Automatic Clustering of Chinese Characters and Words

Chao-Huang Chang (張照煌) and Cheng-Der Chen (陳正德)
Advanced Technology Center (E000)
Computer & Communication Research Laboratories
Industrial Technology Research Institute
Chutung, Hsinchu 31015, Taiwan, R.O.C.
E-mail: changch@e0sun3.ccl.itri.org.tw

Abstract

Research on Chinese part-of-speech tagging has been very active recently. However, there are several problems the research must confront before a successful tagger can be realized. Among them are word definition, segmentation, lexicon, tag set, tagging guideline, and tagged corpora. We propose machine-clustered word classes as an alternative for part-of-speech to be used in class n-gram models. Chinese characters and words are automatically clustered into a predefined number of classes using a simulated annealing approach. The 1991 United Daily text corpus of approximately 10 million characters is used to collect the statistics of character and word collocation. We will show and discuss some preliminary experimental results, which are considered promising and interesting.

1 Introduction

Word n-gram models are useful in many NLP applications. However, they have a lot of parameters, which need huge training data to estimate and take very much memory and disk space to store. Thus, class n-gram models [1] have been proposed to reduce the number of parameters. One of well-known class n-gram models is the Tri-POS model [6], which defines word classes based on syntactic categories, i.e., parts-of-speech. It has been successfully used in many western languages [5]. However, Chinese language models using part-of-speech information have only a very limited success [4,11] due to the following difficulties encountered in Chinese part-of-speech tagging [2]:

1. To define an appropriate Chinese part-of-speech tag set [10,15,16];

2. To find a Chinese lexicon with complete part-of-speech informations;
3. To solve the word segmentation problems [3], e.g., word definition, unregistered words, and compounds;
4. For human to do Chinese part-of-speech tagging; the parts-of-speech of numerous words are either arguable or difficult to decide.
5. To find manually tagged Chinese corpora, counterparts of the Brown and LOB corpora in Chinese.

In the paper, machine-generated *disjoint word classes* are proposed as an alternative for parts-of-speech. The five problems listed above are solved at the same time.

1. Automatic clustering of Chinese words is used to generate the disjoint word classes. Thus, the size of the tag set and the definition of classes are decided automatically.
2. Each word in the lexicon is assigned to one of the machine-generated disjoint word classes.
3. Any raw output of a word segmentation program can be used to train the automatic clustering system. Words are defined as any segmented character strings.
4. There is no need for human to tag the disjoint word classes.
5. Any unsegmented, untagged text corpus can be used for training and/or testing.

The main concept is: Let machines do things in their own way. Chinese words, parts-of-speech are not well-defined concepts even for human, not to mention machines. We consider that this is why machines can not do word segmentation and part-of-speech tagging satisfactorily.

The other advantages include: (1) scalable: The number of classes (i.e., granularity) can be adjusted easily according to applications; (2) adaptable: The word classes can be retrained on corpora of different domains.

2 Automatic Clustering of Words

2.1 The Problem

Let

$T = w_1, w_2, ..., w_L$ be a text corpus with L words;

$V = v_1, v_2, ..., v_{NV}$ be the vocabulary composed of the NV distinct words in T;

$C = C_1, C_2, ..., C_{NC}$ be the set of classes, where NC is a predefined number of classes.

The word clustering problem can be formulated as follows:

Given V and C (with a fixed NC), find a class assignment ϕ from V to C which maximizes the estimated probability of T, $\hat{p}(T)$, according to a specific probabilistic language model.

For a bigram class model, find

$$\phi : V \to C$$

to maximize

$$\hat{p}(T) = \prod_{i=1}^{L} p(w_i|\phi(w_i))p(\phi(w_i)|\phi(w_{i-1})))$$

Alternatively, *perplexity* [7] or *average mutual information* [1] can be used as the characteristic value to optimize.

Perplexity, PP, is a well-known quality metric for language models in speech recognition. It is also called the average word branching factor of the model.

$$PP = \hat{p}(T)^{-\frac{1}{L}}$$

In a sense, the language model reduces the difficulty of recognition task from distinguishing NV words to distinguishing PP words. The perplexities PP for the word and class bigram models are:

$$PP = exp(-\frac{1}{L}\sum_{i=1}^{L} ln(p(w_i|w_{i-1})))$$

and

$$PP = exp(-\frac{1}{L}\sum_{i=1}^{L} ln(p(w_i|\phi(w_i))p(\phi(w_i)|\phi(w_{i-1}))))$$

respectively, where w_j is the j-th word in the text and $\phi(w_j)$ is the class that w_j is assigned to.

For class N-gram models with fixed NC, lower perplexity indicates better class assignment of the words. The word classification problem is thus defined: Find the class assignment of the words to *minimize* the perplexity of the training text.

2.2 Disjoint Word Classes

Linguistic objects in natural languages can be classified into four categories (Table 1): (I) linguistically defined, ambiguous; (II) linguistically defined, disjoint; (III) artificially defined, ambiguous; and (IV) artificially defined, disjoint.

	ambiguous	disjoint
linguistic	Chinese word, part-of-speech	Chinese character, English word
artificial	-	machine-generated word cluster, word equivalence class

Table 1: Taxonomy of Linguistic Objects

Most of Chinese NLP researchers have dealt with the problems from the linguistic point of view. We have been trying to identify Chinese words from text, to tag the part-of-speech for the words. However, these concepts (Chinese words, part-of-speech) are often poorly defined or highly ambiguous (Type I). The computer is not good at resolving ambiguity according to linguistic criteria. Thus, Type II objects are much easier to process than Type I objects. The concept of word classes has been proposed to reduce the number of parameters in statistical language models. Lee *et al.* [12] approximates Chinese word bigram by the idea of word-lattice-based character bigram, because Chinese characters are disjoint, i.e., unambiguous (Type II) while Chinese words are boundary ambiguous (Type I). We can not find examples of Type III objects. Kupiec[9] defined unambiguous word equivalence class (Type IV) based on possible tags of a word for the part-of-speech tagging problem.

We propose the following directions for Chinese class n-gram models:

1. Using disjoint classes instead of ambiguous (overlapping) classes;
2. Using artificial (machine-generated) classes rather than linguistically defined classes.

In other words, Type IV objects are preferred over Type-I and Type-II objects.

2.3 A Simulated Annealing Approach

The word classification problem can be considered as a combinatorial optimization problem to be solved with a simulated annealing algorithm. Jardino and Adda [7] used a simulated annealing approach to automatically classify words in a French corpus of 40,000 words and a German corpus of 100,000 words. A simulated annealing algorithm needs four components [8]:

(1) a specification of **configuration**, (2) a random **move generator** for rearrangements of the elements in a configuration, (3) a **cost function** or objective function to evaluate a configuration, (4) an **annealing schedule** that specifies time and duration to decrease the control parameter (or temperature).

For the word classification problem, the configuration is clearly the class assignment ϕ. The move generator is also straightforward – randomly choosing a word to be reassigned to a randomly chosen class. Perplexity can serve as the cost function to evaluate the quality of word classification[7]. The annealing schedule follows that of Metropolis algorithm. Thus, the clustering procedure is:

1. *Initialize*: Assign the words randomly to the predefined number of classes to have an initial configuration;
2. *Move*: Reassign a randomly selected word to a randomly selected class (Monte Carlo principle);

3. *Accept or Backtrack*: If the perplexity is changed within a controlled limit (decreases or increases within limit), the new configuration is accepted; otherwise, undo the reassignment (Metropolis algorithm, see below);

4. *Loop*: Iterate the above two steps until the perplexity converges.

Metropolis algorithm [7]: The original Monte Carlo optimization accepts a new configuration only if the perplexity decreases, suffers from the local minimum problem. Metropolis *et al.* (1953) proposed that a worse configuration can be accepted according to the control parameter *cp*. The new configuration is accepted if $exp(\Delta PP/cp)$ is greater than a random number between 0 and 1, where ΔPP is the difference of perplexities for two consecutive steps. *cp* is decreased logarithmically after a fixed number of iterations.

In the following two sections, we use similar simulated annealing techniques to automatically cluster Chinese characters and words in the 1991 United Daily corpus.

3 Clustering Chinese Characters

3.1 The Corpus and Character Bigrams

The statistics of Chinese character bigram is based on the 1991 UD corpus (1991ud) of approximately 10,000,000 characters. There are totally 5,403 character types: the 5401 commonly used characters in the Big-5 character set, a type for all 7,650 (13,051-5,401) other Chinese characters in Big-5, and another type for special symbols, such as punctuation marks and foreign characters (Arabics, English). There are 723,681 nonzero entries in the full 5403x5403 bigram and 9,529,107 ($= L$) occurrences of character types.

To keep the clustering experiments running within reasonable time, using the whole UD corpus for full character or word bigrams is not satisfactory. A smaller subcorpus, day7, containing one day of news, was extracted from the 1991 UD corpus. There are 147,976 nonzero entries in the full 5403x5403 bigram and 540,561 ($= L$) occurrences of character types.

3.2 Experimental Results: Clustering 100 Simple Characters

To illustrate the clustering process, the first 100 Big-5 characters are chosen as objects to classify (Table 2).

一乙丁七乃九了二人儿入八几刀刁力匕十卜又
三下丈上丫丸凡久么也乞于亡兀刃勺千叉口土
士夕大女子孑孓寸小尢尸山川工己已巳巾干廾
弋弓才丑丐不中丰丹之尹予云井互五亢仁什仃
仆仇仍今介仄元允內六兮公冗凶分切刈勻勾勿

Table 2: The 100 Characters to be Classified

The statistics for these 100 characters are extracted from the full 5403-character bigram for `1991ud`. That is, a 100x100 submatrix is extracted from the 5403x5403 matrix. This is an approximation of a text composed of these 100 characters. There are 1,968 nonzero entries in the 100x100 bigram and 144,261 ($= L$) occurrences of them.

The control parameter cp is initialized to 0.1 and divided by two after every 1,000 iterations.

First, we tried to cluster the 100 characters into 10 classes (see Figure 1 for the converging process). One of the classes, Class-0, is used to represent unknown characters and characters with zero frequency (i.e., never occurs in the training text). We observe that:

- Initial configuration: Class-0 contains the zero-frequency characters; all other characters are assigned to Class-1. This practice follows the suggestion made by Jardino and Adda [7]. Initial perplexity is 32.950.
- Perplexity decreases quickly at first several runs (each run corresponds to a fixed cp and 1,000 iterations), from 32.950, 22.052, 19.396, to 18.608 after the third run ($cp = 0.025$).
- If the classical Monte Carlo method is used – a new configuration is accepted only if the perplexity decreases, it will get stuck at a much worse local minimum.
- After only 12 runs ($cp = 4.88 \cdot 10^{-5}$), the perplexity converges to its final value 17.719.
- The clustering result is very encouraging. The final configuration is shown in Table 3.
  - Class-3 consists of six digits.

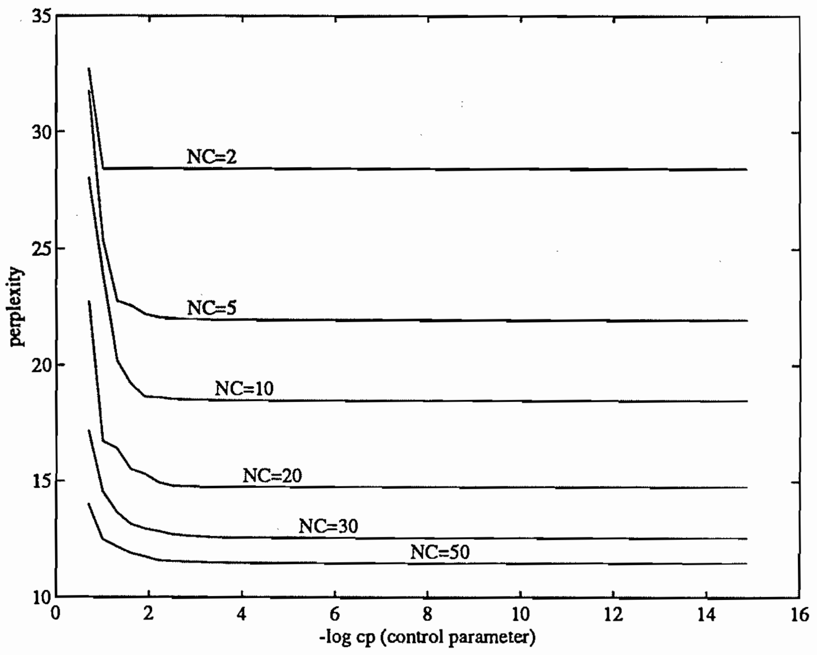


Figure 1: Clustering 100 Chinese characters: different NCs

- Class-9 consists of two characters 十 and 千, which are quite similar syntactically and semantically.
- Class-1 contains several first characters of measure words, 丈, 寸, 元, 公, 分, etc. Note that the classification is based on character bigram, so 公can be considered as a part of measures such as 公分.
- Class-0 is artificially made for unseen characters.
- Class-6 has a large family; somehow, members in the meaningful groups (1) 上中下 (2) 大中小 (3) 女子 (4) 乙丁 (5) 山川 are together.
- The other classes are less meaningful.
- Two digits 一, 九 are not assigned to Class-3.

Figure 1 also compares the converging processes for different values of NC and Table 4 shows the converged perplexities. As expected, the perplexity decreases when the number of classes increases. The perplexity of the character bigram model, i.e., $NC = 100$, is 11.250. If we

Class	Members
Class-0 :	儿刁匕兀尤尸巳廾弋仃仆冗刈
Class-1 :	丈夕寸弓丰元公分匀
Class-2 :	一孑什仄
Class-3 :	七二八三五六
Class-4 :	人又也乞刃叉己井互仍
Class-5 :	了巾之凶
Class-6 :	乙丁乃入几刀力卜下上丫丸凡久亡勺土大女子 小山川工已干中丹予云亢仁仇今介内兮
Class-7 :	九孓切勿
Class-8 :	么于口士才丑丐不尹允勾
Class-9 :	十千

Table 3: 10 Output Character Clusters for the 100 Characters

classify the 100 characters into 30 (NC) classes, the perplexity is just 12.556. There is not much difference. This shows the feasibility of class n-gram models: reducing the number of parameters significantly, having competitive performance, requiring much less resources, and robust.

NC	Perplexity
1	32.959
2	28.442
5	21.947
10	17.719
20	14.760
30	12.556
50	11.472
100	11.250

Table 4: Perplexities for different NCs

3.3 Experimental Results: Clustering 5401 Chinese Characters

Running the full 5401-character bigram for a large corpus ($L > 1,000,000$) takes huge amount of time. Note that Jardino and Adda [7] used much smaller corpora (40,000 and 100,000 words, respectively). However, the algorithm has a time complexity proportional to L. They reported that it took 7 hours on a 486-33 PC to classify 14,000 words into 120 classes using a 75,000-word training set. We have designed an incremental version of the system which is much faster than the original version which recomputes all probabilities in each trial. In our experience, it took 20.06 CPU hours on a DEC 3000/500 AXP workstation to classify 5403 characters into 200 classes (with 50,000 trials in each of 64 iterations) using 540,000-character **day7** corpus.

We have conducted three experiments on the **day7** corpus.

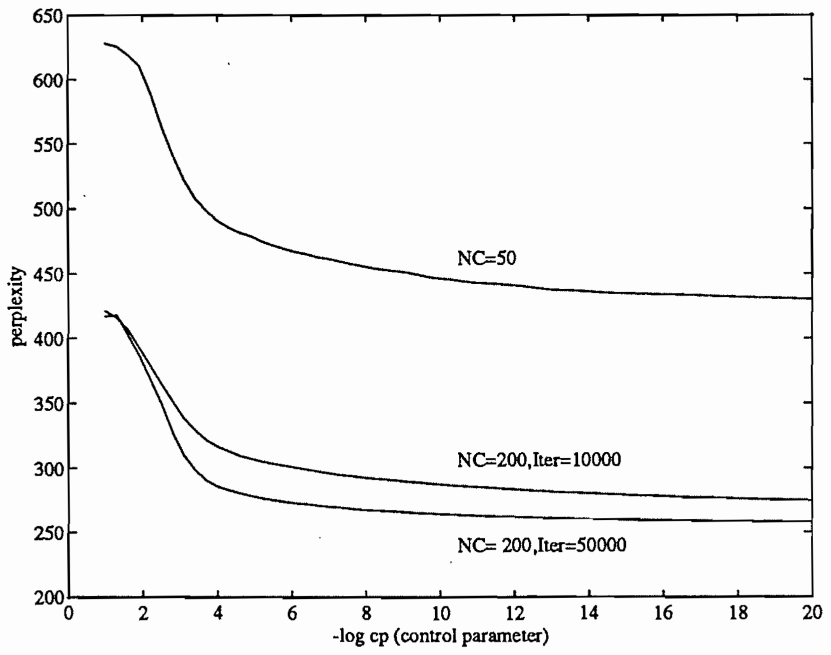


Figure 2: Clustering 5,401 Chinese characters: the converging process

Experiment 1:

- Number of Classes: 50
- Initial cp: 0.1
- Initial perplexity: 675
- Characters with frequency less than 3 are assigned to Class-0
- Number of iterations in each run: 10000
- The perplexity converging process (Figure 2): 675, 628, 626, 619, 610, 589, 562, 541, 522, ..., 430 (*cp* too small, less than 10^{-20})
- The final configuration:

- Class-0 contains 2,293 characters.
- Class-1 contains human-related characters, 人于大女予仁及夫巴引比火父王仔冊卉古史尼母田全, ...
- Class-14: 一今每赤兩昔昨晝翌規逐傍幾瑕睦逾鐮
- Class-18 contains: (numbers) 七九二八三五六四彷零綽銖嘰墩餘.
- Class-20 : 凡允尤乍另因如汎而但否坊迄呵咋俟染倘唯捨毫嗯搽寥漂誰縫譬. Many of them are conjunctions.
- Class-21 consists of 中什他它全各夷她舟你我杜牠那妳拒信俄帝怎某洪美耐范苗英哥哪栽泰爹翁您這敞粵豪緯薰蟹驅. Almost all pronouns are included in this class.
- Class-22 and Class-24 are composed of measure words among others:
  (Class-22) 元分斗冊呎坪姐秒哩畝措楞歲號蝶噸點;
  (Class-24) 天日月乎旦兆吋年次佰岸枚肢頁個株般隻晚棵番週塊漸磅篇趟艘顆曜籃藩蹴
- Class-26 contains several function words, 又也才仍勿只亦刪即均希更並尚彼則卻既皆盼衹若俾豈焉都竟就廁猶滲頗躺還雖
- Class-28 contains several surnames: 于戈王古吉安江余吳呂宋李貝阮林邱侯姜柯柳胡韋唐孫徐桂張梁郭陳麥傅游雲馮黃楊鄒廖趙劉潘蔡鄭黎墨燕盧蕭薛鍾簡魏羅, among others.
- Class-37 consists of orders of ten, 十千卅廿仟百彿巷第萬億蝴
- Somehow, we consider that 50 is not enough for character class discrimination.

