exploit both question titles and question descriptions in question similarity, it is necessary to identify the relationship between titles and descriptions. We developed four baseline scores to

understand the relationship between them: Title-Title', Title-Description', Description-Title', and Description-Description'. In this section, a new question is referred to as *a query*, and a question in the CQA collection is referred to as *a document*.

- **Title-Title' (TT')**: A similarity score between a query's title and a document's title using the Okapi BM25.
- **Title-Description' (TD')**: A similarity score between a query's title and a document's description using the Okapi BM25.
- **Description-Title' (DT')**: A similarity score between a query's description and a document's title using the Okapi BM25.
- **Description-Description' (DD')**: A similarity score between a query's description and a document's description using the Okapi BM25.

With these four scores, the question similarity score is calculated as follows:

$$QuestionSimilarity = \alpha \frac{score(TT')}{max(score(TT'))} + \beta \frac{score(TD')}{max(score(TD'))} \\ + \gamma \frac{score(DT')}{max(score(DT'))} + \delta \frac{score(DD')}{max(score(DD'))}$$

where  $\alpha, \beta, \gamma$ , and  $\delta$  are free parameters for weighting scores and they are experimentally determined. All the scores are normalized ranging from 0 to 1 by dividing their maximum scores.

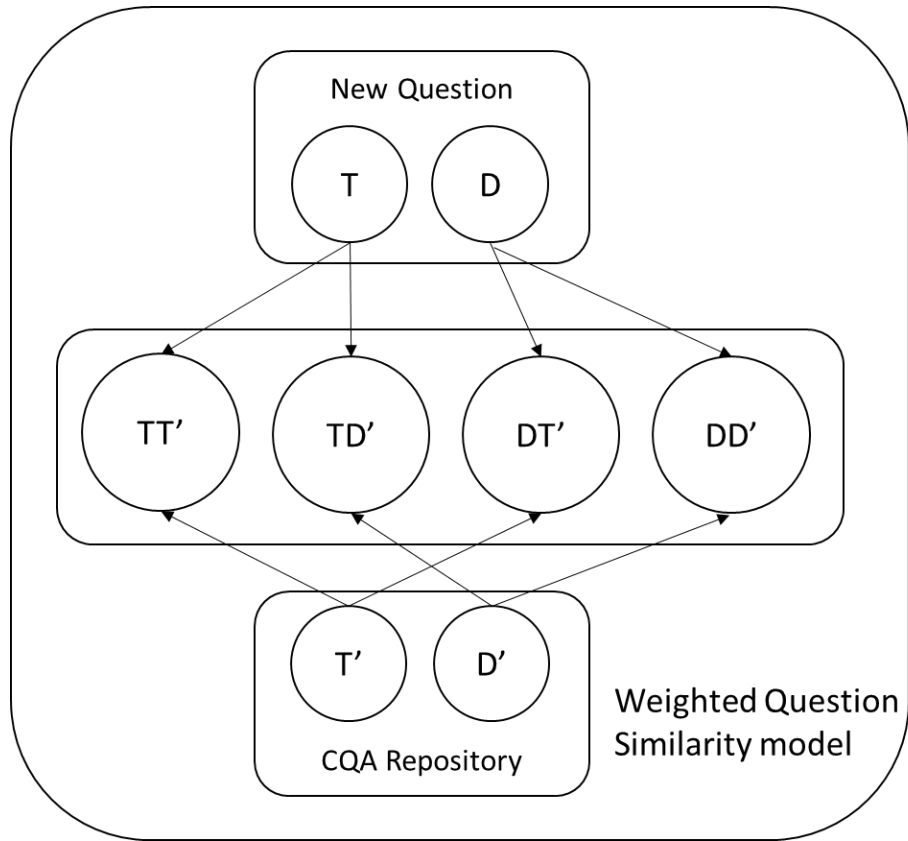


Figure 4-2. A weighted question similarity model

## 4.2 Experimental Setup and Results

In this section, we describe experimental setup and results including the details of the data sets, data preprocessing, and evaluation results for the question retrieval.

#### 4.2.1 Data Sets and Preprocessing

In the experiment we used data sets from Yahoo! Answers and Stack Overflow. The Yahoo! Answers data set<sup>6</sup> includes 4 million questions posted from July 2005 to Dec 2006, and the Stack Overflow data set<sup>7</sup> includes 11 million questions posted from April 2010 to January 2016. Table 4-1 illustrates the details of the data sets used in the experiment. There is a difference between the number of question titles and question descriptions in the Yahoo! Answers collection since some of the question descriptions are empty.

Table 4-1. The details of data sets

	<b>Yahoo! Answers</b>	<b>Stack Overflow</b>
Period	July 2005 – Dec 2006	April 2010 – January 2016
Number of Titles	4,483,032	11,203,031
Number of Descriptions	2,559,603	11,203,031
Number of Answers	4,408,264	18,218,676

All raw texts are lemmatized and part-of-speech tagged by the Stanford NLP module. Only nouns, verbs, and adjectives are used in calculating the question similarity since it is found in the initial experiment that these parts of speeches are meaningful

<sup>6</sup> <http://webscope.sandbox.yahoo.com/>

<sup>7</sup> <https://archive.org/details/stackexchange>

while the others disturb the similarity accuracy. Stop word removal is applied to the all the raw texts.

For the evaluation, twenty questions are sampled from Yahoo! Answers collection and Stack Overflow collection respectively as queries and they are excluded from the CQA collections. We separately conducted an experiment for the Yahoo! Answers collection and Stack Overflow collection. In the evaluation three measures are used: Mean Average Precision (MAP), Precision at 10, and Mean Reciprocal Rank (MRR). MAP is an average precision across queries, which determines a precision at each point when a new relevant document gets retrieved. Precision at 10 computes how many relevant documents are retrieved in top 10 results. MRR considers an average rank of the first relevant document across queries. Human annotators cannot look at more than million documents in the collection. For this reason, top 20 results are annotated and the other results are automatically assumed to be irrelevant, which is called a pooling method.

#### **4.2.2 Results**

In this experiment, we used two different data sets: Yahoo! Answers and Stack Overflow. Each data set has different characteristics with regard to contents. For example, the Yahoo! Answers data include empty descriptions because descriptions are not a requirement of questions. On the contrary, the Stack Overflow data must include descriptions by its community rules. For this reason, free parameters for the weighted



score were determined independently. Free parameters for Yahoo! Answers were experimentally determined as  $\alpha=1$ ,  $\beta=0$ ,  $\gamma=0.8$ , and  $\delta=0$ . Table 4-2 shows the result of the Yahoo! Answers collection with 20 sample queries. The Title-Title' was the best score, and the Description-Title' was the second highest score among the four baselines. Based on this observation, the parameters were adjusted. As a result, the weighted score, a combined weighted score of the best two relationships of titles and descriptions, outperformed the Title-Title' and other baselines in both MAP and MRR while precision at 10 was decreased.

