A Scalable Approach for Chinese Term Extraction

Jyh-Jong Tsay and Jing-Doo Wang

Department of Computer Science and Information Engineering
National Chung Cheng University, Chiayi, Taiwan 62107, ROC.
{tsay, jdwang}@cs.ccu.edu.tw

Abstract

Term extraction is very helpful for Information Retrieval(IR) systems to have higher precision in retrieval, and that this capability is in demand for all of the Internet searching tools. In this paper, we develop a scalable approach via String B-tree(SB-tree) to identify significant terms from large amount of Chinese text data, which does not use a dictionary. Our approach consists of four steps : (i) texts information database, (ii) SB-tree construction, (iii) candidate significant term extraction and (iv) significant term validation. Our experiment uses three year news from Central News Agency(CNA) as the source to extract significant terms. The total number of the news and characters are $220,395$ and $80,046,457$ respectively. With the training corpus from such a long time period, we not only have robust statistic of terms, i.e. term frequency and document frequency, but also can detect some events via the distribution of significant terms according to different scale of time interval. What we have done is somewhat a fundamental work of text data warehouse.

1 Introduction

Under the trend of global networking through the Internet, the growth in the number of electronic documents in Chinese and oriental languages have been rapid, and these have mostly been published in Japan, Korea, Singapore, Hong Kong, Mainland China, Taiwan, etc. These documents are mostly non-structured and usually demand efficient Information Retrieval(IR) techniques for retrieval. There is a increasing need to retrieve large number of such documents quickly and intelligently through world-wide information networks. Term extraction is very helpful for IR systems to have higher precision in retrieval, and that this capability is in demand for all of the Internet searching tools. It is important to note that without efficient term extraction, many IR applications, for instance, text classification[19, 9, 14, 3], information filtering[13] and text summary, cannot obtain satisfactory results.

For Chinese IR systems[5], automatic term extraction from text is quite difficult especially for unknown words, such as names, locations, translated terms, technical terms, abbreviation, etc. It is generally believed that due to the inherent differences in languages such as the lack of explicit separators, i.e. blanks or delimiters, in written oriental sentences to indicate word boundaries, the techniques developed for retrieval occidental documents can not be directly applied to retrieval of oriental language documents. There are significant approaches[19, 18, 4, 2, 12] developed for Chinese term extraction recently. In [19, 18], they use a multi-linear term-phrase technique[11] to split adjacent character sequences(or character sequences in oriental languages). Adjacent character sequences are then merged pairwisely to form longer character sequences if they satisfy the criteria of the merging rules. This process repeats until no more adjacent character sequences can be merges. Those adjacent character sequences fail to satisfy the merging rules are discarded or put in the final term list depending on the frequency of occurrence. This approach is simple and straightforward, but it is not suitable for incremental text data. Chien[4] use *PAT tree* to extract terms from Chinese texts effectively. The PAT tree is incremental and is efficient in representing online corpus, especially when the corpus is dynamic, but PAT tree approach suffers from the limit of fixed memory size because the size of PAT tree may be about 10 times of the size of original text. Although CPAT tree(Compact PAT tree)[2] is proposed to improve PAT tree by reducing the main memory requirement of original PAT trees, its original PAT tree must be built as a temporal media in main memory before constructing a CPAT tree. We cannot build a main memory having an unbounded capacity and, therefore, we need external storage.

In this paper, we develop a scalable approach via String B-tree(SB-tree)[7] to identify terms from large amount of text data, which does not use a dictionary. Note that the String B-tree(SB-tree) is the first external-memory data structure that has the same worst-case performance as regular B-trees but handles unbounded-length strings and performs much more powerful search operations such as the ones supported by suffix trees[8]. Our term extraction process consists of four steps : (i) texts information database, (ii) SB-tree construction, (iii) candidate significant term extraction and (iv) significant term validation. We first store the basic information of each text in the database. Secondly, we individually construct, for each class of the news, an SB-tree and an Reverse SB-tree by inserting the suffix strings[8] of each sentence and its corresponding reversed sentence into the SB-tree and the Reverse SB-tree respectively. Thirdly, we extract candidate significant terms by scanning the leaves of the SB-tree and the Reverse SB-tree to decide the right and the left boundaries of significant terms, and put the extracted candidate terms into Right-Justify table and Left-Justify table in the database respectively. We finally have the significant terms which appear in the Right-Justify table and their reversed strings also appear in Left-Justify table.