Experiments 2 and 3:

- Number of Classes: 200
- Initial cp: 0.1
- Initial perplexity: 675
- Characters with frequency less than 3 are assigned to Class-0
- Number of iterations in each run: 10000 and 50000
- The perplexity converged to 274 and 258, respectively (see Figure 2).

1	山川井卉古丞后州臣村谷里坡河城娃拷泉洛皇厝哥埔島...
3	乙丁丹匹丙仙尼玉瓜甲仲仰吉宇汝亨佛佐君廷...
11	次批屈套座陣貫番項種樣椿篇趟輛輩
19	元呎辰坪畝歲銖噸鎊
23	巴冬寺春玻秋胡迪夏徐桂秦細荷陰紫詠嘉蜜擠澎踴蟬羅籐蘇
25	于王丘朱江汎艾余吳呂吟巫李沈狄貝邢阮卑庚林邵邱侯姜柯柳洪范郁韋倪唐孫袁娼寇張曹梁莫莊貪郭陳麥傅馮黃楊鄒廖蒼蜿赫趙劉潘蔣蔡褒鄭黎儒墾盧穆蕭賴鮑薛謝魏譚龔蠶
26	天日晚溢
51	尺斤司室頃寓債
52	士員
57	屯市禾京省郊莞橙縣韓
63	予於握鄰
69	下上濤
70	倆們搔
72	午月周巷秒週
73	各專殖農漁闔
78	千仟百佰炫
81	北竹西東股南園基魁糖轄曜壢藩
87	七九八
94	台左屏桃焊菩臺閩緬墨
106	十卅廿第
114	兆萬億
124	每那某哪這
128	才只皆祇都就僅還
135	他它她你我牠妳茨您渥嗯誰懊蟹
138	年屆
139	今去昔拜昨翌傍
140	一兩幾
141	向在迄披拘赴從猜被喘趁跟憑
144	令把使垂堵逢替給雇僱對蓋讓
160	名位段隻梯瓶舶啼磅
173	乃凡尤而但倘俾惟譬
189	二三五六四零餘
190	又也仍亦均並尚卻既若豈竟猶滲
196	至初迎近牲剩逾厲嚣

Table 5: Part of Output Character Clusters

- See Table 5 for part of the final configuration with $Iter = 50000$.

  The listed classes have obvious meanings, the others are less clear. However, we observe that function-word and content-word characters are usually grouped separately.

Test Set Perplexity

To evaluate the performance of classification, we use another subcorpus day5 (544,606 characters) to compute a test set perplexity. For character clustering, smoothing is not necessary since the character set is mapped to a fixed set of 5403 character types. Table 6 shows the test set perplexities for different NC and number of trials per iteration (Iter). In general, classifications with higher NC have lower test set perplexity because the character set is closed. Clustering with higher Iter also reduces test set perplexity but with limit.

NC	Iter	Train PP	Test PP
50	10000	430.266	461.355
200	10000	274.916	290.208
200	50000	258.124	274.470

Table 6: Test Set Perplexities

4 Clustering Chinese Words

4.1 The Corpus and Word Bigrams

The statistics of Chinese word bigram is based on the above-mentioned (day7) corpus. The corpus is segmented automatically into clauses, then into words by our Viterbi-based word identification program VSG. There are totally 42,537 clauses, 355,347 ($= L$) words (189,838 1-character, 150,267 2-character, 10,783 3-character, 4,460 4-character), belonging to 23,977 word types (3,377 1-character, 16,004 2-character, 2,461 3-character, 2,135 4-character). There are 203,304 nonzero entries in the full 23,977x23,977 bigram, which is stored in compressed form.

4.2 Experimental Results: Clustering Chinese words

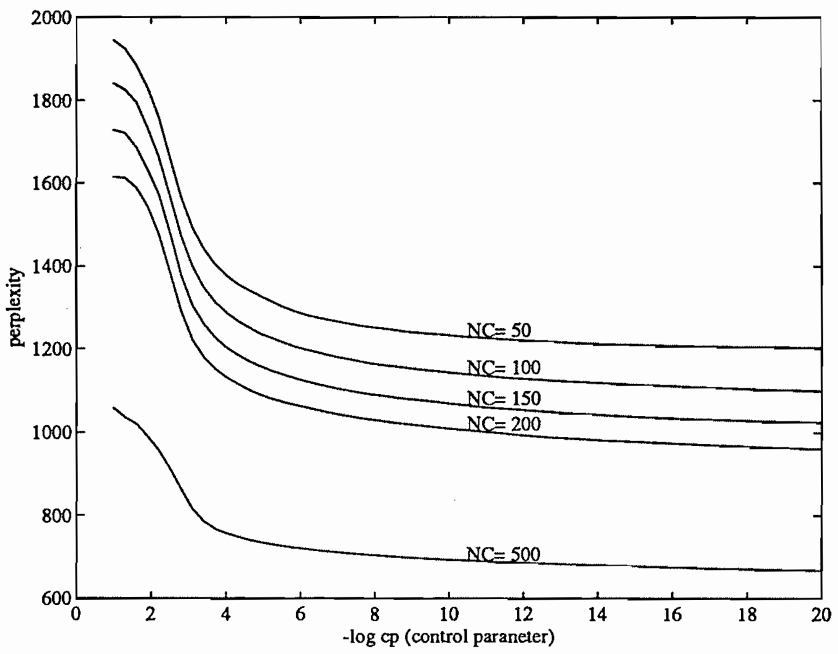


Figure 3: Clustering Chinese words: the converging process

Initial Configuration

Words with frequency less than m (default value: 5) are assigned to Class-0, the unseen word class [7]. Punctuation marks are assigned to a special class Class-1. 1–4 character number words are assigned to Classes 2–5, respectively. (Note that the numbers are composed by lexical rules in our word segmentation program VSG.) All other words are assigned to Class-6. Initial perplexity is 2,048.

Word Clustering: Changing NC and Iter

Numbers of experiments have conducted on the day7 corpus with different parameters. The following table shows the converged training set perplexity.

As expected, classifications with higher NC or higher Iter have lower training set perplexities. However, it is not the case for test set perplexity. Figure 3 shows the perplexity converging

NC	*10,000 iter.*	*20,000 iter.*	*50,000 iter.*
50	1305	1234	1203
100	1212	1140	1099
200	1068	1019	960
500	723	699	667

Table 7: Training Set Perplexities

processes for $iter = 50000$.

Testing Set Perplexities

For the problem of unseen words and bigrams, we adopt a similar linear smoothing scheme to that of Jardino and Adda [7]. (For details, see the original paper.) The interpolation parameters α and β are set to $1 - 10^{-5}$ and 0.1, respectively.

Four subcorpora are used: `day7` for training, `day5`, `day8`, `day9` for testing. Simple statistics are summarized in Table 8.

corpus	#clauses	#vocabulary	#words
day7	42,539	23,977	355,347
day5	44,334	24,706	360,464
day8	27,946	18,948	232,818
day9	26,579	19,111	221,105

Table 8: Four Subcorpora: `day5`, `day7`, `day8`, `day9`

The test set perplexities are summarized in Table 9 (for $Iter = 50,000$).

It appears that the most appropriate number of classes is about 150 to 200 for the size of training corpus. The clustering with $NC = 500$ is apparently overtrained (Figure 4).

Word Classification Results

As we all know, the smallest meaningful unit in Chinese is words rather than characters. The classification results for words are also more meaningful, if the clustering is well trained. When

NC	day5	day8	day9
50	1543	1548	1489
100	1491	1495	1437
150	1482	1487	1427
200	1478	1489	1436
500	1655	1637	1594

Table 9: Test Set Perplexities: day5, day8, day9

we set Iter to 10,000, the result did not have clear meaning. However. we now set Iter to 50,000, the classification results are encouraging.

For $NC = 200$, 15,900 words are assigned to the six special classes ($15070, 5, 17, 131, 266$, and 411 words, respectively). The other words are assigned to the other 194 classes, approximately according to part-of-speech. Part of the output clusters are shown in Table 10. The following are some observations:

- Title nouns: Class-7
- Place nouns: Class-10 (counties, cities, towns), Class-25 (nations, capitals, etc.), Class-79 (organizations)
- Time nouns: Class-12 (seasons), Class-44 (weekdays), Class-196
- Personal Names: Class-14, Class-42 (including some foreign names)
- Common Nouns: Class-6, Class-9, Class-11, Class-13, Class-27, Class-33
- Verbs: Class-8, Class-22 (是-related), Class-37 (有-related), Class-38 (count-related), Class-45 (1-character verbs)
- Adverbs: Class-15
- Prepositions: Class-16, Class-26,
- Adjectives: Class-17
- Conjunctions: Class-23, Class-29, Class-34
- Modals: Class-67
- Number nouns: Class-2, Class-3, Class-4, Class-5, Class-91
- Measure nouns: Class-36, Class-70, Class-74

6	中華文化/內心/公務/巴勒斯坦/心靈/文教/水箱/人力/大自然/...
7	分析師/民航局/主任/主委/主持人/主席/召集人/次長/局長/助理/巡官/抱/委任/拜訪/指認/ 前任/秋山/紀/首相/負責人/祕書/祕書長/秘書/副處長/強暴/組長/理事長/ 部長/處長/發言人/ 隊長/會長/董事長/署長/監委/總書記/總裁/總經理/鎮民/邊界
8	分贈/大戰/干預/代表會/住戶/似/局勢/攻擊/供應/花費/指示/相對/修訂/...
9	心意/人/人選/老婦/見面/角落/例/門窗/巷口/冒險/故鄉/時效性/...
10	中衛/斗六/三重/大甲/白河/台中/台北/台南/宜蘭/花蓮/東勢/板橋/恆春/ 南投/南京/苗栗/桃園/埔里/ 草屯/馬公/高雄/貢寮/通霄/雲林/新竹/嘉義/彰化/龜山/豐原/蘭陽/觀音
11	份子/缸/時/娘娘/創刊/學期
12	乙/末/白/光華/冬/出任/辰/協/帕/奈/昇/娃/冠/秋/美/春/胖/英/ ...
13	公職/勾結/手法/手續/支票/方針/比重/人數/上車/生動/成長率/收入/...
14	布/弗/主/古/宋/希/余/克拉/卓/周/坦/姜/姚/恆/柳/倪/范/夏/ 徐/特/莊/郭/傅/游/渥/湯/程/馮/黃/翡翠/潘/謝/韓/蘇/灌
15	引人/十分/大為/大幅/必經/有何/伺機/呈/具/受/招/欣賞/派人/...
16	不惜/大年/已於/去年度/自/名列/於/原定/將於/對付/糖果/...
17	大/可怕/多/多人/年長/形/看好/高點/淡/煙花/僵持/適合/遲/舉/懷念/轟轟烈烈
22	不至於/才是/正是/何況/係/是/是有/將是/就是/顯得
23	不但/上將/老是/似乎/並/並且/究竟/固然/始終/便/則/皆/既/根本/殊/送完/從來/猶/確/還將/雖
25	中美/中華民國/太平洋/巴國/.../日本/上海/本市/本地/本國/白宮/全美/全球/ 加拿大/加國/外地/米蘭/多倫多/我國/東方/東亞/東京/波士頓/帝國/科威特/ 美國/政大/海地/國內/國民黨/麻州/華盛頓/菲律賓/越南/新加坡/新華社/ 詹氏/歐美/澎湖/廣大/廣西/德州/德國/墾丁/澳洲/雙子星/羅馬尼亞/...
26	引用/大肆/充當/加入/向/在於/存入/呈報/享譽/往/委託/返回/按/前往/洽/查出/致函/留在/...
27	中餐/午餐/文章/日子/水果/牙刷/人潮/口袋/口號/生命/交往/仲裁/...
29	不料/尤其/凡是/也就是/未料/由於/以防/另一方面/外傳/而且/而後/至於/但/但是/因此/否則/並稱/例如/...
33	夫妻/一則/女子/母雞/目前為止/半/外匯存底/有心/年底/攻勢/車子/車主/典型/男友/男人/男子/...
34	尤其是/一旦/乃是/且/以及/以免/以使/以便/以致/有如/而/而是/自從/合計/如/直至/直到/卻因/俾/致使/...
36	寸/句/外國語/次/個/套/埠/條/塊/層樓/輛/艘
37	不無/不敷/充滿/有/自成/均有/具有/沒/祇有/倍感/起出/將有/創下/處於/連續/曾有/無/...
38	.../共有/共計/多達/法例/珍禽/約/降至/高出/高溫/高達/售/款項/超過/達/逾/...
42	于/伊/朱/吳/呂/孟/林/林森/威廉/約翰/洪/柴/馬/郝/梅傑/寇/現任/萊/鼎/維/赫/蔣/蔡/黎/穆/鮑/羅/魏/籣
44	今天/日前/凡/早年/次日/即日/明天/周四/初一/前天/昨/昨天/昨日/除夕/ 週五/週一/當日/...
45	切/引/丟/住/坐/忘記/扮/改/判/走/刺/忽/花錢/阻礙/指控/限/乘/哭/...
67	不必/不可能/不能/不得不/勿/一再/一度/已經/必須/本著/未/未曾/可/可以/可望/有沒有/並未/沒有/...
70	公斤/分鐘/斤/月分/人次/千年/小時/世紀/年/年中/年代/年後/頁/餘年/擊敗
74	元/公分/公升/公尺/公克/公里/公頃/公噸/二元/平方公尺/立方公尺/冊/...
79	公司/公所/公會/出口區/加油站/局/周轉/值/計程車/商/酒樓/教育局/會館/誠/署/戲院/餐廳/黨部
91	千億/平方/百億/尾/巷/億/數百萬/餘/點/厘
196	中午/午/午夜/升/一時/下午/四時/至/年初/初/凌晨/耗資/起至/晚上/晚間/傍晚/墩

Table 10: Part of Output Word Clusters

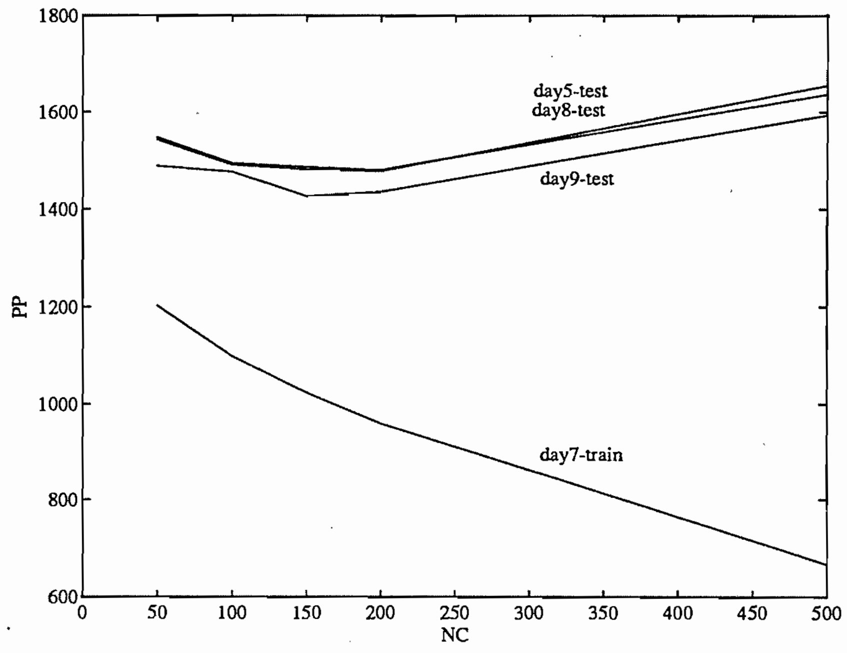


Figure 4: Training vs. Test Set Perplexity

- Equi-length words are usually grouped together. For example, one-character verbs, two-character verbs, one-character nouns, two-character nouns are grouped separately.

4.3 How to Use the Classification Results

The classification results of words (or characters) can be used in language models for speech recognition or OCR postprocessing. The class-ids for the words can be stored in the system dictionary. Words in a new input sentence are just automatically mapped to the classes through dictionary look-up. Thus, a class n-gram language model can be easily built up based on the machine-generated classes. As we mentioned, the number of classes can be adjusted according to the size of training corpus and application needs. As a common rule of thumb, the size of training data should be at least 10 times the number of parameters. Thus, if we have a corpus of size L for an N-gram model, the appropriate number of classes NC can be computed using the equation:

$$L = 10 \cdot NC^N$$

For example, if bigram ($N = 2$) models are used, NC is 100 for $L = 100,000$ and 1000 for $L = 10,000,000$.

For evaluating the performance of such language models, the test set perplexity can be used. In a sense, the language model using the day7-trained character clustering reduces the difficulty of recognition task from 5403 (character types) to 461 ($NC = 50$) and 274 ($NC = 200$). Similarly, the day7-trained word clustering reduces task difficulty from 24,706 to 1,543 ($NC = 50$) and 1,478 ($NC = 200$).

5 Other Approaches for Automatic Word Clustering

5.1 Brown *et al.* (1992)

Brown *et al.* [1] presented several statistical algorithms for automatic word classification. They use the average mutual information $I(C_i, C_j)$ as the characteristic value to maximize for a class bigram model:

$$I(C_i, C_j) = \sum_{C_i C_j} P(C_i C_j) \log \frac{P(C_j | C_i)}{P(C_j)}$$

They proposed two word clustering algorithms [1]:

Greedy-style Merging (1) Assign each word to a unique class and compute $I(C_i, C_j)$; (2) Merge two classes if the loss in $I(C_i, C_j)$ is least; (3) C classes remains after $V - C$ times of merging; (4) For each word in vocabulary, move it to a class to maximize the average mutual information. The merging steps produce a binary tree according to statistical similarity. However, they reported that this algorithm is not practical for a vocabulary with more than 5,000 words.

Add-One Merging For a larger vocabulary, (1) Assign each of the NC most frequently used words to a unique class; (2) Assign the next unclassified word with largest frequency to a new class $C_{(NC+1)}$, and merge two classes if the loss in $I(C_i, C_j)$ is least; (3) After $V - C$ steps, the NV words in the vocabulary are assigned to NC classes. A 260,741-word vocabulary had been classified into 1,000 classes this way.

Using a 1001-word window and the concept of semantic stickiness, they had classified English words semantically and had interesting results [1].

5.2 Ney and Essen (1991)

Ney and Essen [13] proposed a decision-directed, iterative unsupervised learning procedure: (1) Choose an initial mapping $\phi : V_i \rightarrow C_j = \phi(V_i)$ (2) Update the bigram and word counts $N(C_jW)$ and $N(C_j)$; (3) Compute the probability estimates $P(W|C_j) = N(C_jW)/N(C_j)$; (4) Find the optimal class $\phi(V_i)$ for each predecessor word V_i

$$\phi(V_i) = argmax_{C_j} \sum_{W} N(V_iW) \log P(W|C_j)$$

A German corpus of 95,671 words ($NV = 14080$) and a English corpus of 1,157,260 words ($NV = 49615$) have been classified into 128 classes this way.

