M.S. THESIS

Duplicate news filtering using sentence level heterogeneous graph

문장단위의 이종 그래프에 기반한 뉴스 중복 제거

FEBRUARY 2015

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Richeng Xuan

M.S. THESIS

Duplicate news filtering using sentence level heterogeneous graph

문장단위의 이종 그래프에 기반한 뉴스 중복 제거

FEBRUARY 2015

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Richeng Xuan

Duplicate news filtering using sentence level heterogeneous graph

문장단위의 이종 그래프에 기반한 뉴스 중복 제거

지도교수 이상구

이 논문을 공학석사 학위논문으로 제출함

2014 년 12월

서울대학교 대학원

전기.컴퓨터 공학부

현일성

현일성의 공학석사 학위논문을 인준함

2015 년 1월

위 원 장	김형주	(인)
부위원장	이상구	(인)
위 원	이창건	(인)

Abstract

With the flourishing development of the media of the network, dealing with the abusing news is becoming an essential requirement for portal news websites. However, previous research has only been attempting to improve the detecting efficiency or accuracy during finding near-duplicate news. Most of them rarely think about which news should be deleted or retained. Thus, we propose a heterogeneous graph-based news filtering framework using novel sentence level graph model for a new generation of duplicate news filtering, which is composed of two basic algorithms. First, extract and identify more duplicate news pairs by using sentence-level near-duplicate news detection algorithm; and second, calculate an accurate representative score by using the graph-ranking based on representative news selection algorithm. The proposed framework has been tested using real world dataset and the experimental result show that the proposed algorithms can improve the accuracy of descriptive news selection effectively.

Keywords: News filtering, Heterogeneous graph , Duplicate detection, Representative news

Student Number: 2013-22516

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

With the flourishing development of the media of the network, the growing abusing news on Web gets prosperous with every passing day and arouses the attention from Press, Research circle too, Near-duplicate news is one of the most representative type among the different types of abusing news. When users access news website, obviously, no one will show interest on the dozens of similar news which are often times "almost same". Therefore, it is an is becoming increasing essential research and development on news filtering.

In order to filter the duplicate news, transform the original text in the news can be the first step. Among several effective ways to model a text document, the Bag-of-words model is the commonest one. In the Bag-of-words model, words are assumed to appear independently and order is immaterial. This model is widely used in information retrieval and text mining. News can be represented with a vector having terms as theirs dimensions and term weight. For example,

text A	Kim likes movies. Park likes movies too.
text B	Kim also likes baseball.

Table 1.1 Two simple text examples

we have two simple texts as in Table 1.1. Base on 7 distinct words in two texts.

$\vec{T_a}$ = [1, 2, 2, 0, 0, 1, 1]

$\vec{T_b}$ = [1, 1, 0, 1, 1, 0, 0].

Each number in the vector represents the occurrences time of each word which can be known as term frequency(TF). We can also obtain TF-IDF value by multiplying the term frequency(TF) value and inverted index frequency(IDF) value. In general, the product of TF and IDF is used for term weight.

Therefore, the correlation between two vectors can represent the similarity of two different news. *Similarity Measure* is a real-valued function that quantifies the similarity between two news which contains many different methods to measure the similarity or distance of two news. *Cosine similarity* is one of the most popular similarity measure applied to text documents. This is quantified as the cosine of the angle between vectors, that is, the so-called *Cosine similarity*.

Given two text $\vec{T_a}$ and $\vec{T_b}$, their cosine similarity is

$$Sim(T_a, T_b) = \frac{\vec{T_a} \cdot \vec{T_b}}{\left|\vec{T_a}\right| \times \left|\vec{T_b}\right|}$$

Take the previous text examples and put them into the formula,

$Sim(T_a, T_b) = \frac{1\times1+2\times1+2\times0+0\times1+0\times1+1\times0+1\times0}{\sqrt{1^2+2^2+2^2+1^2+1^2}\times\sqrt{1^2+1^2+1^2+1^2}} = 0.754$,

So, the similarity between text A and text B is 0.754

After computing similarity, we usually define a threshold value to determine whether the two news is duplicated, if the similarity value exceeds the threshold value, we will say two articles are duplicated text.

1.2 Motivation

There are many studies on *near-duplicate detection* use the similarity of the entire article to figure out the records that share the same content. But it is impossible in the case of news text since the news media could be doing some cheating. They often modify news to reduce the similarity of the entire article so that they can raise their on-line visibility since the smaller similarity makes it easier to pass filtering process of the portal site. The simplest way among different modifications to decrease the similarity is add a paragraph to the news they want to post it repeatedly.

In many cases, the quality of portal filtering processing is not that good at all. So, even we find the correct duplicate news, we still won't know whether it should be abolished or not. It will lessen the user's trust in the portal site if we remove the high quality news instead of the plagiarize one. However, most previous studies are always focused on duplicate detection instead of choosing the best one to be retained, which could be more important than detection in the process of news filtering.

PageRank [2] is a famous ranking algorithm based on graph theory. Recent years, PageRank has been widely used in various fields. PageRank for Text document was also proposed by some researchers [4, 5, 6]. These studies are provided to extract the "representative" sentence using the similarity of sentences. Early graph ranking approaches were limited in homogeneous graph such as the network of citation between research paper. But recent studies are often focus on heterogeneous graph [11, 13, 14]. These approaches mainly focus on the scientific research paper ranking problem. Although these ways improved the rankings of papers and their authors in mutually reinforcing way, their models and algorithms were specific to paper and author graph. Since these models

only depend on the semantic relation between authors and papers, they can't be applied in other problems and heterogeneous graphs.

In this paper, we propose a graph based news filtering framework to detect duplicate news by using sentence level similarity and to select descriptive news by using novel heterogeneous graph ranking model. Our framework consists of two components. The first one is to find all similar pairs and then set the duplicate news in the same group, which is called *Near-duplicate Detection*. The second one, called *representative selection*, is divided to three steps: firstly, rank news in each group; secondly, select the top-k "representative" news; thirdly, remove the other in each group.