Table 4-2. The question retrieval result in Yahoo! Answers

<b>Yahoo! Answers</b>	<b>MAP</b>	<b>Precision@10</b>	<b>MRR</b>
Title-Title'	0.505	<i>0.320*</i>	0.600
Title-Description'	0.229	0.155	0.283
Description-Title'	0.397	0.180	0.483
Description-Description'	0.176	0.120	0.210
Proposed Model	<i>0.523*</i>	0.285	<i>0.653*</i>

*\*best result*

For the Stack Overflow data set, we also used a combined weighted score of the best two relationships of titles and descriptions. Free parameters for Stack Overflow were also experimentally determined as  $\alpha=1$ ,  $\beta=0.8$ ,  $\gamma=0$ , and  $\delta=0$ . Table 4-3 shows the result of the Stack Overflow collection with 20 sample queries. The Title-Title' was also the best

score, and the Title-Description' was the second highest score in the four baseline scores. The weighted score slightly outperformed the Title-Title' score in both MAP and MRR while precision at 10 were decreased.

Table 4-3. The question retrieval result in Stack Overflow

<b>Stack Overflow</b>	<b>MAP</b>	<b>Precision@10</b>	<b>MRR</b>
Title-Title'	0.499	0.295*	0.572
Title-Description'	0.331	0.125	0.372
Description-Title'	0.103	0.070	0.086
Description-Description'	0.235	0.045	0.271
Proposed Model	0.508*	0.270	0.585*

*\*best result*

From the results, we observed that the Title-Title' was the best among the baselines in both data sets. However, the second highest combination was different for each data set. The Description-Title' was the second highest baseline in Yahoo! Answers data whereas the Title-Description' was the second highest baseline in Stack Overflow data. From this observation, we found that exploiting the question descriptions increased the ranks of the relevant questions while reducing the recalls of them as compared with Title-Title'. For example, in the Stack Overflow collection, the ranks of the 5 queries were increased and those of 3 queries were decreased. 12 fewer relevant questions were retrieved by the weighted model than by Title-Title'. In the Yahoo! Answers collection,

the ranks of the 3 queries were increased and 5 fewer relevant questions were retrieved by the weighted model than by Title-Title'.

Table 4-4 shows an example of the rank rises of the relevant questions by the weighted model in the Yahoo! Answers collection. The question 1 and the question 2 are retrieved by the similarity with the query. The question 1 is ranked seventy-fifth based on the Title-Title' and is ranked thirteenth based on the Description-Title'. The question 2 is ranked eighth based on the Title-Title' and is ranked six hundred seventy-fourth based on the Description-Title'. With the weighted model, their ranks can rise to the first and the second respectively. Without combining the Title-Title' and Description-Title', either the question 1 or the question 2 cannot be retrieved as a similar question.

Table 4-4. Examples of the rank rises of the relevant questions by the weighted model in the Yahoo! Answers collection

Query	<b>Title:</b> In the san francisco bay area, does it make sense to rent or buy? <b>Description:</b> the prices of rent and the price of buying does not make sense to me, mostly the rent will not cover the mortgage. Is it better to rent a house or to buy?	<b>Rank (TT')</b>	<b>Rank (DT')</b>	<b>Rank (Weighted)</b>
<b>Question 1</b>	<b>Title:</b> Why is it better to rent a house than buy? If in Bay Area, California? <b>Description:</b> None <b>Best Answer:</b> One, the prices are insane. Two, your house payments will be higher than rent. Three, you have to spend more on your own repairs Four, property taxes Five, homeowners insurance Much cheaper to rent than own right now. Save your money until the market evens out and see where you are.	75	13	1
<b>Question 2</b>	<b>Title:</b> Should I buy or rent in San Francisco? <b>Description:</b> None <b>Best Answer:</b> BUY BUY BUY!!!!!!!!!!!!!!	8	674	2

### **4.3 Tag Recommendation in CQA**

Tag Recommendation is one of the solutions to the tag synonym and the tag explosion in Community Question Answering (CQA) services that employ tagging systems. The tag recommendation reduces the chance of creating new tags from users and can help to choose appropriate tags to their questions by recommending proper tags. The strategy of the tag recommendation for a target document (i.e. an unseen document) is to find similar documents to the target document and then to recommend the tags used in the similar documents (i.e. candidate documents) based on the ranking score. The following sections describe methodology and experiment results of the tag recommendation.

#### **4.3.1 Topic Models**

A topic model is a statistical model for discovering latent topics in a collection of documents based on the idea that words with similar meaning will occur in similar documents [34, 72, 73]. Latent Dirichlet Allocation (LDA), one of the topic models, is a generative probabilistic model which finds topic probabilities over documents given that documents are represented as random mixtures over latent topics where each topic is characterized by a distribution over words [34]. In this chapter we assume that a document is a bag of words and consists of a title and a description of a question. Question titles are usually short but contain the keywords describing the askers' interests whereas question descriptions are rather long and include detailed information. We

combine the words in the title and the description for the document representation. Given a corpus  $D$  containing  $M$  documents where each document contains  $N$  words  $\{w_1, w_2, \dots, w_n\}$  over  $K$  topics, the generative process of LDA is described as follows:

1. Choose  $\theta_i \sim \text{Dirichlet}(\alpha)$
2. Choose  $\varphi_k \sim \text{Dirichlet}(\beta)$
3. For each of the  $N$  words  $w_n$ :
  - a. Choose a topic  $z_{i,j} \sim \text{Multinomial}(\theta_i)$
  - b. Choose a word  $w_{i,j} \sim \text{Multinomial}(\varphi_{z_{i,j}})$

where  $i \in \{1, \dots, M\}, j \in \{1, \dots, N\}, k \in \{1, \dots, K\}$ , and  $\alpha$  and  $\beta$  are hyperparameters for the topic distribution per document  $\theta$  and the word distribution per topic  $\varphi$  respectively. In LDA, we used the variational inference to learn the hidden topics  $z$  from the corpus. The hidden topics  $z$  consists of document-topic distributions and topic-word distributions. From the hidden topics  $z$ , we can predict the topic distributions for unseen documents. This prediction results in a vector of topic distributions of the documents and the words. Table 4-5 shows an example of document-topic distribution from the hidden topics  $z$  in LDA. Each document is a distribution on topics and the sum of the probabilities of topics in each document is equal to 1.

Table 4-5. An example of document-topic distribution in LDA

	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5	...	
Document 1	0.0015	0.0276	0.0021	0.0034	0.0043	...	$\sum P(x) = 1$
Document 2	0.0042	0.0002	0.0660	0.0192	0.0002	...	$\sum P(x) = 1$
...	...	...	...	...	...	...	

#### 4.3.2 A Ranking Function using Highest Topic Filtering (HTF)

The Latent Dirichlet Allocation (LDA) results in a vector of topic distributions for existing documents. The topic distributions for the target document can be inferred using the trained LDA model. The similarity scores between existing documents and the target document are calculated based on the similarity between their document-topic distributions. We used Jensen–Shannon divergence (JSD) to calculate the similarity between document-topic distributions. JSD is a well-known method for measuring the similarity between two probability distributions, which is a symmetrized and smoothed version of Kullback-Leibler divergence. JSD is defined as

$$\text{JSD}(P \parallel Q) = \frac{1}{2} \sum_i P(i) \log \frac{P(i)}{M(i)} + \frac{1}{2} \sum_i Q(i) \log \frac{Q(i)}{M(i)}$$

$$M(i) = \frac{1}{2} (P(i) + Q(i))$$



where  $P$  and  $Q$  are topic distributions of documents and  $M$  is the average of the two distributions. The measure is 0 only for identical distributions and approaches infinity as the two differ more and more.