5.3 Schutze (1993)

Schutze [14] proposed the idea of *category space.* A category of each word is represented by a vector in a multidimensional real-valued space. The category space is built by collecting various word distributional information. Four bigram matrices are built with distances -2, -1, 1, 2. A sparse matrix algorithm SVDPACK is then used to compute fifteen singular values for each word. A word is represented as a 15-dimensional vector in the category space. Close neighbors in the space are grouped into a word class (part-of-speech). A linear-time clustering algorithm called Buckshot was used to cluster the category space. The experiment was conducted on a 5000-word vocabulary. 278 high-frequency closed-class words such as prepositions are assigned distinct classes. The other 4,722 words were clustered into 222 classes. A second singular value decomposition was then performed on 22,771 words from New York Times based on the resulting 500 classes. Finally, a second Buckshot was used to classify the words into 200 output clusters.

6 Concluding Remarks

We have proposed using machine-generated disjoint word classes as an alternative for the popular word class – part-of-speech in Chinese class n-gram models. A simulated annealing approach is used to cluster Chinese characters and words into a predefined number of classes. Encouraging and interesting experiment results on the 1991 UD corpus have been shown and discussed. Future works include (1) more experiments on word clustering with different parameters; (2) studying more efficient algorithm for simulated annealing; (3) using windows to study semantic clustering; (4) applying to language models for speech recognition or OCR; and (5) studying and applying different clustering approaches.

Acknowledgements

This paper is a partial result of the project no. 37H2100 conducted by the ITRI under sponsorship of the Minister of Economic Affairs, R.O.C.

References

[1] P.F. Brown, V.J. Della Pietra, P.V. de Souza, J.C. Lai, and R.L. Mercer. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479, 1992.

[2] C.-H. Chang and C.-D. Chen. HMM-based part-of-speech tagging for Chinese corpora. In *ACL-93: Workshop on Very Large Corpora*, Ohio, USA, June 1993.

[3] K.J. Chen and S.-H. Liu. Word identification for Mandarin Chinese sentences. In *Proc. COLING-92*, Nantes, France, 1992.

[4] T.-H. Chiang, J.-S. Chang, M.-Y. Lin, and K.-Y. Su. Statistical models for word segmentation and unknown word resolution. In *Proc. of ROCLING V*, pages 121–146, Taipei, Taiwan, September 1992.

[5] K. Church. A stochastic parts program and noun phrase parser for unresticted text. In *Proc. of ICASSP-89*, pages 695–698, Glasgow, Scotland, 1989.

[6] A. Derouault and B. Merialdo. Natural language modeling for phoneme-to-text transcription. *IEEE Trans. PAMI*, 8(5):742–749, 1986.

[7] M. Jardino and G. Adda. Automatic word classification using simulated annealing. In *Proc. of ICASSP-93*, pages II:41–44, Minneapolis, Minnesota, USA, 1993.

[8] S. Kirkpatrick, C.D. Gelatt, Jr., and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[9] J. Kupiec. Robust part-of-speech tagging using a hidden Markov model. *Computer Speech and Language*, 6:225–242, 1992.

[10] H.-J. Lee and C.-H. Chang Chien. A Markov language model in handwritten Chinese text recognition. In *Proc. of Workshop on Corpus-based Researches and Techniques for Natural Language Processing*, Taipei, Taiwan, September 1992.

[11] H.-J. Lee, C.-H. Dung, F.-M. Lai, and C.-H. Chang Chien. Applications of Markov language models. In *Proc. of Workshop on Advanced Information Systems*, Hsinchu, Taiwan, May 1993.

[12] L.-S. Lee et al. Golden Mandarin (II) - an improved single-chip real-time Mandarin dictation machine for Chinese language with very large vocabulary. In *Proc. of ICASSP-93*, pages II:503–506, April 1993.

[13] H. Ney and U. Essen. On smoothing techniques for bigram-based natural language modelling. In *Proc. of ICASSP-91*, pages 825–828, Toronto, September 1991.

[14] Hinrich Schutze. Part-of-speech induction from scratch. In *Proc. of ACL-93*, pages 251–258, Columbus, Ohio, USA, June 1993.

[15] K.-Y. Su, Y.-L. Hsu, and C. Saillard. Constructing a phrase structure grammar by incorporating linguistic knowledge and statistical log-likelihood ratio. In *Proc. of ROCLING IV*, pages 257–275, Pingtung, Taiwan, 1991.

[16] M.S. Sun, T.B.Y. Lai, S.C. Lun, and C.F. Sun. The design of a tagset for Chinese word segmentation. In *First International Conference on Chinese Linguistics*, Singapore, June 1992.

A STORAGE REDUCTION METHOD FOR CORPUS-BASED LANGUAGE MODELS

Hsin-Hsi Chen and Yue-Shi Lee

Department of Computer Science and Information Engineering

National Taiwan University

Taipei, Taiwan, R.O.C.

Abstract

There are many progresses in corpus-based language models recently. However, the storage issue is still one of the major problems in practical applications. This is because the size of the training tables is in direct proportion to the parameters of the language models and the number of the parameters is in direct proportion to the power of these language models. In this paper, we will propose a storage reduction method to solve the problem that results from the large training tables. We use mathematical functions to simulate the distribution of the frequency value of the pairs in the training tables. For the good approximation, the pairs are grouping by their frequency. The experimental results show that although there is a little error rate introduced by the curve function, this scheme is still satisfactory because it performs the closed performance and no extra storage is required in pure curve-fitting model. Besides, we also propose a neural network approach to deal with the pairs classification which is a problem for all class-based approaches. The experimental results show that the neural network approach is suitable to deal with this problem in our storage reduction method.

1. Introduction

There are many progresses in corpus-based language models recently. However, the storage issue is still one of the major problems. The conventional corpus-based language models record the statistical information extracted from the corpora in the training tables. The size of the training tables is in direct proportion to the parameters of the language models and the number of the parameters is in direct proportion to the power of these language models. Thus, the size of disk space becomes one of the major factors to limit the power of the language models. For example, assume the vocabulary size is about 10^5. If we want to reduce the error rates of applications with word bigram Markov language model, then the possible way to achieve this goal is to enlarge the window size. However, it is difficult to do that from word bigram (the number of parameters is about 10^{10}) to trigram (the number of parameters is about 10^{15}), or more high degree Markov language models in practice because of the tremendous number of parameters.

To overcome this difficulty, class-based approaches and neural network approaches are proposed in recent years. The basic concept of class-based approaches is to use classes instead of words. Because the number of classes is less than the number of words, the class-based approaches will need less disk space than the word-based approaches. In order to group or classify the words into classes, some criteria such as lexical, syntactic and semantic relationships between words are presented.

If two words appear in the same or closed context, then these words have some lexical relationship and belong to the same class. Jelinek, et al. [1] used a very large number of classes on the order of the vocabulary size, and set up an adaptive language model to incorporate unknown words to suitable classes on the basis of the lexical relationships. This method does not touch on how to determine the initial vocabularies, i.e., initial classes, and it takes much time to find the synonym classes for the unknown words. Martelli [2] and Brown, et al. [3] proposed equivalent criteria and co-occurrence relationships respectively to assign words into classes. These methods are able to extract some classes that have the flavor of either syntactically based groupings or semantically based groupings, depending on the nature of the underlying statistics. However, their experiments are still very small and do not have satisfactory results.

Classes that correspond to the grammatical part-of-speech (or semantic tag) are called syntactic (or semantic) relationship classes. That is, if there are two words in the same class, then these words will have the same part-of-speech (or semantic tag). There are some applications [4-6] using these approaches to reduce the number of parameters. However, these approaches have some drawbacks shown below:

(1) The original performance of the word-based language models may be decreased very much. These may also limit the range of the applications.

(2) These methods need the syntactic (or the semantic) corpus tagged by human.

(3) Because the practice system using these approaches has to do automatic syntactic (or semantic) tagging, it will introduce some extra error results.

Nakamura, et al. [7] proposed a neural network approach, i.e., NET-gram, to overcome the large parameters problem. Training results show that the performance of the NET-gram is comparable to that of the statistical model. However, it still has the problems of the limited network size and the longer training time.

Generally speaking, the goal of these two approaches is to reduce the number of parameters in order to reduce the usage of storage. But these two approaches have suffered from some problems respectively. This paper will propose a storage reduction method to solve the problem that results from the large training table. This method should satisfy the following four conditions:

(1) The original performance of the word-based language models can not be decreased too much by applying this storage reduction method.

(2) This method need not be interfered by human.

(3) The range of the applications will not be limited by applying this method.

(4) The processing speed of the language models must be faster than the original word-based language models by applying this method.

2. The Storage Reduction Method

Our basic idea in the storage reduction is to use a mathematical function to simulate the distribution of the frequency value of the pairs in the training table. This idea comes from theoretical models of natural distributions [8]. If there is a function F that can simulate the distribution of the frequency value of the Markov word bigram pairs, then we can use this function F to evaluate the approximate frequency value for all bigram pairs, that is, F($word_1$, $word_2$) = frequency value. Similarly, given a function G that simulates the distribution of the frequency value of the Markov word trigram pairs, the approximate value for all trigram pairs can be computed by the function G($word_1$, $word_2$, $word_3$).

If such a function can be found, then this function can be used instead of training table. However, two major problems will be introduced and should be considered.

(1) The distribution of the frequency value of the pairs is usually very random. It is difficult to find a function that is very closed to the distribution of the value of the pairs.

(2) The dimension of the pairs is too high. Assume that the vocabulary size is V and Markov word bigram language model is used. The dimension of the pairs is about V^2. The computation time to find this function is very long because of the large dimension.

2.1 Grouping the Pairs

To overcome the difficulties, grouping the pairs is needed. The grouping criterion is that the pairs are grouped to the same class if the frequency values of the pairs are the same.

We use BDC segmented corpus as our training corpus and word association language models as our language model. BDC corpus includes 7010 sentences about 50000 words. With word association model, 156080 different pairs are generated from this corpus. Tables 1 and 2 show two different groupings. Basically, the grouping result of the 100 classes is extended from that of the 76 classes. Its purpose is: we try to disperse the risk caused by the mathematical function. Assume ten pairs are grouped in the same class C. If the curve (mathematical function) F returns the wrong frequency for the class C, then these ten pairs will have the wrong frequency under this curve. Thus, we extend some of the last classes in the first grouping and generate the second grouping.

Table 1. The 76-Class Grouping

Class	Frequency	# Of Pairs	Class	Frequency	# Of Pairs	Class	Frequency	# Of Pairs
1	1	137644	27	27	9	53	58	1
2	2	12206	28	28	6	54	62	1
3	3	2840	29	29	8	55	64	1
4	4	1220	30	30	5	56	67	1
5	5	603	31	31	7	57	69	1
6	6	398	32	32	4	58	71	1
7	7	236	33	33	3	59	73	1
8	8	168	34	34	6	60	75	1
9	9	136	35	35	11	61	77	2
10	10	97	36	36	2	62	79	1
11	11	70	37	37	5	63	82	1

Table 1. The 76-Class Grouping (*continued*)

Class	Frequency	# Of Pairs	Class	Frequency	# Of Pairs	Class	Frequency	# Of Pairs
12	12	56	38	39	4	64	84	1
13	13	44	39	40	5	65	87	1
14	14	37	40	41	2	66	93	1
15	15	27	41	42	3	67	94	1
16	16	30	42	43	3	68	95	2
17	17	25	43	44	2	69	97	1
18	18	22	44	45	2	70	108	2
19	19	18	45	46	1	71	127	1
20	20	13	46	47	1	72	130	1
21	21	13	47	48	1	73	135	1
22	22	14	48	49	3	74	220	1
23	23	12	49	52	2	75	237	1
24	24	15	50	53	1	76	280	1
25	25	6	51	54	1			
26	26	5	52	55	1			

Table 2. The 100-Class Grouping

Class	Frequency	# Of Pairs	Class	Frequency	# Of Pairs	Class	Frequency	# Of Pairs
1	1	137644	35	35	11	69	52	1
2	2	12206	36	36	2	70	52	1
3	3	2840	37	37	1	71	53	1
4	4	1220	38	37	1	72	54	1
5	5	603	39	37	1	73	55	1
6	6	398	40	37	1	74	58	1
7	7	236	41	37	1	75	62	1
8	8	168	42	39	1	76	64	1
9	9	136	43	39	1	77	67	1
10	10	97	44	39	1	78	69	1
11	11	70	45	39	1	79	71	1
12	12	56	46	40	1	80	73	1
13	13	44	47	40	1	81	75	1
14	14	37	48	40	1	82	77	1

Table 2. The 100-Class Grouping (*continued*)

Class	Frequency	# Of Pairs	Class	Frequency	# Of Pairs	Class	Frequency	# Of Pairs
15	15	27	49	40	1	83	77	1
16	16	30	50	40	1	84	79	1
17	17	25	51	41	1	85	82	1
18	18	22	52	41	1	86	84	1
19	19	18	53	42	1	87	87	1
20	20	13	54	42	1	88	93	1
21	21	13	55	42	1	89	94	1
22	22	14	56	43	1	90	95	1
23	23	12	57	43	1	91	95	1
24	24	15	58	43	1	92	97	1
25	25	6	59	44	1	93	108	1
26	26	5	60	44	1	94	108	1
27	27	9	61	45	1	95	127	1
28	28	6	62	45	1	96	130	1
29	29	8	63	46	1	97	135	1
30	30	5	64	47	1	98	220	1
31	31	7	65	48	1	99	237	1
32	32	4	66	49	1	100	280	1
33	33	3	67	49	1			
34	34	6	68	49	1			

2.2 Curve Fitting

At this step, a tool Cricket Graph 1.3.2 in Macintosh is used to find a suitable mathematical function. Some possible curve fitting results for the 76-class grouping are shown in Figure 1 - Figure 7.