1.3 Outline

The rest of this paper is organized as follows. In Chapter 2, we review the related work in near duplicate detection and representative selection. Chapter 3 defines the problem. Chapter 4 presents the proposed method and algorithm. And the experiment result is shown in Chapter 5. Finally, Chapter 6 concludes this paper.

Chapter 2

Related Work

2.1 Near-duplicate detection

Near-duplicate detection has attracted much attention over the past few years and it is becoming an increasingly important topic in the present time of the Web news explosion. Today, the pace of life is increasing with technological advancements

Border [1] first defined the resemblance and containment between two documents. He proposed calculate method of similarity between documents which is the basis of near-duplicate document filtering.

Partial duplicate often appears on Web page and news. Qi. Zhang et al. [7] divided partial duplicate detection task into two subtasks: sentence-level near-duplicate detection and sequence matching. They proposed MapReduce based algorithm to detect large web collection. Their study proved effective sentence-level near-duplicate detection.

2.2 Graph-based representative selection

In recent years, several graph-based algorithms have been studied and claimed to be reasonable and effective in many domains, especially on ranking and summarization.

Chu et al. [10] presented a representative selection technique. They built a non-directed, non-weighted relationship graph between near-duplicate photos and then selected one representative photo by using "centrality value". This technique is an effectively demonstrates that graph-based method can be used in representative selection field although it seems less complicated.

The most popular ranking algorithm based on graph is PageRank [2], which is one of the most important ranking techniques being used in today's search engines. It is simple, robust, reliable and efficient. PageRank is defined formally as the stationary distribution of a stochastic process whose states are the nodes of a web graph.

2.2.1 TextRank

TextRank [4] is one of the most impressive graph-ranking based summarization algorithm in the Multi-document summarization. The TextRank can extract and identify the most representative sentence. This algorithm is used on undirected weighted graph that contains sentence vertex and similarity weighted edge. To put it simply, TextRank is an algorithm that PageRank being applied to text graph. Some different variants based on TextRank have been published by many researchers after TextRank was proposed, such as TimedTextRank [5].

In order to make it easier to apply TextRank on news, we consider a random walk on undirected weighted news graph G_N and a transition matrix from news to news N. Graph edge weight is the similarity value between different news,

therefore N can be calculated by this weight and damping factor α. We do not make a normal random work step with probability α, but instead jump to any vertex, chosen uniformly at random. We make use of damping factor to rewrite the transition matrix N to $\overline{N}$ as follows:

$$\overline{N} = (1 - \alpha)N + \frac{\alpha}{|n|} \tag{2.1}$$

Where $|n|$ is the number of news in G_N, $\frac{1}{|n|}$ is the uniform probability to every vertex. Then we can obtain TextRank on G_N by

$$n^{t+1} = \overline{N}^T n^t \tag{2.2}$$

Vector n contains the ranking score of news in G_N, n^t is the value of ranking score in time t. Given an initial value to n, update n iteratively until the convergence criterion $n^t - n^{t-1}$ reaches the low threshold value.

2.2.2 CoRank

Zhou, et al. [11] proposed the document and author ranking method based on heterogeneous graph. They built authors and documents graph using tree different relationships: a social network connecting authors, the citation network connecting document, and the co-authorship network that ties the previous two networks together. They also proposed CoRank algorithm that includes tree random walks to get ranking score. CoRank is the earliest work when ranking on heterogeneous graphs.

We add the sentence vertexes to the news graph, just like CoRank adds the authors vertex to built the heterogeneous graph. Figure 2.1 is a simple example: a document similarity connecting news(red), the sentence similarity connecting sentence(purple), and the contain relation that ties previous two together. This algorithm can rewrite the Equation (2.2) as follows:

$$n^{t+1} = (1 - \lambda)(\overline{N}^T)^m n^t + \lambda N S^T (S N^T N S^T)^k s^t \tag{2.3}$$

Vector s contains the ranking scores of all sentences in the G_S, λ is the coupling weight of intra-class step and inter-class step. Intra-class step is the one step random walk on inward graph. Inter-class step is the one step of external graph. Therefore, if random walker finds himself on the news graph, the current vertex is a news in G_N, then with probability $1-\lambda$ take m intra-class steps, while with probability λ take k inter-class steps.

Similarly, we can define the sentence side equation as follows:

$$s^{t+1} = (1-\lambda)(\overline{S}^T)^n s^t + \lambda S N^T (N S^T S N^T)^k n^t \tag{2.4}$$

Where n is the number of times random walker take the intra-steps in the G_S. In general, CoRank has 4 parameters, m, n, k, λ. These parameters will determine the algorithm result. We will give detailed description in the next chapter.

2.2.3 FutureRank

FutureRank [13] is another state-of-art graph-based co-ranking algorithm on heterogeneous graph. FutureRank adds *recency* value to paper ranking and simplifies the calculation formula in CoRank. The *recency* is one of time weight that current time $Time_{current}$ minus the publication time of the papers $Time_i$.

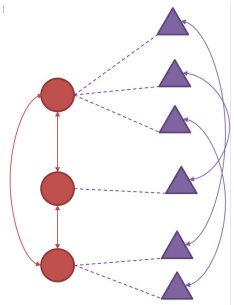


Figure 2.1 An example of heterogeneous graph

$Time_{current}$ is also a query time in ranking search result:

$$n_i^{Time} = e^{-\rho*(Time_{current}-Time_i)} \tag{2.5}$$

The ranking score of paper in FutureRank has four part: α part from papers; β part from authors; γ part from *recency* value; and $(1-\alpha-\beta-\gamma)$ part is uniform probability like damping factor in PageRank. Apply this algorithm in news, the ranking score of news is as fellows:

$$\begin{aligned} n^{t+1} &= \alpha N^T n^t \\ &+ \beta N S^T s^t \\ &+ \gamma n^{Time} \\ &+ (1-\alpha-\beta-\gamma)\frac{1}{|n|} \end{aligned} \tag{2.6}$$

