Novelty Detection by Latent Semantic Indexing

by

Xueshan Zhang

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Statistics

Waterloo, Ontario, Canada, 2013

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

As a new topic in text mining, novelty detection is a natural extension of information retrieval systems, or search engines. Aiming at refining raw search results by filtering out old news and saving only the novel messages, it saves modern people from the nightmare of information overload. One of the difficulties in novelty detection is the inherent ambiguity of language, which is the carrier of information. Among the sources of ambiguity, synonymy proves to be a notable factor. To address this issue, previous studies mainly employed WordNet, a lexical database which can be perceived as a thesaurus. Rather than borrowing a dictionary, we proposed a statistical approach employing Latent Semantic Indexing (LSI) to learn semantic relationship automatically with the help of language resources.

To apply LSI which involves matrix factorization, an immediate problem is that the dataset in novelty detection is dynamic and changing constantly. As an imitation of real-world scenario, texts are ranked in chronological order and examined one by one. Each text is only compared with those having appeared earlier, while later ones remain unknown. As a result, the data matrix starts as a one-row vector representing the first report, and has a new row added at the bottom every time we read a new document. Such a changing dataset makes it hard to employ matrix methods directly. Although LSI has long been acknowledged as an effective text mining method when considering semantic structure, it has never been used in novelty detection, nor have other statistical treatments. We tried to change this situation by introducing external text source to build the latent semantic space, onto which the incoming news vectors were projected.

We used the Reuters-21578 dataset and the TREC data as sources of latent semantic information. Topics were divided into years and types in order to take the differences between them into account. Results showed that LSI, though very effective in traditional information retrieval tasks, had only a slight improvement to the performances for some data types. The extent of improvement depended on the similarity between news data and external information. A probing into the co-occurrence matrix attributed such a limited performance to the unique features of microblogs. Their short sentence lengths and restricted dictionary made it very hard to recover and exploit latent semantic information via traditional data structure.

Acknowledgements

The beginning of this study was accidental. During a seminar on domain adaptation for learning in changing environments, the speaker raised an example of Amazon user reviews. His purpose was to show that people used different vocabularies for books and refrigerators, but the texts appeared on screen, with some words highlighted in red and green in the middle of a slide full of mathematical formulas, suddenly aroused my long-time passion for language, and the wild idea that I could do something to influence netizen opinions and behaviours by statistics. Technical development had vastly popularized the usage of internet in China; together brought a period of chaos since many people suddenly found they had the chance and freedom to speak to millions. Rationality and the respect to diverse opinions, which are critical in the development of a civil society, gave place to emotional reactions and verbal attacks. I was thinking that if the minority rational opinions which were overwhelmed could be picked out and read by more people, their ways of looking at a problem might not be limited to those angry words prevalent in the current internet environment. I chatted with Professor Mu Zhu, my supervisor, about my thoughts, more as a vision of future work which was too far from my current field of study. However, right on that night, Mu sent me a long email suggesting statistical novelty detection and clustering, the research of the former finally led to this paper. Though some remedial work on natural language processing was necessary, the majority of this project was an application of my familiar statistical methods. I thank Mu sincerely not only for his insights into the nature of statistics, which were acknowledged by a lot many students, but - most importantly - his concrete encouragements to students to pursue what they truly love, without which I could not understand what was the feeling to do something one really likes, nor could I find the linkage between my leisure interests and my major study.

If Mu was the one who opened the door for me, then my friend Wu Lin paved the way. As a novice who abruptly entered the vast territory of text mining, I was totally lost at the beginning since even the very basic thing: the data format, was different from those I was used to; not to mention the multitude of possible preprocessing treatments which turned lines of words into clean matrices. Wu shared his years of text mining experience with me, tutoring me in popular text models as well as python programming. When I dealt with

foreign database proficiently at the end of my research, I felt heartily grateful to his timely help.

I would also thank Professor Ghodsi and Professor Yingli Qin for their assistance and valuable advice on my thesis. As a preliminary attempt, this piece of work is far from perfect, but your help made it closer.

The thesis marks the end of my two-year study in Waterloo, which could not be as smooth as it was without the consistent support from Mary Lou Dufton. She gave me an enthusiastic welcome when I arrived alone in this foreign city, and has always been there when I need help.

I owe a debt of gratitude to my friends in Waterloo who made my life colourful and let me fall in love with Canada. Many thanks to Liang Li, who helped me not only throughout the study, but is always available even across the Pacific. Finally, I owe my deepest gratitude to my parents who brought life and love to me, and bear the feeling of missing only to let me chase my heart.

To my parents

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Novelty detection is a new topic in text mining. Pioneered by Yi Zhang's work [34] in 2002, it can be viewed as a natural extension of information retrieval systems, or search engines. It is inspired by the practical need that when tracking news events, discussions, scientific proceedings, or social network news feed, people often prefer a refined search result consists exclusively of the very latest updates, while all known, repetitive pieces are sifted out. However, the current search engine selects its results based only on the candidate documents' relevance to the query, without any consideration on novelty. Therefore, a user often have to face dozens of news articles, coming from different news agencies but reporting the same event backgrounds and repeating preliminary findings, while the truly up-to-the-minute advance is lying somewhere in the mess, taking forever to find. This predicament, called information overload, is hardly a stranger to any contemporary person.

Novelty detection aims to help out by filtering out old information with the aid of reading history and, most importantly, a measurement of novelty. The key problems lie in the definition and quantification of newness. It is immediate to see that novelty is not an objective concept with direct mathematical description. It varies from people to people, and from topic to topic. Basically, the sources of variability and subjectivity can be concluded as follows.

Firstly, different people have different background knowledge to a same subject, which will surly have an effect on the judgement of novelty. Such variety is due to educational level, personal interest, or just accidental factors. Personal traits and individual understanding of new information may play a role as well.

For example, when judging the newness of an economics article reporting the recently proposed BRICS Development Bank, there may be at least three responses. For those readers who even do not know what BRICS stands for, the entire passage, including a title picture of BRICS member country presidents' photos, a report on the agreement of establishing a new development bank, and a brief introduction to BRICS countries at the end, is all brand-new. For frequent readers in international affairs, the last part may be a platitude: they are only interested in the new bank and its possible influence. If some of them happened to miss the March news in Asia, the face of China's new president would be a surprise. These are just a few possible cases. Real situation will certainly be more diversified.

Therefore, a single novelty recommendation cannot satisfy everybody. The system should be personalized, which could be achieved by recording personal reading history on the local computer or a cloud account and employing adaptive filtering which takes reader feedbacks into consideration. Some explorations have also been made on setting dynamic threshold by [34] and [25]. In our system, which is a moderate first attempt, we simplified the situation by fixing the reading history, assuming that the input texts are all the information that a user will have, and produce our results based exclusively on these data.

Secondly, people have varied criteria and requirement for novelty on different topics, based on topic types, importance, even the media where the topic is reported.

For instance, in a disputation on an abortion pill, when the debate is held at national level and debaters involved House representatives, the president and the opposition party, with the practice of other countries such as France, Germany, and the attitude of Roman Catholic Church are quoted as support, a message that a figure of authority has finally taken his stance - even though his/her opinion itself may be exactly the same as an existing

view of others - are usually regarded as new, since here the speakers weigh as much as the content of the speech. However, if the debate happens on the Internet among common people, then the opinions will be much more important than the speakers. That an ordinary person simply clicked "like" without posting any original ideas should not bring a novelty alert to the followers of the discussion.

In consequence, a mature novelty detection system should carefully analyse the features of different subjects and return specialized reports. It calls for an understanding of the characteristics of traditional media topics and Internet posts, as well as knowledge in psychology. This may as well be set as one of the ultimate goals of a novelty detection system.

The last but not the least, language, as the carrier of information, has its inherent ambiguity. It proves to be not only a challenge in novelty detection, but a knotty point in all text mining tasks and one of the core problems of natural language processing. We briefly list some sources of ambiguity or their corresponding natural language processing research topics below.

- Word level:
  - Synonymy and polysemy.
  - Word structure. Grammatical conjugation.
- Sentence level:
  - Part-of-speech tagging.
  - Syntactical parsing.
- Passage or corpus level:
  - Lexical similarity.
  - Coreference identification. Cross-document coreference study.

In this thesis, we would like to focus our discussion on synonymy and polysemy. Words are the bricks of passages, as well as the most fundamental component of meanings. In novelty detection, polysemy is seldom a problem. Since each topic is restricted to its own area, a word often has only one definite meaning. However, its twin brother: synonymy, proves to a notable factor which determines performance. Here is an example taken from two successive news reports from Xinhua News Service on Princess Diana's car accident.

19970831 01:10 am	*Diana was fatally injured and later* ***died*** *in a hospital early Sunday in a car* ***accident*** *in Paris*
19970831 01:29 am	*Princess Diana* ***Dead*** *in Paris Car* ***Crash***

Apparently, the information in the second sentence is entirely included in the first report. However, if one apply cosine similarity to these two sentences directly (using bag-of-words expression and term frequency-inverse document frequency weights, which will be discussed in detail in Chapter 3), he/she may find a similarity score as merely 0.13. That is because in the plain vector-space-model, only the terms that directly appear in a sentence are given positive weights; those do not show up just have weights zero. Therefore, only when two sentences have exactly the same word, the word's weights would be counted in the cosine score, or the appearance of a word in one sentence would be totally cancelled out just because it does not occur in the other sentence as well. Let us look at the above example. If we use 0-1 to represent a word's presence, then parts of the vector representations would look like this.