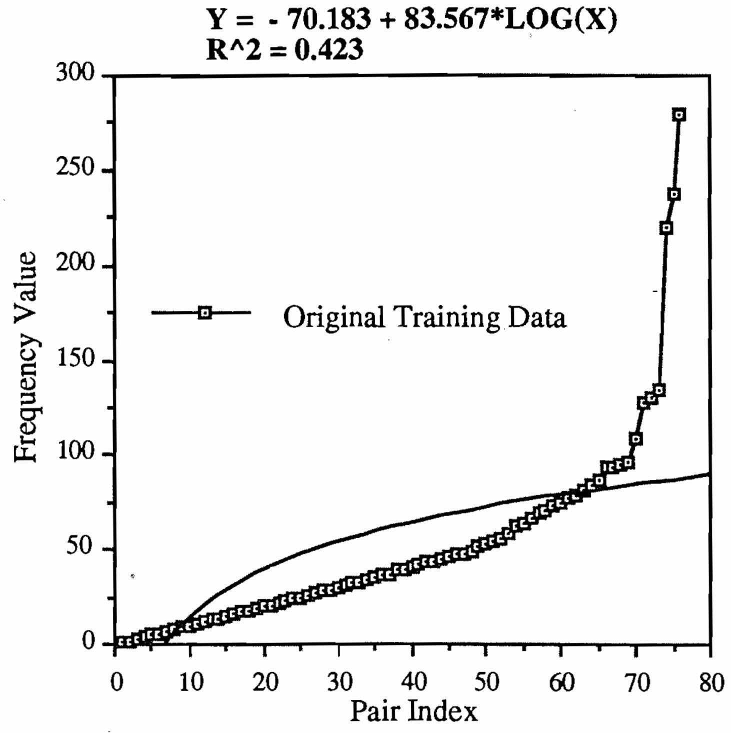


Figure 1. The 76-Class Grouping and the Log Function Are Used

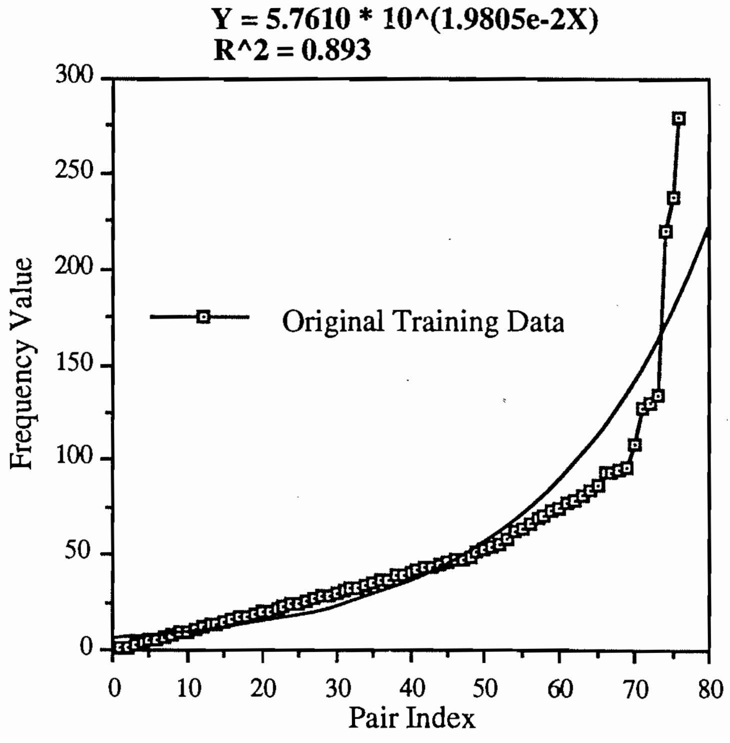


Figure 2. The 76-Class Grouping and the Exponential Function Are Used

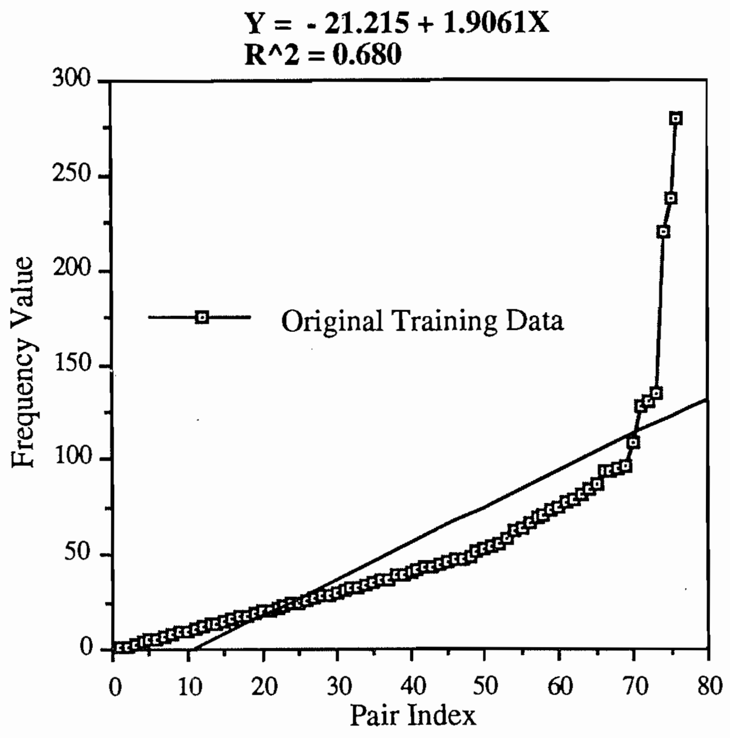


Figure 3. The 76-Class Grouping and the 1st Degree Polynomial Function Are Used

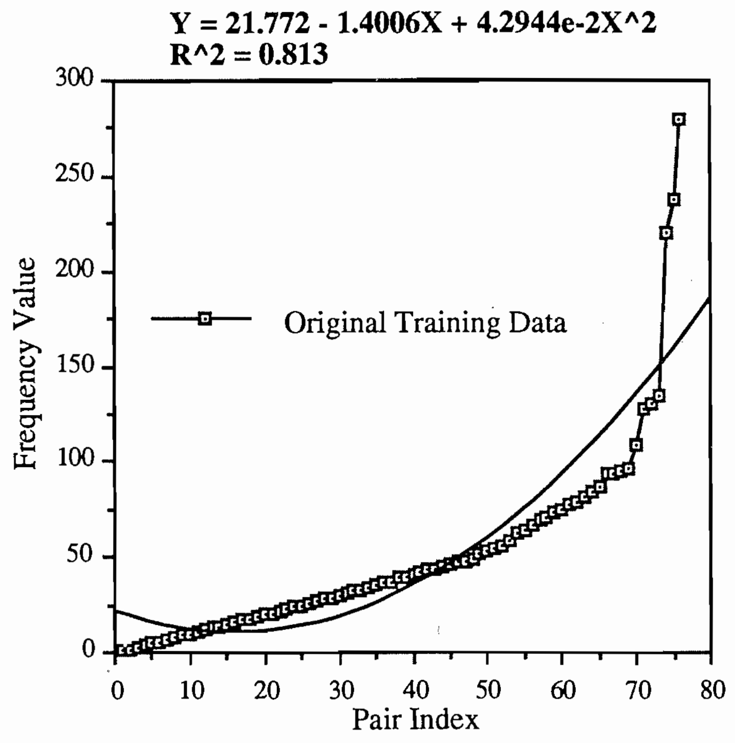


Figure 4. The 76-Class Grouping and the 2nd Degree Polynomial Function Are Used

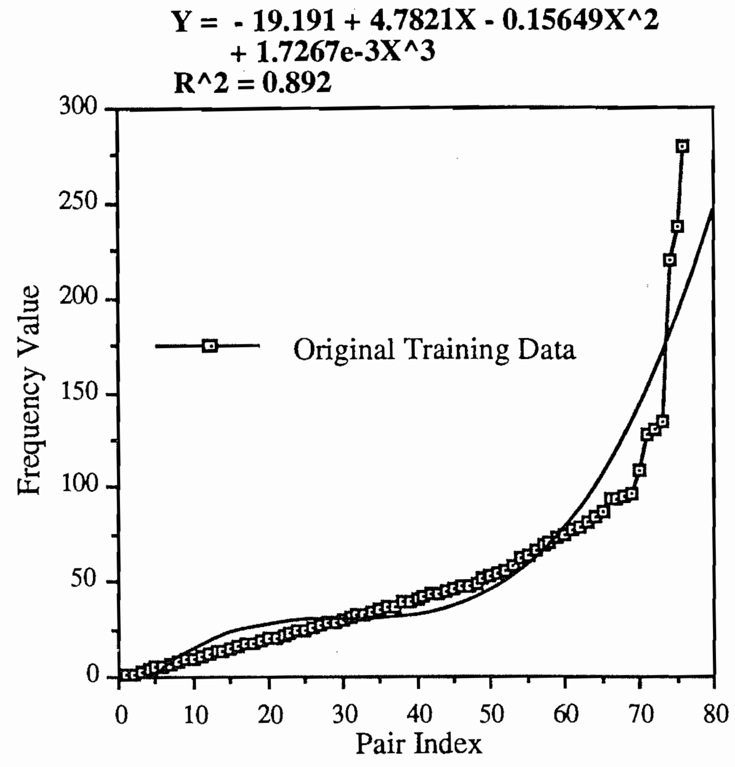


Figure 5. The 76-Class Grouping and the 3rd Degree Polynomial Function Are Used

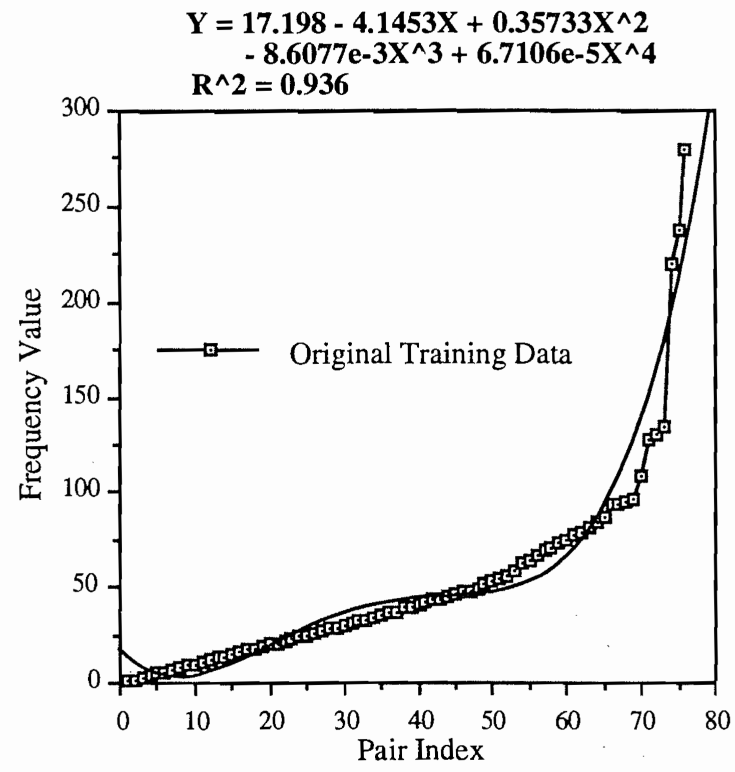


Figure 6. The 76-Class Grouping and the 4th Degree Polynomial Function Are Used

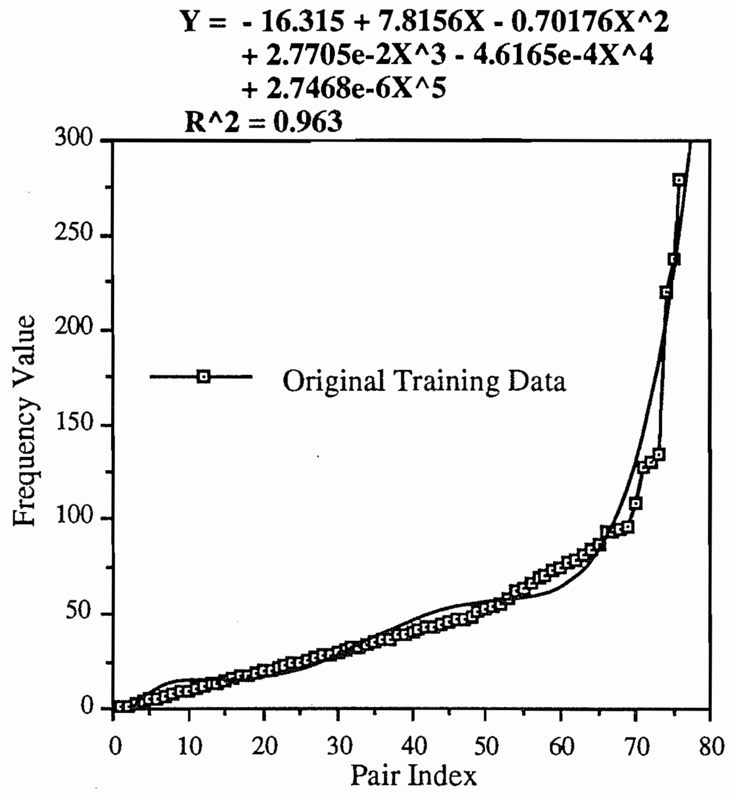


Figure 7. The 76-Class Grouping and the 5th Degree Polynomial Function Are Used

R^2 in these figures denotes the degree of the similarity between the training table and the curve function. The function

Y=-16.315+7.8156X-0.70176X^2+2.7705e-2X^3-4.6165e-4X^4+2.7468e-6X^5

has the highest R^2. Thus, it is selected as our testing function for 76-class case in the following experiment. Note that if the function returns a negative value then the value will be reset to 1. Similarly, we can find the curve function to fit the training table for 100-class grouping. The function

Y=-12.619+4.9581X-0.30176X^2+ 8.9514e-3X^3-1.1579e-4X^4+5.3913e-7X^5

is selected as our testing function for 100-class case in the following experiment (R^2: 0.941). The negative function value is treated in the same way, i.e., it will also be reset to 1.

2.3 Experimental Results

In the experiments, we use word association language model (called MM language model later) and forward training model to generate Chinese sentences described in Lee [9]. The forward training model means that the direction of word association is forward in training. For example, given a sentence $S=w_1, w_2, w_3$. Only word association pairs (w_1,w_2), (w_1,w_3) and (w_2,w_3) in forward direction will be generated. Consider a word sequence $S=w_1, w_2, ..., w_n$ as one of the arrangement of the words. The probability of the word sequence is measured as follows:

$$P(S)=P(w_1, w_2, ..., w_n)$$
$$\cong\prod_{i=1}^{n-1}\prod_{j=i+1}^{n} P_f(w_i,w_j)$$

where $P_f(w_i,w_j)$ is the probability of the word association between word w_i and w_j under forward training model.

$P_f(w_i,w_j)$ is defined as follows:

$$P_f(w_i,w_j)=\frac{F_f(w_i,w_j)}{\sum_{i=1}^{n}\sum_{j=1}^{n} F_f(w_i,w_j)}$$

where $F_f(w_i,w_j)$ is the frequency of words w_i and w_j that appear in the same sentence under forward training model.

Besides, there are two constraints used in our experiments to improve the system performance:

(1) Word/Word Linear Relation.

(2) POS/POS Linear Relation, where POS denotes part of speech.

Type (1) constraint is a set of constraint pairs (w_1,w_2). The pair (w_1,w_2) means word w_2 follows by word w_1 in the training corpus. Similarly, type (2) constraint is a set of constraint pairs (POS1,POS2). In this case, POS1 and POS2 appear in succession in the training corpus. Type (1) and type (2) constraints are enforced on the language models to eliminate the illegal combinations. Consider type (1) constraint and an arrangement of the words $S=w_1, w_2, ..., w_n$. This arrangement will be discarded if there exists any (w_i,w_{i+1}) $(1 \leq i \leq n-1)$ pair in the arrangement such that it does not satisfy the constraint. In order to use POS as constraints, the BDC corpus is tagged with BDC tag set. The experimental results are shown in Table 3.

Table 3. Experiment Results

Experiment	# of Test Sentences	Sentence Length	Original Language Model	Curve Fitting Model	Decrease (Correct Rate)
1	1000	1~6	82.8%	79.5%	3.3%
2	633	7~9	72.5%	68.8%	3.7%
3	1000	1~6	99.8%	99.7%	0.1%
4	633	7~9	99.5%	99.4%	0.1%
5	1000	1~6	83.2%	80.0%	3.2%
6	1000	1~6	82.8%	77.4%	5.4%
7	633	7~9	72.5%	66.3%	6.2%
8	1000	1~6	99.8%	99.5%	0.3%
9	633	7~9	99.5%	99.2%	0.3%
10	1000	1~6	83.2%	77.9%	5.3%

The experiments adopt different grouping and different language models shown as follows:

(1) Experiments 1 and 2

The 76-class grouping and MM language model are used.

(2) Experiments 3 and 4

The 76-class grouping and MM Language Model with type (1) constraint are used.

(3) Experiment 5

The 76-class grouping and MM language model with type (2) constraint are used.

(4) Experiments 6 and 7

The 100-class grouping and MM language model are used.

(5) Experiments 8 and 9

The 100-class grouping and MM language model with type (1) constraint are used.

(6) Experiment 10

The 100-class grouping and MM language model with type (2) constraint are used.

The curve fitting model is the original word association language model, but it uses the function instead of the training table. The experimental results show that there is a little error rate introduced by the curve function. The second grouping (100 classes) has a worse performance than the first grouping (76 classes) in our experiments because the latter has higher R^2. We evaluate this storage reduction method by the four conditions mentioned in Section 1:

(1) The result seems satisfactory because the method performs the closed performance and it just uses a little disk space. The training table, the index table, and the type (1) constraint table in the original language model occupy 2.23 M, 0.16 M and 0.61 M bytes respectively. In the pure curve fitting model, i.e., no type (1) constraints, no extra disk storage is required.

(2) The grouping is done with the frequency values of the word pairs so that no human interference is required.

(3) The approach can be applied to different language models such as Markov model, word association model, hybrid model, and so on.

(4) The frequency value is computed by function application, not by table look up. In the conventional storage management approach, it may take much time in disk I/O when the frequency value is retrieved.

3. The Classification Problem

The result seems satisfactory, but a classification problem is introduced, that is, how to know what class a given word pair ($word_1$,$word_2$) belong to. It is a problem for all class-based approaches. The next subsections will propose a neural network approach to deal with this problem.

3.1 Neural Network Approach

The overall neural network architecture is shown in Figure 8.

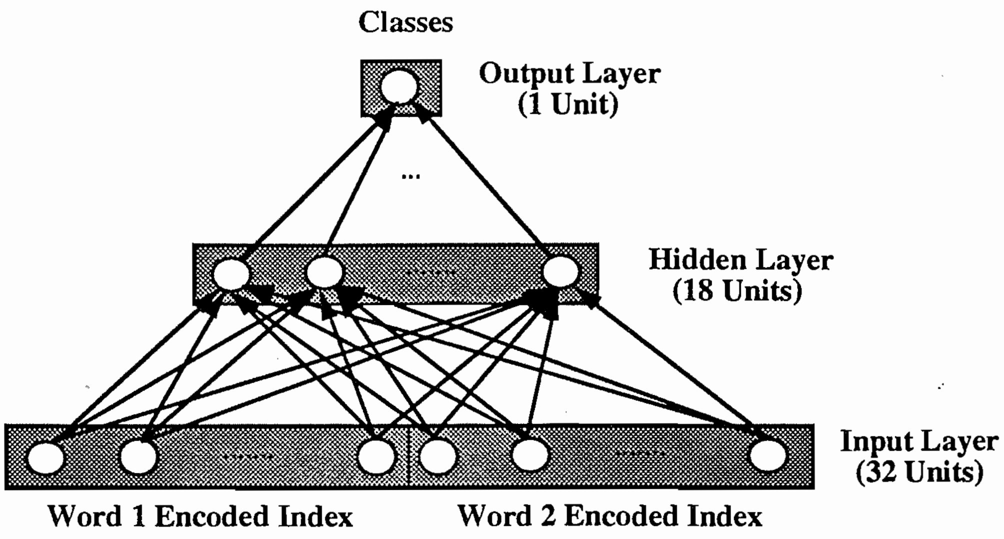


Figure 8. The Overall Neural Network Architecture

This neural network is a 3-layer feed-forward network with one hidden layer. In the input layer, word 1 and word 2 are all encoded into 16 bit vectors. For example, if the indices of words 1 and 2 are 8 and 15 respectively, then word 1 will be encoded into (0000000000001000) and word 2 will be encoded into (0000000000001111). The training adopts the back-propagation algorithm [10, 11], which uses the gradient descent to change link weights to reduce the difference between the network output and the desired output. In this algorithm, sigmoid function is used as nonlinear activation function. The sigmoid function is shown below:

$$F(y)=(1+e^{-\beta y})^{-1}$$

This function is continuous and varies monotonically from 0 to 1 as y varies from $-\infty$ to ∞. The gain of the sigmoid function, β, determines the stepness of the transition region. In our experiment, β is set to 1.0. This task is a many-to-one mapping problem. Thus, it is easy to train. In the output layer, there is only one unit. Because this unit will output a value whose range is from 0 to 1, we define the classes over this range. That is, if there are five classes, then the ranges of these classes are assigned to the five open intervals, (0.0,0.2), (0.2,0.4), (0.4,0.6), (0.6,0.8) and (0.8,1.0) respectively. After convergence, the training process confirms that the critical values such as 0.0, 0.2, 0.4, 0.6, 0.8 and 1.0 cannot appear.

3.2 Experimental Results

Four experiments are considered. Each selects different set of test data.

Experiment 1:	Group or classify the last 50 classes in 76 classes, i.e. 125 pairs.
Experiment 2:	Group or classify the first 5 classes in 76 or 100 classes, but only use (1/10000) of the pairs, i.e., 17 pairs.
Experiment 3:	Group or classify the first 5 classes in 76 or 100 classes, but only use (1/1000) of the pairs, i.e. 154 pairs.

Experiment 4: Group or classify the first 5 classes in 76 or 100 classes, but only use (1/100) of the pairs, i.e. 1545 pairs.

In these experiments, each pair is identified correctly. The results show that the neural network approach is suitable to deal with the classification problem, but it still has the longer training time problem.

4. Zero Frequency Problem

Last section proposes a neural network approach to deal with the pair classification problem. It can correctly identify the pairs with nonzero frequency. However, there still exists a serious problem called *zero frequency problem.* That is, we cannot tell out the word pairs that do not appear in the training table completely. In our experiments, 99.88% of the total belong to this kind of pairs, i.e., 132646496: 132802576. For the treatment of this problem, a preprocess procedure is taken. If the training data is stored in a 2-dimensional matrix (11524x11524 in our experiment), then it must be a sparse matrix. We reassign the word indices to move the zero pairs to the left-upper of the matrix. Figures 9 and 10 demonstrate the distribution of the word pairs of the training table before and after the preprocessing. Points in these figures denote the word pairs with nonzero frequency.