However, equally comparing every topic distribution may lose the relevance of the topic. Table 4-6 describes an example of document similarity between the target document and existing documents. Based on the JSD, the document 1 is most similar to the target document. While the document 2 and the document 3 are less similar than the document 1, their first highest topic distributions are the same with the target document. Document 2's second highest topic distribution is also the same with the target document. Even though the highest topic distributions are the same, the scores may indicate less similar. To emphasize the most relevant topics and exploit the highest topic distributions, we proposed a method that filters the highest topic distributions, named Highest Topic Filtering (HTF). The HTF method filters the highest topic distributions in advance of calculating document similarity scores using the JSD. For HTF with the first highest topic distribution, candidate documents are filtered to have the same first highest topic distribution with the target document. For HTF with the second highest topic distribution, candidate documents are filtered to have the same first and second highest topic distribution with the target document. Therefore, it confines the candidate documents by filtering the documents that have the same highest topic distributions with the target document.

Table 4-6. Document similarity between the target document and existing documents

	Topic 1	Topic 2	Topic 3	Topic 4	...	Similarity
Document 1	<b>0.0276*</b>	0.0021	0.0034	0.0043**	...	0.0791
Document 2	0.0002	<b>0.0660*</b>	0.0192**	0.0002	...	0.3026
Document 3	0.0100	<b>0.2747*</b>	0.0001	0.0557**	...	0.4423
Target Document	0.0054	<b>0.0276*</b>	0.0157**	0.0145	...	0 (identical)

(\*First highest topic distribution, \*\*Second highest topic distribution)

After the Highest Topic Filtering, we calculate the document similarity between existing documents and the target document using the JSD. Candidate documents are selected based on the similarity scores to calculate tag scores. In calculating tag scores, we consider both the document similarity and the tag occurrence in the candidate documents. The score of each tag  $t$  is calculated as

$$\text{TagScore}(t) = \sum_{i=1}^n \frac{1}{\text{JSD}_i} \cdot \text{occurrence}(t, i)$$

where  $n$  is the number of candidate documents,  $\text{JSD}_i$  is the similarity between the document  $i$  and the target document, and  $\text{occurrence}(t, i)$  is an indicator function that is equal to 1 if the tag  $t$  occurs in a document  $i$  and 0 otherwise. The value of JSD is taken as a multiplicative inverse to give weight to similar documents. Therefore, the more frequently tags occur in more similar documents, the higher tag scores become. As a result, the higher tag score is more likely to be relevant to the target document. By

ranking the tags in descending order of their tag scores, the top ranked tags are selected for recommendation. We called this ranking function  $TagScore_{HT}$ . Algorithm 2 describes the procedure of  $TagScore_{HT}$ . A function  $argmaxY$  returns the indices of  $Y$  number of highest topic distributions of a document.

---

**Algorithm 2** A ranking function  $TagScore_{HT}$

---

$tag_t$ : a score of a tag  $t$   
 $td_k$ : Distribution of  $k$ -th topic  
 an existing document  $P_i \leftarrow (td_1, td_2, \dots, td_k)$   
 a target document  $Q \leftarrow (td_1, td_2, \dots, td_k)$   
 Initialize the score of each tag  $t$   $tag_t \leftarrow 0$

2: **procedure**  $TagScore_{HT}(P, Q)$   
 3:   **for**  $i = 1, 2, \dots, n$  **do**  
 4:     **if**  $argmaxY(P_i) == argmaxY(Q)$  **then**  
 5:       **for each** tag  $t$  **do**  
 6:           $tag_t += occurrence(t, i) \cdot \frac{1}{JSD_i}$   
 7:       **end for**  
 8:     **end if**  
 9:   **end for**  
 10: **end procedure**

---

#### 4.4 Experimental Setup and Results

In this section, we describe experimental setup and results including the details of the data sets, data preprocessing, LDA parameters, evaluation metrics and results for the tag recommendation.

#### 4.4.1 Data sets and Preprocessing

In this experiment we used mutually exclusive training and test sets. For a training set, we used data sets from Stack Overflow. The Stack Overflow data is publicly released and includes about 13 million questions posted from July 2008 to December 2016. For a test set, we manually collect 100 questions with the most views, answers, and votes in a month in the Stack Overflow website posted from April 2017 to May 2017.

All the data is preprocessed before training the LDA model. The raw data is stored in the eXtensible Markup Language (XML) format, and texts are wrapped with HTML tags such as `<code>`, `<pre>`, `<div>`, etc. First, we parse HTML tags to extract only normal texts, and discard any code snippets that are wrapped with the tag `<code>`. Code snippets are noisy when they are handled as normal texts since most programming languages have similar syntax and keywords. After extracting normal texts, all the texts are lemmatized by the Stanford NLP module. Stop words such as ‘a’, ‘the’, and ‘of’ are also removed from the texts.

The time complexity of LDA is  $O(NKM)$  where  $N$  is the number of unique words,  $K$  is the number of topics, and  $M$  is the number of documents. For the Stack Overflow data used in this experiment, the number of unique words is 6 million and the number of documents is 13 million. The number of topics may vary from tens to hundreds depending on the data. It may take a huge amount of time to train the LDA model and it is intractable to compute with all the Stack Overflow data. Inevitably, we reduced the

dimensions of words and documents. We found that a small subset of tags appeared in most of the documents. As shown in Figure 4-3, a small number of the most frequently used tags can cover most of the documents. Therefore, we selected the most frequently used 300 tags that can cover 11,790,316 (91.25%) documents of the Stack Overflow data, and randomly sampled 1,000 documents for each tag. The number of training documents was reduced to 300,000 and the number of unique words was also reduced to 212,144 in order of frequency to train the LDA model.

We also developed a validation set in order to validate the Highest Topic Filtering (HTF) method for the tag recommendation. The validation set consists of 2,028 questions whose voting scores are greater than 400. The voting score is the sum of upvotes and downvotes, and a question can be upvoted or downvoted by users. We assume that voting scores indicate the quality of both questions and their tags. Therefore, the validation set is assumed to have more relevant tags to their question and is used to find what number of highest topic distributions would achieve the best performance in the tag recommendation. The details of the data sets used in the experiment are described in Table 4-7.