	Princess	Diana	**died**	**dead**	Paris	car	**accident**	**crash**
Sentence 1	0	1	**1**	**0**	1	1	**1**	**0**
Sentence 2	1	1	**0**	**1**	1	1	**0**	**1**

From the above table we know: only the weights of *Diana*, *Paris*, and *car* participate in the calculation of similarity score. Even though *died* and *dead*, *accident* and *crash* have exactly the same meanings, their appearances were cancelled out because of no co-occurrence. Although a word stemmer can help us with verb conjugations, it does nothing with verb-adjective changes as well as synonyms. Therefore, an effective synonym-matching algorithm would enhance the accuracy of text similarity measures, thus boost the performance of the whole system.

In the Novelty Track of the 2002-2004 Text Retrieval Conference (TREC), participants from Carnegie Mellon University [6] and Chinese Academy of Sciences [24][32] attempted to deal with this problem by adding WordNet [30], a lexical database which can be perceived as a thesaurus. It contains 155,287 words organized in a network with nodes as 117,659 sets of cognitive synonyms. Distances between nodes can be used as a measure of similarity. These teams achieved moderate results; however, the effect of introducing the WordNet module remained unknown. Since each team had different pre-processing techniques, novelty measures, and selection rules, we could not find any pair of teams which adopted the same approaches in all components but only differed in whether used the WordNet. On the other hand, as statisticians, we prefer learning from data, instead of borrowing an existing dictionary.

In this thesis, we tried to address the synonymy problem by the well-established statistics method: Latent Semantic Indexing using the framework of kernels for texts. External databases, such as the Reuters-21578, were introduced to derive the latent semantic structure. We also ameliorated some technical treatments to the datasets to improve statistical soundness.

The layout of this thesis is as follows. Chapter 2 provides a detailed introduction to novelty detection. Chapter 3 illustrates the kernel framework we adopted and builds the kernel function step by step. Chapter 4 states our experiments and their results. The last chapter concludes the thesis by a discussion.

Chapter 2

Novelty Detection

2.1 Problem Description

In practice, a complete novelty detection process has two steps:

1. Given a topic, return all relevant texts.
2. Rank all relevant texts in chronological order. Examine them one by one. For any text, given only the texts that have been seen before, determine whether it contains novel information.

Obviously, the two steps are distinct in nature. The first one is a traditional **information retrieval** problem, requiring an evaluation of the texts' relevance to the query. The second one resembles a **filtering** task, in which all previous information is used as the mesh of a sieve to sift out repetitive texts and retain new ones. Accordingly, these two tasks should be treated separately. This thesis focuses on the latter. From now on, when we mention "Novelty Detection", we mean specifically the second problem.

An important note should be made here at the beginning of our discussion. The novelty detection is essentially an on-line problem, which is different from the majority of statistical learning tasks that we are familiar with. In the latter, the data set is fixed. It comes as an intact matrix on which statistical distribution is modelled or algebra operation is conducted. However, in novelty detection, the data set is growing over time. It starts as a one-row matrix representing the very first report on a topic. Every time we identify a text as novel or repetitive, it is added as a new row to the matrix. Therefore, if regular statistical learning methods such as matrix factorization are applied directly, the changing data set would require frequent re-factorization, which is neither efficient nor effective.

Such a situation greatly limits the range of novelty detection algorithms. In the past studies, treatments were largely confined to simple vector comparison. Methods involving matrix operation which have proven to be very effective in text mining, such as Latent Semantic Indexing (LSI), had never been employed. In this thesis, we attempted to propose a possible solution to the predicament: introducing external text data to establish a latent semantic space, on which we projected the incoming news files.

Here is another note before the formal discussion. Novelty detection in sequenced text data and novelty detection for the identification of outliers or abnormalities are two disparate subjects. The latter, as discussed in [14], [15], [26], and [21], works on static unordered datasets in classification or identification systems. It aims to detect whether an input is homogeneous with the training data which is known to a classifier. After the earlier discussion, we can clearly tell that this is an entire different task from the novelty detection problem that we are dealing with.

With these in mind, we can proceed to get our hands dirty by beginning the introduction on problem setting.

2.2 Unit of Detection

We chose to perform novelty detection on sentence level. Although document is the natural unit of written language, it is not ideal in novelty mining. As pointed out in the TREC

experience [22] [31] [23], every document contains some new information, especially in the news domain. As a consequence, a novelty detection system would return most of the documents, leaving readers the task to manually identify the novel parts in these passages, no matter whether the fractions of them are large or not. This is certainly not the scenario we want.

Therefore, sentence was adopted as the unit of retrieval. The TREC Novelty Track dataset, which we used, was prepared in the following fashion. Firstly, documents for a chosen topic were retrieved based on relevance (Step 1 in 2.1), ranked in time order, and then split into sentences. Each sentence was assigned a number composed of the published date and time of its source document, as well as the sequence of the sentence in the document. As a result, sentences coming from earlier documents were ranked at the front. When coming from the same documents, sentences were ranked in written order. On this sentence dataset, the human assessors of the TREC Novelty Track took a step further to pick out relevant sentences, and judged their novelty. Our experiment aimed to model their judgement of novelty based on the choice of relevant sentences.

2.3 Datasets

We used two datasets in the entire study. The main dataset came from the 2003-2004 TREC Novelty Track, which is the most reliable database in novelty detection. Another dataset, Reuters-21578, was adopted as external information to boost the performance of Latent Semantic Indexing.

2.3.1 TREC Novelty Track Data

After Yi Zhang's pioneer work in 2002, the Text REtrieval Conference (TREC) held three Novelty Tracks from 2002 to 2004 in a row. These competitions not only popularized the research subject, but established many standards that were followed by later studies. One

of them was the 2003-2004 TREC Novelty Track dataset (referred to as the TREC data below).

As we have seen in Chapter 1, the determination of "ground truth" for novelty detection is a tough problem. The TREC data did a good job. After an unsuccessful first trial in 2002, the event organizers in the National Institute of Standards and Technology (NIST) adopted the AQUAINT Corpus of English News Text [11] in the 2003 and 2004 conferences. This collection is more than suitable in that it consists of news articles from three main news agencies with overlapping time periods: New York Time News Service (Jun 1998 - Sep 2000), Associated Press Worldstream News Service (Jun 1998 - Sep 2000), and Xinhua News Service (Jan 1996 - Sep 2000). Therefore, it contains similar reports from different sources to the same event: an ideal copy of the real life situation faced by novelty detection. The organizers selected 50 topics per year: 100 in total. Each topic is an independent unit. A first assessor composed the topic query and descriptions, chose sentences according to the process in 2.2, and assessed the sentences' novelty. A second assessor would evaluate the novelty again independently. Most of the second assessors in 2003 and all of them in 2004 had compiled their own topics, thus were experienced in the selection. The assessor effect was carefully analyzed by NIST researchers in event summaries [22] [31] [23], according to which the human assessors had reached a reasonable level of agreement (some statistics in Table 2.2). In the final dataset, the judgements of the primary assessor were taken as ground truth for evaluation. The second assessors' judgement was not published, and was used as a ceiling for system performance in the conferences.

	2003	2004	Total
Event	28	25	53
Opinion	22	25	47
Total	50	50	100

Table 2.1: Fact table 1 of the TREC data. Numbers of event and opinion topics.

In addition, topics were divided into two types: event and opinion. In 2003, 28 topics were events; 22 were opinions. In 2004, each counted a half: 25. (Table 2.1) Event topics

included cloning of the sheep Dolly, the bombing at the 1996 Olympics in Atlanta, etc. Opinion topics were about controversial subjects such as gun control, same-sex marriage and the Lewinsky scandal. The examples on abortion pills and Diana's accident in Chapter 1 was just taken from the real dataset. From Table 2.2, we can tell that these two types varied in statistics, which may imply some difference in novelty selection as we discussed in Chapter 1. We will take this into account in the following studies.

Task	Topic Type	2003	2004
Relevant	All topics	0.39	0.20
	Events	0.47	0.25
	Opinions	0.38	0.15
Relevant agreement	All topics	0.69	0.60
	Events	0.82	0.68
	Opinions	0.63	0.50
Novelty	All topics	0.68	0.40
	Events	0.61	0.38
	Opinions	0.73	0.42
Novelty agreement	All topics	0.56	0.35
	Events	0.65	0.45
	Opinions	0.48	0.29

Table 2.2: Fact table 2 of the TREC data. Median fraction of sentences which were relevant and novel, and proportion of agreement by the secondary assessor.