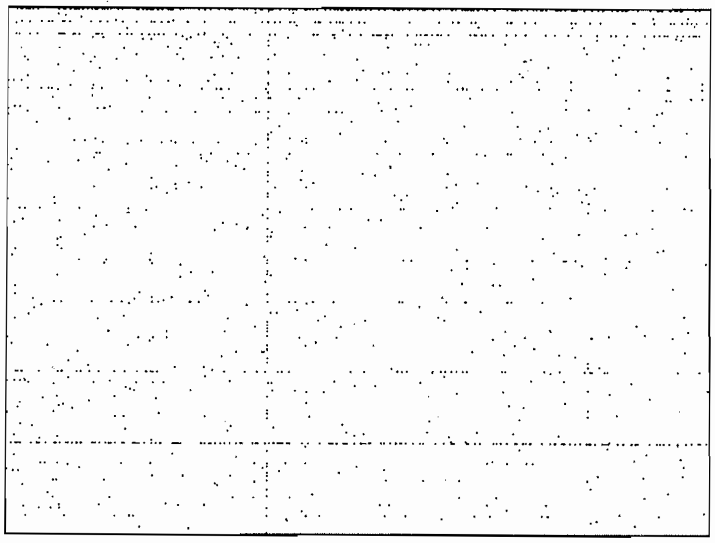


Figure 9. The Distribution of Word Pairs before Preprocessing

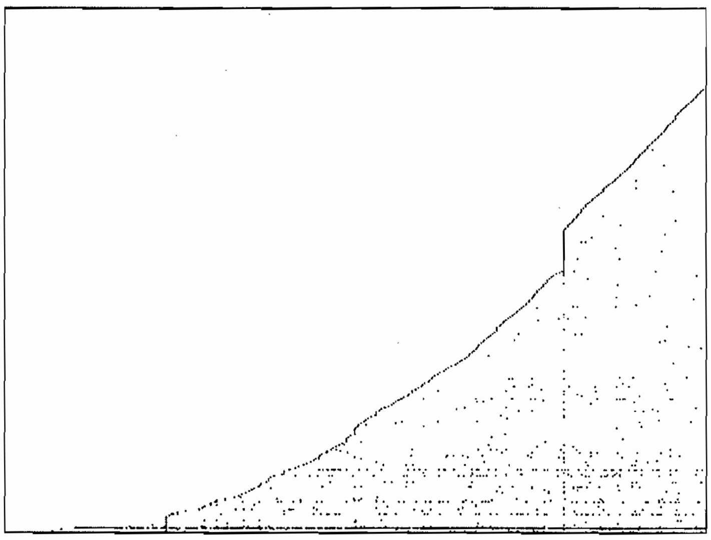


Figure 10. The Distribution of Word Pairs after Preprocessing

The experimental results show 72.5% of zero points can be rearranged to the left-upper corner. We can only record their indices (row number, column number), which occupy little space. However, the number of the remainder zero points is still very large (36477786). A sampling method is proposed to reduce the quantity in the neural network training. Given a continuous zero points P1, P2, ..., P*n*, only P1 and P*n* are taken as samples. At this step, 5862 samples (0.016%) are selected. The integration of this method to the large training data management can refer to [12].

5. Concluding Remarks

In this paper, we propose a storage reduction method to solve the problem that the training table is too large. Mathematical function is used to simulate the distribution of the frequency value of the pairs in the training table. The experimental results show that although there is a little error rate introduced by the curve function, this approach has the advantages of little space requirement, no human interference, no application limitation and faster processing speed. The neural network approach is also proposed to deal with the pairs classification problem. The experimental results show its feasibility.

References

[1] F. Jelinek, *et al.*, "Classifying Words for Improved Statistical Language Models," *Proceedings of IEEE ICASSP,* 1990, pp. 621-624.

[2] A. Martelli, "Stochastic Modeling of Language Via Sentence Space Partitioning," *Proceedings of Third Conference of the European Chapter of the ACL,* 1987, pp. 91-93.

[3] P.F. Brown, *et al.*, "Class-Based N-Gram Models of Natural Language," *Computational Linguistics*, Vol. 18, No. 4, 1992, pp. 467-479.

[4] H.C. Danon and M.E. Beze, "Three Different Probabilistic Language Models: Comparison and Combination," *Proceedings of IEEE ICASSP*, 1991, pp. 297-300.

[5] P. Dumouchel, *et al.*, "Three Probabilistic Language Models for a Large-Vocabulary Speech Recognizer," *Proceedings of IEEE ICASSP*, 1988, pp. 513-516.

[6] G. Maltese and F. Mancini, "An Automatic Technique to Include Grammatical and Morphological Information in a Trigram-Based Statistical Language Model," *Proceedings of IEEE ICASSP*, 1992, pp. 157-160.

[7] M. Nakamura, *et al.*, "Neural Network Approach to Word Category Prediction for English Texts," *Proceedings of COLING*, 1990, pp. 211-219.

[8] T.C. Bell, *et al.*, *Text Compression*, Prentice Hall Advanced Reference Series, Computer Science, 1990.

[9] Y.S. Lee, *Generating Chinese Sentences: A Corpus-Based Approach*, Master Thesis, Department of Computer Science and Information Engineering, National Taiwan University, 1993.

[10] P.D. Wasserman, *Neural Computing: Theory and Practice*, Van Nostrand Reinhold, 1989.

[11] D.R. Hush and B.G. Horne, "Progress in Supervised Neural Networks," *IEEE Signal Processing Magazine*, January, 1993, pp. 8-39.

[12] H.H. Chen and Y.S. Lee, "Very Large Training Data Management in Corpus-Based Language Modeling," Submitted to *IEEE Transaction on Signal Processing*.

A Probabilistic Chunker

Kuang-hua Chen and Hsin-Hsi Chen

Department of Computer Science and Information Engineering

National Taiwan University

Taipei, Taiwan, R.O.C.

Abstract

This paper proposes a probabilistic partial parser, which we call chunker. The chunker partitions the input sentence into segments. This idea is motivated by the fact that when we read a sentence, we read it chunk by chunk. We train the chunker from Susanne Corpus, which is a modified but shrinked version of Brown Corpus, underlying bi-gram language model. The experiment is evaluated by outside test and inside test. The preliminary results show the chunker has more than 98% chunk correct rate and 94% sentence correct rate in outside test, and 99% chunk correct rate and 97% sentence correct rate in inside test. The simple but effective chunker design has shown to be promising and can be extended to complete parsing and many applications.

1. Introduction

A probabilistic approach to natural language processing is not new [1]. Recently, many parsers based on this line have been proposed [2-9]. Garside and Leech [2] apply the constituent-likehood grammar of Atwell [10] to probabilistic parsing. Magerman and Marcus [3] adopt the chart-based probabilistic parsing. Zuijlen [4] tells out three probabilistic applications in parsing task. He also claims the probabilistic method should be controlled, otherwise it is not useful to us. Some papers [5-9] employ probabilistic context-free grammar to parsing task. The probabilistic context-free grammar is a modified version context-free grammar, which associates each grammar rule with a probability. The fact that these papers [11, 12, 13] use probabilistic approach to process speech also shows this approach has wide applications. Although these parsers apply different approaches, they all try to completely parse an input sentence into an annotated tree.

Abney [14] proposes a two-level architecture to tackle with the parsing task. The first level is a chunker, which is responsible for segmenting the input sentence into chunks. The second is an attacher, which is accountable for uniting the chunks to a parsing tree. This idea is motivated by the intuition:

(1) When we are about to read a sentence, we usually read it chunk by chunk.

We examplify the intuition by (2).

(2) [When we] [are about to] [read a sentence,] [we usually read it] [chunk by chunk].

The words between the left square bracket and the right square bracket form a chunk. Between chunks, we pause a while, when we read it. Abney further applies the context-free grammar to forming the backbone of chunker and attacher. Therefore, Abney's chunker and attacher are special LR-style parsers.

In this paper, we will propose a probabilistic chunker underlying bi-gram language model as a partial parser. The reason to call it partial parser is the fact that the chunker only segments the sentence into chunks. Instead of producing the hierarchical annotated tree, the chunker only produces the linear chunk sequence. The parameters of underlying bi-gram language model are trained from Susanne corpus [15, 16], which contains one tenth of Brown Corpus [17] and adopts the LOB corpus [18] tagging style. The Susanne corpus has more syntactic information and semantic information than Brown corpus, including parsing trees and trace marks.

This kind of partial parsers has many applications [19-22]. Church [19] applies the idea of partially parsing to designing a probabilistic NP detector. Church et al. [20] use Fidditch parser to extract typical arguments of verbs. Hindle [21] also employs Fidditch parser to extract arguments of verb for noun classification. Smadja [22] applies partial parser to collocation extraction. Our partial parser, chunker, not only provides the linear chunk sequence, but also the head of each chunk. This information can be applied to extracting the argument structure of verb and collocation. In addition, the chunker may be extended to a complete parser.

Section 2 will give a brief introduction to Susanne Corpus. Section 3 will describe the task and the language model. We will present the experiment procedure in Section 4 and show the preliminary results of the experiment in Section 5. In Section 6, we will describe the applications of chunker and future developments. Finally, we will give a brief conclusion.

2. Susanne Corpus

The Susanne Corpus is the modified and the condensed version of Brown Corpus. It only contains the 1/10 of Brown Corpus, but involves more information than Brown Corpus. The Corpus consists of four kinds of texts: 1) A: press reportage; 2) G: belles letters, biography, memoirs; 3) J: learned writing; and 4) N: adventure and Western fiction. The Categories of A, G, J, and N are named from each of the Brown Corpus. Each Category consists of 16 files and each file contains about 2000 words.

The following shows a snapshot of Susanne Corpus.

(3)
```
A01:0010a    -   YB      <minbrk>         -        [Oh.Oh]
A01:0010b    -   AT      The      the     [O[S[Nns:s.
A01:0010c    -   NP1s    Fulton   Fulton  [Nns.
A01:0010d    -   NNL1cb  County   county  .Nns]
A01:0010e    -   JJ      Grand    grand   .
A01:0010f    -   NN1c    Jury     jury    .Nns:s]
A01:0010g    -   VVDv    said     say     [Vd.Vd]
A01:0010h    -   NPD1    Friday   Friday  [Nns:t.Nns:t]
A01:0010i    -   AT1     an       an      [Fn:o[Ns:s.
A01:0010j    -   NN1n    investigation    investigation   .
A01:0020a    -   IO      of       of      [Po.
A01:0020b    -   NP1t    Atlanta Atlanta  [Ns[G[Nns.Nns]
A01:0020c    -   GG      +<apos>s         -        .G]
A01:0020d    -   JJ      recent   recent  .
A01:0020e    -   JJ      primary primary  .
A01:0020f    -   NN1n    election         election        .Ns]Po]Ns:s]
A01:0020g    -   VVDv    produced         produce [Vd.Vd]
A01:0020h    -   YIL     <ldquo> -        .
A01:0020i    -   ATn     +no      no      [Ns:o.
A01:0020j    -   NN1u    evidence         evidence        .
A01:0020k    -   YIR     +<rdquo>         -        .
A01:0020m    -   CST     that     that    [Fn.
A01:0030a    -   DDy     any      any     [Np:s.
A01:0030b     -    NN2     irregularities  irregularity    .Np:s]
A01:0030c     -    VVDv    took     take    [Vd.Vd]
A01:0030d     -    NNL1c   place    place   [Ns:o.Ns:o]Fn]Ns:o]Fn:o]S]
A01:0030e     -    YF      +.       -       .O]
```

The snapshot shows each line of the corpus includes six fields: 1) reference; 2) status; 3) wordtag; 4) word; 5) lemma; and 6) parse. Reference field shows the information of file name, the original line number in the Brown Corpus and word index in the Corpus (indexed with lower-case letter). Status field denotes the "abbreviation" or "symbol" information. Wordtag field points out what part of speech of the word should be. The tagging set, which is an extension and a modification of the tagging set of LOB Corpus, consists of 358 tags. Lemma field shows the base form of the word. Parse field is the core of the corpus, which shows the grammatical structure of the text and the current word is represented by "." symbol. Table 1 gives an overview of the Susanne Corpus. The details can refer to [15, 16].

Table 1. The Overview of Susanne Corpus

Categories	Files	Paragraphs	Sentences	Words
A	16	767	1445	37180
G	16	280	1554	37583
J	16	197	1353	36554
N	16	723	2568	38736
Total	64	1967	6920	150053

3. Task Description and Language Model

Parsing can be viewed as optimizing. Suppose a n-word sentence, w_1, w_2, ..., w_n (including pucntuation marks) , the parsing task is to find a parsing tree T, such that $P(T|w_1, w_2, ..., w_n)$ has the maximal probability. The annotated form of parsing tree T is changeable freely according to the task demand. We define T here to be a sequence of chunks, c_1, c_2, ..., c_m, and each c_k $(0 < k \leq m)$ contains one or more words $w_j\,(0 < j \leq n)$. For example, the sentence "parsing can be viewed as optimizing ." consists of 7 words. Its one possible parsing result under our guideline is:

(4) [Parsing] [can be viewed] [as optimization] [.]
c_1 c_2 c_3 c_4

Now, the parsing task is to find the best chunk sequence, C^*, such that

(5) $C^* = \arg\max_{C_i} P(C_i | w_1^n)$

The C_i is one possible chunk sequence, $c_1, c_2, \ldots, c_{m_i}$, where the m_i is the number of chunks of the possible chunk sequence. To resolve the optimization problem, we may adopt various language models. Here bi-gram language model is applied. Therefore, we further reduce $P(C_i | w_1^n)$ as (6),

(6)
$$
\begin{aligned}
&P(C_i | w_1^n) = P_i(c_1^{m_i} | w_1^n) \\
&\cong \prod_{k=1}^{m_i} P_i(c_k | c_{k-1}, w_1^n) \times P_i(c_k | w_1^n) \\
&\cong \prod_{k=1}^{m_i} P_i(c_k | c_{k-1}) \times P_i(c_k)
\end{aligned}
$$

where $P_i(\cdot)$[1] denotes the probability for the i'th chunk sequence. Once a probability $P_i(\cdot)$ is zero, the formula (6) will be zero. We then transform (5) to (7). In addition, when $P_i(\cdot)$ is zero, we define $\log(P_i(\cdot))$ to be zero.

(7)
$$
\begin{aligned}
&\arg\max_{C_i} P(C_i | w_1^n) \\
&\cong \arg\max_{C_i} \prod_{k=1}^{m_i} P_i(c_k | c_{k-1}) \times P_i(c_k) \\
&= \arg\max_{C_i} \sum_{k=1}^{m_i} [\log(P_i(c_k | c_{k-1})) + \log(P_i(c_k))]
\end{aligned}
$$

In order to make the expression (7) match the intuition of human being, namely, 1) the scoring metrics are all positive, 2) large value means high score, and 3) the scores are between 0 and 1, we define a score function S(·) shown as (8).

(8) $S(P(\cdot)) = 0$ when $P(\cdot) = 0$;
$S(P(\cdot)) = 1.0/(1.0+\mathrm{ABS}(\log(P(\cdot))))$ otherwise.

We then rewrite (7) as (9).

1 In general, $P(\cdot)$ repesents the probabilities of some events.

(9)

$$
\begin{aligned}
&\arg\max_{C_i} P(C_i \mid w_1^n) \\
&\cong \arg\max_{C_i} \prod_{k=1}^{m_i} P_i(c_k \mid c_{k-1}) \times P_i(c_k) \\
&= \arg\max_{C_i} \sum_{k=1}^{m_i} [\log(P_i(c_k \mid c_{k-1})) + \log(P_i(c_k))] \\
&= \arg\max_{C_i} \sum_{k=1}^{m_i} [S(P_i(c_k \mid c_{k-1})) + S(P_i(c_k))]
\end{aligned}
$$

The final language model is to find a chunk sequence C^*, which satisfies the expression (9).

4. Experiment Procedure

There are three parts in the experiment: the first part is training; the second is testing; the third is evaluating. Training process is to extract bi-gram data from Susanne corpus; testing process is 1) to tag the input raw data from the Susanne corpus, and then output tagged data; 2) to chunk the input data and produce the chunked data. Evaluating process is to compare the chunked data to Susanne corpus, and reports the correct rate. These are shown in the Figure 1.

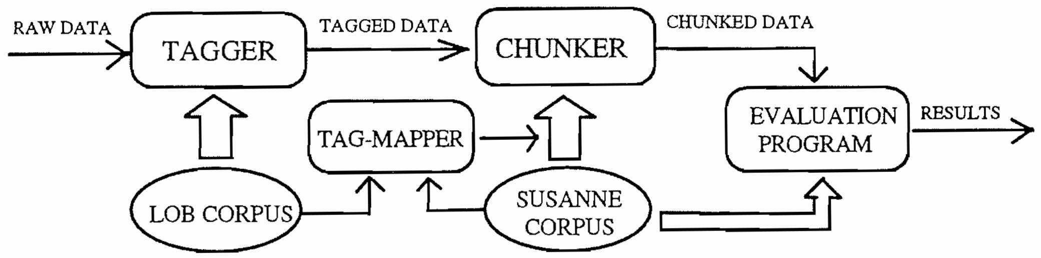


Figure 1. Experiment Procedure

The tagger is trained from LOB corpus [18]. This corpus contains 1 million words of English texts. Since the tag set of LOB corpus is different from that of the Susanne corpus, we first write a mapping program, TAG-MAPPER, to recover the LOB tags from the Susanne tags. The program maps 358 tags which Susanne corpus defines to 134 tags LOB corpus defines[2]. Then,

[2] Susanne corpus tags genitive case noun as [John_NP 's_GG], but LOB corpus tags it as [John's_PN$]. Two tags of Susanne corpus may be mapped to one tag of LOB corpus.

according to the criteria of (10), we extract the bi-gram chunk data from 3/4 of Susanne corpus (the rest is for outside test).

(10) a. The chunk is similiar to the phrase with content word as its head.

b. The considered content words are noun, verb, adjective, and preposition.

c. When a considered phrase is complex, a chunk contains at most two level sub-tree.

When we extract the bi-gram chunk data, we map them to the LOB tags and store them in datafile. Then, we sort this chunk data and build the "chunk grammar". As the results, the number of chunk grammar rules is 8675.

The second part is to test the Susanne corpus. The original 3/4 of Susanne corpus is used for inside testing; the rest of it for outside testing. The chunker runs on Sun SPARC-1 workstation. The processing time is shown in Table 2. In Table 2, Time/W means the time taken to process a word; Time/C means the time taken to process a chunk; and Time/S means the time taken to process a sentence.

Table 2. The Processing Time

	OUTSIDE TEST			INSIDE TEST		
	Time/W	Time/C	Time/S	Time/W	Time/C	Time/S
A	0.00944	0.0182	0.2268	0.01006	0.0264	0.2653
G	0.00889	0.0172	0.2174	0.00933	0.0252	0.2249
J	0.00902	0.0181	0.2738	0.00888	0.0263	0.2316
N	0.00988	0.0180	0.1634	0.00972	0.0220	0.1426
Average	0.00931	0.0179	0.2204	0.00950	0.0250	0.2161

According to Table 2, to process a word needs 0.00931 seconds for outside test, 0.00950 seconds for inside test, and 0.00941 on average. To process all Susanne corpus needs about 1412 seconds, or 23.6 minutes. Figure 2 depicts this results.