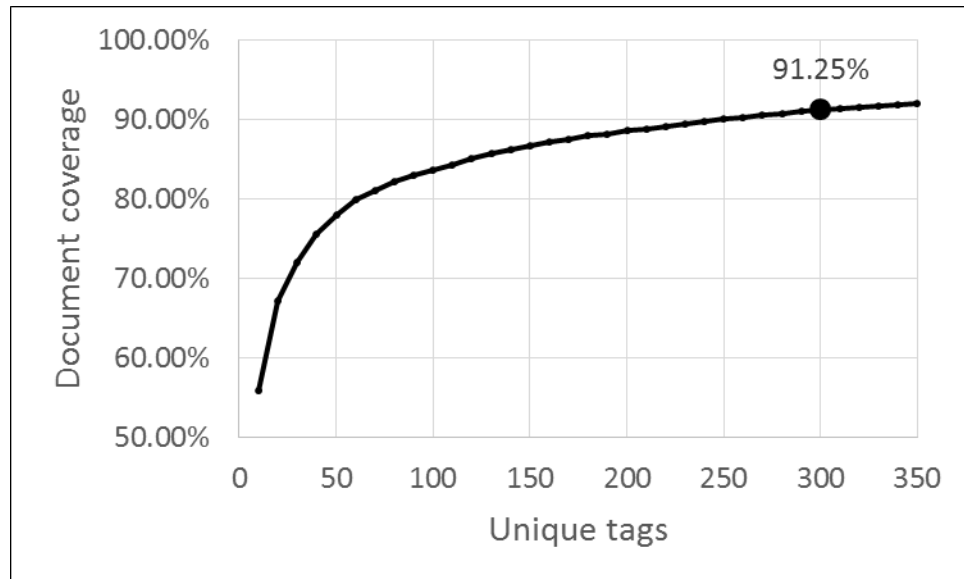


Figure 4-3. Document coverage by the most frequent tags

Table 4-7. The details of the data sets

Dataset	# of Documents	# of Tags	Period
Raw data	12,921,385	47,391	07/31/08 – 12/11/16
Training	300,000	23,659	04/13/09 – 12/11/16
Validation	2,028	1,702	07/31/08 – 11/01/16
Test	100	184	04/13/17 – 05/19/17

#### 4.4.2 Evaluation Metrics

We assumed that all tags appeared in the training set and the test set are relevant and those tags were considered the answer tags to evaluate our tag recommendation method. We evaluate the performance of our tag recommendation method using the validation set and the test set. First of all, we discovered similar documents of a target document in the test set using the LDA model trained with the sample data of the raw data set. Then, we calculated scores of tags based on the document similarity and the frequency of tags appeared in the candidate documents. We finally recommended the top 10 tags in order of the ranking score.

To evaluate our tag recommendation method, we used the following ranking evaluation metrics.

**P@k:** Precision at cut-off  $k$  measures the percentage of answer tags that are matched with the recommended tags in the top  $k$  positions of the predicted rank. We evaluate the average of precision at cut-off  $k$  over the test set for  $k = 5$ .

**R@k:** Recall at cut-off  $k$  measures the percentage of answer tags that are selected out of the recommended tags in top  $k$  positions. We evaluate the average of recall at cut-off  $k$  over the test set for  $k = \{5, 10\}$ .

**MAP:** Mean average precision measures an average precision across queries, and is defined as

$$\text{MAP} = \frac{\sum_{q=1}^Q \text{AveP}(q)}{Q}$$

where  $Q$  is the number of queries and  $\text{AveP}(q)$  denotes the average precision for query  $q$ , computed as

$$\text{AveP}(q) = \frac{\sum_{k=1}^n P@k \cdot \text{rel}(k)}{\text{number of answer tags}}$$

where  $\text{rel}(k)$  is an indicator function that is equal to 1 if the tag at rank  $k$  is an answer tag and 0 otherwise.

**MRR:** Mean reciprocal rank measures the average of the reciprocal rank across queries. It is defined as

$$\text{MRR} = \frac{1}{Q} \sum_{q=1}^Q \frac{1}{\text{rank}_q}$$

where  $\text{rank}_q$  denotes the rank position of the first relevant tag for the query  $q$ .

**nDCG:** Discounted cumulative gain (DCG) measures the ranking quality based on its position by discounting the gain at lower ranks. It is defined as

$$\text{DCG} = \sum_{k=1}^n \frac{2^{\text{rel}_k} - 1}{\log_2(k + 1)}$$

where  $\text{rel}_k$  is the graded relevance of the tag at position  $k$ . The relevance scores of tags are binary in this experiment. The normalized DCG (nDCG) normalizes this score across queries by Ideal DCG (IDCG), which are defined as



$$\text{nDCG} = \frac{\text{DCG}}{\text{IDCG}}$$

$$\text{IDCG} = \sum_{k=1}^{|REL|} \frac{2^{rel_k} - 1}{\log_2(k + 1)}$$

where  $|REL|$  represents the list of answer tags.

The cut-off level  $n$  is set to 10 for MAP, MRR, and nDCG in this experiment.

#### 4.4.3 Training Topic Models

To determine the proper number of topics  $K$ , we examined the log likelihood of the LDA for different  $K$  topics. We eventually chose the number of topics  $K = 160$  since the log likelihood of the LDA was highest at the point of  $K = 160$  as shown in Figure 4-4.

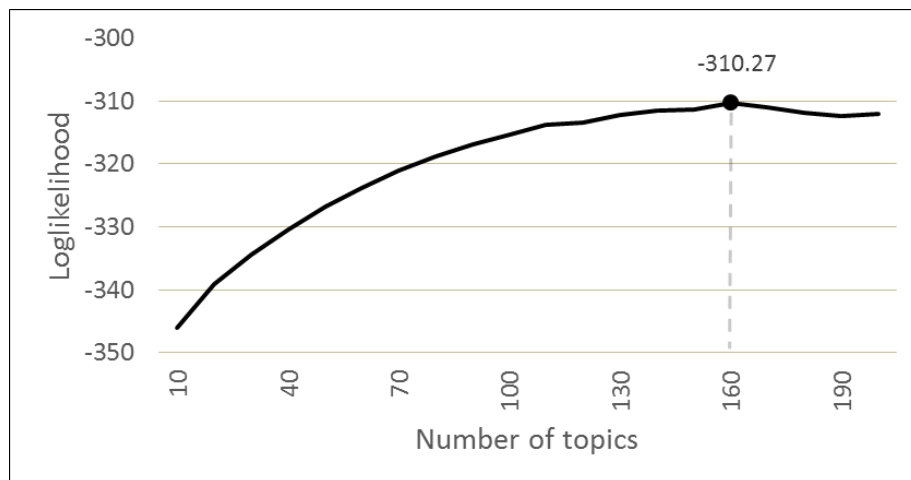


Figure 4-4. A log likelihood of LDA for a different topic  $K$

After determining  $K = 160$ , we run the LDA with 100 iterations. As a result, distributions of the 160 topics for each document were learned. We observed that the LDA discovered latent topics well.

*Table 4-8* lists the topic number, the category we named, and the top 10 words of the topic discovered by the LDA. For example, Topic 0 includes the top 10 words ‘library’, ‘include’, ‘c++’, ‘compile’, ‘static’, ‘standard’, ‘compiler’, ‘boost’, ‘definition’, and ‘callback’. We named this topic ‘C++’. Topic 3 includes the top 10 words ‘run’, ‘apply’, ‘job’, ‘apache’, ‘dataframe’, ‘execution’, ‘spark’, ‘driver’, ‘scala’, and ‘cluster’. We called this topic ‘Apache Spark’. Other topics are also categorized well by looking top 10 words for each topic.