Our experiment uses three year news, 1991/1/1-1993/12/31, from Central News Agency(CNA)[1] as source texts to extract significant terms. The total number of the news and characters are $220,395$ and $80,046,457$ respectively. The total size of original texts are about $152.7MB$, which is far more than previous related researches[19, 4, 12]. The maximum length of extracted terms is 20 and the number of significant terms extracted is $908,413$. With such a large scale and a long time period of training corpus, we can extract very specific and significant terms, such as "大陸地區人民在台灣地區定居或居留許可辦法"(the rules of the permission for the people in mainland China to settle down or to take up residence in Taiwan), and such terms are very helpful to improve Chinese text classification accuracy [17, 15, 14, 16]. With the training corpus from such a long time period, we not only have robust statistic of terms, i.e. term frequency and document frequency, but also can detect some events via the distribution of significant terms according to different scale of time interval. Note that we can observe various objects according to the statistic of significant terms with different scale of time interval, e.g. monthly and yearly, and the other attributes of terms associated with the original texts in the database if necessary.

The remainder of this paper is organized as follows. Section 2 describes the String B-tree(SB-tree). Section 3 describes our term extraction approach includes (i) texts information database, (ii) SB-tree construction, (iii) candidate significant term extraction and (iv) significant term validation. Section 4 gives our experimental results. Section 5 gives conclusion and further research.

2 String B-Tree(SB-tree)

The String B-tree(SB-tree)[7] is the first external-memory data structure that has the same worst-case performance as regular B-trees but handles unbounded-length strings and performs much more powerful search operations such as the ones supported by suffix trees. SB-Tree can be seen as a link between some traditional external-memory and string-matching data structures, and is a combination of B-Trees and Patricia tries for internal-node indices that is made more effective by adding extra pointers to speed up search and update operations.

We briefly describe SB-tree data structure[7] as following : each string in the input set is stored in a contiguous sequence of disk spaces and represent the strings by their logical pointers to the external-memory addresses of their first character, for example, as shown at the bottom in Figure 1(a). We can therefore locate the disk page containing the ith character of a string by performing a constant number of simple arithmetical operations on its logical pointer. Note that a single disk page contains only 4 logical pointers, as shown in Figure 1(a), and we are not able to retrieve strings' characters if we only read this page. The keys for each internal node of SB-tree are composed of the leftmost and the rightmost strings of its children. When managing keys in the form of logical pointers to arbitrarily-long strings, it is extremely inefficient if we compare any two strings character-by-character because the worst cost is proportional to the length of the two strings involved each time and the same input characters are (re)examined several times. In [6], Ferragina *et al.* transform the B-tree-like data structure into a simplified version of the SB-tree by properly organizing the logical pointers inside its nodes by means of Patricia tries. They plug a Patricia trie into each B-tree node in order to organize its strings properly and support searches that compared only one string in the keys for each internal node in the worst case. Note that the searches compare with only one string in the worst case rather than $\log_2 |B|$ ones required for a binary search, where $|B|$ is the number of suffix strings represented in that node of SB-tree. The value of B is 4 in Figure 1(a).

As shown in Figure 1(b), for example, the Patricia trie[8] with four suffix strings, "a#", "aa#", "aab#" and "ab#", represents the 4th node of the SB-tree in Figure 1(a). Note that all of the indexed suffix strings are appended an marker "#" to identify the ending. The detailed steps of the construction of Patricia tries are ignored here. When a node pointed to a suffix string is inserted into a Patricia trie, it will be inserted into the neighborhood of the node with a longest bit stream similarity and will be tagged with the minimal comparison bit to discriminate them. The advantage of Particia