2.3.2 Reuters-21578

The Reuters-21578 [12] (referred to as Reuters below) was published in 1990 for the first time containing 21,578 documents that appeared on the Reuters Newswire in 1987. Further formatting, data cleaning, and tagging was carried out from 1990 to 1996. It is one of the standard collections in text categorisation. The version we adopted was a part of the

Natural Language Toolkit (NLTK) package in Python, known as the "ApteMod" corpus. It has 10, 788 documents in total. As one of the most popular versions, this sample ameliorates the high skewness in the original data source to some extent. In our study, we ignored the category tags, using only the text content as background information from which to learn language knowledge.

2.4 Data Pre-processing

We performed the standard pre-processing operations for texts data. Steps are listed as follows.

1. **Stripping all the punctuation and capitalisation.**

2. **Tokenisation.** Sentences were broken into single words.

3. **Stop word removal.** Commonly used English words, such as "the", "has", "to" were removed, since they hardly contain useful information in sentence comparison. For the TREC data, we used the stop word list on [1]. For Reuters, the original list in NLTK package was applied. In addition, we removed all single-letter words in both datasets.

4. **Number removal for Reuters.** We kept all numbers in the TREC data, since numbers such as ages, dates, population are informative; but for Reuters, the background data, these numbers are useless.

5. **Stemming.** As discussed in Chapter 1, grammatical conjugation, inflection and derivation stand among sources of language ambiguity. Stemmers are designed to reduce the inflected and derived words to their root form. It may not be performed by mapping each word to its exact morphological root in the dictionary, but by simply deleting specific suffixes. For instance, in the Porter Stemmer we adopted [17], the ending "-y" is replaced with "-i"; endings "-e", "-ed", and "-s" are stripped. As a result, word "early" becomes "earli"; "bridge" turns into "bridg"; "beautiful" matches

with "beauty" since they are both changed into "beauti"; "look" and "looked" both become "look".

After these preparations, we were ready to apply the kernel framework. Before entering a detailed discussion in Chapter 3, let us first take a look at the performance measure.

2.5 Performance Measure: Average Precision

We chose average precision as our measurement of evaluation. Prevalent in Information Retrieval fields, this quantity resembles the ROC curve in spirit. The subsequent discussion follows Mu Zhu's article [36].

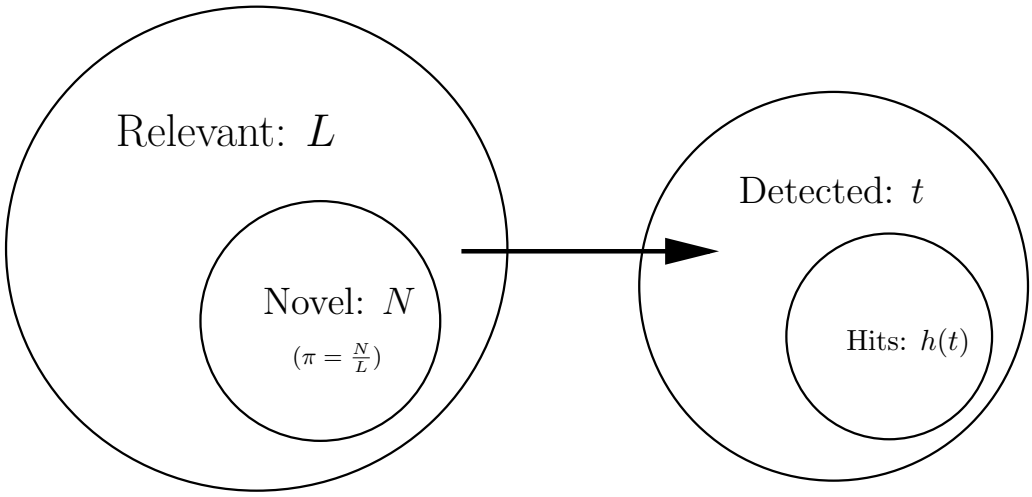

Figure 2.1: Illustration of a novelty detection operation. Among L collections, N are novel. A system detects t out of L, of which $h(t)$ are truly novel, i.e. the "hits".

A typical detection operation can be illustrated as Figure 2.1. Among the total collection of L sentences, N are novel. A novelty detection system identifies t, of which $h(t)$

are confirmed as correct. They are called "hits". The algorithm does this by assigning a novelty score to each sentence and ranking them. The higher the score, the more likely the sentence is what we want. Then after a threshold is given by system users, those sentences with above-threshold novelty scores are returned.

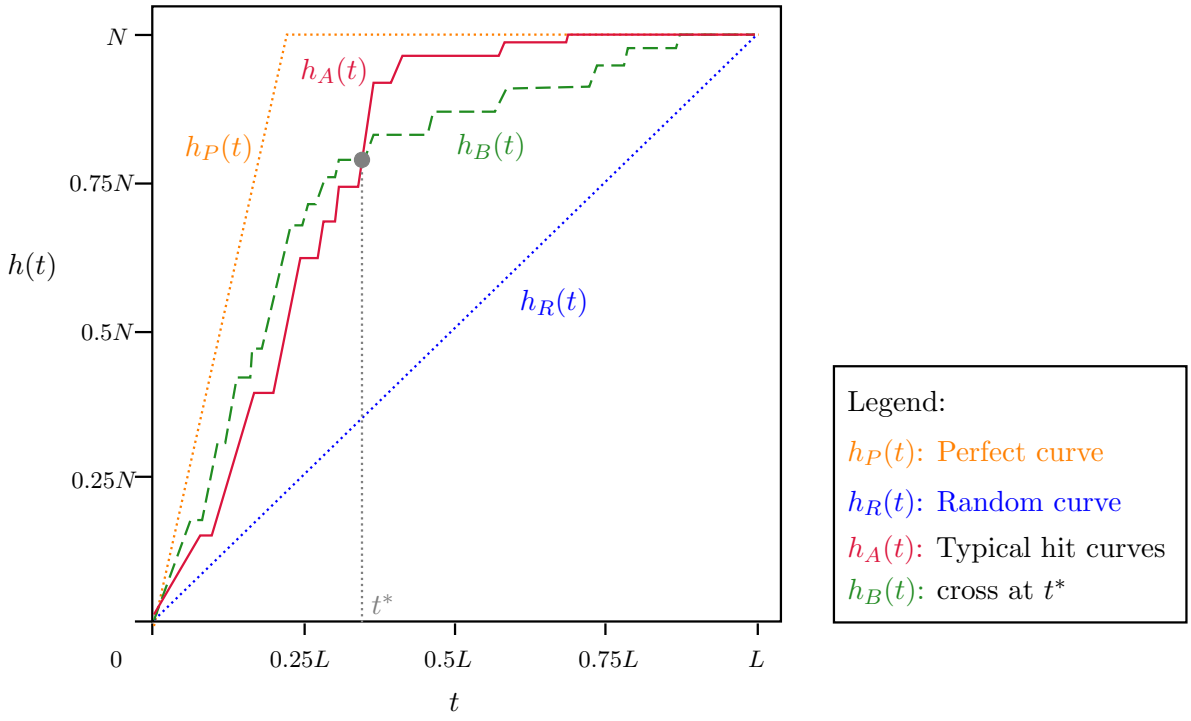

Figure 2.2: Artificial example of some hit curves. The curve $h_P(t)$ corresponds to perfect detection; $h_R(t)$ is produced by random selection. Curves $h_A(t)$ and $h_B(t)$ stand for typical hit curves produced in reality. Neither one has an uniform advantage; note that they cross each other at t^*.

If we pretend to detect only the most likely sentence at the first round; then add the next most promising one, one by one, we will get a hit curve by drawing $h(t)$ against t. Figure 2.2 shows some typical (though hypothetical) hit curves. The dotted orange curve on the top, $h_P(t)$, is an ideal curve corresponding to a perfect detection: all novel items

are ranked before non-novel ones, thus every sentence detected is an actual hit until all novel ones are exhausted. The dotted blue curve at the bottom, $h_R(t)$, is produced by random selection. The two curves in the middle: the blue solid one $h_A(t)$ and the green dashed one $h_B(t)$, stand for typical detection algorithms. Neither of them has an uniform advantage over the other. Although $h_B(t)$ climbs faster at the beginning, it is surpassed by $h_A(t)$ after their crossing at t^*. Of course, we would prefer an algorithm whose hit curve maintains a quick increasing rate - better be above all other curves for any t.

Unfortunately, this is hardly the case happening in reality. For most of the time, we have to compare two intersecting curves or even more. Such comparison can be troublesome and time-consuming, especially at the development stage when we are fine-tuning the algorithms. Therefore, a simple numeric performance measure is preferred. Such is the motivation of recall, precision, and their close relative: average precision.

Here is the definition of recall and precision. Let

$$y = \begin{cases} 1 & \text{if } \omega \text{ is relevant} \\ 0 & \text{otherwise} \end{cases} \qquad \text{and} \qquad \hat{y} = \begin{cases} 1 & \text{if the algorithm detects } \omega \\ 0 & \text{otherwise.} \end{cases}$$

Then

$$\text{Recall} = \Pr(\hat{y} = 1 | y = 1) \qquad \text{and} \qquad \text{Precision} = \Pr(y = 1 | \hat{y} = 1).$$

At a particular detection level of t, the two measurements are simply

$$r(t) = \frac{h(t)}{N} \qquad \text{and} \qquad p(t) = \frac{h(t)}{t}.$$

Because of the well-known recall-precision trade-off [36], the two measurements are hard to be optimized at the same time. Generally, $r(t)$ increases with t while $p(t)$ decreases with t. As an inconvenient consequence, both recall and precision must be considered simultaneously in algorithm evaluation. Therefore, a single numeric metric which integrates the two measurements, namely average precision, is proposed, and has become the most popular performance measure.