Table 4-8. Top 10 words for different topics discovered by LDA ( $K = 160$ )

Topic 0	Topic 3	Topic 28	Topic 49	Topic 51	Topic 58	Topic 82
C++	Apache Spark	C#	Java	Image	Email	iOS
library	run	api	application	image	send	reference
include	apply	c#	java	picture	address	ios
c++	job	allow	spring	description	email	target
compile	apache	rest	configuration	pixel	contact	force
static	dataframe	net	eclipse	processing	mail	iphone
standard	execution	dll	context	opencv	verify	xcode
compiler	spark	token	container	photo	attachment	layer
boost	driver	register	tomcat	png	outlook	swift
definition	scala	assembly	deploy	bitmap	receiver	ipad
callback	cluster	twitter	maven	blob	gmail	unity

#### 4.4.4 Results

In this section, we present the results of the tag recommendation. We tested four different ranking functions: TagScore (without filtering highest topics), TagScore<sub>HTF=1</sub> (by filtering the first highest topic), TagScore<sub>HTF=2</sub> (by filtering the two highest topics), and TagScore<sub>HTF=3</sub> (by filtering the three highest topics). 100 candidate documents were

selected based on document similarity. From the candidate documents, the tag scores were calculated based on the tag occurrences and the document similarity where the tags occur. We evaluated the proposed method for the most frequently used 300 tags and for all tags.

Table 4-9 shows the tag recommendation result of the top 300 tags for the validation set. In this result, we did not consider the rest of the tags other than the top 300 tags in the evaluation.  $\text{TagScore}_{\text{HT}=1}$  improved the tag recommendation in all evaluation metrics as compared to TagScore (without filtering highest topics).  $\text{TagScore}_{\text{HT}=2}$  also achieved the better performance in all evaluation metrics than TagScore did. On the contrary,  $\text{TagScore}_{\text{HT}=3}$  decreased the performance and achieved the lowest result.

Table 4-10 shows the tag recommendation result of the top 300 tags for the test set.  $\text{TagScore}_{\text{HT}=1}$  outperformed TagScore in all the evaluation metrics. However,  $\text{TagScore}_{\text{HT}=2}$  decreased R@5 and R@10 while increasing P@5, MAP, MRR, and nDCG as compared to TagScore.  $\text{TagScore}_{\text{HT}=2}$  was also slightly higher than  $\text{TagScore}_{\text{HT}=1}$  in MAP, MRR, and nDCG but was lower than  $\text{TagScore}_{\text{HT}=1}$  in R@5 and R@10. For the test set, we observed that  $\text{TagScore}_{\text{HT}=1}$  recommended correct tags more

often than the other ranking functions did, and that  $\text{TagScore}_{\text{HT}=2}$  ranked correct tags more highly than the others did.  $\text{TagScore}_{\text{HT}=3}$  decreased the evaluation result.

Table 4-9. Tag recommendation result for the validation set with regard to top 300 tags

Ranking Function	P@5	R@5	R@10	MAP	MRR	nDCG
TagScore	0.112	0.328	0.415	0.177	0.274	0.268
$\text{TagScore}_{\text{HT}=1}$	0.168	0.490	0.628	0.386	0.495	0.471
$\text{TagScore}_{\text{HT}=2}$	<b>0.182</b>	<b>0.518</b>	<b>0.630</b>	<b>0.429</b>	<b>0.530</b>	<b>0.505</b>
$\text{TagScore}_{\text{HT}=3}$	0.066	0.183	0.208	0.139	0.188	0.176

Table 4-10. Tag recommendation result for the test set with regard to the top 300 tags

Ranking Function	P@5	R@5	R@10	MAP	MRR	nDCG
TagScore	0.181	0.559	0.622	0.283	0.420	0.389
$\text{TagScore}_{\text{HT}=1}$	<b>0.204</b>	<b>0.564</b>	<b>0.668</b>	0.399	0.544	0.508
$\text{TagScore}_{\text{HT}=2}$	<b>0.204</b>	0.536	0.618	<b>0.417</b>	<b>0.549</b>	<b>0.519</b>
$\text{TagScore}_{\text{HT}=3}$	0.083	0.224	0.233	0.186	0.240	0.214

In addition to the result of the top 300 tags, we tested our tag recommendation method for all tags to evaluate how effective it is. Table 4-11 shows the result of the tag

recommendation for all tags. The result for all tags showed the similar improvement in the result for the top 300 tags.  $\text{TagScore}_{\text{HT}=1}$  achieved the best performance in  $\text{R@5}$  and  $\text{R@10}$ .  $\text{TagScore}_{\text{HT}=2}$  achieved the best performance in  $\text{P@5}$ ,  $\text{MAP}$ ,  $\text{MRR}$ , and  $\text{nDCG}$ .  $\text{TagScore}_{\text{HT}=3}$  achieved very low results in both tests because there were less than 100 candidate documents that have the three same highest topic distributions. Evaluation results described that  $\text{TagScore}_{\text{HT}=2}$  achieved the best result in the experiment, and that our method slightly improved the result in the recall-related metrics which are  $\text{R@5}$  and  $\text{R@10}$ , and significantly improved the result in the rank-related metrics which are  $\text{MAP}$ ,  $\text{MRR}$ , and  $\text{nDCG}$ .

Table 4-11. Tag recommendation result for the test set with regard to all tags

Ranking Function	P@5	R@5	R@10	MAP	MRR	nDCG
TagScore	0.176	0.314	0.358	0.170	0.404	0.262
$\text{TagScore}_{\text{HT},Y=1}$	0.194	<b>0.343</b>	<b>0.405</b>	0.237	0.521	0.342
$\text{TagScore}_{\text{HT},Y=2}$	<b>0.198</b>	0.323	0.372	<b>0.247</b>	<b>0.525</b>	<b>0.345</b>
$\text{TagScore}_{\text{HT},Y=3}$	0.076	0.128	0.138	0.101	0.222	0.137

Furthermore, we deeply examined each question of the test set to find whether our recommendation could potentially help users choose relevant tags.

Table 4-12 describes an example of the test set with its tag recommendation result.

This post was asking how to preserve the current local time data in AngularJS. The actual tags attached to the post were ‘javascript’, ‘angularjs’, ‘internationalization’, and ‘timezone’. Using our method, we recommended ‘javascript’, ‘angularjs’, ‘date’, ‘datetime’, ‘angular2’, and so on. From these recommended tags, the asker of this post could add an additional tag ‘date’, ‘datetime’, or ‘angular2’. The asker may not even know if ‘date’ and ‘datetime’ exist in a list of user-defined tags and may also not think to provide a version number of the AngularJS such as ‘angular2’. The tag recommendation has potential advantages in choosing relevant tags without searching for the list of tags and letting users provide more detailed and additional information in their posts. These results imply that our proposed method, a ranking function that filters the highest topic distributions, can improve the tag recommendation result and help users improve the quality of their posts.



Table 4-12. An example of the test set with the recommendation result of  $TagScore_{HT=2}$ 

<b>Title</b>	angular \$http.post changing date to UTC date
<b>Description</b>	<p>I was trying to post some data to my REST api which has date. Now while I debug, my date parameter is a JS Date object with correct date in my timezone: Tue Apr 04 2017 00:00:00 GMT+0530</p> <p>after it leaves my code, and I see the same in network tab, it is converted to UTC date: "2017-04-03T18:30:00.000Z"</p> <p>I searched for the solution according to which I need to include locale file of angular in my index.html which I did: but it doesn't help. I've seen solutions like adding date format to filter or something, but I want a global solution. Any help? Thanks :)</p>
<b>Actual tags</b>	javascript, angularjs, internationalization, timezone
<b>Recommended tags</b>	javascript, angularjs, date, datetime, angular2, jquery, google-chrome, firefox, internet-explorer, node.js

## **Chapter 5**

### **Conclusion**

In this dissertation, we conduct an analysis on textual contents in social media and online communities and have proposed three methods: a company name discrimination method, a question retrieval method, and a tag recommendation method.