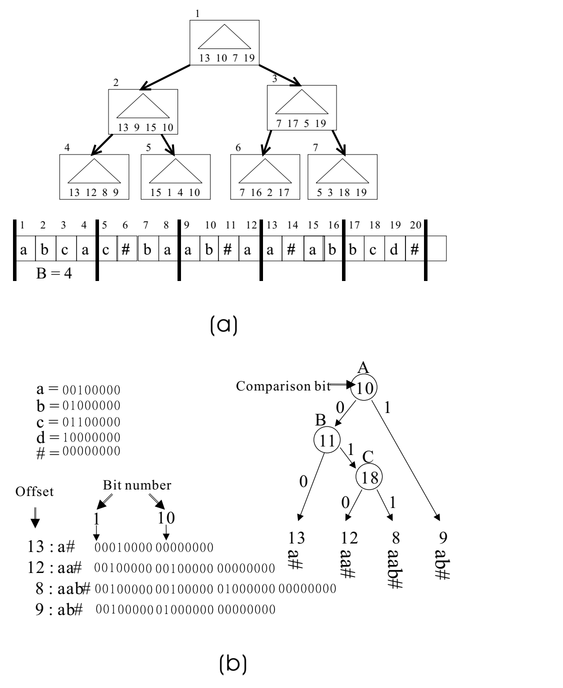


Figure 1: (a) An SB-tree example (b) Patricia trie : the 4th node of SB-tree

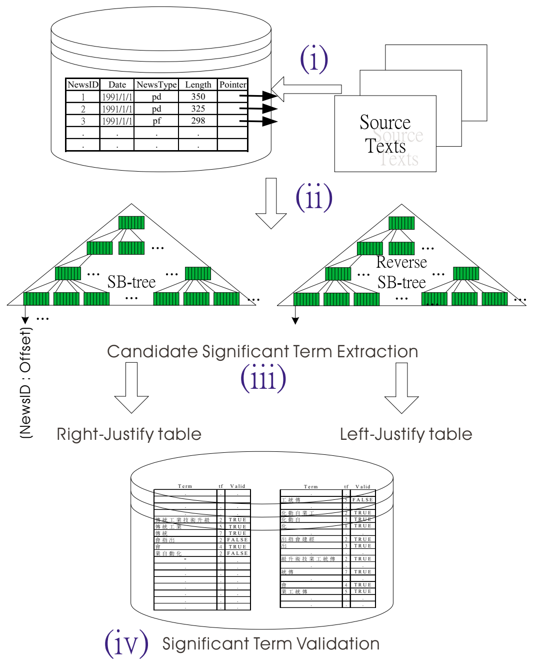


Figure 2: Overview of the Significant Term Extraction Process

trie is useful for speeding up searching.

3 Our Approach

Our term extraction process consists of four steps : (i) texts information database, (ii) SB-tree construction, (iii) candidate significant term extraction and (iv) significant term validation. As shown in Figure 2, we first store the information of each text in the database, such as text identifier(NewsID), construction date(Date), the type of news(NewsType), the length and the pointer of string to the beginning of the text, etc. Secondly, for each class of the texts, we individually construct an SB-tree and an Reverse SB-tree by inserting the suffix strings[8] of each sentence and its corresponding reversed sentence into SB-tree and Reverse SB-tree respectively. Thirdly, we extract candidate significant terms by scanning the leaves of the SB-tree and the Reverse SB-tree respectively to decide the right and the left boundaries of significant terms, and put the extracted candidate terms from SB-trees and Reverse SB-trees into Right-Justify table and Left-Justify table in the database respectively. We finally have the candidate significant terms in the Right-Justify table as significant terms if their reversed strings also appear in Left-Justify table. We describe the above four steps in Section 3.1, Section 3.2, Section 3.3 and Section 3.4 respectively.

3.1 Text Information Database

This is an preprocessing step for the SB-tree construction. In this paper, we have the information of each text store in the database, such as text identifier*(NewsID)*, construction date*(Date)*, the type of news*(NewsType)*, the length and the pointer of the string to the beginning of the text. After this step, we can access each text and find its construction date via its *NewsID* in the database. The suffix strings of each sentence in one text can be represented as the format *(NewsID : Offset)*, where "Offset" is the distance of the suffix string to the beginning of that text. With the data of *(NewsID : Offset)*, we can retrieve the suffix strings and find the construction date of each string such that we can have information not only ordinary statistic, i.e. term frequency and document frequency, but also the distribution of term frequency according to different time interval, i.e. monthly or yearly. Note that there are many other attributes can associated with texts in the database according to any specific domains if necessary.

3.2 SB-tree Construction

The purpose of SB-tree construction is to sort the suffix strings of the sentences in the texts such that the similar strings can be grouped together and the repeat-

(1) 傳統工業技術升級$	(1) 級升術技業工統傳$
(2) 統工業技術升級$	(2) 升術技業工統傳$
(3) 工業技術升級$	(3) 術技業工統傳$
(4) 業技術升級$	(4) 技業工統傳$
(5) 技術升級$	(5) 業工統傳$
(6) 術升級$	(6) 工統傳$
(7) 升級$	(7) 統傳$
(8) 級$	(8) 傳$
SB-tree	Reverse SB-tree

Figure 3: Suffix Strings examples(SB-tree)

ed patterns can be extracted by scanning the leaves of SB-tree. In this paper, we individually construct an SB-tree and an Reverse SB-tree, for each class of texts, by inserting the suffix strings of the sentences of the texts into SB-tree and the suffix strings of the corresponding reversed sentences of the texts into Reverse SB-tree gradually. The suffix strings of the sentence "傳統工業技術升級" for SB-tree and its corresponding reversed sentence "級升術技業工統傳" for Reverse SB-tree are listed in Figure 3. Note that the number of the suffix strings in the texts is equal to the number of the leaves of the SB-tree. Each leaf of SB-tree has the format of data as ($NewsID : Offset$), where *NewsID* represents the identifier of the text that the string belong to, and the *Offset* means the distance of the string to the beginning of the text. Therefore, the suffix string can be accessed when significant term extraction as described in Section 3.3. In Figure 4, for example, there are the partial of sorted suffix strings in the SB-tree and in Reversed SB-tree. For simplicity, we have only suffix strings from one text whose *NewsID* is 1 to construct SB-tree and Reverse SB-tree.