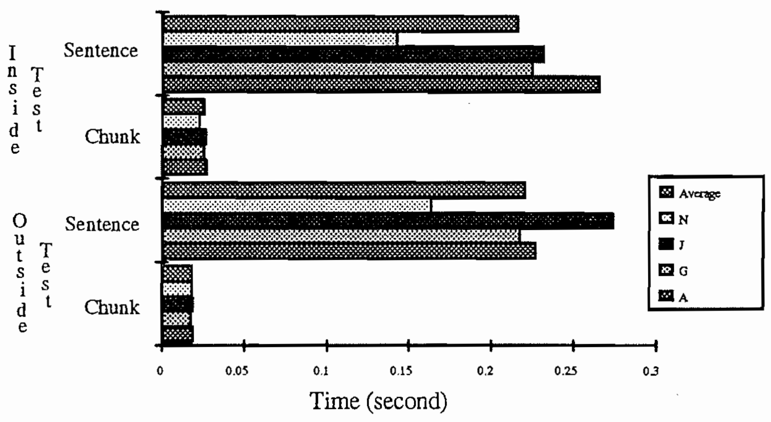
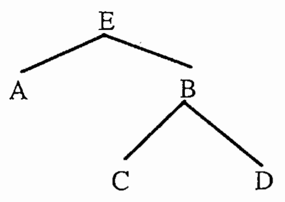


Figure 2. The Processing Time for Sentence and Chunk

The evaluating part is to compare the parsing results of our chunker with the denotation made by the Susanne corpus. The criterion is that the content of each chunk should be dominated by one non-terminal node in Susanne parse field.



[A C D]	right
[A] [C D]	right
[A C] [D]	wrong
[A] [C] [D]	right

Figure 3. The Evaluation Criterion

Figure 3 further explains this criterion. For a parsing tree [E [A] [B [C D]]], as shown in the left part of Figure 3, there are four possible chunk sequences. The third chunk sequence violates the criterion, since the contents of the first chunk are dominated by the different non-terminal nodes.

5. Preliminary Results

As the Section 4 points out, we begin the inside test by using the 3/4 of Susanne corpus and outside test by using the rest of the corpus. Evaluating the results by the criterion mentioned previously, we have the preliminary results shown in Table 3.

Table 3. Experimental Results

Category	TEST	OUTSIDE TEST		INSIDE TEST	
		Chunks	Sentences	Chunks	Sentences
A	# of correct	4866	380	10480	1022
	# of incorrect	40	14	84	29
	#	4906	394	10564	1051
	correct rate	0.99	0.96	0.99	0.97
G	# of correct	4748	355	10293	1130
	# of incorrect	153	32	133	37
	#	4901	387	10426	1167
	correct rate	0.97	0.92	0.99	0.97
J	# of correct	4335	283	9193	1032
	# of incorrect	170	15	88	23
	#	4505	298	9281	1055
	correct rate	0.96	0.95	0.99	0.98
N	# of correct	5163	536	12717	1906
	# of incorrect	79	42	172	84
	#	5242	578	12889	1990
	correct rate	0.98	0.93	0.99	0.96
Average	# of correct	19112	1554	42683	5090
	# of incorrect	442	103	477	173
	#	19554	1657	43160	5263
	correct rate	0.98	0.94	0.99	0.97

There are two kinds of correct rates. The first is chunk correct rate, which is measured by the correct segmented chunks over the total segmented chunks. The second is sentence correct rate, which is measured by the correct segmented sentences over the total sentences. A wrong segmented chunk means the whole sentence is not chunked properly. From Table 3, we know the overall sentence correct rate is over 94% and the chunk correct rate is over 98%. The difference between the inside test and outside test is not trivial. We compare the training data extracted from all Susanne corpus and the 3/4 of corpus, and find that the data from the latter cover the 80% of data from the former. The rest 20% data capture the gap of correct rate between inside test and outside test. But the 94% chunk correct rate have shown the work is promising. Figure 4 shows the correct rates of these experiments and gives an overview of these experiments.

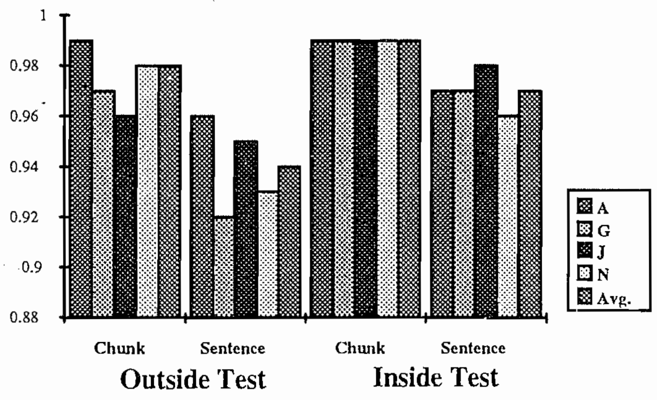


Figure 4. The Correct Rate of Experiments

For further analyzing the experiment, we define the chunk length.

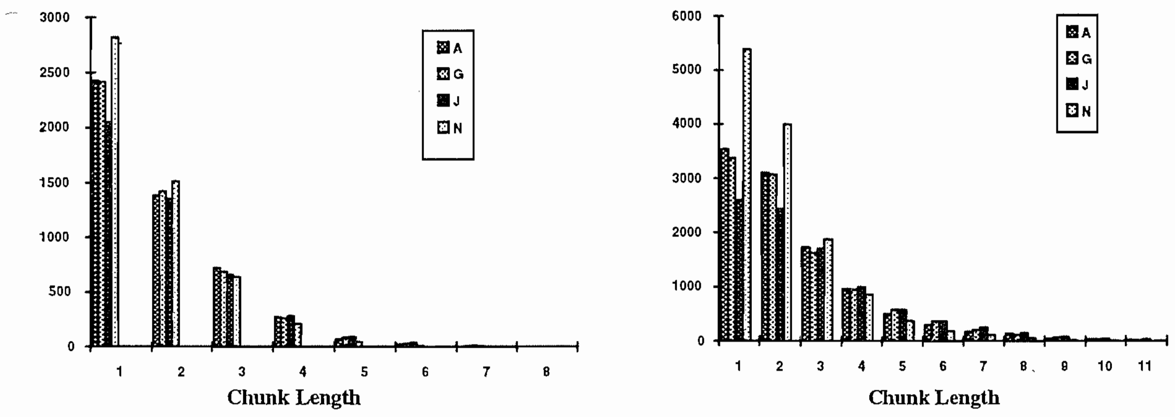
(11) Chunk length is the number of the words in a chunk.

We analyze the distribution of chunk length and list it in Table 4.

Table 4. The Distribution of Chunk Length

Chunk Length	OUTSIDE TEST				INSIDE TEST			
	A	G	J	N	A	G	J	N
1	2427	2411	2054	2823	3540	3380	2602	5390
2	1385	1420	1355	1511	3109	3070	2439	3999
3	721	688	659	635	1730	1630	1711	1873
4	276	260	283	208	959	952	997	854
5	67	83	95	46	509	590	587	378
6	24	31	43	11	302	363	368	186
7	3	7	13	7	169	210	253	117
8	3	1	3	1	143	115	151	55
9					52	74	85	20
10					28	28	52	13
11					23	14	36	4

The number of one-word chunks covers 43% of all kinds of chunks. This can be viewed in Figure 5. At the first glance, this result seems to challenge our probabilistic chunker. We further analyze what grammatic component constitutes the one-word chunks. The analysis is listed in Table 5.



(a) Outside Test (b) Inside Test

Figure 5. The Distribution of Chunk Length

In Table 5, WH-PN means wh-pronoun. OTHERS includes interjection, punctuation marks, letters, formulas, and foreign words. Ql/Qn represents the qualifiers and quantifiers. The rest types of one-word chunk are easy to understand.

Table 5. The Types of One-Word Chunks

Chunk Type	OUTSIDE TEST				INSIDE TEST			
	A	G	J	N	A	G	J	N
Noun	851	698	481	934	1399	1082	746	2224
Verb	672	674	549	957	1532	1639	1314	2390
Conj.	172	167	162	151	98	135	62	99
Prep.	145	169	227	109	106	92	91	64
Adjective	113	169	164	95	125	158	145	174
Adverb	143	145	117	288	90	81	88	274
Ql/Qt	96	94	87	70	43	62	64	41
WH-PN	46	46	18	24	76	59	2	43
OTHERS	189	249	249	195	69	72	89	76

We then scrutinize the table and know the most of the one-word chunks consist of noun, verb, and verbial adjective. This is because pronoun and proper name form the bare subject or object; verb is presented in the form of third person and singular, past tense, or base form; adjective forms the verbial adjective phrase, like beautiful in the sentence "Mary is beautiful". Figure 6 gives a clear view on the distribution. Noun and verb consist of 72% of one-word chunks. This shows our approach is useful to segment the sentence into the suitable chunks.

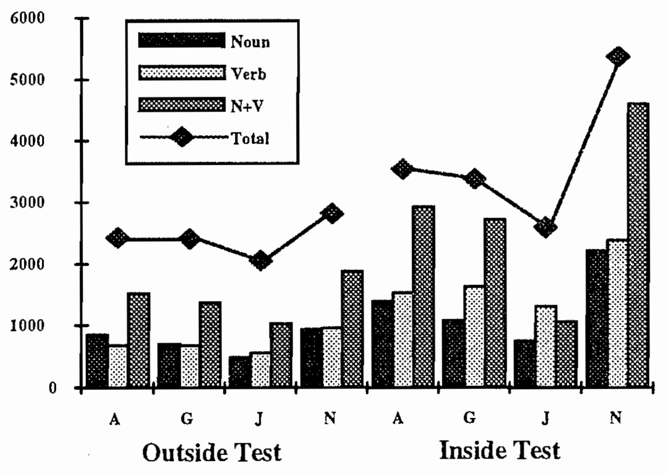


Figure 6. The Distribution of Noun and Verb Chunk

In Appendix, we list a sample output of the partial parsing.

6. Applications

Recently, the partial parsers have been applied to many problems as a preprocessor [19-22]. The applications include extracting argument structure of verbs [19, 20], grouping words [21], gathering collocations [22], and so on. Our probabilistic chunker is also capable of resolving these problems. We may modify the current version of chunker. The modified chunker not only partitions the input text, but also associates each chunk with a phrase mark (or a chunk mark). If it is a one-word chunk, the word itself is the chunk mark. For other chunks, the chunker finds the most manifest word in this chunk as the chunk mark. Generally speaking, the word is the head of this chunk. (12) is a possible chunked sentence.

(12) [We_PP1AS] [saw_VBD] [NN(2): a_AT woman_NN] [IN(1): with_IN a_AT telescope_NN] [._.]

In (12), every chunk is associated with a mark and its position in the chunk (it is unnecessary to associate one-word chunk with this information). According to the information, we may extract argument structure of verb with SVO and other heuristic rules. Furthermore, we can group noun or verb according to the extracted argument structure.

In addition to these applications, we may construct a recursive probabilistic chunker to be a complete parser. We may reorganize the parsing task as a sequence of actions, chunking and raising interleavingly. The parsing task is finished, when no more chunking is needed. This idea is shown in Figure 7.

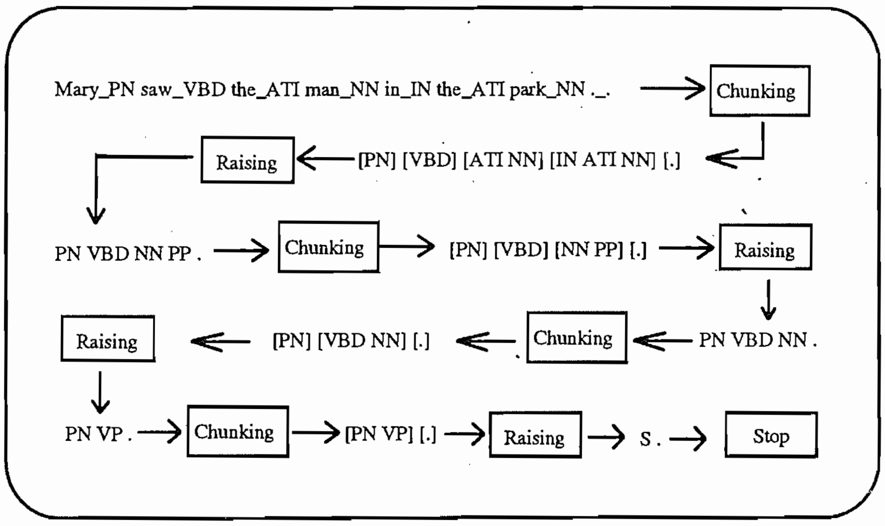


Figure 7. A Recursive Chunker as a Parser

We formally define parsing as (13)-(16) based on the idea.

(13) Parsing is a sequence of actions consisting of chunking and raising interleavingly.

(14) Chunking is an action of segmenting input components into a sequence of chunks.

(15) Raising is an action of lifting the head from input chunks.

(16) Parsing is finished, when no chunking can be operated on.

7. Concluding Remarks

To process real text is indispensable for a practical natural language system. Probabilistic method provides a robust way to tackle with the unrestricted text. This is why probabilistic method dominates the recent research directions of natural language processing. In the field of parsing techniques, many parsers based on this line are proposed. Some of them are LR-style [5-9]; some of them are chart-based [3]; some adopt constituent-likehood grammar [2]. These approaches are more complexive. For example, it is necessary for the probabilistic LR parsing to extract hierarchical context-free grammar rules from corpus and to calculate the probability associated with each rule. Once there are left-recursive rules, we must transform them or use equations to solve these intermixing probabilities [7]. In this paper, we report a probabilistic chunker to execute the partial parsing. Comparing to these approaches mentioned above, ours is simple and easy to extend to construct a complete parser. In training process, the mere work we do is to extract bi-gram (according to the language model; maybe tri-gram) linear data from a parsed corpus. Through the evaluation procedure, the correct rate is promising. The preliminary experimental results show the chunker has the 98% correct rate for chunk and 94% for sentence in outside test. It depicts our finding is worthy looking forward to. In addition, we also provide the future development and the possible applications of the finding.

Acknowledgements

We are grateful to Dr. Geoffrey Sampson for his kindly providing Susanne Corpus and the details of tag set to us. Research on this paper was partially supported by National Science Council grant NSC82-0408-E002-029.

References

[1] P. Suppew, "Probabilistic Grammars for Natural Languages," *Synthese* 22, 1970, pp. 95-116.

[2] R. Garside. and F. Leech, "A Probabilistic Parser," *Proceedings of Second Conference of the European Chapter of the ACL*, 1985, pp. 166-170.

[3] D.M. Magerman and M.P. Marcus, "Pearl: A Probabilistic Chart Parser," *Proceedings of Fifth Conference of the European Chapter of the ACL*, 1991, pp. 15-20.

[4] J.M.V. Zuijlen, "Probabilistic Methods in Dependency Grammar Parsing," *Proceedings of International Workshop on Parsing Technologies*, 1989, pp. 142-151.

[5] T. Fujisaki, "A Stochastic Approach to Sentence Parsing," *Proceedings of 22th Annual Meeting of the ACL*, 1984, pp. 16-19.

[6] T. Fujisaki, *et al.*, "Probabilistic Parsing Method for Sentence Disambiguation," *Proceedings of International Workshop on Parsing Technologies*, 1989, pp. 85-94.

[7] S.K. Ng and M. Tomita, "Probabilistic LR Parsing for General Context-Free Grammars," *Proceedings of International Workshop on Parsing Technologies, 1991, pp. 154-163.*

[8] A. Corazza, *et al.*, "Stochastic Context-Free Grammars for Island-Driven Probabilistic Parsing," *Proceedings of International Workshop on Parsing Technologies*, 1991, pp. 210-217.

[9] J. Wright and E.N. Wrigley, "Adaptive Probabilistic Generalized LR Parsing," *Proceedings of International Workshop on Parsing Technologies*, 1991, pp. 100-109.

[10] E.S. Atwell, "Constituent-Likelihood Grammar," *(ICAME News)*, No. 7, 1983, pp. 34-66.

[11] K. Kita, *et al.*, "Parsing Continuous Speech by HMM-LR Method," *Proceedings of 27th Annual Meeting of the ACL*, 1989, pp. 126-131.

[12] J.H. Wright and E.N. Wigley, "Probabilistic LR Parsing for Speech Recognition," *Proceedings of International Workshop on Parsing Technologies*, 1989, pp. 105-114.

[13] S. Seneff, "Probabilistic Parsing for Spoken Language Applications," *Proceedings of International Workshop on Parsing Technologies*, 1989, pp. 209-218.

[14] S. Abney, "Parsing by Chunks," in *Principle-Based Parsing,* Berwick, Abney and Tenny (Eds.), Kluwer Academic Publishers, 1991, pp. 257-278.

[15] G. Sampson, "The Susanne Corpus," *ICAME Journal*, No. 17, 1993, pp. 125-127.

[16] G. Sampson, *English for the Computer*, Oxford University Press (Forthcoming).

[17] N. Francis and H. Kucera, *Manual of Information to Accompany a Standard Sample of Present-day Edited American English, for Use with Digital Computers*, Department of Linguistics, Brown University, Providence, R. I., U.S.A., original ed. 1964, revised 1971, revcised and augmented 1979.

[18] S. Johansson, *The Tagged LOB Corpus: Users' Manual*, Bergen: Norwegian Computing Centre for the Humanities, 1986.

[19] K.W. Church, "A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text," *Proceedings of Second Conference on Applied Natural Language Processing*, 1988, pp. 136-143.

[20] K.W. Church, *et al.*, "Parsing, Word Associations and Typical Predicate-Argument Relations," *Proceedings of International Workshop on Parsing Technologies*, 1989, pp. 389-398.

[21] D. Hindle, "Noun Classification from Predicate-Argument Structures," *Proceedings of 28th Annual Meeting of ACL*, 1990, pp. 268-275.

[22] F. Smadja, *Extracting Collocations from Text, An Application: Language Generation* Ph.D. Dissertation, Columbia University, 1991.

Appendix

Five examples are demonstrated in the appendix. In each example, the first part is the sentence extracted from Susanne Corpus but tagged with the LOB tagging set; the second part is the corresponding chunked results by our Chunker.

1. The_ATI Fulton_NP County_NPL Grand_JJ Jury_NN said_VBD Friday_NR an_AT investigation_NN of_IN Atlanta_NP recent_JJ primary_JJ election_NN produced_VBD <ldquo>_*' +no_ATI evidence_NN +<rdquo>_**' that_CS any_DTI irregularities_NNS took_VBD place_NPL +._.

[The_ATI Fulton_NP County_NPL]

[Grand_JJ Jury_NN]

[said_VBD]

[Friday_NR]

[an_AT investigation_NN]

[of_IN Atlanta_NP]

[recent_JJ primary_JJ election_NN]

[produced_VBD]

[<ldquo>_*' +no_ATI evidence_NN +<rdquo>_**']

[that_CS any_DTI irregularities_NNS]

[took_VBD]

[place_NPL]

[.]