α, β, γ is the weight of tree parts, and it is easy to find that sum of tree parameters must be less than 1. All of matrix in Equation (2.6) is original transition matrix without incorporated damping factor. In FutureRank random jump probability is represented by the fourth part of Equation (2.6). For instance, the transition matrix N doesn't use Equation (2.1). FutureRank removes the m, n, k parameters in CoRank, because the default values used in paper is $m = 2, n = 2, k = 1$. FutureRank just uses the default parameter value to simplify the equation. In fact, If we set β to 0, Equation (2.6) may equal to CiteRank [12]; If we set γ to 0, Equation (2.6) may equal to Equation (2.3) in CoRank; If we set γ and β to 0, Equation (2.6) may equal to Equation (2.2) in TextRank. CiteRank is one of raking publication algorithm based on publication citation graph and the publication time.

The ranking score of researcher is different from CoRank, it removes the effect of papers. The ranking score only depends on researcher's network.

$$s^{t+1} = S^T s^t \tag{2.7}$$

2.2.4 MutualRank

MutualRank [14] is one of state-of-art heterogeneous graph ranking frameworks. This framework integrates mutual reinforcement relationship among authors, publications of authors and venues to achieve a more accurate and fair ranking result. Venue information is also important to ranking paper and authors since we often assign some importance value to a paper in top tier conferences with few citations and undistinguished authors. However, previous studies on ranking papers and authors only utilized the paper and author information.

MutualRank combines HITS [3] and variant CoRank algorithm to reduce the unreasonably high ranked old papers skillfully. MutualRank decomposes the paper ranking score into *authority* and *soundness*, just like how HITS algorithm define *authority* and *hub*. They use *soundness* instead of *hub* to make semantics of this value clearly in this graph.

Currently, the framework has 4 vertex types, each of them has a ranking score. MutualRank updates *authority* value as follows:

$$\begin{aligned} paut^{t+1} &= \lambda_1 paut^t \\ &+ \lambda_2(1-\lambda_1)\overline{P}^T psnd^t \\ &+ \frac{(1-\lambda_1)(1-\lambda_2)\overline{PR}^T rimp^t}{2} \\ &+ \frac{(1-\lambda_1)(1-\lambda_2)\overline{PV}^T vprs^t}{2} \end{aligned} \tag{2.8}$$

Where vector $paut^{t+1}$ is the *authority* values in time $t+1$, $psnd^t$ is the *soundness* values in time t, $vprs^t$ is important of venues, $rimp^t$ is prestige of author. The first two line in Equation (2.8) is same to HIST, only add the weight λ_1 and λ_2. The author influence is represented as the third line in Equation (2.8), it is same to author factor in CoRank and FutureRank. The fourth line in Equation (2.8) is the venue effect which is get from venue paper network. We

can find the third line and the fourth line are divided by two, that is because MutualRank has two vertex to represent paper score. Half value of author and venue is used for reinforcing $psnd^{t+1}$.

$$\begin{aligned} psnd^{t+1} &= \lambda_1 psnd^t \\ &+ \lambda_1(1-\lambda_2)\overline{PT}^T paut^t \\ &+ \frac{(1-\lambda_1)(1-\lambda_2)\overline{PR}^T rimp^t}{2} \\ &+ \frac{(1-\lambda_1)(1-\lambda_2)\overline{PV}^T vprs^t}{2} \end{aligned} \tag{2.9}$$

Although MutualRank uses venue information and combines HITS algorithm to get a more complete model. However this unified mutual reinforcement model is highly specific to paper, author and venue heterogeneous graph ranking. Therefore, It can't be applied to other multi-network data well. Since the *authority* and *soundness* can get same value in the undirected graph, MutualRank can't be applied in the undirected graph well. Unfortunately, almost all text graphs are undirected graphs.

2.2.5 Other Approach

Wu. Xindong et al.[8] proposed a classification based news filtering system, which called NFAS. This system can recognize Web news page automatically. They trained a classifier base on the combination of URL, structure and content attributes. Although these classification approaches are pretty good option in news filtering, they still have to extract a lot of features to improve accuracy.

Chapter 3

Preliminaries

In this Chapter, we define the problem and summarize notations. Table (3.1 shows the notations will be used in this chapter.

3.1 Problem Definition

The problem of finding a slightly altered news pairs is termed *near-duplicate news detection*. In an other words, *near-duplicate news detection* is the problem that find all pairs of news that their similarities are almost max value.

Definition 1. ***Near-Duplicate News Detection***: Given a set news, a similarity function $Sim(n_i, n_j)$ and a threshold t, find all similar news for every news n_i such that there similarity is bigger than given threshold t.

After find the duplicate news, we can grouping similar news to get the news clusters. Each cluster contain near-duplicate news and different with other clusters.

Definition 2. ***Representative News Selection***: Given a set of similar

n, s, m	Vector of news, sentence, media ranking score
n_i, s_i, m_i	ith news, sentence, media
n^t, s^t, m^t	Ranking score of news, sentence, media on time t
$\|n\|, \|s\|$	Number of news .
$N, \overline{S}, \overline{SN}$	Transition Matrix
G_n, G_s, G_m	Graph of news, sentence, media
V_n, V_s, V_m	Vertex of news, sentence, media
E_n, E_s, E_m	Edge of news, sentence, media
$\lambda_1, \lambda_2, \lambda_3,$	Parameters of representative selection algorithm
$W(n_i, n_j)$	Edge weight of news n_i and n_j
$S(s_i, s_j)$	Similarity between sentence i and sentence j
n_i^{Time}	Time weight value of n_i
$T(n_i)$	Publication time of n_i
tr_{length}	Threshold of length in duplicate detection
tr_{ratio}	Threshold of length in duplicate detection

Table 3.1 Notations summarization

news, find the news n_i, which can represent all of the news set.