The definition of average precision is as follows:

$$\mathrm{AP} = \sum_{t=1}^{L} p(t)\Delta r(t).$$

As an example, table 2.3 is helpful for understanding.

Score Rank	t	Hit	$h(t)$	$p(t)$	$r(t)$	$\Delta r(t)$
1	1	1	1	$1/1$	$1/N$	$1/N$
2	2	1	2	$2/2$	$2/N$	$1/N$
3	3	0	2	$2/3$	$2/N$	$0/N$
4	4	0	2	$2/4$	$2/N$	$0/N$
5	5	1	3	$3/5$	$3/N$	$1/N$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

$$\mathrm{AP} = \left(\frac{1}{1} \times \frac{1}{N}\right) + \left(\frac{2}{2} \times \frac{1}{N}\right) + \left(\frac{2}{3} \times \frac{0}{N}\right) + \left(\frac{2}{4} \times \frac{0}{N}\right) + \left(\frac{3}{5} \times \frac{1}{N}\right) + \cdots .$$

Table 2.3: Calculation of average precision for a hypothetical algorithm

In conclusion, average precision is proposed in need of a single-number numeric performance measure which can be calculated regardless of a particular threshold. Such properties are highly desirable in algorithm evaluation and comparison. Therefore, we will adopt average precision as our measure of performance in the subsequent study.

Chapter 3

Kernel Methods

Kernel methods for text data is introduced in [21]. We employed it as a handy framework to integrate the vector space model, choice of inverse document frequency, with Latent Semantic Indexing.

3.1 Vector Space Model

With the pre-processed dataset (Chapter 2.4) in hand, an important question is how to convert these lines of text to a form that we are familiar with. Vector space model, or bag-of-words approach, is used most commonly today. Although relatively simple, it proves to be a powerful representation of language contents. Moreover, it takes one of our most acquainted formats: the matrix.

A note should be made here to avoid confusion. In Chapter 2.2, we chose sentence as the unit of detection. To keep our terms consistent with those commonly used in vector space model, we may use *document* to refer to the unit of input, which is sentence. In fact, after the original articles were broken into sentences, each sentence can be perceived as an independent document. These sentences together form the input of the following model.

As its name suggests, the vector space model represents every input sentence as a vector. Each element is associated with a unique word, or called term exchangeably. The value is how many times this word appears in the sentence, named term frequency. When processing a number of inputs, a *dictionary* recording all the words appearing in these sentences is needed. In reality, where new sentences are coming in as real-time updates, the dictionary is usually comprehensive and predefined. However in our experiment, the complete dataset was already in hand, thus we could build a dictionary whose index only includes the words that are used in a topic. Correspondingly, all the sentences in a topic are referred to as a *corpus*.

When the dictionary is ready, a sentence in a corpus can be mapped to a vector in a space in which each dimension corresponds to one term in the dictionary

$$\phi : d \longmapsto \phi(d) = (\mathrm{tf}(t_1, d),\, \mathrm{tf}(t_2, d),\, \ldots,\, \mathrm{tf}(t_T, d)) \in \mathbb{R}^T,$$

where $\mathrm{tf}(t_i, d)$ is the term t_i's frequency in the document d. T is the size of the dictionary as well as the number of dimensions in the image space. As the words used in a sentence are usually very small compared to the dictionary size, the vector is often sparse with most of its entries equal to 0.

Sentences mapped to this space form a *term-document matrix*

$$\mathbf{D} = \begin{pmatrix} \mathrm{tf}(t_1, d_1) & \mathrm{tf}(t_2, d_1) & \ldots & \mathrm{tf}(t_T, d_1) \\ \mathrm{tf}(t_1, d_2) & \mathrm{tf}(t_2, d_2) & \ldots & \mathrm{tf}(t_T, d_2) \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{tf}(t_1, d_L) & \mathrm{tf}(t_2, d_L) & \ldots & \mathrm{tf}(t_T, d_L) \end{pmatrix},$$

whose rows are indexed by the documents and whose columns are indexed by the terms. The total number of sentences in the current collection is denoted as L.

3.2 Novelty Metric

One of the key questions in novelty detection is how to determine the novelty score. Among the 2002-2004 TREC Novelty Track and subsequent studies, cosine similarity was most

frequently used to construct the metric.

Given a document d_{i+1} and its previous documents $d_1, \ldots, d_i$, the novelty score of d_{i+1} can be defined as

$$\mathrm{N}(d_{i+1}) = 1 - \max_{1 \le j \le i} \cos(\phi(d_{i+1}), \phi(d_j)) = 1 - \max_{1 \le j \le i} \frac{\langle \phi(d_{i+1}), \phi(d_j) \rangle}{||\phi(d_{i+1})|| \cdot ||\phi(d_j)||},$$

where $\langle \cdot, \cdot \rangle$ is the inner-product; $|| \cdot ||$ is the norm in the vector space.

In some research, sentences that are not identified as novel are removed from the collection. That is, d_{i+1} is only compared with a subset of novel sentences in $d_1, \ldots, d_i$. Such a practice comes from the requirement of many real-world on-line systems: due to a multitude of incoming documents and limited storage, only necessary information is stored. However, at the development stage, this procedure causes difficulty.

Firstly, to spot novel sentences, a threshold is needed, which is undesirable for the purpose of comparing algorithms. For different algorithms, the thresholds under which they perform best may differ wildly. To conduct a fair comparison, we need to experiment firstly with thresholds for each algorithm to find their most suitable ones, which adds to the complexity of our experiments. Moreover, by adopting the LSI approach in 3.4.2, we actually already have a tuning parameter: the number of chosen singular vectors. Adding the threshold as a second parameter requires a grid search in parameter tuning, which raises the time cost considerably. These extra efforts may obscure the original purpose of our experiment as well.

Secondly, by doing so, the system's earlier performance will largely affect its later judgements. If sentence S is novel but neglected by the system, then the next sentence similar to S would be identified as novel, although it may just be a repetition of S. A lapse at the beginning may cause a huge difference in the system's whole performance, which is not fair.

Therefore, we retained all read documents in the system since we hardly had a storage burden for the experiment dataset. For any document, it was compared with all previous sentences. By doing so, we not only focused ourselves on algorithm comparison, but also reduced the effect of false judgement to the minimum.

3.3 Kernel Function

Sometimes, the observations show a much clearer pattern after space transformation. When inner product is defined on the image space, it is desirable that we can get it directly from the data without computing the mapping explicitly. This is the role of a kernel function.

Kernel Function [21] A *kernel* is a function κ that for all $\mathbf{x}, \mathbf{z} \in X$ satisfies

$$\kappa(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle,$$

where ϕ is a mapping from X to an (inner product) feature space F

$$\phi : \mathbf{x} \longmapsto \phi(\mathbf{x}) \in F.$$

For text data, ϕ maps a document d to $\phi(d)$ in the vector space $\mathbb{R}^T$, thus the *vector space kernel* is

$$\kappa(d_1, d_2) = \langle \phi(d_1), \phi(d_2) \rangle = \sum_{j=1}^{T} \mathrm{tf}(t_j, d_1)\, \mathrm{tf}(t_j, d_2).$$

A *kernel matrix* can be created as

$$\mathbf{K} = \mathbf{D}\mathbf{D}',$$

whose (i, j)th entry gives $\kappa(d_i, d_j)$, which is just the cosine similarity between d_i and d_j.

3.4 Designing the Semantic Kernel

Though convenient, the vector space model has three main shortcomings. Firstly, it neglects word order, thus some grammatical relations are lost. Secondly, all terms are given equal weights. Although high-frequency words in the stop word list are removed, the rest of the terms still differ in overall frequency, thus have differences in importance. Thirdly, as demonstrated in the example of Diana's Car Accident in Chapter 1, the plain vector space

model does not consider information passed by synonymy. In computing cosine similarity, only terms that are non-zero in both documents are counted. If a synonymy pair is used separately in two sentences, their existences will just cancel out. Among these three drawbacks, the first one is inherent in the vector space representation. We aimed to ameliorate the other two using the semantic kernel that we will introduce in the next.

Consider a linear transformation of the vector $\phi(d)$:

$$\tilde{\phi}(d) = \phi(d)\mathbf{S}.$$

The corresponding kernel took the form

$$\tilde{\kappa}(d_1, d_2) = \langle \tilde{\phi}(d_1), \tilde{\phi}(d_2) \rangle = \phi(d_1)\mathbf{S}\mathbf{S}'\phi(d_2)'$$

We called $\mathbf{S}$ as the *semantic matrix*. It could be created as a composition of several steps. To tackle the latter two problems mentioned at the beginning of this section, we constructed $\mathbf{S}$ as

$$\mathbf{S} = \mathbf{R}\mathbf{P}.$$

Here, $\mathbf{R}$ was a diagonal *weighting matrix* indicating the relative importance of different words, and $\mathbf{P}$ was a *proximity matrix* giving the semantic distances between different words in the corpus.