3.3 Candidate Significant Term Extraction

The process of the candidate significant term extraction with right(left) boundary validation is scanning from the least leaves of SB-tree(Reversed SB-tree) orderly, and extract the longest common prefix of adjacent strings as candidate significant term using stack operations, push and pop, to gather the related statistic information of each candidate significant term, and put to Right(Left)-Justify table in the database. We verify the right and the left boundaries of significant terms via scanning the leaves of SB-trees and Reversed SB-trees respectively.

We have significant terms as the terms with both right and left boundary validation. To have candidate significant terms with boundary validation, we have the repeated patterns that have, at least, two different kinds of successor Chinese characters as candidate significant terms. As shown in Figure 4(a), for example, "傳統", "傳統工業" and "傳統工業技術升級" are considered as candidate significant terms with right boundary validation and are inserted into Right-Justify table. Notice that the "傳統工業技", "傳統工業技術" and "傳統工業技術升" are not considered as candidate significant term because they have only one successor Chinese character "術","升" and "級" respectively. This process determines the right boundary of significant terms. To determine the left boundary of significant terms, we apply the same skill to Reverse SB-tree. For example, as shown in Figure 4(b), there are candidate significant terms, "級升", "級升術技業工" and "級升術技業工統傳", with left boundary validation in Left-Justify table. Similarly, the "級升術", "級升術技" and "級升術技業" are not considered as candidate patterns because they have only one successor Chinese character "技","業" and "工" respectively.

We show, for example, the partial steps of the candidate significant term extraction. As shown in Figure 5(a), the pattern "傳統工業" is an candidate significant term and has 4 items in IDlist. When pointing to the next string, as shown in Figure 5(b), we have the common prefix of two adjacent suffix strings as "傳統工業技術升級" whose length, 8, is longer than the length, 4, of "傳統工業" at the top record of stack. We then push the record with the "傳統工業技術升級" into stack with related IDList. When move to the next suffix string, as shown in Figure 5(c), we have the common prefix of two adjacent suffix strings as "傳統" whose length, 2, is shorted than the length, 10, of "傳統工業技術升級" at the top record of stack. Therefore, we pop up individually the records, "傳統工業技術升級" and "傳統工業", whose length are longer than 2, and output the statistic of the records to database, and add the IDLists of "傳統工業" and "傳統工業技術升級" to the record of "傳統" and push into stack as shown in Figure 5(c). Note that "傳統" is an substring of "傳統工業技術升級" and "傳統工業". Therefore, the IDList of "傳統" contains the union of two IDLists of "傳統工業技術升級" and "傳統工業". Note that the statistic of one string is covered in that of its substrings. For example, if the term frequency of "傳統工業技術升級" is 2, then the term frequency of "傳統" and "傳統工業" must be 2 at least. It is the reason that we adapt stack operations to handle the statistic of the strings.

3.4 Significant Term Validation

We have the terms in Right-Justify table as significant terms if their reversed string exists in Left-Justify table. As shown in Figure 6, for example, there are terms in Right-Justify table and Left-Justify table constructed after candidate significant terms extraction. The "傳統工業技術升級" in Right-Justify is an significant term because its reversed string "級升術技業工統傳" appears in Left-Justify table. On the other hand, the

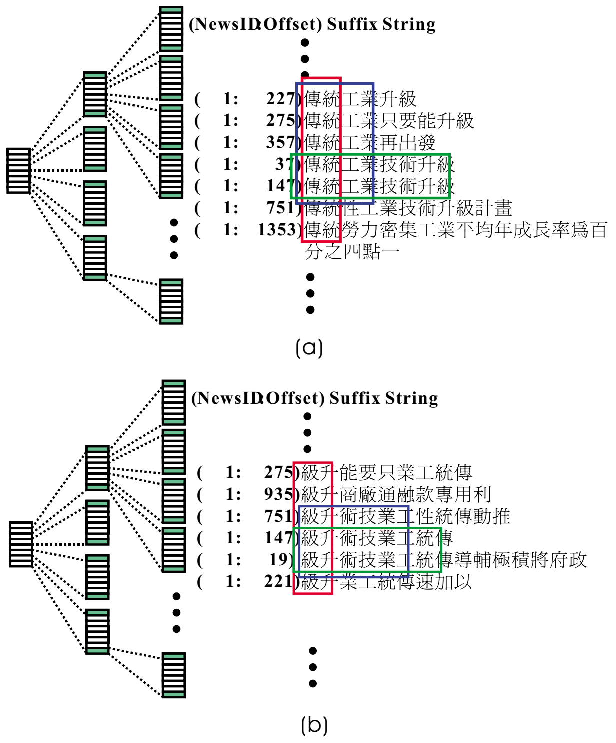

(NewsID:Offset) Suffix String
(1: 227) 傳統工業升級
(1: 275) 傳統工業只要能升級
(1: 357) 傳統工業再出發
(1: 37) 傳統工業技術升級
(1: 147) 傳統工業技術升級
(1: 751) 傳統性工業技術升級計畫
(1: 1353) 傳統勞力密集工業平均年成長率爲百分之四點一
(a)
(NewsID:Offset) Suffix String
(1: 275) 級升能要只業工統傳
(1: 935) 級升商廠通融款專用利
(1: 751) 級升術技業工性統傳動推
(1: 147) 級升術技業工統傳
(1: 19) 級升術技業工統傳導輔極積將府政
(1: 221) 級升業工統傳速加以
(b)

Figure 4: (a) SB-tree (b) Reverse SB-tree

” 業自動化” is not an significant term because there is no ” 化動自業” in Left-Justify table.