According to TextRank theory, we can get "representative" score in sentence network. TextRank have a assumptions that similar sentence have similar information. So, if we make edge between similar sentences, the highly ranked sentence will have more information. We can assume that highly ranked news is representative news.

Chapter 4

Framework

Our Framework makes duplicate news groups using sentence-level duplicate detection and ranks the news using heterogeneous graph model. As a result, we can select top-k news in each cluster and remove the other news from the clusters.

4.1 Near-Duplicate Detection

Finding sentence-level similarity could be useful; however, the sentence-level time complexity is unacceptable. The time complexity of finding sentence-level similarity is $O(K^2N^2)$, which is much worse than that of document-level algorithms that have time complexity of $O(N^2)$. It is the reason sentence-level detection has not been popular in the past years. However, as research on vector similarity join algorithm has been growing, the similarity join algorithms have improved drastically. Hence, we now have sufficient ability to deal with sentence-level algorithms.

	s_1	s_2	s_3	s_4	s_5
s_1	•				
s_2		•			
s_3			•	•	
s_4					
s_5					
s_6					•

Table 4.1 Example of similarity sentence matrix

The method is as follows. We get similarities of all pairs of news article sentences using MMjoin algorithm [9]. Then we obtain a similarity matrix as shown in Table (4.1. If a similarity between s_4 from a news article and s_3 from another news article is bigger than a certain threshold, we note the coordinates of the pair. Two news articles are considered duplicates if $MaxdiagonalLength$ or $MaxdiagonalRatio$ is larger than some threshold. $MaxdiagonalLength$ is the number of continuous sentence pairs that are similar. In Table (4.1, $MaxdiagnalLength$ is 3. Then, we can get the ratio using the follow formula:

$$MaxdiagonalRatio = \frac{MaxdiagonalLength}{AVG(|s_i|, |s_j|)}$$

4.2 Representative Selection

4.2.1 Graph Model

Denote the heterogeneous graph of news articles, their sentences and offices as $G(V, E) = G(V_n \cup V_s \cup V_m, E_n \cup E_s \cup E_{ns} \cup E_{sm} \cup E_{nm})$(see Figure 4.1 for illustration). There are mainly two types of edges: those with real-numbered weights and those that denote some relation between the connecting vertices. Those with weights have the following meanings: the weights of those connected

to a pair of news articles denote document similarities (red); the weights of those connected to a pair of sentences denote sentence-level similarities (purple). An edge that connects a news article to a sentence means that the news article contains the sentence (dotted purple). An edge that connects a news article to a media means that the media had provided the news article. An edge that connects a media and a sentence means that the sentence cites the media at some point.

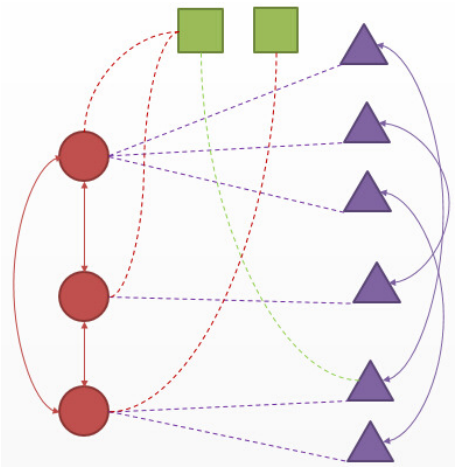


Figure 4.1 Heterogeneous graph of our framework

$G_n = G(V_n, E_n)$ is a weighted undirected-graph of news. V_n denotes a news vertex and E_n is a set of edges that connect news articles. The weight of each edge in E_n in G_n satisfies the following

$$W_n(n_i, n_j) = Sim(n_i, n_j) \tag{4.1}$$

Similar to the news graph, $G_s = G(V_s, E_s)$ is a weighted undirected-graph of sentences that contains the sentence-level similarity relation. The weight of each edge in E_s in G_s is set as

$$W_s(s_i, s_j) = Sim(s_i, s_j) \tag{4.2}$$

$G_{ns} = G(V_n \cup V_s, E_{ns})$ is an unweighed undirected-graph, where each edge represents 'containment'. Each edge in E_{ns} connects a sentence to its original news article.

$$W_{ns}(n_i, s_j) = \begin{cases} 1 & n_i contains s_j \\ 0 & otherwise \end{cases} \tag{4.3}$$

$G_{nm} = G(V_n \cup V_m, E_{nm})$ is an unweighed undirected-graph, where each edge represents 'provision'. An edge between n_i and m_i is established, if n_i was provided by m_i.

$$W_{nm}(n_i, m_j) = \begin{cases} 1 & m_j provide n_i \\ 0 & otherwise \end{cases} \tag{4.4}$$

$G_{sm} = G(V_s \cup V_m, E_{sm})$ is an unweighed undirected-graph, where each edge represents 'citation'. In reality, many news articles make direct references to news articles provided by other media offices. For example,

"*According to the* ***Daily Journal****, Memphis radio station WHBQ reported Monday morning that Freeze had signed an extension with Ole Miss that would pay him 4 million annually, with his assistant coaches each getting 500,000 a year.*"

Above sentence makes a reference to a media called *Daily Journal*, and this citation information is important in ranking news articles, because it is intuitive to reward the media with credibility when some news article mentions "according to Science Magazine", even if the news article provide no substance. The weights between sentence vertexes and media vertexes are assigned as follows.

$$W_{sm}(s_i, m_j) = \begin{cases} 1 & s_i citem_j \\ 0 & otherwise \end{cases} \tag{4.5}$$

4.2.2 Algorithm

Based on the assumptions that

"*Highly ranked sentence appear in highly ranked news, while highly ranked news contain highly ranked sentence, a news is ranked higher if it contains many sentences that appear in many other highly ranked news.*"

"*Highly ranked media provide highly ranked news, while highly ranked news was published by highly ranked media*", and "*a certain sentence is ranked higher if it reference the highly ranked media, that referenced by many highly ranked sentence*",

we can define the following.