2. The_ATI jury_NN further_RBR said_VBD in_IN term_NR +<hyphen>_*- +end_NN presentments_NNS that_CS the_ATI City_NPL Executive_JJB Committee_NN +,_, which_WDTR had_HVD over<hyphen>all_JJB charge_NN of_IN the_ATI election_NN +,_, <ldquo>_*' +deserves_VBZ the_ATI praise_NN and_CC thanks_NNS of_IN the_ATI City_NPL of_IN Atlanta_NP +<rdquo>_**' for_IN the_ATI manner_NN in_IN which_WDTR the_ATI election_NN was_BEDZ conducted_VBN +._.

[The_ATI jury_NN]

[further_RBR said_VBD]

[in_IN term_NR +<hyphen>_*- +end_NN]

[presentments_NNS]

[that_CS the_ATI City_NPL Executive_JJB Committee_NN +,_,]

[which_WDTR had_HVD]

[over<hyphen>all_JJB charge_NN of_IN the_ATI election_NN +,_,]

[<ldquo>_*' +deserves_VBZ]

[the_ATI praise_NN and_CC thanks_NNS]

[of_IN the_ATI City_NPL of_IN Atlanta_NP +<rdquo>_**']

[for_IN the_ATI manner_NN]

[in_IN which_WDTR]

[the_ATI election_NN]

[was_BEDZ conducted_VBN]

[.]

3. The_ATI September_NR +<hyphen>_*- +October_NR term_NR jury_NN had_HVD been_BEN charged_VBN by_IN Fulton_NP Superior_JJ Court_NN Judge_NPT Durwood_NP Pye_NP to_TO investigate_VB reports_NNS of_IN possible_JJ <ldquo>_*' +irregularities_NNS +<rdquo>_**' in_IN the_ATI hard_RB +<hyphen>_*- +fought_VBN primary_NN which_WDTR was_BEDZ won_VBN by_IN Mayor_NPT +<hyphen>_*- +nominate_RB Ivan_NP Allen_NP Jr_NPT +._.

[The_ATI September_NR +<hyphen>_*- +October_NR]

[term_NR]

[jury_NN]

[had_HVD been_BEN charged_VBN]

[by_IN Fulton_NP Superior_JJ Court_NN]

[Judge_NPT Durwood_NP Pye_NP]

[to_TO investigate_VB]

[reports_NNS of_IN possible_JJ]

[<ldquo>_*' +irregularities_NNS +<rdquo>_**']

[in_IN the_ATI]

[hard_RB +<hyphen>_*- +fought_VBN]

[primary_NN]

[which_WDTR was_BEDZ won_VBN]

[by_IN Mayor_NPT +<hyphen>_*- +nominate_RB]

[Ivan_NP Allen_NP Jr_NPT]

[.]

4. <ldquo>_*' +Only_RB a_AT relative_JJ handful_NN of_IN such_ABL reports_NNS was_BEDZ received_VBN +<rdquo>_**' +,_, the_ATI jury_NN said_VBD +,_, <ldquo>_*' +considering_IN the_ATI widespread_JJ interest_NN in_IN the_ATI election_NN +,_, the_ATI number_NN of_IN voters_NNS and_CC the_ATI size_NN of_IN this_DT city_NPL +<rdquo>_**' +._.

[<ldquo>_*' +Only_RB a_AT relative_JJ handful_NN of_IN such_ABL reports_NNS]

[was_BEDZ received_VBN +<rdquo>_**' +,_,]

[the_ATI jury_NN]

[said_VBD +,_,]

[<ldquo>_*' +considering_IN the_ATI widespread_JJ interest_NN in_IN the_ATI election_NN +,_,]

[the_ATI number_NN of_IN voters_NNS]

[and_CC the_ATI size_NN of_IN this_DT city_NPL +<rdquo>_**']

[.]

5. The_ATI jury_NN said_VBD it_PP3 did_DOD find_VB that_CS many_AP of_IN Georgia_NP registration_NN and_CC election_NN laws_NNS <ldquo>_*' +are_BER outmoded_JJ or_CC inadequate_JJ and_CC often_RB ambiguous_JJ +<rdquo>_**' +._.

[The_ATI jury_NN]

[said_VBD]

[it_PP3]

[did_DOD find_VB]

[that_CS]

[many_AP]

[of_IN Georgia_NP]

[registration_NN]

[and_CC election_NN]

[laws_NNS]

[<ldquo>_*' +are_BER]

[outmoded_JJ or_CC inadequate_JJ]

[and_CC often_RB ambiguous_JJ +<rdquo>_**']

[.]

A PRELIMINARY STUDY ON UNKNOWN WORD PROBLEM IN CHINESE WORD SEGMENTATION

Ming-Yu Lin, Tung-Hui Chiang and Keh-Yih Su

Department of Electrical Engineering
National Tsing Hua University
Hsinchu, Taiwan 30043, R.O.C.

Abstract

Unknown word, in general, is the main factor that causes the performance of word segmentation to be unsatisfied. To recognize the words which are derived from highly productive morphemes, a set of 17 morphological rules is proposed in this paper to recognize those regular unknown words. In addition, an unknown word model is further proposed to deal with the unknown words of irregular forms such as proper name etc. With the unknown word resolution procedures, the error reduction rate of 78.34% in word and 81.87% in sentence are obtained in the task of smoothing technical manuals. To examine the procedures in more general task, a corpus of newspaper is also tested and the error reduction rate of 40.15% in word and 34.78% in sentence are observed.

1. Introduction

"Word" is the basic unit used in most Chinese information processing tasks, such as machine translation or spoken language processing. However, there is no obvious delimiter marker, except for some punctuation markers, to specify the boundaries of words. Therefore, word segmentation is essential in almost all Chinese language processing systems.

Several models for word segmentation were proposed in our previous work [Chia 92a], in which the comparisons between rule-based and statistics-based approaches were made. From that work, over 99% word segmentation accuracy rate was observed when there is not any unknown word in the corpus; while only 95-96% could be obtained in case unknown words existed. Unfortunately, in Chinese, many morphemes have high derivative abilities such that they can combine with other words or morphemes to form compounds or complex words. To enumerate all such kinds of words in the dictionary is impossible and impractical. What is more, many new words are generated every day, so it is very difficult to keep the dictionary up-to-date. Thus, the problems caused by unknown words are inevitable in processing Chinese information. Hence, how to identify unknown words is the most important issue in real Chinese language processing systems. Motivated by that, the focus is shifted to the study of unknown words in this paper.

There are two kinds of unknown words: one is *regular*, such as time, date, reduplication, etc.; while the other is *irregular*, such as proper names, compound nouns, of which the unknown words must be determined by their context instead of simple rules. Regular unknown words are likely to be predicted according to some morphological rules. However, the words of irregular forms are usually difficult to be identified from rules. They must be examined through the analysis with more high level knowledge sources, such as syntax and semantics.

In this paper, a set of 17 morphological rules are first introduced to tackle the problem caused by the regular unknown words. After applying the morphological rules, 1.81% error rate in word is observed. Compared with the error rate of 7.48% in the baseline system, it corresponds to 75.8% error reduction rate. Afterwards, an irregular unknown word model is proposed to recognize the irregular unknown words, with which 78.34% in error reduction is obtained.

This paper is organized as follows. The system architecture, the databases including the lexicon, the morphological rules, and the tasks are described in Section 2. In Section 3, the overview of the baseline models, which were derived in our previous works [Chia 92a], are given. Then, the effects of the morphological rules are investigated in Section 4. In addition, we incorporate part of speech information into the system to explore the performance both in word and lexical tag in Section 5. Furthermore, an unknown word model is proposed in Section 6 to resolve some problems caused by the unknown words in irregular forms. Finally, a summary is addressed in the last section.

2. System Architecture

The flow of the word segmentation in our system is shown in Figure 1. The system consists of four phases of processes, including the baseline word segmentation, morphological analysis, tagging, and unknown words identification. The input character string is first processed by the baseline segmentation model, in which all possible segmentation patterns are generated by looking up the dictionary and assigned the corresponding preference scores depending on the model used. Then the best N (N is set to 10 in the current implementation) word hypothesis sequences are passed to the morphological analyzer. The morphological rules are then employed to detect some particular forms of unknown words in this phase. Again, the top N candidates are output for being tagged with their lexical tags. Afterwards, the best tagged result is dispatched into the unknown word module to examine other types of unknown words. Finally, the best hypothesis is picked up as the final output.

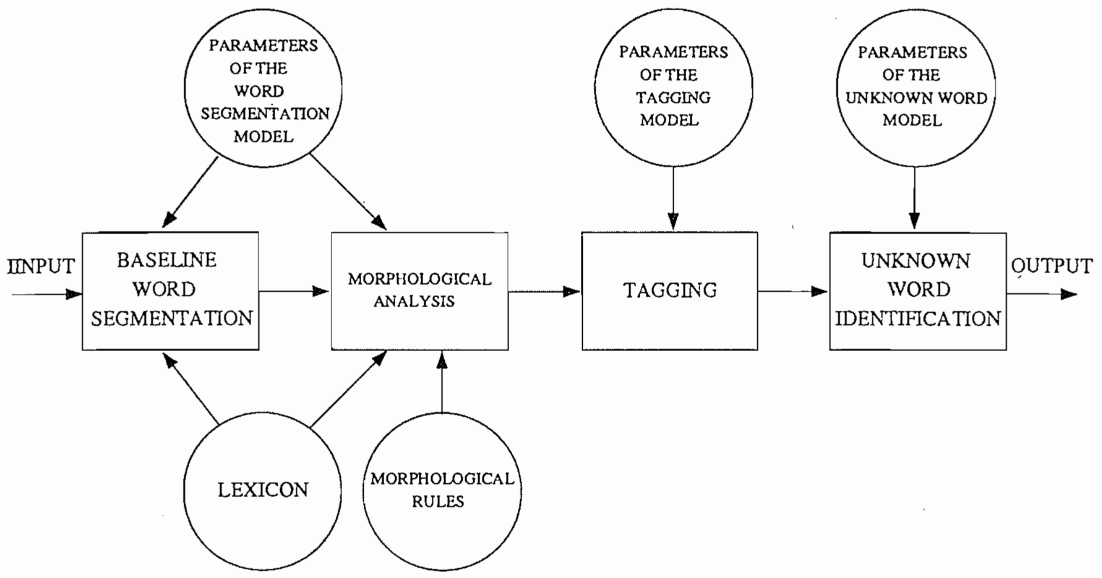

PARAMETERS OF THE WORD SEGMENTATION MODEL
PARAMETERS OF THE TAGGING MODEL
PARAMETERS OF THE UNKNOWN WORD MODEL
INPUT
BASELINE WORD SEGMENTATION
MORPHOLOGICAL ANALYSIS
TAGGING
UNKNOWN WORD IDENTIFICATION
OUTPUT
LEXICON
MORPHOLOGICAL RULES

FIGURE 1
The block diagram of the system architecture.

Lexicon

The electronic dictionary used in our system is provided by Behavior Design Corporation (BDC) [BDC 92], in which there are 89,590 entries of definition. For each word, the possible lexical tags that can be attached to it are encoded in the dictionary. Currently, there are 49 different categories of tags used in the dictionary. The statistics of the dictionary are listed in Table 1.

# of characters / word	# of entries
1	1,734
2	35,492
3	19,650
4	24,054
5	6,140
6	2,020
>= 7	500
Total	89,590

TABLE 1
The statistics of the dictionary.

Morphological Rules

There are 17 morphological rules [Lin 93] available in the system which are written by linguistic experts according to a large corpus. Two of these morphological rules are only related to some particular affixes. The rest 15 rules, on the contrary, must refer to the lexical tags. All these morphological rules are listed in appendix A.

Corpus

To evaluate the performance of different segmentation models, a corpus of 9,677 sentences extracted from technical manuals are collected. This corpus is further divided into a training corpus of 7,742 sentences, i.e., 4/5 of the original set, and a testing corpus of the remaining 1,935 sentences in the following simulations. Along with the simulations performed in [Chia 92a], an *ideal* corpus is formed by extracting the sentences which contain unknown words out of the original corpus. Therefore, the original corpus is also called the *real corpus* in contrast. The effect of using the proposed models both in the ideal and the real corpora are investigated and compared in the paper. The statistics of the corpora are listed in Table 2.

	Ideal Corpus		**Real Corpus**	
	Training Set	Testing Set	Training Set	Testing Set
# of sentences	3,711	911	7,742	1,935
# of words	37,720	9,238	87,715	21,964
# of characters	62,423	15,374	148,221	37,261
Ave. # of words /sentence	10.16	10.14	11.33	11.35
Ave. # of characters /sentence	16.82	16.88	19.15	19.26

TABLE 2
The statistics of the corpora.

3. Overview of the Baseline Model

Since the baseline models have been derived in our previous work [Chia 92a], instead of repeating the detail derivations of those models, only the final forms of the computational models are listed in this paper.

Let c_1^n denote the input character sequence of n Chinese characters and $W_i = w_{i,1}, w_{i,2}, \cdots, w_{i,M_i}$ be the i-th word segmentation pattern, where M_i denotes the total number of words in W_i, the model derived in [Chia 92a] is summarized as follows:

$$\operatorname*{argmax}_{w_{i,1}^{i,M_i}} \left\{ \prod_{k=1}^{M_i} P\left(w_{i,k} \mid l_{i,k-1}\right) \right\}, \tag{1}$$

where $l_{i,k-1}$ denotes the length, i.e., the number of characters, of $w_{i,k-1}$. In other words, the correlation of the word and the length of its left contextual word is considered in the model.

To compare the common rule-based approach with the baseline models, the approach using the rule that the longest word is most preferred is also implemented in this simulation. The results for these approaches both in the real and ideal corpora are shown in Table 3.

Model	**Error rate in the training set**		**Error rate in the testing set**	
	word (%)	sentence (%)	word (%)	sentence (%)
Max. Match	2.15	9.84	2.63	11.09
$P(w_k \mid l_{k-1})$	0.12	0.62	0.69	2.63

(a)

Model	**Error rate in the training set**		**Error rate in the testing set**	
	word (%)	sentence (%)	word (%)	sentence (%)
Max. Match	8.74	56.74	9.47	58.14
$P(w_k \mid l_{k-1})$	6.91	52.34	7.48	54.16

(b)

TABLE 3
The results of various word segmentation models in (a) the ideal corpus and (b) the real corpus.

Comparing the results in Table 3(a) and 3(b), it is apparent that the existence of unknown words is the main issue which causes the performance to degrade evidently. The results performed in the real corpus are 6.79% worse in word accuracy, and 51.53% degradation in sentence accuracy compared with those in the ideal case. After analyzing the errors caused by these models, two kinds of error patterns are founded. The first one is the mis-combined error, denoted by **s_ns**, such as |一 |個 |人 |→ |一 |個人 |, where two or more words which should be separated are regarded as a word. The other pattern, denoted by **ns_s**, is the over-segmentation error, where a word is mis-segmented apart into several morphemes or words, such as |轉換器 |→ |轉換 |器 |. The statistics of these two error patterns for the baseline models in the real corpus are listed in the following table.

Models	**Errors in the training set**		**Errors in the testing set**	
	s_ns	ns_s	s_ns	ns_s
$P(w_k \mid l_{k-1})$	260	5,904	109	1,533

TABLE 4
The statistics of the error patterns for the baseline model in the real corpus.

In Table 4, it is obvious that the error is caused mainly from over-segmentation of words. Therefore, to combine those over-segmented words into a word will improve the performance effectively. To do this, the approaches with the morphological rules and an unknown word model are introduced later in this paper.

4. The Morphological Analysis

As mentioned above, many morphemes in Chinese have high derivative capability so that they can combine with other words or morphemes to form new words, such as 總，化 . Therefore, the words formed in such a way are unable and impractical to be enumerated in the lexicon. Since the word formation processes associated with those morphemes are quite regular, they are; therefore, predictable according to a few rules. Motivated by this concern, a set of morphological rules are introduced in our system. Currently, there are 17 morphological rules in the system. They are divided into two parts according to whether part of speech is applied or not. The first part consists of two morphological rules which only relate to some particular affixes. On the other hand, the remaining 15 rules in the second part are applied with the lexical tags. Interested readers for the morphological rules are referred to appendix A or [Lin 93].

Like most rule-based approaches, the use of the morphological rules will results in the problem of redundancy, and inconsistency. Those redundant and, especially, the inconsistent rules have to be withdrawn from the rule-base to improve the performance of the system both in terms of the accuracy and efficiency. In this paper, a sequential forward selection method is used in rule ordering, which will be described in the following subsection.

4.1. Rule Ordering

To examine the effectiveness of the morphological rules, the sequential forward selection (SFS) procedure [Devi 82, Liu 93] is applied to determine the ordering of morphological rules. SFS is a simple bottom up search procedure where one rule at a time is added to the current rule set. At each stage, the rule to be included in the rule set is selected from the remaining available rules, so that the new enlarged set of the rules yields a maximum value of the criterion function used. The rule ordering procedure with the SFS is shown as follows.

Assume that G1 is the original rule set and G2 is the set including the rules which are ordered through the SFS algorithm. Initially, G1 consists of all morphological rules and G2 is an empty set.

```
SFS(n rules) {
        G1= {n rules};                                   /* initialization for G1 set */
        G2= {};                                          /*
        initialization for G2 set */
        /* the loop of moving the best rule in G1 to G2 */
        loop( while there is any rule in G1) {
```

```
        mincost=minimum_value;          /* initialize the variable for minimum cost */
        /* the loop of computing the cost of embodying each of rules in G1 to G2 */
        loop ( for each rule_i in G1) {
            cost=WordSegmentation(corpus, {rule_i}+G2);
                      /* computing the cost returned by word segmentation procedure for using
                         the new rule set which is composed of rule_i and those rules in G2 */
            /* find the rule with minimum cost */
            if (cost<mincost) then
                swap(cost,mincost);          /* swap the minimum cost with the current one */
                best_rule=rule_i;            /* current rule is assigned to be the best one */
            endif
        }
        move_rule(G1,G2,best_rule)           /* move the best rule from G1 to G2 */
    }
}
```

ILLUSTRATION 1
The rule ordering procedure with the sequential forward selection algorithm.

Note that the cost function returned by the WordSegmentation() function is computed according to the following formula:

$$cost = w_r \times (1 - P_r) + w_p \times (1 - P_p), \tag{2}$$

where $P_r = \frac{No.\,of\ words\ identified\ correctly}{No.\,of\ words\ in\ the\ corpus}$, named as the *recall rate*, is the percentage of the words in the corpus which are identified correctly by the system; $P_r = \frac{No.\,of\ words\ identified\ correctly}{No.\,of\ words\ identified\ by\ the\ system}$, known as the *precision rate*, is the percentage of the words identified by the system being correct; w_r, w_p are defined as the weights to the error rate of recall and precision respectively and they are both defined to be *0.5* in the following test. Thus, the performance of both the recall rate and the precision rate are taken into account through the cost function defined above.