Firstly, we propose a semi-supervised system for a company name discrimination on tweets based on topic signatures extracted from news articles. The proposed system is a fully automated system that requires only a search keyword when a new company is added so that no human coder is necessary. From the experiment we found that news articles could be used to disambiguate word senses of tweets as an external source. In addition, we have conducted an experiment for measuring the effectiveness of various features in news articles for the company name discrimination. The snippet, lead paragraph, and body feature obtain high retrieval rates, which means that they can be useful features for extracting topic signatures. In the experiment, only the snippet and body feature were selected as candidate features because almost every lead paragraph has the same contents as the snippet does. The best threshold for extracting the topic signature was determined as 2% for the snippet feature and as 15% for the body feature. The classification result for each feature was 63.2% accuracy for the snippet feature and was 61.1% for the body feature. As compared with the random baseline, the accuracy was increased by 10.4% with the body feature and by 12.5% with the snippet feature.

Although the body feature extracted topic signature words twice as much as those of the snippet feature, the snippet feature achieved 2.1% higher accuracy than that of the body feature. This study observed that topic signatures extracted from news articles improve the accuracy of the company name discrimination in Twitter.

Secondly, we propose a weighted question retrieval model to find similar questions and recommend their best answers in large-scale CQA archives. The proposed model exploits question titles, descriptions, and the relationship between them while most research uses only question titles or combines titles and descriptions as questions. The experiment results showed that our weighted question retrieval model outperformed the baseline that uses only question titles in MAP and MRR. From the experiment result we found that exploiting the question descriptions increased the ranks of the relevant questions while reducing the recalls of them as compared with the baseline using only titles. CQA services have their own characteristics of descriptions. This makes different weights to each CQA service. The weighted question retrieval model fits when ranks are more important than recalls.

Lastly, we propose a tag recommendation method in software information sites using topic models. To recommend relevant tags, we used our ranking function that filters the highest topic distributions based on the document similarity and occurrence of the tags. We evaluated the performance of our tag recommendation method using various ranking evaluation metrics. The experiment results showed that the proposed method slightly

improved the recall-related metrics and considerably improved the rank-related metrics. It has potential advantages in choosing relevant tags without searching for the list of tags letting users provide more detailed and additional information in their posts. Therefore, our proposed method, a ranking function that filters highest topic distributions, can improve the tag recommendation and help users improve the quality of their posts.

## References

- [1] Tim O'Reilly. 2005. What Is Web 2.0? Design Patterns and Business Models for the Next Generation of Software. (September 2005). Retrieved November 5, 2017 from <http://www.oreillynnet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html>
- [2] Graham Vickery, and Sacha Wunsch-Vincent. 2007. *Participative web and user-created content: Web 2.0 wikis and social networking*. Organization for Economic Cooperation and Development (OECD), Paris, France.
- [3] Kelli D. Washington, and Richard K. Miller. 2013. *The 2013 Entertainment, Media & Advertising Market Research Handbook* (13th. ed.). Richard K Miller & Associates, Loganville, GA.
- [4] Wu He, Shenghua Zha, and Ling Li. 2013. Social media competitive analysis and text mining: a case study in the pizza industry. *International Journal of Information Management*, 33, 3 (June 2013), 464-472.
- [5] Bernard J. Jansen, Mimi Zhang, Kate Sobel, and Abdur Chowdury. 2009. Twitter power: tweets as electronic word of mouth. *Journal of the American Society for Information Science and Technology*, 60, 11 (July 2009), 2169-2188.
- [6] Twitter Usage Statistics. 2014. Internet Live Stats Site. Retrieved from <http://www.internetlivestats.com/>

- [7] Yandong Liu, Jiang Bian, and Eugene Agichtein. 2008. Predicting information seeker satisfaction in community question answering. In *Proceedings of the 31st. Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'08)*. 483-490.
- [8] Mohan John Blooma, and Jayan Chirayath Kurian. 2011. Research Issues In Community Based Question Answering. In *Pacific Asia Conference on Information Systems (PACIS2011) Proceedings*. 29.
- [9] Akihiro Tamura, Hiroya Takamura, and Manabu Okumura. 2005. Classification of multiple-sentence questions. In *Proceedings of International Joint Conference on Natural Language Processing (IJCNLP)*. 426-437.
- [10] F. Maxwell Harper, Daphne Raban, Sheizaf Rafaeli, and Joseph A. Konstan. 2008. Predictors of answer quality in online Q&A sites. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'08)*. 865-874.
- [11] Alton Y.K. Chua, and Snehasish Banerjee. 2013. So fast so good: An analysis of answer quality and answer speed in community Question-answering sites. *Journal of the Association for Information Science and Technology*. 64, 10 (July 2013), 2058-2068.
- [12] Jiwoon Jeon, W. Bruce Croft, Joon Ho Lee, and Soyeon Park. 2006. A framework to predict the quality of answers with non-textual features. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'06)*. 228-235.

- [13] Jiwoon Jeon, W. Bruce Croft, and Joon Ho Lee. 2005. Finding similar questions in large question and answer archives. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM'05)*. 84-90.
- [14] Baichuan Li, and Irwin King. 2010. Routing questions to appropriate answerers in community question answering services. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM'10)*. 1585-1588.
- [15] Liu Yang, Minghui Qiu, Swapna Gottipati, Feida Zhu, Jing Jiang, Huiping Sun, and Zhong Chen. 2013. CQArank: jointly model topics and expertise in community question answering. In *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM'13)*. 99-108.
- [16] Xiaoyong Liu, W. Bruce Croft, and Matthew Koll. 2005. Finding experts in community-based question-answering services. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM'05)*. 315-316.
- [17] Imrul Kayes, Nicolas Kourtellis, Daniele Quercia, Adriana Iamnitchi, and Francesco Bonchi. 2015. The social world of content abusers in community question answering. In *Proceedings of the 24th International Conference on World Wide Web (WWW'15)*. 570-580.

- [18] Robert Krovetz, and W. Bruce Croft. 1992. Lexical ambiguity and information retrieval. *ACM Transactions on Information Systems*. 10, 2 (April 1992), 115-141.
- [19] Roberto Navigli. 2009. Word sense disambiguation: a survey. *ACM Computing Surveys*. 41, 2 (February 2009), 1-69.
- [20] William A. Gale, Kenneth W. Church, and David Yarowsky. 1992. A method for disambiguating word senses in a large corpus. *Computers and the Humanities*. 26, 5/6 (December 1992), 415-439.
- [21] Montse Cuadros, and German Rigau. 2006. Quality assessment of large scale knowledge resources. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP'06)*. 534-541.
- [22] Shari Landes, Claudia Leacock, and Randee I. Tengi. 1998. Building semantic concordances. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA. 199-216.
- [23] Vasudev Bhat, Adheesh Gokhale, Ravi Jadhav, Jagat Pudipeddi, and Leman Akoglu. 2014. Min(e)d your tags: Analysis of question response time in StackOverflow. In *Proceedings of 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM'14)*. 328-335.
- [24] Baichuan Li and Irwin King. 2010. Routing questions to appropriate answerers in community question answering services. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM'10)*. 1585-1588.