$$\begin{aligned} n^{t+1} &= \lambda_1\lambda_3\overline{N}^T n^t \\ &\quad + (1-\lambda_1)\lambda_2\lambda_3 N S^T s^t \\ &\quad + (1-\lambda_1)(1-\lambda_2)\lambda_3 N M^T m^t \\ &\quad + \lambda_1(1-\lambda_3)\lambda_3 n^{Time} \end{aligned} \tag{4.6}$$

$$\begin{aligned} s^{t+1} &= \lambda_1\overline{S}^T s^t \\ &\quad + (1-\lambda_1)\lambda_2\overline{SM^T}^T m^t \\ &\quad + (1-\lambda_1)(1-\lambda_2) S N^T n^t \end{aligned} \tag{4.7}$$

$$\begin{aligned} m^{t+1} &= \lambda_1 m^{score} \\ &\quad + (1-\lambda_1)\lambda_2 M\overline{N^T}^T n^t \\ &\quad + (1-\lambda_1)(1-\lambda_2)\overline{MS}^T s^t \end{aligned} \tag{4.8}$$

From the above equations, we find that it is better not to apply damping factor to the transition matrices NS, NM, SN, and MN. Since relations between news articles and media and between sentences and news articles are binary, applying random jumps to these relations is meaningless. n^{Time} is the time value of news articles, which is used in the "personalized" PageRank vector. In the original PageRank, this "personalized" vector is a score vector to rank the results in favor of user preferences. The n^{Time} values in our framework are precomputed personalized vector. The score of n_i^{Time} is based on time of n_i and the other news.

$$n_i^{Time} = \frac{1}{1.0 + e^{d(T(n_0) - T(n_i))}} \tag{4.9}$$

where $T(n_0)$ is the time of the first published news article in each cluster, and d is the sigmoid parameter to control the time weight n^{Time}. If d is big, large time gaps will relatively lose their significance. This time weight function is different from time wight of FutureRank [13] we described in Equation (2.6) in page 9. m^{score} in Equation (4.8) is vector of office scores we precomputed based on the number of "main news" which we will describe in the next chapter. the main process of updating the three vectors at each iteration is as follows:

1. Values of news articles n at time t can influence the values of sentences s and medias m at time$t+1$. n^{t+1} will keep values as much as λ_1 for themselves first, which is updated by time weight n^{Time} by λ_3 proportions. The remaining part $(1-\lambda_1)n^t$ is divided into two parts, one part is $\lambda_2(1-\lambda_1)$ and the other is $(1-\lambda_1)(1-\lambda_2)$. The first part is used for reinforcing media m^{t+1} which appears at the second line of Equation (4.8).The second part is used for reinforcing sentence s^{t+1} which also appears at the third line of Equation (4.7).

2. Values of sentences s at time t can influence the values of news articles n and medias m at time $t+1$. Similar to news article vector, s^t is divide into 3

parts: values in proportions as much as λ_1 are channeled for themselves, while $\lambda_2(1-\lambda_1)$ will be channeled to news articles and $(1-\lambda_1)(1-\lambda_2)$ for medias.

3. Values of medias m at time t can influence the values of sentences s and medias m at time$t+1$. In contrary, s^t do not remain any part for themselves and gets updated by the media scores m^{score} that we precomputed value in λ_1 proportions. This can be found in the first line of Equation (4.7). The remaining is divided into 2 parts: $\lambda_2(1-\lambda_1)$ for sentences and $(1-\lambda_1)(1-\lambda_2)$ for news articles.

Algorithm 1 below summarized the whole process that update tree vectors:

Algorithm 1 Ranking score of each vertex

Require:

Transition matrix, $\overline{N}, NS, NM, \overline{SM}, SN, MN, \overline{MS}$,;

Time weighting, n_{Time};

Media score, m^{score};

A small threshold, ϵ ;

Parameters, $\lambda_1, \lambda_2, \lambda_3$;

$n \Leftarrow 1$

$s \Leftarrow 1$

$m \Leftarrow 1$

while $n' - n > \epsilon$ **do**

  $n' \Leftarrow n$

  $s' \Leftarrow s$

  $n \Leftarrow \lambda_1\lambda_3\overline{N}^T n' + \lambda_1(1-\lambda_2)\lambda_3\overline{NS^T}^T s' + (1-\lambda_1)(1-\lambda_2)\lambda_3\overline{NM}^T m' + (1-\lambda_3)n^{Time}$

  $s \Leftarrow \lambda_1\overline{S}^T s' + \lambda_1(1-\lambda_2)\overline{SM^T}^T m' + (1-\lambda_1)(1-\lambda_2)\overline{SN}^T n'$

  $m = \lambda_1 m^{score} + \lambda_1(1-\lambda_2)\overline{MN^T}^T n' + (1-\lambda_1)(1-\lambda_2)\overline{MS}^T s'$

end while

return n, s, m

Chapter 5

Experiment

In this chapter, we will evaluate several algorithms based on real world datasets. We will compare the previous algorithms to our algorithms, and then evaluate performance of our proposed framework according to the several performance criteria.

5.1 Data Preparation

For experiments, we use data from the most popular agglomerative news provider in Korea. Everyday, the website receives about 10,000 30,000 news articles from various sources; but many of them are considered near-duplicate. Because of the huge number of news articles, this portal site just selects about one thousand news articles to be displayed on the front page. We called these news articles "Main News". It is the important concept we introduced in Chapter 4.2.2.

For the experiment, the data consist of 7 days worth of news articles. The

Date	# News	# Main news	# Token
03.24	33149	1217	174350
03.25	34187	1276	178870
03.26	34307	1251	163132
03.27	35332	1204	172452
03.28	30877	1208	178963
03.29	13603	804	122352
03.30	17412	768	142548

Table 5.1 News dataset summarization

data summary is in Table 5.1.

5.2 Evaluation

The performance comparison between our algorithm and the previous algorithm is carried out in two ways. We compare how well do the algorithms detect near-duplicates on document level, and then we compare how well they are able to select the representative news articles along with the performance of a simple PageRank (TextRank), CoRank and FutureRank.