4 Experimental Results

We use three year news, 1991/1/1-1993/12/31, from Central News Agency(CNA)[1] as source texts to extract significant terms. As shown in Table 1, there are 24 classes of the news. The total number of the news and characters are $220,395$ and $80,046,457$ respectively. The average number of characters per news is 363.2 and the average number of sentences per news is 29.1. The statistic of SB-tree is shown in Table 1. The total size of original texts are about $152.7MB$, which is far more than previous related researches[19, 4, 12]. The total size of SB-tree is about $1.8GB$ and the obtained average ratio value(the space of SB-tree needed with respect to original text size) is 11.9.

In this paper, the maximum length of extracted terms is 20 and we filter out the terms whose term frequency are less than 10. The total number of candidate significant terms and significant terms are $1,157,361$ and $908,413$ respectively. The length distribution of terms is shown in Figure 2. Note that the number of significant terms whose length is 3 is the largest set of terms. This observation is not coincided with [10], which states the bi-grams, whose length is 2, is the largest set in modern Chinese words. To show the run-

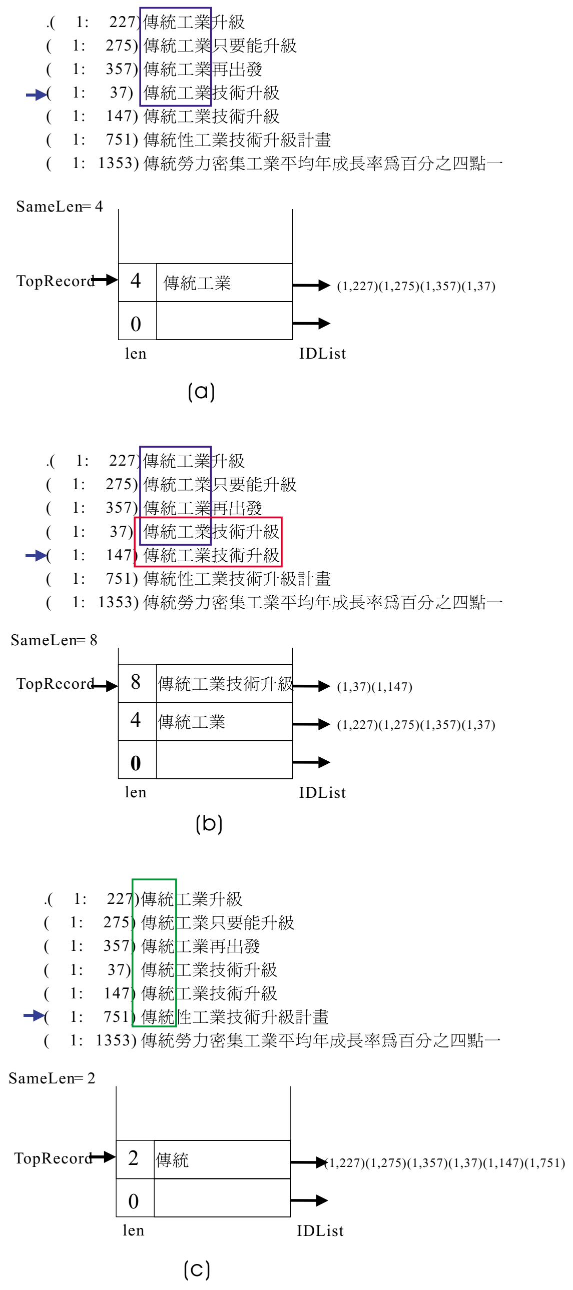


Figure 5: Scanning SB-Tree

Term	tf	Valid
.	.	.
.	.	.
.	.	.
傳統工業技術升級	2	TRUE
傳統工業	5	TRUE
傳統	7	TRUE
會指出	2	FALSE
會	4	TRUE
業自動化	2	FALSE
"	.	.
.	.	.
.	.	.
.	.	.
.	.	.
.	.	.
.	.	.
.	.	.