- [25] James Sinclair, and Michael Cardew-Hall. 2008. The folksonomy tag cloud: when is it useful?. *Journal of Information Science*. 34, 1 (February 2008), 15-29.
- [26] Sanjay C. Sood, Sara H. Owsley, Kristian J. Hammond, and Larry Birnbaum. 2007. TagAssist: Automatic Tag Suggestion for Blog Posts. In *Proceedings of the International Conference on Weblogs and Social Media (ICWSM'07)*.
- [27] William B. Frakes, and Thomas P. Pole. 1994. An empirical study of representation methods for reusable software components. *IEEE Transactions on Software Engineering*. 20, 8 (August 1994), 617-630.
- [28] Arash Joorabchi, Michael English, and Abdulhussain E. Mahdi. 2015. Automatic mapping of user tags to Wikipedia concepts: The case of a Q&A website – StackOverflow. *Journal of Information Science*. 41, 5 (October 2015), 570-583.
- [29] Marieke Guy, and Emma Tonkin. 2006. Folksonomies: Tidying up tags. *D-Lib Magazine*. 12, 1 (January 2006).
- [30] Anton Barua, Stephen W. Thomas, and Ahmed E. Hassan. 2014. What are developers talking about? an analysis of topics and trends in stack overflow. *Empirical Software Engineering*. 19, 3 (June 2014), 619-654.
- [31] Pingyi Zhou, Jin Liu, Zijiang Yang, and Guangyou Zhou. 2017. Scalable tag recommendation for software information sites. In *Proceedings of 2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER'17)*. 272-282.

- [32] Jiaul H. Paik. 2013. A novel TF-IDF weighting scheme for effective ranking. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'13)*. 343-352.
- [33] S. E. Robertson, and S. Walker. 1994. Some Simple Effective Approximations to the 2-Poisson Model for Probabilistic Weighted Retrieval. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'94)*. 232-241.
- [34] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*. 3 (January 2003), 993-1022.
- [35] Ian Porteous, David Newman, Alexander Ihler, Arthur Asuncion, Padhraic Smyth, and Max Welling. 2008. Fast collapsed gibbs sampling for latent dirichlet allocation. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'08)*. 569-577.
- [36] Matthew D. Hoffman, David M. Blei, Francis Bach. 2010. Online learning for latent dirichlet allocation. In *Proceedings of Advances in Neural Information Processing Systems (NIPS'10)*. 856-864.
- [37] Raymond J. Mooney. 1996. Comparative experiments on disambiguating word senses: An illustration of the role of bias in machine learning. In *Proceedings of the 1996 Conference on Empirical Methods in Natural Language Processing (EMNLP'96)*. 82-91.

- [38] Roberto Navigli, and Simone Paolo Ponzetto. 2012. Joining forces pays off: Multilingual joint word sense disambiguation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL'12)*. 1399-1410.
- [39] Rada Mihalcea. 2004. Co-training and self-training for word sense disambiguation. In *Proceedings of the 8th Conference on Natural Language Learning (CoNLL'04)*. 33-40.
- [40] Eneko Agirre, Olatz Ansa, David Martinez, and Eduard Hovy. 2001. Enriching WordNet concepts with topic signatures. In *Proceedings of the NAACL Workshop on WordNet and Other Lexical Resources*. 23-28.
- [41] Eneko Agirre, and Aitor Soroa. 2007. SemEval-2007 task 02: evaluating word sense induction and discrimination systems. In *Proceedings of the 4th International Workshop on Semantic Evaluations*. 7-12.
- [42] Enrique Amigo, Javier Artiles, Julio Gonzalo, Damiano Spina, Bing Liu, and Adolfo Corujo. 2010. WePS-3 evaluation campaign: overview of the online reputation management task. In *Proceedings of International Conference on Cross-Language Evaluation Forum (CLEF2010)*.
- [43] Surender Reddy Yerva, Zoltan Miklos, and Karl Aberer. 2010. It was easy, when apples and blackberries were only fruits. In *Proceedings of International Conference on Cross-Language Evaluation Forum (CLEF2010)*.

- [44] Minoru Yoshida, Shin Matsushima, Shingo Ono, Issei Sato, and Hiroshi Nakagawa. 2010. ITC-UT: tweet categorization by query categorization for on-line reputation management. In *Proceedings of International Conference on Cross-Language Evaluation Forum (CLEF2010)*.
- [45] M. A. García-Cumbreras, M. García-Vega, F. Martínez-Santiago, and J. M. Peréa-Ortega. 2010. SINAI at weps-3: Online reputation management. In *Proceedings of International Conference on Cross-Language Evaluation Forum (CLEF2010)*.
- [46] Manos Tsagkias, Krisztian Balog. 2010. The University of Amsterdam at WePS3. In *Proceedings of International Conference on Cross-Language Evaluation Forum (CLEF2010)*.
- [47] Paul Kalmar. 2010. Bootstrapping websites for classification of organization names on twitter. In *Proceedings of International Conference on Cross-Language Evaluation Forum (CLEF2010)*.
- [48] Sandhya Sachidanandan, Prathyush Sambaturu, and Kamalakar Karlapalem. 2013. NERTUW: Named Entity Recognition on Tweets using Wikipedia. In *Proceedings of Concept Extraction Challenge at the 3rd Workshop on Making Sense of Microposts (#MSM2013)*. 67-70.
- [49] Yegin Genc, Winter Mason, and Jeffrey V. Nickerson. 2013. Classifying Short Messages using Collaborative Knowledge Bases: Reading Wikipedia to Understand Twitter. In *Proceedings of Concept Extraction Challenge at the 3rd Workshop on Making Sense of Microposts (#MSM2013)*. 50-53.

- [50] Mena B. Habib and Maurice van Keulen. 2013. A generic open world named entity disambiguation approach for tweets. In *Proceedings of the 5th International Conference on Knowledge Discovery and Information Retrieval (KDIR'13)*. 267–276.
- [51] Agustin D. Delgado Munoz, Raquel Martinez Unanue, Alberto Perez Garcia-Plaza, and Victor Fresno. 2012. Unsupervised real-time company name disambiguation in twitter. In *ICWSM Workshop on Real-Time Analysis and Mining of Social Streams (RAMSS)*. 25-28.
- [52] Kamel Nebhi. 2012. Ontology-based information extraction from twitter. In *Proceedings of the Workshop on Information Extraction and Entity Analytics on Social Media Data – COLING 2012*. 17–22.
- [53] Chenliang Li, Jianshu Weng, Qi He, Yuxia Yao, Anwitaman Datta, Aixin Sun, and Bu-Sung Lee. 2012. Twiner: named entity recognition in targeted twitter stream. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'12)*. 721-730.
- [54] Guangyou Zhou, Li Cai, Jun Zhao, and Kang Liu. 2011. Phrase-based translation model for question retrieval in community question answer archives. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (HLT'11)*. 653-662.