5.2.1 Near-duplicate detection

We evaluated near-duplicate detection on varying levels of similarities in the news data: sentence, paragraph, and document. To carry out the evaluation, we broke down each news article into paragraphs and sentences. Figure 5.1 shows the number of duplicate articles at each level of similarities. The bottom (red) parts of bar represents news that can be detected as near-duplicates, and top (blue) half represent the number of non-duplicate articles. From Figure 5.1, we

could see that the ratio of duplicates found increased as the level of duplicate detection became grainer.

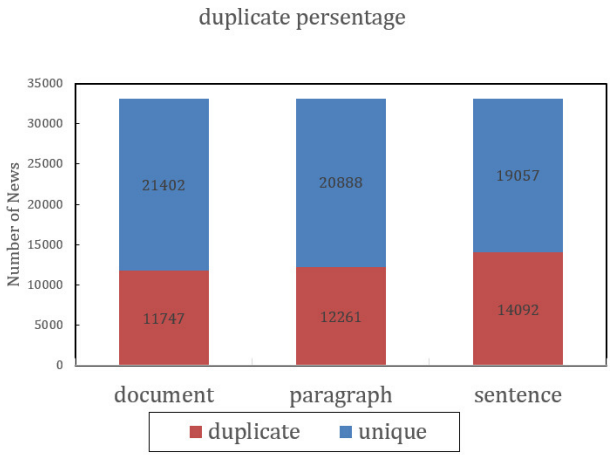


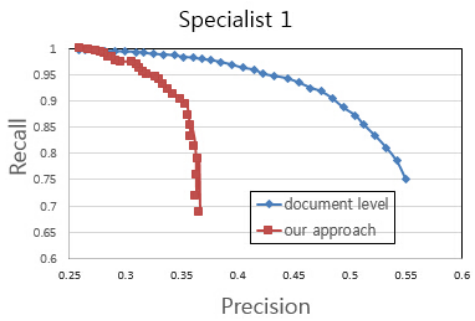
Figure 5.1 Duplicate Percentage of news

In the second part of the experiment, we created the answer set by checking duplication of each article manually which we random sampled 10% from the one day news. Three specialists of communication sciences extracted duplicate news pairs with their own viewpoints. We also get the intersection of the specialists to reduce in order to bolster their credibility. The summary of answer set are in Table 5.2. Compare with other specialists, specialist 1 tagged much more duplicate news pairs from sample news, because specialist 1 tagged two news as duplicate when the photo of two news were similar. Since there are only 3498 pairs in sample news where similarity of two news is bigger than 0.05, the specialist tagged most of news pair which have same word.

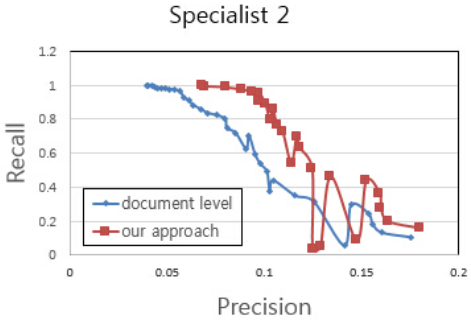
We compared document-level approach, which described in introduction, with our approach, which is described in chapter 4.1. Result for recall and precision are in Figure 5.2 and Figure 5.3. Different values of document-level approach have been obtained by varying the similarity threshold and different

specialist	# duplicate pairs
1	2913
2	222
3	359
1 and 2	212
1 and 3	350
2 and 3	111
1 and 2 and 3	108