(a)

Term	tf	Valid
.	.	.
工統傳	5	FALSE
⋮	⋮	⋮
化動自業工	2	TRUE
化動自	7	TRUE
化	8	TRUE
⋮	⋮	⋮
出指會建經	2	TRUE
出	3	TRUE
⋮	⋮	⋮
級升術技業工統傳	2	TRUE
⋮	⋮	⋮
統傳	7	TRUE
⋮	⋮	⋮
會	4	TRUE
業工統傳	5	TRUE
.	.	.
.	.	.

(b)

Figure 6: (a) Right-Justify table (b) Left-Justify table

ning time of our approach, we run our program on a PC with Pentium II 233 CPU, 128MB RAM and IBM IDE Harddisk. The total of time for building and scanning of SB-tree are about 138 hours and 142 hours respectively. Note that the time for building and scanning of Reverse SB-tree are similar to that of SB-tree and, therefore, are not listed for simplicity.

With such a large scale and a long time period of training corpus, we can extract very specific and significant terms, such as "大陸地區人民在台灣地區定居或居留許可辦法"(the rules of the permission for the people in mainland China to settle down or to take up residence in Taiwan), and such terms are very helpful to improve Chinese text classification accuracy [17, 15, 14, 16]. With the distribution of significant terms according to different scale of time interval, we can have information not only term frequency but also historical event detection. As shown in Table 3, for example, "伊拉克駐聯合國大使安巴里" and "伊拉克駐聯合國大使哈姆東" represent two persons," 安巴里" and "哈姆東", that had been the ambassador of Iraq in the United Nations. Note that the value in the column of "Y1991" means the term frequency of that term in 1991. We can conjecture that "哈姆東" substitute for "安巴里" to be the ambassadors of Iraq in the United Nations in 1992. There is an similar observation that we can figure out that "許水德" take the place of "宋楚瑜" to be the head secretary of the central committee of "國民黨" in 1993. Furthermore, we can detect the event precisely with fine-grained time interval. As shown in Table 4, for example, we can guess that "ＡＰＥＣ高峰會" was hold at November in 1993, and "蔣公一百零六歲" was at October in 1991, and "一九九一台北金馬國際影展" was hold at October or November in 1991, "八十二學年度大學聯招試務" and "第十六屆威廉瓊斯杯國際籃球邀請賽" were

Term	Y1991	Y1992	Y1993
伊拉克駐聯合國大使安巴里	39	22	0
伊拉克駐聯合國大使哈姆東	0	2	22
中國國民黨中央委員會祕書長宋楚瑜	200	285	26
中國國民黨中央委員會祕書長許水德	0	0	253

Table 3: Event detection

hold at July in 1993. Note that we can observe various objects according to the statistic of terms with different scale of time interval and the attributes of terms associated with original texts in the database. What we have done is somewhat a fundamental work of text data warehouse.

5 Conclusion and Future Research

In this paper, we develop a scalable approach via String B-tree(SB-tree) to identify terms from large amount of text data, which does not use a dictionary. Note that SB-tree can grow incrementally, is I/O efficient and is scalable to store large amount of data. Our term extraction process consists of four steps : (i) texts information database, (ii) SB-tree construction, (iii) candidate significant term extraction and (iv) significant term validation. Our experiment uses three year news, 1991/1/1-1993/12/31, from Central News Agency(CNA) as source texts to extract significant terms. The total number of the news and characters are $220,395$ and $80,046,457$ respectively. The total size of original texts are about $152.7MB$, which is far more than previous related researches. The maximum length of extracted terms is 20 and the number of significant terms extracted are $908,413$. With the training corpus from such a long time period, we not only have robust statistic of terms, i.e. term frequency and document frequency, but also can detect some events via the distribution of significant terms according to different scale of time interval. Note that we can observe various objects according to the statistic of significant terms with different scale of time interval, e.g. weekly, monthly and yearly, and the other attributes of terms associated with the original texts in the database if necessary. What we have done is somewhat a fundamental work of text data warehouse.