3.4.1 Weighting Matrix: Inverse Document Frequency

Inverse document frequency is one of the most intuitive quantities to measure the amount of information a word contains. Consider the two words "model" and "shrinkage". Obviously, the former appears much more often in a common language database than the latter. For an article containing "model", we cannot determine whether it is talking about engineering, mathematics, or finance; but if it contains "shrinkage" instead, then it is probably a piece on statistics. Therefore, we can weight a word's importance by counting in how many documents it appears. The frequency should be an inverse of importance.

Document Frequency

$$\mathrm{df}(t) \stackrel{\mathrm{def}}{=} |\{d; \mathrm{tf}(t,d) \neq 0\}|.$$

Inverse Document Frequency

$$w(t) = \mathrm{idf}(t) \stackrel{\mathrm{def}}{=} \log \frac{L}{|\{d; \mathrm{tf}(t,d) \neq 0\}| + 1},$$

where L is the corpus size. The number '1' is added to the denominator to avoid a division-by-zero. The logarithm function is used to adjust the scale: by doing this, the maximum inverse document frequency would be approximately twice as the average. The base of the log function does not matter. In our experiments, we used the natural logarithm.

Substitute this definition of $w(t)$ into $\mathbf{R}$:

$$\mathbf{R} = \begin{pmatrix} w(t_1) & & & \\ & w(t_2) & & \\ & & \ddots & \\ & & & w(t_T) \end{pmatrix}.$$

The kernel function became

$$\tilde{\kappa}(d_1, d_2) = \phi(d_1)\mathbf{R}\mathbf{R}'\phi(d_2)' = \sum_{j=1}^{T} w(t_j)^2 \mathrm{tf}(t_j, d_1)\mathrm{tf}(t_j, d_2),$$

as a weighted multiplication of $\phi(d_1)$ and $\phi(d_2)$. We called $\mathrm{tf}(t,d) \times w(t) = \mathrm{tf}(t,d) \times \mathrm{idf}(t)$ as the *tf-idf* weight of term t in document d.

Generally, inverse document frequency is calculated with respect to the whole corpus: specifically to the novelty detection problem, to the whole topic. This is the practice adopted in all past studies. However, we suspected that this operation would lead to over-fitting. When document d_i is being read, we do not know anything about d_{i+1} to d_L. Computing the inverse document frequency of terms in d_i using the information in later sentences equals to "peeking" at these data that are waiting to be detected.

Based on the above consideration, it is better to calculate the inverse document frequency from some background data. Although Reuters was introduced as external source in the later section, it was mainly on financial news while the TREC focused on political and social events. Considering that the TREC topics were consistent in flavour in 2003 and 2004, we chose to utilize the topic themselves as the source of this background information. The 100 topics were divided into two parts by year; topics in the same year were treated as a big corpus to calculate the yearly overall inverse document frequency, which was used for the other year's topics. In short, the 2003 topics were weighted by the corresponding terms' inverse document frequency in 2004, and vice versa.

3.4.2 Proximity Matrix: Latent Semantic Indexing

In the last section, the weighting matrix $\mathbf{R}$ was diagonal, thus the effect of right-multiplying it to the term-document matrix was clear: to convert a simple vector multiplication to a weighted one. However, the proximity matrix, $\mathbf{P}$, could be any matrix, hence its mathematical meaning was obscure. To better interpret it, we expanded the expression

$$\tilde{\kappa}(d_1, d_2) = \phi(d_1)\mathbf{RPP'R'}\phi(d_2)',$$

and got

$$\begin{aligned}\tilde{\kappa}(d_1, d_2) &= \sum_{k=1}^{T}\left[\sum_{i=1}^{T}\sum_{j=1}^{T} w(t_i)w(t_j)p_{ik}p_{kj}\mathrm{tf}(t_i, d_1)\mathrm{tf}(t_j, d_2)\right] \\ &= \sum_{i=1}^{T}\sum_{j=1}^{T}\left[w(t_i)w(t_j)\mathrm{tf}(t_i, d_1)\mathrm{tf}(t_j, d_2)\left(\sum_{k=1}^{T} p_{ik}p_{kj}\right)\right].\end{aligned}$$

As a result,

$$\sum_{k=1}^{T} p_{ik}p_{kj}$$

played a role in adjusting the significance of tf-idf(t_i, d_1) $\times$ tf-idf(t_j, d_2). If we took the elements in $\mathbf{P}$ as distances, then it could be understood as a sum of the lengths of all

possible routes from term i to term j via one other term (including themselves) as transit, which coincided with our understanding of a "proximity" matrix.

Latent Semantic Indexing (LSI) is a prominent machine learning method to reveal the semantic relationship between different terms in a corpus. It was inspired by the need in Information Retrieval. A document may describe the same concept as the query only using different terms, thus a synonym matching is desirable. Roughly speaking, the LSI projects words and documents onto a latent semantic space where items with similar meanings are closer by making the best use of Singular Value Decomposition (SVD). We will illustrate this idea in detail using an example borrowed from the original paper in which LSI was proposed [7].

Here were nine sentences categorized into two fields and a query phrase. The first five sentences, marked as the "a" group, talked about human computer interaction which exactly matches the query. The latter four sentences, called the "b" group, were used as reference and lay in the domain of graph theory. Words appearing in more than one sentence were highlighted. If a word-for-word comparison was carried out, then only sentences a1, a2, and a4 were likely to be returned since they shared the exact terms used in the query; a3 and a5 would be missed.

Query	***Human computer*** *interaction*
a1	***Human*** *machine* ***interface*** *for Lab ABC* ***computer*** *applications*
a2	*A* ***survey*** *of* ***user*** *opinion of* ***computer system response time***
a3	*The* ***EPS user interface*** *management* ***system***
a4	***System*** *and* ***human system*** *engineering testing of* ***EPS***
a5	*Relation of* ***user****-perceived* ***response time*** *to error measurement*
b1	*The generation of random, binary, unordered* ***trees***
b2	*The intersection* ***graph*** *of paths in* ***trees***
b3	***Graph minors*** *IV: Widths of* ***trees*** *and well-quasi-ordering*
b4	***Graph minors****: A* ***survey***

To simplify, we only used the term-document matrix containing these highlighted words; inverse document frequency was not multiplied. We denoted this matrix as $\mathbf{D}$. The latent semantic indexing conducted singular value decomposition on $\mathbf{D}'$:

$$\mathbf{D}' = \mathbf{U}\Lambda\mathbf{V}'.$$

Then, we projected the original document and term vectors onto the space spanned by $\mathbf{U}_k$ and $\mathbf{V}_k$. $\mathbf{U}_k$ and $\mathbf{V}_k$ were the first k dimensions of $\mathbf{U}$ and $\mathbf{V}$. We chose $k = 2$ for the convenience of graph presentation.

$$d \longmapsto \phi(d)\,\mathbf{U}_k, \qquad term \longmapsto \mathbf{V}_k'\,\phi(term).$$

Figure 3.1 showed the geometric representation for terms and documents on the projected space.