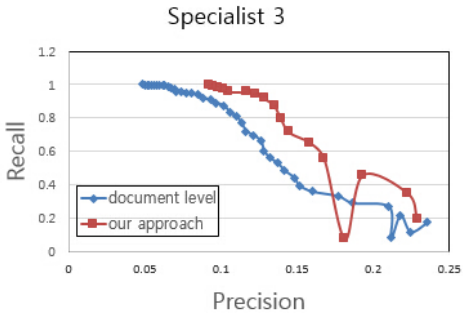
Table 5.2 Answer set



(a) specialist 1

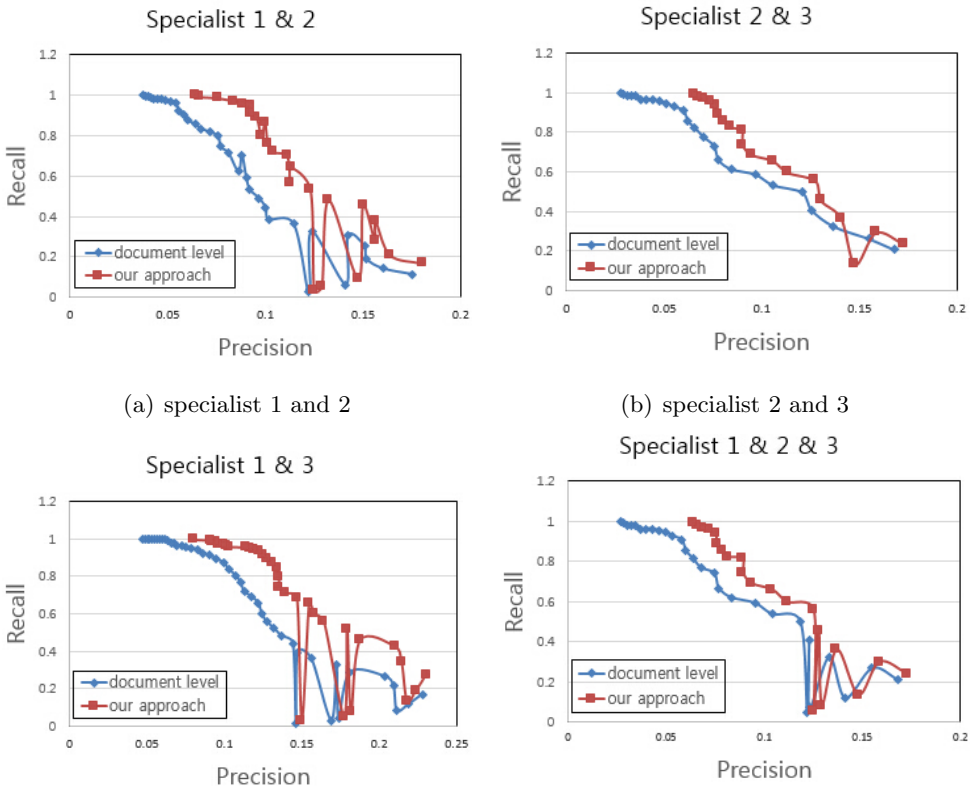


(b) specialist 2



(c) specialist 3

Figure 5.2 Accuracy of each specialist



(a) specialist 1 and 2

(b) specialist 2 and 3

(c) specialist 1 and 3

(d) specialist 1 and 2 and 3

Figure 5.3 Accuracy of each intersection

values of our approach have been obtained by varying length and ratio. From the result, we can see that, our approach outperform in intersections, specialist 2 and specialist 3. Since the specialist 1 see more values on photo in news and two approaches not using photo information, the result in Figure 5.2(a) is meaningless.

Using Figure 5.3, we determined an appropriate threshold which maximizes the F-1 score. The maximum F-1 score was achieved when $tr_{length} = 2, tr_{ratio} = 0.08$.

Then we applied our approach to find duplicate in total dataset on $tr_{length} = 2, tr_{ratio} = 0.08$. We used a simple clustering algorithms to make news article clusters.

5.2.2 Representative Selection

We can get news clusters after near-duplicate detection. Each cluster contains similar news, and the size of the clusters is in the range from 1 to 80. We rank representative scores of news articles in each cluster by methods listed below:

1.**Length**, the byte size of news content and title.

2.**Time**: the publication time of news.

3.**Regression**: linear regression using publication time, media ranking.

4.**PageRank(TextRank)**: ranking by simple TextRank on Graph G_n. Using Equation (2.2) in Page 7.

4.**T_SUM**: ranking by simple TextRank on Graph G_s. The rank of each news is calculated as the sum of the ranks of all sentence.

5.**CoRank**: ranking by CoRank on Graph G_n,G_s and G_{sn}. Using Equation (2.3), (2.4) in Page 7.

6.**FutureRank**: ranking by FutureRank on Graph G_n,G_s and G_{sn}. Using

Equation (2.6), (2.7) in Page 9.

7.**OurApproach**: ranking by our approach on Graph G_n,G_s and G_{sn}. Using Equation Equation (4.6), (4.7), (4.8) in Page 19.

We rank and select top-k news articles in each cluster. Then we count how many "Main News" articles remain. The number is then divided by total "Main News" count (summarized in Table 5.1) to obtain recall.

Figure 5.4 shows the total results of above method. This figure plots the recalls of varying algorithms for each group in top-1 to 4 values. Parameter tunning is an intrinsic difficult problem for all graph-based ranking algorithm. We learned parameter by using Bayesian optimization approach [15]. Because of the news graph is very different from paper and author graph, we obtain the entirely different from paper values. For CoRank $m = 1, n = 25, k = 0, \lambda = 0.77$. For FutureRank $\lambda_1 = 0.001, \lambda_2 = 0.089, \lambda_3 = 0.871$. For our approach $\lambda_1 = 0.522, \lambda_2 = 0.002, \lambda_3 = 0.885$. The evaluation shows that our approach outperforms the other 6 ranking methods. It also shows that a graph approach can be outperform regression approaches.

Note the actual agglomerative news website actually employs personnels to manually filter and select news article by publication times, media ranking and subjective perceptions. Thus, the result of the regression methods is probably what it would be like if the manual filtering jobs at the portal website has been replaced by an automated software. The results clearly show that our approach also simulates the "human sense" correctly. This "human sense" can not represented by data. Thus this measure can't be used in machine.

In order to compare the ranking algorithms, we use the performance metric which called *recommendation intensity*[14]. Let N_k be the list of top-k returned news. For each news n_i in N_k, the *recommendation intensity* of n_i can denoted as $RI(n_i)@k$, is define as

$$RI(n_i)@k = \begin{cases} 1 + \dfrac{(k - o_r)}{k} & n_i \in MainNews \\ 0 & otherwise \end{cases} \tag{5.1}$$

Where o_r is the ranked order of n_i in N_k. If a news $n_i$is recommended by a method and it in the "Main News", this method wins a score of 1. If this matched news is n_r-th one on the top-k list N_k, this method also wins an additional score of $\dfrac{(k - o_r)}{k}$. The *recommendation intensity* of a top-k result list N_k returned by a ranking method, denoted as $RI@k$, is formulated in Equation 5.2.

$$RI@k = \sum_{n_i \in N_k} RI(n_i)@k \tag{5.2}$$

Figure 5.4 shows the result of each ranking algorithm under different top-k. It shows the recommendation intensity curves of all the graph based algorithms discussed before. Our approach is consistently the most effective method.