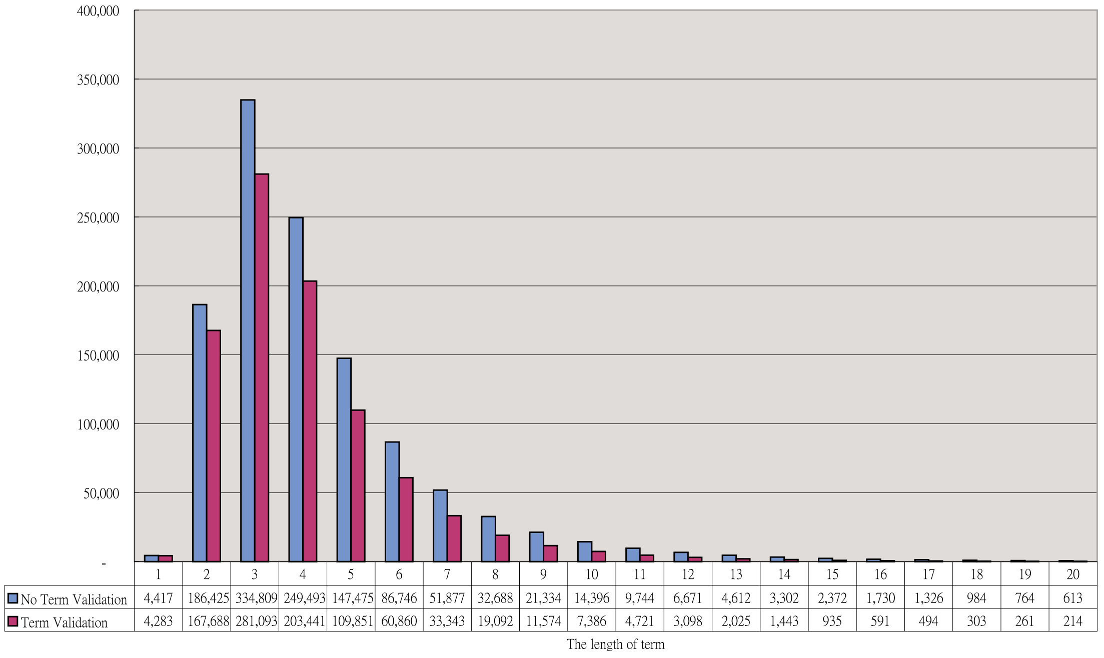
References

[1] Central news agency. http://www.cna.com.tw/index.html.

[2] Chun-Liang Chen, Bo-Ren Bai, Lee-Feng Chien, and Lin-Shan Lee. CPAT-tree-based language models with an application for text verification in chi-

	NewsType	NewsCnt	CharCnt	SentCnt	Text size(KB)	SB-tree size(KB)	Ratio	Time of building SB tree	Time of scaning SB tree
1	ad	6,827	2,577,328	212,599	5033.8	60,504	12.0	02:28:10	01:12:09
2	af	1,234	355,272	26,023	693.9	7,952	11.5	00:06:46	00:12:44
3	dd	14,361	5,620,440	484,629	10977.4	133,212	12.1	08:02:34	03:15:48
4	df	2,840	1,078,566	83,260	2106.6	25,008	11.9	00:35:33	00:40:36
5	ed	24,361	9,032,388	707,166	17641.4	215,260	12.2	17:48:06	05:51:29
6	ef	9,872	3,431,820	252,956	6702.8	80,396	12.0	03:51:15	02:23:31
7	fd	8,014	2,579,848	193,272	5038.8	60,868	12.1	02:18:51	02:04:25
8	ff	1,762	537,279	34,874	1049.4	12,416	11.8	00:10:40	00:27:53
9	hd	7,760	2,930,638	243,596	5723.9	69,048	12.1	03:00:30	01:55:20
10	hf	2,835	1,108,674	84,914	2165.4	25,776	11.9	00:37:31	01:03:45
11	jd	15,372	5,377,119	451,385	10502.2	126,548	12.0	07:01:56	03:34:04
12	jf	6,991	2,272,156	173,429	4437.8	52,524	11.8	02:04:51	01:29:41
13	ld	11,554	3,866,604	351,700	7552.0	90,660	12.0	04:22:28	02:39:04
14	lf	5,938	1,954,531	161,638	3817.4	45,404	11.9	01:33:34	01:57:59
15	md	1,748	640,287	55,744	1250.6	14,924	11.9	00:13:51	01:02:33
16	mf	8,812	2,775,522	196,610	5420.9	64,560	11.9	02:44:45	02:02:02
17	pd	34,151	14,867,802	1,267,673	29038.7	359,200	12.4	39:47:10	30:48:40
18	pf	34,144	12,929,745	950,558	25253.4	308,116	12.2	36:55:44	55:41:28
19	rd	2,084	742,956	64,642	1451.1	17,236	11.9	00:17:23	02:12:14
20	rf	975	321,424	24,702	627.8	7,332	11.7	00:05:43	00:57:31
21	sd	3,391	671,875	54,606	1312.3	15,500	11.8	00:14:08	02:49:15
22	sf	5,456	893,375	65,193	1744.9	18,924	10.8	00:21:11	03:41:23
23	td	8,168	2,949,131	232,566	5760.0	69,028	12.0	03:07:37	07:02:47
24	tf	1,745	531,677	39,728	1038.4	12,244	11.8	00:10:45	01:52:21
		220,395	80,046,457	6,413,463	156,341 =152.7MB	1,892,640 =1.8GB	11.9	138:01:19	142:16:33

Table 1: The statistic of the news and SB-tree



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
🟦 No Term Validation	4,417	186,425	334,809	249,493	147,475	86,746	51,877	32,688	21,334	14,396	9,744	6,671	4,612	3,302	2,372	1,730	1,326	984	764	613
🟥 Term Validation	4,283	167,688	281,093	203,441	109,851	60,860	33,343	19,092	11,574	7,386	4,721	3,098	2,025	1,443	935	591	494	303	261	214