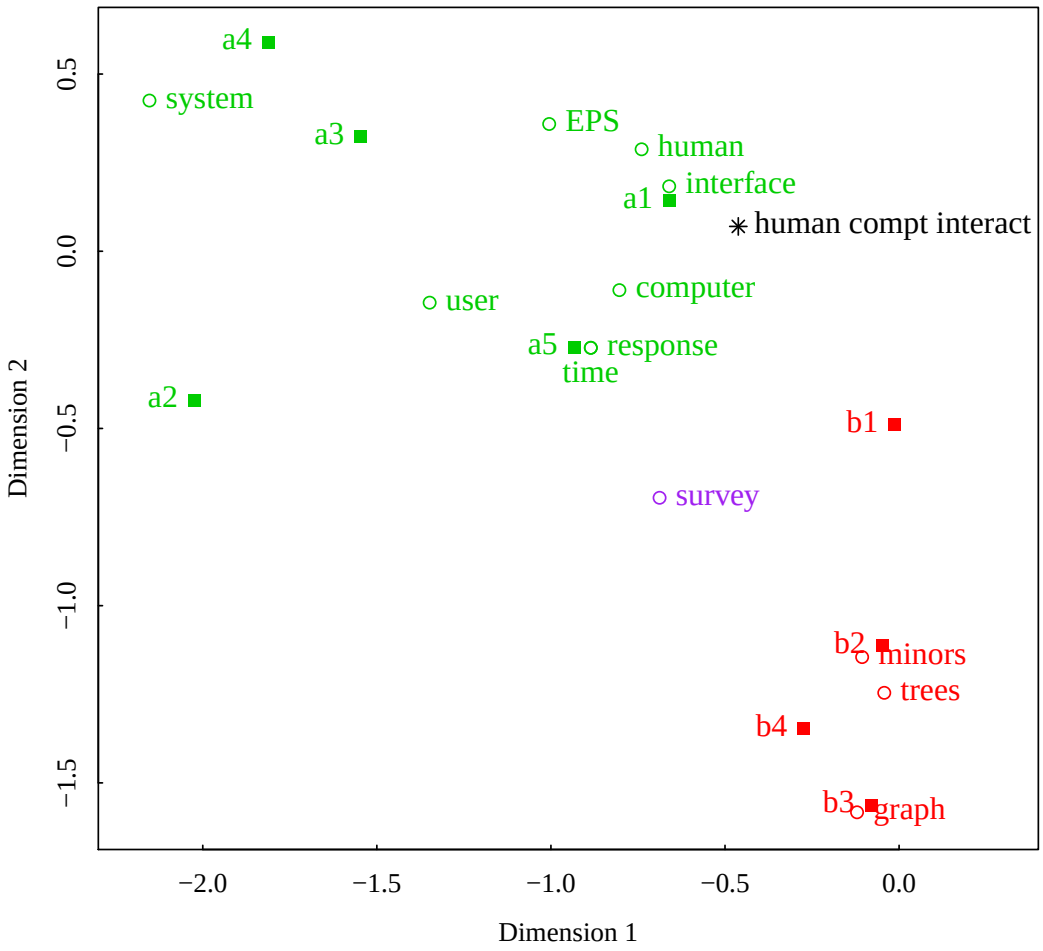

Figure 3.1: Example of LSI. Documents are represented as filled squares; terms are open circles; the query is the star mark. The documents and terms appearing only in group "a" are drawn in green; those in group "b" are red; term "survey" which is used in both groups is purple; the query is black. We can see that there is a natural cluster of these two groups. The query lies closer to the points in group "a". Even documents a3 and a5 which do not share common words with the query are in its vicinity.

From Figure 3.1, we could see that the terms and documents were naturally clustered into two groups, which coincided with their appearance in the original categorization.

Furthermore, the query "human computer interaction" lay in the vicinity of group "a", but far from group "b". Even documents a3 and a5 which did not share any common terms with the query were close to it in the figure. Actually, documents a1 to a5, but not b1 to b4, were in the cone area with a cosine of 0.9 to the query [7]. We could fairly hope that it would be promising to achieve our goal of synonymy matching by LSI.

As discussed in 2.1, we could not apply LSI directly to the changing term-document matrix in novelty detection. Instead, an external matrix, denoted as $\tilde{\mathbf{D}}$, was utilized to derive the latent semantic space.

$$\tilde{\mathbf{D}}' = \tilde{\mathbf{U}}\tilde{\Lambda}\tilde{\mathbf{V}}'$$

Putting everything together, our final expression of document d was

$$d \longmapsto \phi(d)\,\mathbf{R}_{\text{idf}}\,\tilde{\mathbf{U}}_k,$$

where $\mathbf{R}_{\text{idf}}$ was the inverse document frequency weighting matrix designed in 3.4.1; $\tilde{\mathbf{U}}_k$ was the proximity matrix by which documents were projected onto the semantic space learned from external knowledge; k was determined by the cumulative proportion of $\tilde{\Lambda}$'s diagonal elements. The final kernel function held the form

$$\tilde{\kappa}(d_1, d_2) = \phi(d_1)\,\mathbf{R}_{\text{idf}}\,\tilde{\mathbf{U}}_k\tilde{\mathbf{U}}_k'\,\mathbf{R}_{\text{idf}}'\phi(d_2).$$

Chapter 4

Experiments

4.1 System Summary

Here we summarize the system we constructed in the previous chapters.

System 1 Inverse Document Frequency

1: **for** year $y \in \{2003, 2004\}$ **do**
2: Calculate $\mathbf{R}_{\text{idf}}$
3: **for** topic i in the other year **do**
4: Extract submatrix $\mathbf{R}_{\text{idf}\{y \to i\}}$ whose columns have terms in $\text{Dict}(i)$
5: **end for**
6: **end for**

System 2 Latent Semantic Indexing

Choose the external source to apply LSI
Build tf-idf matrix $\tilde{\mathbf{D}}$ for the selected dataset
for topic $i = 1$ **to** 100 **do**
    Extract submatrix $\tilde{\mathbf{D}}^{(i)}$ whose columns have terms in Dict(i)
    $\tilde{\mathbf{D}}^{(i)\prime} = \tilde{\mathbf{U}}^{(i)}\tilde{\Lambda}^{(i)}\tilde{\mathbf{V}}^{(i)\prime}$
end for

System 3 Main

for topic $i = 1$ **to** 100 **do**
    for document $l = 1$ **to** L_i **do**
        Pre-process d_l
        Build dictionary Dict(i)
    end for
    for document $l = 1$ **to** L_i **do**
        $d_l \longmapsto \phi(d_l) = (\mathrm{tf}(t_1, d_l), \ldots, \mathrm{tf}(t_{T_i}, d_l))$
        if topic i is in year 2003 **then**
            $\phi(d_l) \leftarrow \phi(d_l)\mathbf{R}_{\mathrm{idf}\{2004\to i\}}$
        else
            $\phi(d_l) \leftarrow \phi(d_l)\mathbf{R}_{\mathrm{idf}\{2003\to i\}}$
        end if
        $\phi(d_l) \leftarrow \phi(d_l)\tilde{\mathbf{U}}_k^{(i)}$
        $\mathrm{N}(d_l) = 1 - \max_{1\le j\le l-1} \cos(\phi(d_l), \phi(d_j))$
    end for
    $\mathrm{AP}(i) = \sum_{l=1}^{L_i} p(l)\Delta r(l)$
end for

4.2 Experiment Description

In this study, we attempted to answer two questions:

1. Does Latent Semantic Indexing really help in Novelty Detection?

2. If so, what kind of language information is most effective in constructing the latent semantic space?

Based on the above motivations, we tried four different ways to choose the dataset for LSI, which were listed in Table 4.1. In the table, "E" denoted event topics; "O" stood for opinion ones. Thus "2003E" meant event topics in 2003. When LSI of type "same" was conducted, the tf-idf input in 2003E was projected onto the space formed by 2004E dataset. The rest could be interpreted in the same manner. In the bottom line, "na" stood for the baseline situation where *no* LSI was conducted and novelty scores were computed on plain tf-idf vectors.

	2003E	2003O	2004E	2004O
Reuters	Reuters	Reuters	Reuters	Reuters
same	2004E	2004O	2003E	2003O
all	2004	2004	2003	2003
other	2004O	2004E	2003O	2003E
na	n.a.	n.a.	n.a.	n.a.

Table 4.1: Experiment description: corresponding LSI datasets for topics in each category.

For each of the 20 combinations above, we chose the dimension of $\tilde{\mathbf{U}}_k$ by the following rule. Denote $\tilde{\Lambda}$ in SVD as

$$\tilde{\Lambda} = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_r),$$

where r was the norm of $\tilde{\Lambda}$; $\lambda_1, \lambda_2, \ldots, \lambda_r$ were non-negative.

Then k was selected as

$$k = \min m, \quad s.t. \ \frac{\lambda_1 + \cdots + \lambda_m}{\lambda_1 + \cdots + \lambda_r} \geqslant p.$$

Four different levels of p were tried: 0.75, 0.80, 0.85, 0.90.

4.3 Results

From raw results, we found that the level of p only affected the average precisions' absolute values, but did not show a remarkable influence on their relative patterns. Therefore, we used the mean average precision of different levels to simplify the following presentation.

As LSI was utilized to offer language proximity information to the experiment dataset, it was intuitive to guess that the more alike the two datasets were, the better the detection became. In fact, we did observe such a pattern when comparing LSI types of "same", "all" and "other" shown as follows.

AvePrc	2003E	2004E	2003O	2004O	2003	2004	E	O
same	0.7803	0.6016	0.8070	0.5700	0.7936	0.5858	0.6910	0.6885
all	0.7773	0.6018	0.8095	0.5667	0.7934	0.5842	0.6896	0.6881
other	0.7753	0.5991	0.8079	0.5578	0.7916	0.5785	0.6872	0.6829

Table 4.2: Result of LSI types differing in topic similarity. Categories "2003E", "2003O", "2004E", "2004O" show the mean of respective average precisions. Categories "2003", "2004" are the mean results in each year. Categories "E" and "O" are the mean results in each topic type.

Gray scale shows relative size of numerical values in the same column: full gray scale marks the largest; white means the smallest. It is clear that for all categories except "2003O", utilizing LSI from topics with the same type leads to the highest average precision; "all" ranks in the middle; while "other" is the last.

In Figure 4.1 and Table 4.2, the mean average precisions of each category described in 4.2 were listed. Yearly average and type average were also calculated. We could see the clear decreasing pattern shown when the similarity between LSI data type and experiment data type droped. "2003O" turned out to be the only exception. In fact, from the Fact Table 2.2 in 2.3, it was obvious that the agreement ratios on novel sentence judgements between two human assessors were much lower in the opinion topics than event ones. In

2003, the rate was 65% for events but 48% for topics. In 2004, it was 45% for events versus 29% for topics. This indicated that the detection on opinions might be a tough problem.