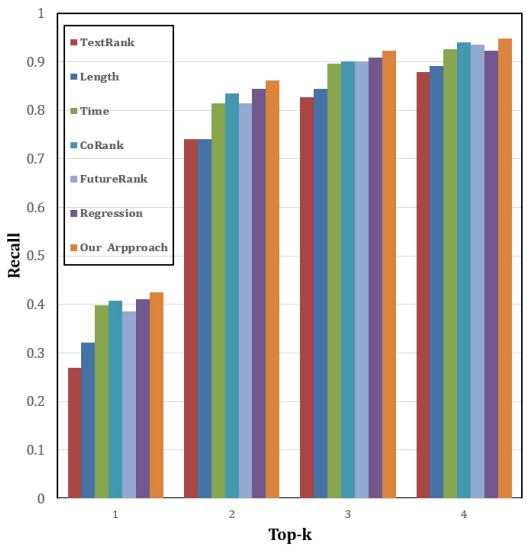


Figure 5.4 Recall of select top-k news in each cluster

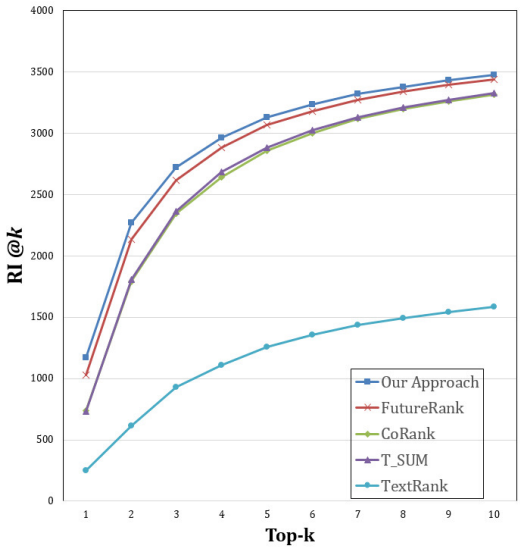


Figure 5.5 Recommendation intensity on each top-k

Chapter 6

Conclusion

In this paper, we presented a framework for filtering news articles composed of two algorithms: finding all duplicate pairs using a sentence-level duplicate detection algorithm, and ranking and selecting the most representative news using a heterogeneous graph-based algorithm. In our framework, the duplicate detection algorithm was able to find duplicate news articles that were not detectable using the previous method. The ranking algorithm was able to combine information about the relationships among the articles, media, sentences and publication times to effectively rank the "representative" scores. To prove our practicality, we performed experimental evaluations. The result of this evaluation showed that our approach had high F-1 score and achieved some improvements over the previous graph-ranking algorithm.

Bibliography

[1] Broder, Andrei Z. "On the resemblance and containment of documents." Compression and Complexity of Sequences 1997. Proceedings. IEEE, 1997.

[2] Page, Lawrence, et al. "The PageRank citation ranking: Bringing order to the web." (1999).

[3] Kleinberg, Jon M. "Authoritative sources in a hyperlinked environment." Journal of the ACM (JACM) 46.5 (1999): 604-632.

[4] Mihalcea, Rada, and Paul Tarau. "TextRank: Bringing order into texts." Association for Computational Linguistics, 2004.

[5] Wan, Xiaojun. "TimedTextRank: adding the temporal dimension to multi-document summarization." Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 2007.

[6] Cai, Xiaoyan, et al. "Simultaneous ranking and clustering of sentences: a reinforcement approach to multi-document summarization." Proceedings of the 23rd International Conference on Computational Linguistics. Association for Computational Linguistics, 2010.

[7] Zhang, Qi, et al. "Efficient partial-duplicate detection based on sequence matching." Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval. ACM, 2010.

[8] Wu. Xindong. et al, "News filtering and summarization on the web," *IEEE Intelligent Systems*, vol.25, no.5, pp. 68-76, 2010.

[9] Lee, Dongjoo, et al. "Efficient filtering techniques for cosine similarity joins." INFORMATION-An International Interdisciplinary Journal 14 (2011): 1265.

[10] Chu, Wei-Ta, and Chia-Hung Lin. "Automatic selection of representative photo and smart thumbnailing using near-duplicate detection." Proceedings of the 16th ACM international conference on Multimedia. ACM, 2008.

[11] Zhou, Ding, et al. "Co-ranking authors and documents in a heterogeneous network." Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on. IEEE, 2007.

[12] Walker, Dylan, et al. "Ranking scientific publications using a model of network traffic." Journal of Statistical Mechanics: Theory and Experiment 2007.06 (2007): P06010.

[13] Sayyadi, Hassan, and Lise Getoor. "FutureRank: Ranking Scientific Articles by Predicting their Future PageRank." SDM. 2009.

[14] Jiang, Xiaorui, Xiaoping Sun, and Hai Zhuge. "Towards an effective and unbiased ranking of scientific literature through mutual reinforcement." Proceedings of the 21st ACM international conference on Information and knowledge management. ACM, 2012.

[15] Hutter, Frank, Holger H. Hoos, and Kevin Leyton-Brown. "Sequential model-based optimization for general algorithm configuration." Learning and Intelligent Optimization. Springer Berlin Heidelberg, 2011. 507-523.

요약

인터넷 미디어의 발전과 함께 점점 많아지는 뉴스를 다루는 일이 중요해지고 있다. 특히 각종 뉴스를 다루는 포털 뉴스 사이트에서는 중복된 기사를 제거하는 문제를 중요하게 여기고 많이 노력하고 있다. 이전의 연구는 오직 중복 뉴스를 찾는 중에 검출 효율성이나 정확성을 향상시키기 위해 노력하고 있다. 그중 대부분은 유사기사가 검출된 다음 어떤 뉴스를 남기고 어떤 뉴스를 제거해야 하는지를 고려하지 않고 있다. 따라서 우리는 이 부분을 고안하기 위하여 새로운 중보기사 제거 방법을 제시하였다. 제안하는 중복기사 방법은 두 가지 알고리즘으로 구성된다. 첫 번째는 유사 기사 검출 알고리즘이고 문장단위의 유사도를 이용하여 더 많은 유사기사를 찾고 더 정확하다는 것을 실험으로 확인 할 수 있었다. 두 번째 그래프기반의 대표기사 선정 알고리즘은 참신한 이종 그래프를 이용하여 뉴스, 문장, 매제 정보를 적절하게 이용하여 서로보강하면서 실험결과에서 더 좋은 정확도와 재현율을 볼 수 있었다. 제안하는 방법은 실세계에 있는 데이터를 이용하여 실험하였고 실험결과는 기존의 알고리즘에 비해 일정한 성능향상이 있다는 것을 확인할 수 있었다.

주요어: 이종 그래프랭킹,중복 검출,대표기사 선정

학번: 2013-22516