Table 2: The length distribution of terms

Term	Y1991	Y1992	Y1993
ＡＰＥＣ高峰會	0	0	11
蔣公一百零六歲	0	12	0
一九九一台北金馬國際影展	10	0	0
八十二學年度大學聯招試務	0	1	14
第十六屆威廉瓊斯杯國際籃球邀請賽	0	0	13

(a)

Term	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12
ＡＰＥＣ高峰會	0	0	0	0	1	0	1	1	1	1	6	0
蔣公一百零六歲	0	0	0	0	0	0	0	0	0	10	2	0
一九九一台北金馬國際影展	0	0	0	0	0	0	0	0	0	4	6	0
八十二學年度大學聯招試務	0	0	0	0	0	0	11	3	0	1	0	0
第十六屆威廉瓊斯杯國際籃球邀請賽	0	0	0	0	0	4	9	0	0	0	0	0

(b)

Table 4: (a)By year (b)By month

nese. In *Research on Computational Linguistics Conference(ROCLING XI)*, pages 189–203, 1998.

[3] Chun-Liang Chen and Lee-Feng Chien. PAT-tree-based online corpus collection and classification. In *The Fourth Internaltional Workshop on Information Retrieval with Asian Languages(IRAL'99)*, pages 78–82, 1999.

[4] Lee-Feng Chien. PAT-tree-based keyword extraction for chinese information retrieval. In *SIGIR'97*, pages 50–58, 1997.

[5] Lee-Feng Chien and Hsiao-Tieh Pu. Important issues on chinese information retrieval. In *Computation Linguistics and Chinese Language Processing*, pages 205–221, 1996.

[6] Paolo Ferragina and Roberto Grossi. An experimental study of sb-trees. In *ACM-SIAM symposium on Discrete Algorithms*, 1996.

[7] Paolo Ferragina and Roberto Grossi. The string b-tree : A new data structure for string search in external memory and its application. *Journal of ACM*, 46(2):236–280, March 1999.

[8] William B. Frakes and Rick Kazman. *Information Retrieval Data Structures & Algorithm*. Prentice Hall, Englewood Cliffs, New Jersey 07632, 1992.

[9] Sen-Yuan Huang, Yu-Ling Chou, and Ja-Chen Lin. Automatic classification for news written in chinese. *Computer Processing of Oriental Languages*, 12(2):143–159, 1998.

[10] K.L Kwok. Comparing representations in chinese information retrieval. In *SIGIR'97*, pages 33–41, 1997.

[11] C. Lin and H. Chen. An automatic indexing and neural network approach to concept retrieval and classification of multilingual(chinese-english) documents. In *http://ai.bpa.arizona.edu/papers/chinese93/chinese93.html*, 1994.

[12] Yih-Jeng Lin, Ming-Shing Yu, Shyh-Yang Hwang, and Ming-Jer Wu. A way to extract unknown words without dictionary from chinese corpus and its applications. In *Research on Computational Linguistics Conference(ROCLING XI)*, pages 217–226, 1998.

[13] Von-Wun Soo, Pey-Ching Yang, Shih-Huang Wu, and Shih-Yao Yang. A character-bases hierarchical information filtering scheme for chinese news filtering agents. In *Information Retrieval with Asian Languages(IRAL 1997)*, pages 96–107, 1997.

[14] Jyh-Jong Tsay and Jing-Doo Wang. Comparing classifiers for automatic chinese text categorization. In *1999 National Computer Symposium*, pages B–274–B–281, 1999.

[15] Jyh-Jong Tsay and Jing-Doo Wang. Term selection with distributional clustering for chinese text categorization using n-grams. In *Research on Computational Linguistics Conference XII*, pages 151–170, 1999.

[16] Jyh-Jong Tsay and Jing-Doo Wang. Improving automatic chinese text categorization by error correction. In *The Fifth Internaltional Workshop on Information Retrieval with Asian Languages(IRAL2000)*, pages 1–8, 2000.

[17] Jyh-Jong Tsay, Jing-Doo Wang, Chun-Fu Pai, and Ming-Kuen Tsay. Implementation and evaluation of scalable approaches for automatic chinese text categorization. In *The 13th Pacific Asia Conference on Language, Information and Computation*, pages 179–190, 1999.

[18] Yuen-Hsien Tseng. Fast keyword extraction of chinese document in a web environment. In *Information Retrieval with Asian Languages(IRAL 1997)*, pages 81–87, 1997.

[19] Yun-Yan Yang. A study of document auto-classification in mandarin chinese. In *Research on Computational Linguistics Conference(ROCLING VI)*, pages 217–233, 1993.