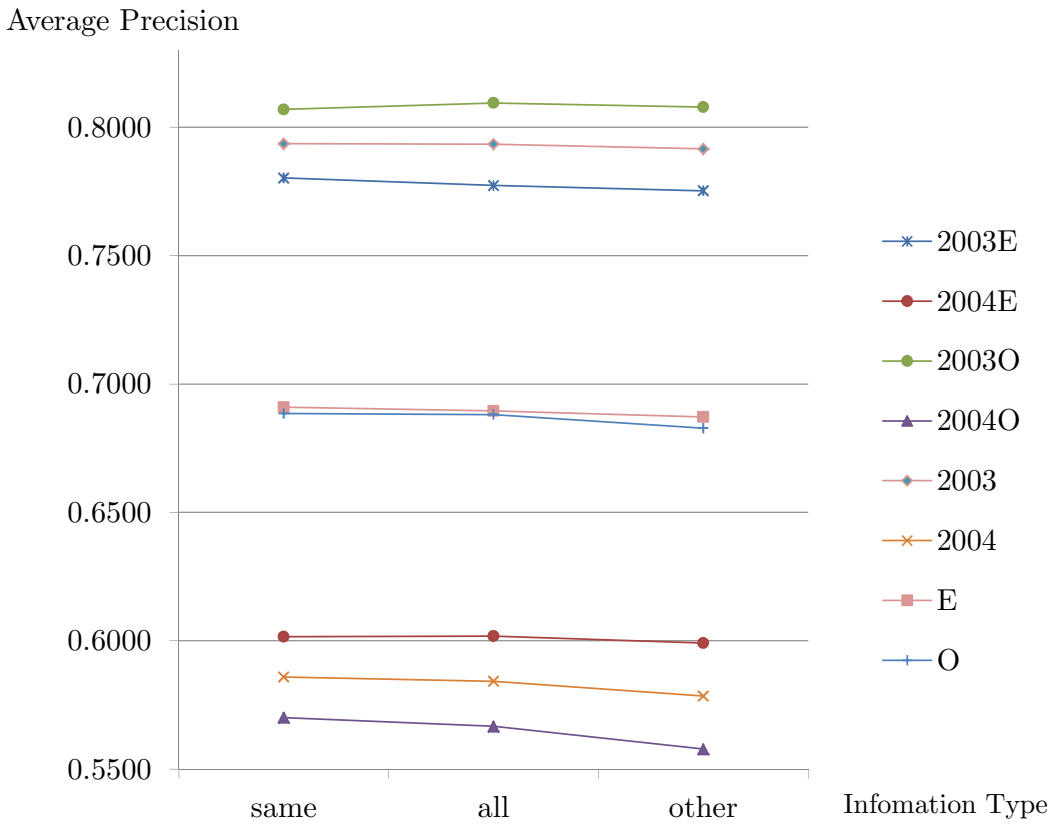

Figure 4.1: Result of LSI types differing in topic similarity. Categories "2003E", "2003O", "2004E", "2004O" show the mean of respective average precisions. Categories "2003", "2004" are the mean results in each year. Categories "E" and "O" are the mean results in each topic type. Apparently, all categories' average precisions decrease as the topic similarity drops from "same" to "all" to "other", with "2003O" as the only exception.

The Reuters data was one of the easiest-to-get text databases. Although it was used prevalently in text categorization, its LSI effect to our experiment topics was hard to

predict, because most of its documents fell in the domain of finance news. Though categories like "organization" and "people" existed, their index codes turned out to be items like "European Investment Bank" and "Prime Minister Brian Mulroney of Canada". Although some opinion topics featuring in finance discussions might favour such background information, the overall result for Reuters LSI did not turn out to be optimistic, shown in Table 4.3 together with our baseline "na" situation.

AvePrc	2003E	2004E	2003O	2004O	2003	2004	E	O
Reuters	0.7636	0.5899	0.8092	0.5752	0.7864	0.5825	0.6768	0.6922
same	0.7803	0.6016	0.8070	0.5700	0.7936	0.5858	0.6910	0.6885
all	0.7773	0.6018	0.8095	0.5667	0.7934	0.5842	0.6896	0.6881
other	0.7753	0.5991	0.8079	0.5578	0.7916	0.5785	0.6872	0.6829
na	0.7721	0.6050	0.8135	0.5776	0.7928	0.5913	0.6886	0.6956

Table 4.3: Result of all LSI types. The performance of Reuters was not satisfactory. Its best scores turned out to be in "2004O" and "O", where Reuters > same > all > other, however none of them beat na the baseline. Compared with this, LSI using the TREC data was a little better: in groups "2003E", "2003", and "E", the LSI outperformed the vanilla case.

From Table 4.3, we could see that when measured by average precision, Reuters, as well as the other LSI methods, did not show the overall advantage we might expect. The best performance of Reuters was in groups "2004O" and "O", where it surpassed all previous LSI datasets, but still fell a little behind of the baseline. However, it was cheerful to find that although in many groups the vanilla cases of na had the highest score, we did have three categories where LSI proved to be helpful. They were groups "2003E", "2003", and "E", in which "2003E" was the best. In these categories, the average precision produced by LSI methods showed a small advantage over the vanilla case.

Although Reuters did not meet our expectation in the category-level, it turned out to be rather effective in boosting those topics that actually benefited from it. To measure the advantage that a method had on the topics it was good with, we picked out all the

topics that it was in the lead, and calculated the weighted sum of differences between this method's topic average precision and its closet component's. The weights were the corresponding topic's corpus size. To our surprise, we found that Reuters jumped to the top in this round. In Figure 4.3 and Table 4.4, results under different LSI proportions were listed, as well as their averages. We could see that the Reuters not only took the lead in the relative ranking in Figure 4.3 , but also proved to have an obvious advantage in absolute numeric values in Table 4.4.

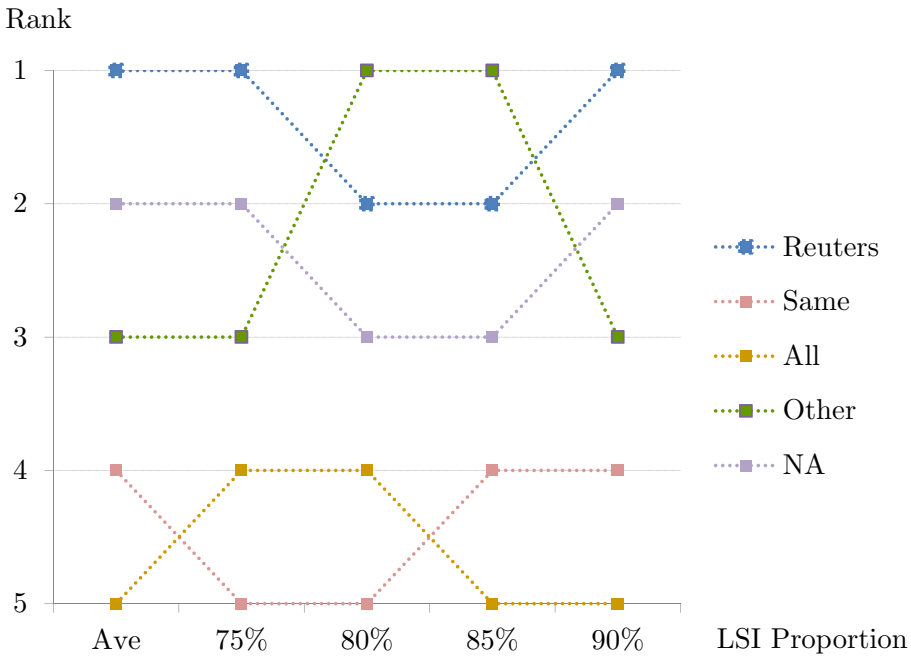

Figure 4.2: Rank of different methods' advantages in the topics that they topped. Coloured lines were drawn to show the methods' performance under different LSI proportions and their averages. The Reuters dominated in the cases of 75%, 90% as well as the average, and achieved the second in other proportions.

	Ave	75%	80%	85%	90%
Reuters	0.0217	0.0185	0.0190	0.0228	0.0220
Same	0.0067	0.0075	0.0102	0.0080	0.0142
All	0.0030	0.0077	0.0148	0.0071	0.0127
Other	0.0136	0.0095	0.0286	0.0261	0.0150
NA	0.0152	0.0164	0.0170	0.0161	0.0180

Table 4.4: Evaluation of different methods' advantages in the topics that they topped. Results under different LSI proportions were listed, and their averages were on the leftmost column. The Reuters dominated in the cases of 75%, 90% as well as the average, and achieved the second in other proportions.

Chapter 5

Discussions

In this thesis, we made some exploratory attempts on removing the synonymy obstacle in novelty detection by introducing Latent Semantic Indexing. External data sources were employed, such as Reuters-21578 and parts of the TREC data which were complementary with the experiment topics. Although Latent Semantic Indexing proved to be an effective tool in general text mining contexts, it only showed limited improvements on some of the topics, which was quite out of our expectation. In this chapter, we try to propose a possible explanation to this result.