- [55] Guangyou Zhou, Yang Liu, Fang Liu, Daojian Zeng, and Jun Zhao. 2013. Improving question retrieval in community question answering using world knowledge. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI'13)*. 2239-2245.
- [56] Kai Wang, Zhaoyan Ming, and Tat-Seng Chua. 2009. A syntactic tree matching approach to finding similar questions in community-based QA services. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'09)*. 187-194.
- [57] Li Cai, Guangyou Zhou, Kang Liu, and Jun Zhao. 2011. Learning the latent topics for question retrieval in community QA. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP'11)*. 273-281.
- [58] Zongcheng Ji, Fei Xu, Bin Wang, and Ben He. 2012. Question-answer topic model for question retrieval in community question answering. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM'12)*. 2471-2474.
- [59] Shuguang Li and Suresh Manandhar. 2011. Improving question recommendation by exploiting information need. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (HLT'11)*. 1425-1434.

- [60] Börkur Sigurbjörnsson, and Roelof Van Zwol. 2008. Flickr tag recommendation based on collective knowledge. In *Proceedings of the 17th International Conference on World Wide Web (WWW'08)*. 327-336.
- [61] Ralf Krestel, Peter Fankhauser, and Wolfgang Nejdl. 2009. Latent dirichlet allocation for tag recommendation. In *Proceedings of the third ACM Conference on Recommender Systems (RecSys '09)*. 61-68.
- [62] Yang Song, Ziming Zhuang, Huajing Li, Qiankun Zhao, Jia Li, Wang-Chien Lee, and C. Lee Giles. 2008. Real-time automatic tag recommendation. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'08)*. 515-522.
- [63] Shaowei Wang, David Lo, Bogdan Vasilescu, and Alexander Serebrenik. 2014. EnTagRec: An enhanced tag recommendation system for software information sites. In *Proceedings of 2014 IEEE International Conference on Software Maintenance and Evolution (ICSME'14)*. 291-300.
- [64] Xin-Yu Wang, Xin Xia, and David Lo. 2015. TagCombine: Recommending tags to contents in software information sites. *Journal of Computer Science and Technology*. 30, 5 (September 2015), 1017-1035.
- [65] Yong Wu, Yuan Yao, Feng Xu, Hanghang Tong, and Jian Lu. 2016. Tag2Word: Using tags to generate words for content based tag recommendation. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (CIKM'16)*. 2287-2292.

- [66] Ralf Krestel, and Peter Fankhauser. 2010. Language models and topic models for personalizing tag recommendation. In *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT'10)*. 82-89.
- [67] Suppawong Tuarob, Line C. Pouchard, and C. Lee Giles. 2013. Automatic tag recommendation for metadata annotation using probabilistic topic modeling. In *Proceedings of the 13th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL '13)*. 239-248.
- [68] Chin-Yew Lin, and Eduard Hovy. 2000. The automated acquisition of topic signatures for text summarization. In *Proceedings of the 18th Conference on Computational Linguistics (COLING'00)*. 495-501.
- [69] Maria Biryukov, Roxana Angheluta, and Marie-Francine Moens. 2005. Multidocument question answering text summarization using topic signatures. *Journal of Digital Information Management*. 3, 1 (March 2005), 27-33.
- [70] Hideo Joho, and Mark Sanderson. 2007. Document frequency and term specificity. In *Proceedings of the Recherche d'Information Assistée par Ordinateur Conference (RIAO'07)*. 350-359
- [71] Beomseok Hong, Youngsub Han, and Yanggon Kim. 2015. A semi-supervised tweet classification method using news articles. In *Proceedings of the 2015 Conference on Research in Adaptive and Convergent Systems (RACS'15)*. 62-67.



- [72] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing By Latent Semantic Analysis. *Journal of the American Society for Information Science*. 41, 6 (September 1990), 391-407.
- [73] Thomas Hofmann. 1999. Probabilistic latent semantic indexing. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '99)*. 50-57.

## Curriculum Vitae

**Name:** Beomseok Hong



**Program of Study:** Information Technology

**Degree and Date to be Conferred:** Doctoral of Science (D. Sc.), 2017

### Collegiate Institutions Attended:

Towson University	Towson, Maryland
Doctor of Science	Aug. 2013 – Dec. 2017
Towson University	Towson, Maryland
Master of Science	Feb. 2011 – May. 2013
National Institute for Lifelong Education	South Korea
Bachelor of Engineering	Mar. 2010 – Feb. 2011
Myongji College	South Korea
Associate Degree of Engineering	Feb. 2007 – Feb. 2010

### Professional Publications

- Yun, Je-kuk, **Beomseok Hong**, Yanggon Kim. "The BGP Monitoring and Alarming System to Detect and Prevent Anomaly IP Prefix Advertisement", Research in Applied Computation Symposium (RACS 2013), Montreal, QC, Canada, 1-4 October 2013.

- Yun, Je-kuk, **Beomseok Hong**, Yanggon Kim. "The Implementation of BGP Monitoring, Alarming, and Protecting System by a BGP-UPDATE-Based Method using ECOMMUNITY in Real Time", THE 2014 INTERNATIONAL CONFERENCE ON SECURITY & MANAGEMENT (SAM 2014), Las Vegas Nevada, USA, 21-24, July 2014.
- Yun, Je-kuk, **Beomseok Hong**, Yanggon Kim. "The Policy-Based AS\_PATH Verification to Monitor AS Path Hijacking", The Eighth International Conference on Emerging Security Information, Systems and Technologies (SECURWARE 2014), Lisbon, Portugal, 16-20, November, 2014. (Best Paper Award)
- Yun, Je-kuk, **Beomseok Hong**, Yanggon Kim. "The Policy-Based AS\_PATH Verification to Prevent 1-Hop AS Path Hijacking by Monitoring BGP Live Streams", International Journal on Advances in Security, Vol.8, 2015.
- **Hong, Beomseok**, Youngsub Han, and Yanggon Kim. "A semi-supervised tweet classification method using news articles." In Proceedings of the International Conference on Research in Adaptive and Convergent Systems, pp. 62-67. ACM, 2015.
- Lee, Hyeoncheol, **Beomseok Hong**, and Kwangmi Ko Kim. "Documents topic classification model in social networks using classifiers voting system." In Proceedings of the 2015 Conference on research in adaptive and convergent systems, pp. 68-73. ACM, 2015.
- **Hong, Beomseok**, and Yanggon Kim. "A Weighted Question Retrieval Model using Descriptive Information in Community Question Answering." In Proceedings of the International Conference on Research in Adaptive and Convergent Systems, pp. 35-39. ACM, 2016.
- **Hong, Beomseok**, Yanggon Kim, and Sang Ho Lee. "Company Name Discrimination in Tweets using Topic Signatures Extracted from News Corpus." Journal of Computing Science and Engineering 10, no. 4 (2016): 128-136.
- **Hong, Beomseok**, and Yanggon Kim. "An Efficient Tag Recommendation Method using Topic Modeling Approaches." In Proceedings of the International Conference on Research in Adaptive and Convergent Systems, pp. 56-61. ACM, 2017.
- Han, Youngsub, **Beomseok Hong**, Hyeoncheol Lee, and Kwangmi Kim. "How do we Tweet? The Comparative Analysis of Twitter Usage by Message Types, Devices, and Sources." The Journal of Social Media in Society 6, no. 1 (2017): 189-219.
- **Hong, Beomseok**, and Yanggon Kim. "An Efficient Tag Recommendation Method using Topic Modeling Approaches." In Proceedings of the International Conference on Research in Adaptive and Convergent Systems, pp. 56-61. ACM, 2017.

Blank White Page