In the Latent Semantic Indexing system summarized in 4.1, the singular value decomposition operation was not done on the original matrix $\tilde{\mathbf{D}}$, but its submatrix $\tilde{\mathbf{D}}^{(i)}$ which only contained those columns whose corresponding terms appeared in the dictionary of topic i, namely Dict(i) . We designed this step due to two reasons. The first was the memory and storage limits. The original tf-idf matrix of Reuters dataset had 10788 documents and 29786 words. It was hard to do SVD on such a high dimensional space. Secondly, even though a SVD was conducted and the experiment dataset was projected onto this original semantic space, those dimensions with terms out of Dict(i) would just remain zero. Therefore, the submatrix $\tilde{\mathbf{D}}^{(i)}$ was extracted to replace $\tilde{\mathbf{D}}$'s role in LSI.

However, this practice may lead to a loss of some critical semantic information in deriving the synonymy structure for novelty detection documents. Note that the left singular

vectors $\tilde{\mathbf{U}}$ of $\tilde{\mathbf{D}}'$ are actually the eigenvectors of $\tilde{\mathbf{D}}'\tilde{\mathbf{D}}$:

$$\tilde{\mathbf{D}}'\tilde{\mathbf{D}} = \tilde{\mathbf{U}}\tilde{\Lambda}^2\tilde{\mathbf{U}}'.$$

While the (i, j)th entry of $\tilde{\mathbf{D}}'\tilde{\mathbf{D}}$ can be written out as

$$(\tilde{\mathbf{D}}'\tilde{\mathbf{D}})_{ij} = \sum_{l=1}^{\tilde{L}} \mathrm{tf}(t_i, \tilde{d}_l)\mathrm{tf}(t_j, \tilde{d}_l).$$

That is, only when there exists a document where the ith and jth terms co-occur, the (i, j)th entry of $\tilde{\mathbf{D}}'\tilde{\mathbf{D}}$ is non-zero. In short, $\tilde{\mathbf{D}}'\tilde{\mathbf{D}}$ describes the co-occurrence pattern in the corpus, based on which the latent semantic structure is derived.

Recall our understanding of the proximity matrix. The strength of two terms' correlation was depicted by all the possible paths connecting them via other terms. These transit terms are important in that they can link two terms that have not occurred together but co-occur with the same third word respectively. In novelty detection, synonyms hardly appear together in the same document since documents are read in the unit of sentences, which are usually pithy. Instead, they tend to occur in different lines of input, relying on the "link" terms to unveil their relationship. Since topic dictionaries often contain much limited terms than a general corpus, many "link" terms may be lost when the Reuters space was truncated to another much smaller one. This may lead to failures to identify synonymy pairs in the subsequent SVD, resulting in unsatisfactory performances.

From this study, we gained a precious experience that sentence data, especially the microblogs, has a huge difference from the traditional datasets. Their short lengths, lack of contexts, as well as incomplete grammatical components make them an unique challenge. The classical vector space model which omits word order and grammatical information may not suit this new kind of text data. Innovative data structure, such as graph models, is in need. Luckily, some promising attempts have been made in this field. For instance, in K. Ganesan, C. Zhai, and J. Han's paper [10], a graphical method, called Opinosis, was developed to record the subject-verb-object structure of incoming sentences in order to produce abstractive summary. These novel representations may truly open a brand-new era for novelty detection.

References

[1] Stopword List. *https://code.google.com/p/text-categorization/source/browse/trunk/res/stopwords-lemur?r=64*, 2013.

[2] James Allan, Rahul Gupta, and Vikas Khandelwal. Temporal summaries of new topics. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 10–18. ACM, 2001.

[3] Ahmed Amrani, Jérôme Azé, Thomas Heitz, Yves Kodratoff, and Mathieu Roche. From the texts to the concepts they contain: a chain of linguistic treatments. In *In Proceedings of TREC*, volume 4, pages 712–722. Citeseer, 2004.

[4] Stephen Blott, Oisin Boydell, Fabrice Camous, Paul Ferguson, Georgina Gaughan, Cathal Gurrin, Gareth JF Jones, Noel Murphy, Noel E O'Connor, Alan F Smeaton, et al. Experiments in terabyte searching, genomic retrieval and novelty detection for TREC 2004. *Proceeding of the TREC*, 2004, 2004.

[5] Jaime Carbonell and Jade Goldstein. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336. ACM, 1998.

[6] Kevyn Collins-Thompson, Paul Ogilvie, Yi Zhang, and Jamie Callan. Information filtering, novelty detection, and named-page finding. In *Proceedings of the 11th text retrieval conference*, 2002.

[7] Scott Deerwester, Susan T. Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

[8] Taoufiq Dkaki and Josiane Mothe. TREC Novelty Track at IRIT–SIG. *Voorhees and Harman*, 13, 1999.

[9] Ao Feng and James Allan. Incident threading for news passages. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 1307–1316. ACM, 2009.

[10] Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. Opinosis: a graph-based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 340–348. Association for Computational Linguistics, 2010.

[11] David Graff. The AQUAINT Corpus of English News Text. Linguistic Data Consortirum, Philadelphia.

[12] David D. Lewis. Reuters-21578, Distribution 1.0. *http://www.daviddlewis.com/ resources/testcollections/reuters21578*, 2004.

[13] Xiaoyan Li and W Bruce Croft. An information-pattern-based approach to novelty detection. *Information Processing and Management*, 44(3):1159, 2008.

[14] Markos Markou and Sameer Singh. Novelty detection: a review: part 1: statistical approaches. *Signal processing*, 83(12):2481–2497, 2003.

[15] Markos Markou and Sameer Singh. Novelty detection: a review: part 2: neural network based approaches. *Signal processing*, 83(12):2499–2521, 2003.

[16] Kok Wah Ng, Flora S Tsai, Lihui Chen, and Kiat Chong Goh. Novelty detection for text documents using named entity recognition. In *Information, Communications & Signal Processing, 2007 6th International Conference on*, pages 1–5. IEEE, 2007.

[17] Martin F Porter. An algorithm for suffix stripping. *Program: electronic library and information systems*, 14(3):130–137, 1980.

[18] Liyun Ru, Le Zhao, Min Zhang, and Shaoping Ma. Improved feature selection and redundancy computing–thuir at trec 2004 novelty track. In *The Thirteenth Text Retrieval Conference (TREC 2004) Notebook*. Citeseer, 2004.

[19] Barry Schiffman and Kathleen McKeown. Machine learning and text segmentation in novelty detection. Technical report, 2004.

[20] Barry Schiffman and Kathleen R McKeown. Columbia university in the novelty track at trec 2004. In *Proceedings of the TREC*, volume 2004, 2004.

[21] John Shawe-Taylor and Nello Cristianini. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

[22] Ian Soboroff and Donna Harman. Overview of the TREC 2003 novelty track. In *Proceedings of TREC*, pages 38–53, 2003.

[23] Ian Soboroff and Donna Harman. Novelty detection: the TREC experience. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 105–112. Association for Computational Linguistics, 2005.

[24] Jian Sun, Wenfeng Pan, and Huaping Zhang. TREC 2003 novelty and web track at ICT. In *Proc. the Twelfth Text Retrieval Conference, Gaithersburg, Maryland*, page 138, 2003.

[25] Wenyin Tang and Flora S Tsai. Adaptive threshold setting for novelty mining. *Text Mining: Applications and Theory*, pages 129–148, 2010.

[26] L Tarassenko, A Nairac, N Townsend, I Buxton, and P Cowley. Novelty detection for the identification of abnormalities. *International Journal of Systems Science*, 31(11):1427–1439, 2000.

[27] Tomoe Tomiyama, Kosuke Karoji, Takeshi Kondo, Yuichi Kakuta, Tomohiro Takagi, Akiko Aizawa, and Teruhito Kanazawa. Meiji university web, novelty and genomics track experiments. *NIST Special Publication*, pages 500–261, 2004.

[28] Flora S Tsai, Wenyin Tang, and Kap Luk Chan. Evaluation of novelty metrics for sentence-level novelty mining. *Information Sciences*, 180(12):2359–2374, 2010.

[29] Ming-Feng Tsai, Ming-Hung Hsu, and Hsin-Hsi Chen. Approach of information retrieval with reference corpus to novelty detection. In *Proceeding of the TREC, 2003*, pages 474–479, 2003.

[30] Princeton University. About WordNet. *http://wordnet.princeton.edu*, 2010.

[31] Ellen M Voorhees. Overview of TREC 2003. In *Proceedings of TREC*, volume 2003, 2003.

[32] Hua-Ping Zhang, Hong-Bo Xu, Shuo Bai, Bin Wang, and Xue-Qi Cheng. Experiments in TREC 2004 novelty track at CAS-ICT. *NIST Special Publication*, pages 500–261, 2004.

[33] Min Zhang, Chuan Lin, Yiqun Liu, Leo Zhao, and Shaoping Ma. THUIR at TREC 2003: Novelty, robust and web. In *the proceedings of the Twelfth Text REtrieval Conference (TREC 2003)*, page 556, 2003.

[34] Yi Zhang, Jamie Callan, and Thomas Minka. Novelty and redundancy detection in adaptive filtering. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 81–88. ACM, 2002.

[35] Le Zhao, Min Zhang, and Shaoping Ma. The nature of novelty detection. *Information Retrieval*, 9(5):521–541, 2006.

[36] Mu Zhu. Recall, precision and average precision. *Department of Statistics and Actuarial Science, University of Waterloo, Waterloo*, 2004.