Through the SFS procedure, the results of cost versus number of rules is illustrated in Figure 2. From this figure, it is noted that the cost is decreasing as the number of rules is increasing up to 11; however, the cost remains a constant as the rule number is in the range from 11 to 16. Finally, it increases slightly when the last rule is incorporated. After checking the results, we find that it is the rule $\mathrm{n} \rightarrow \mathrm{q} +$ 點 (時間) which results in the increment in cost. Therefore, it should be modified or withdrawn from the rule base. In addition, those rules that rank from 11-th to 16-th are never applied in the training corpus; however, they are remained in the rule base because they may be useful in the testing set.

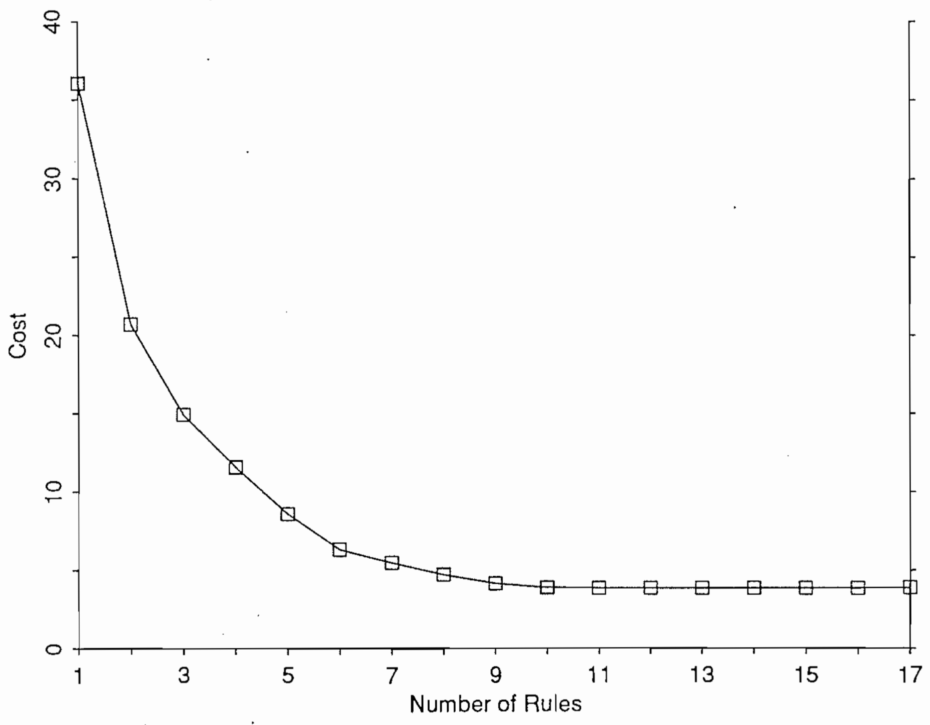


FIGURE 2
Illustration of the cost versus the number of rules through the rule ordering mechanism.

4.2. Summary of the Morphological Analysis

The results of various word segmentation models with the morphological analysis in the ideal corpus as well as the real corpus are shown in Table 5, where the values in the parentheses are the corresponding results of the baseline models.

Model	Error rate in the training set		Error rate in the testing set	
	word (%)	sentence (%)	word (%)	sentence (%)
$P(w_k \mid l_{k-1})$	0.80 (0.12)	4.04 (0.62)	1.44 (0.69)	6.26 (2.63)

(a)

	Error rate in the training set		Error rate in the testing set	
Model	word (%)	sentence (%)	word (%)	sentence (%)
$P(w_k \mid l_{k-1})$	1.32 (6.91)	8.86 (52.34)	1.81 (7.48)	10.70 (54.16)

(b)

TABLE 5
The results of the baseline word segmentation model with the morphological analysis in (a) the ideal corpus and (b) the real corpus.

It is observed that the performance in the ideal corpus degrades slightly. After applying the morphological rules, the possibility of mis-combining the words or morphemes which should be separated will inevitably increase. Therefore, the performance of the ideal corpus degrade. On the contrary, the situation of mis-combination is not so serious in the real corpus. In fact, the results are greatly improved with the morphological analysis, where it corresponds to the error reduction rate of 75.8% in word and 80.43% in sentence. The statistics of the error pattern for morphological analysis are listed in Table 6, where the corresponding results with the baseline models are tabulated in parentheses.

	Errors in the training set		Errors in the testing set	
Models	s_ns	ns_s	s_ns	ns_s
$P(w_k \mid l_{k-1})$	429 (260)	370 (5,904)	168 (109)	110 (1,533)

TABLE 6
The statistics of the error patterns in the real corpus after morphological analysis.

The result shows that the morphological analysis significantly reduces the errors caused by the over-segmentation, which is over 90%. Therefore, the performance is improved dramatically with the morphological rule approach. On the other hand, checking up the s_ns type of error in Table 6, it is observed that this approach has slight side-effect for increasing the mis-combination errors. Those mis-combinations are caused by unconditionally applying the morphological rules without regarding their contexts. For example, the mis-combination of |從 |研究到 |發展 | is caused by applying the rule " v → v(研究)+ 到 ".

To further decrease the error of mis-combination, those morphological rules should be accomplished with a context-sensitive framework, which is similar to the formulae for phrase structure rules in [Chia 92b]. It will be our future work and will not be discussed in this paper. Instead, we will pay attention to the unknown words which are formed irregularly and cannot be recognized through the morphological rules. Because lexical tags will be used as the parameters in the unknown word model, we will describe the tagging process in the next section before starting the unknown word modeling.

5. Tagging Part Of Speech

In the previous study [Chia 92a], we have shown that the incorporation of lexical information is useful in word segmentation. However, the morphological rules are applied before the tagging process. The introduction of the morphological analysis may result in changes of the formation of words or the lexical tags. Accordingly, the effect of the combination of the morphological and lexical knowledge sources is investigated in this section. To do this, we derive the word segmentation model which incorporates the lexical information as follows:

$$\widehat{W} = \underset{W_i}{\operatorname{argmax}} \sum_{T_{j,i}} P(W_i, T_{j,i} \mid c_1^n), \tag{3}$$

where $T_{j,i}$ stands for the j-th lexical sequence corresponding to the i-th word segmentation pattern W_i. To save the time for computation in the above equation, we approximate it in the following form:

$$\widehat{W} = \underset{W_i}{\operatorname{argmax}} \left\{ \max_{T_{i,j}} P(W_i, T_{i,j} \mid c_1^n) \right\}. \tag{4}$$

The term $P(W_i, T_{i,j} \mid c_1^n)$ in Eq.(4) is further derived as follows.

$$\begin{aligned} & P(W_i, T_{i,j} \mid c_1^n) \\ & = P(T_{i,j} \mid W_i, c_1^n) \times P(W_i \mid c_1^n) \\ & \approx P(T_{i,j} \mid W_i) \times P(W_i \mid c_1^n) \\ & = \frac{P(W_i \mid T_{i,j}) \times P(T_{i,j})}{P(W_i)} \times \frac{P(c_1^n \mid W_i) \times P(W_i)}{P(c_1^n)}. \end{aligned} \tag{5}$$

Note that the approximation in the above derivation is based on the fact that the lexical tags are attached only to words; therefore, it is assumed that tagging the part-of-speech is independent of the character string if the word sequence is given. In addition, since the character sequence can be determined uniquely if a word sequence is given, it causes that $P(c_1^n \mid W_i) = 1$ holds for all word segmentation patterns. Besides, the term $P(c_1^n)$is the same constant to each segmentation ambiguity and it does not affect the result in Eq.(4) if being neglected. Hence, the criterion in Eq.(4) is rewritten in the following form:

$$\widehat{W} = \underset{W_i}{\operatorname{argmax}} \left\{ \max_{T_{i,j}} P(W_i \mid T_{i,j}) \times P(T_{i,j}) \right\}. \tag{6}$$

Concerning the i-th word segmentation pattern $W_i = w_{i,1}, w_{i,2}, \cdots, w_{i,M_i}$ of M_i words, let $t_{i,j,k}$ denote the part-of-speech that is attached to $w_{i,k}$ in the j-th lexical sequence $T_{i,j} = t_{i,j,1}, t_{i,j,2}, \cdots, t_{i,j,M_i}$. To make the computation in Eq.(6) feasible, $P(W_i \mid T_{i,j})$ and $P(T_{i,j})$ are approximated as follows.

$$\begin{aligned} P(W_i \mid T_{i,j}) &= P\left(w_{i,1}^{i,M_i} \mid t_{i,j,1}^{i,j,M_i}\right) \\ &\approx \prod_{k=1}^{M_i} P_i(w_k \mid t_{j,k}). \end{aligned} \tag{7}$$

$$P(T_{i,j}) = P_i\left(t_{j,1}^{j,M_i}\right) \approx \prod_{k=1}^{M_i} P_i\left(t_{j,k} \mid t_{j,k-1}\right). \tag{8}$$

It is noted that $P_i(\cdot)$, which denotes the probability function that relates to the i-th word segmentation pattern, is introduced in the above equations to prevent from notational confusion. Therefore, the word segmentation model with lexical knowledge incorporated is represented as the following formula:

$$\arg\max_{w_{i,1}^{i,M_i}} \left\{ \max_{t_{i,j,1}^{i,j,M_i}} \left[\prod_{k=1}^{M_i} P\left(w_{i,k} \mid t_{i,j,k}\right) \times P\left(t_{i,j,k} \mid t_{i,j,k-1}\right) \right] \right\}. \tag{9}$$

The results with the morphological and lexical analysis in the ideal corpus as well as the real corpus are shown in Table 7, where the values in the parentheses are the results with the model $P(w_k \mid l_{k-1})$ before incorporating the lexical information.

	Error rate in the training set		**Error rate in the testing set**	
Model	word (%)	sentence (%)	word (%)	sentence (%)
$P(w_k \mid t_k) \times P(t_k \mid t_{k-1})$	0.69 (0.80)	3.48 (4.04)	1.45 (1.44)	7.14 (6.26)

(a)

	Error rate in the training set		**Error rate in the testing set**	
Model	word (%)	sentence (%)	word (%)	sentence (%)
$P(w_k \mid t_k) \times P(t_k \mid t_{k-1})$	1.24 (1.32)	8.58 (8.86)	1.74 (1.81)	11.16 (10.70)

(b)

TABLE 7
The word and sentence error rate of various word segmentation models with the morphological analysis in (a) the ideal corpus and (b) the real corpus.

From Table 7, it is observed that the results are improved slightly, except for the testing set in the ideal corpus. However, the improvement is not significant enough to show the superiority to incorporate the lexical information. Since the morphological rules are applied in a context-free manner, the errors of mis-combination resulting from the morphological analysis cannot be recovered even with a tagger. Besides, by using this tagging model, the number of parameters is much larger than those of the baseline models so that the over-tuning phenomena is more apparent. Hence, the results in the training set can be improved more significant than those in the testing set.

To couple the tagger into the system is, however, essential because the lexical information is required in the following unknown word model. To examine the effectiveness of the tagger, the error of the tagging process is also listed in Table 8. Note that in this paper, a correct tagging to a word is defined when both the word segmentation and the lexical tag are correct simultaneously.

Model	Error rate in the training set		Error rate in the testing set	
	tag (%)	sentence (%)	tag (%)	sentence (%)
$P(w_k \mid t_k) \times P(t_k \mid t_{k-1})$	7.56	46.86	8.92	51.70

(a)

Model	Error rate in the training set		Error rate in the testing set	
	tag (%)	sentence (%)	tag (%)	sentence (%)
$P(w_k \mid t_k) \times P(t_k \mid t_{k-1})$	7.77	51.52	9.50	58.40

(b)

TABLE 8
The tag error of the morphological analysis in (a) the ideal corpus and (b) the real corpus.

6. Unknown Word Modeling

The unknown words in the corpus can be categorized into the following classes.

1. The words should be contained in the dictionary, such as 若要，不僅，爭議，驚魂未定 . For the corpus of technical manuals, there are 263 words in the training set and 72 words in the testing set of this class; while in the newspaper corpus, 141 and 40 words in the training set and the testing set are categorized to this class respectively.
2. The words should be combined through the morphological rules, such as 牛肝，牛心 . For the technical manuals, there are 35 words in the training set and 7 words in the testing set of this class. In the newspaper corpus, 12 and 2 words in the training set and the testing set belong to this class respectively.
3. Abbreviations, such as 國大，立委，台獨 . No word in the corpus of technical manuals are classified to this class. However, for the newspaper corpus, there are 6 words in the training set and 2 words in the testing set of this class.
4. Proper nouns, biographical names, and geographical names, such as 德國聯邦共合國，大牛，胡適. There are 2 words in the training set and none in the testing set of this class. As to the newspaper corpus, 39 and 6 words in the training set and testing set belong to this class respectively.
5. Others: this class includes the words of typographical errors in the corpus, such as 吩附 (咐) and missing lexical tags in the dictionary, such as 閣 . In addition, several words in the dictionary are in conflict with the principals of word formation

announced by Computational Linguistics Society R.O.C., which should be withdrawn from the dictionary.

Due to the incompleteness of the dictionary and the morphological rules, the words in class 1 and 2 are regarded as unknown words. They should be restored by renewing the dictionary or modifying the morphological rules. In this paper, the words in class 3 and 4 are what we are really interested in. Nevertheless, the words of class 1 and 2 will always appear unless a dictionary in unlimited size is available.

Since the morphological rules are written for detecting unknown words which are formed regularly, they cannot identify those words which are neither formed regularly nor able to be enumerated entirely in the dictionary. Therefore, a statistical model is further proposed in this paper to tackle the problems caused by these kinds of unknown words. In viewing the word segmentation results, several unknown words are segmented into a series of separate characters, such as |國 |大 |黨 |部 |透露 |層 |峰 |消息 |. In the current task, over 72% of the irregular type of unknown words belong to this case. Therefore, we will attack this kind of error in this paper. To deal with this kind of unknown words, only the region in which all words are of single character is considered to have the possibility of possessing a unknown word in our model. It means that the unknown words of length over than 2, such as 曼德拉 and 德國聯邦共合國, are not taken into account currently.

To consider the region of interest R_u as shown below, which is composed of N_u separate characters, $w_1, w_2, \cdots, w_{N_u}$, with their corresponding tags $t_1, t_1, \cdots, t_{N_u}$, the contexts associated with this region are w_b, w_e, and their tags are t_b, t_e respectively. Here, we further assume that there is only one unknown word resides in the suspected region in the unknown word model. Accordingly, two decisions relating to this suspected region have to be made by the unknown word fixing strategy: (1) to decide whether there is any unknown word in R_u; (2) to determine the way of combination of the unknown word if the previous answer is "yes."

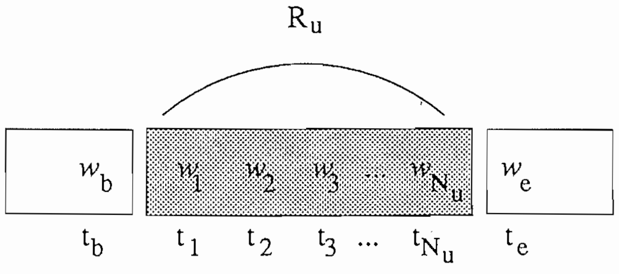


FIGURE 3
The suspected unknown word region.

To answer the first question, a likelihood ratio γ is defined as follows:

$$\gamma = \frac{P\left(E_{uw} = 1 \mid (w_b, t_b), \left(w_1^{N_u}, t_1^{N_u}\right), (w_e, t_e)\right)}{P\left(E_{uw} = 0 \mid (w_b, t_b), \left(w_1^{N_u}, t_1^{N_u}\right), (w_e, t_e)\right)}, \tag{10}$$

where E_{uw} is an indicator; $E_{uw} = 1$ denotes the existence of unknown words, otherwise $E_{uw} = 0$. The number of parameters associated with the Eq.(10) are too many to be handled in practice. Hence, it is approximated as the following equation:

$$\gamma = \frac{P\Big(E_{uw} = 1 \mid t_b, t_1^{N_u}, t_e\Big)}{P\Big(E_{uw} = 0 \mid t_b, t_1^{N_u}, t_e\Big)}$$

$$\approx \frac{\left[\prod_{i=0}^{N_u+1} P(t_i \mid t_{i-1}, E_{uw} = 1)\right] \times P(t_e, E_{uw} = 1)}{\left[\prod_{i=1}^{N_u+1} P(t_i \mid t_{i-1}, E_{uw} = 0)\right] \times P(t_e, E_{uw} = 0)}; \quad (where\ t_0 = t_b,\ t_{N_u+1} = t_e). \tag{11}$$

Currently, it is regarded that there is an unknown word in the region if $\gamma > 1$; otherwise, the suspected region is considered without any unknown word.

If the suspected region is considered with an unknown word, each possible way of combination associated with the unknown word shown below is given a preference score according to a scoring function. To clearly describe this function, we take the second case (UW 2.2) for example. The score of the case (UW 2.2), where the unknown word is combined by w_2 and w_3, is defined as follows:

$$P\Big(UW = w_2w_3 \mid (w_b.t_b), \Big(w_1^{N_u}, t_1^{N_u}\Big), (w_e, t_e), E_{uw} = 1\Big), \tag{12}$$

where $UW = w_2w_3$ represents the event that the unknown word is composed of w_2 and w_3.

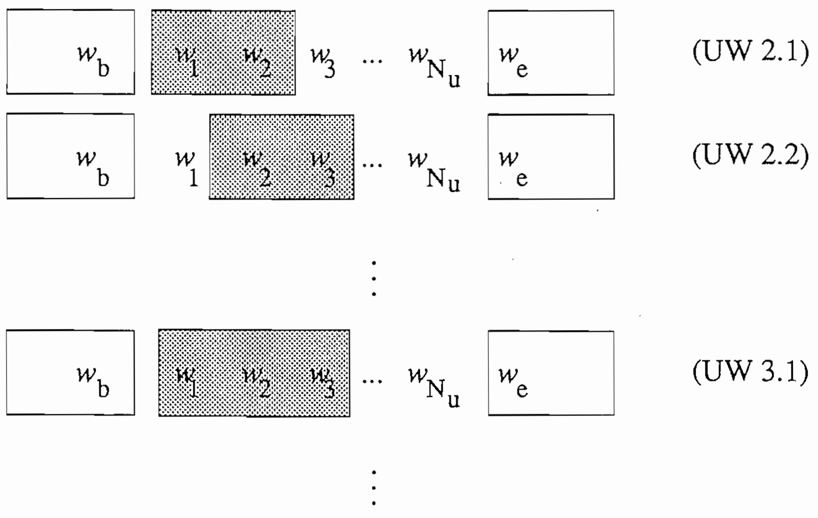


FIGURE 4
The possible types of combination with an unknown word existing in suspected region; the shaded regions indicate the possible positions and formations of the unknown word.

Again, Eq.(12) is too complex to make the computation feasible. In implementation, it is simplified as the following formula:

$$P\Big(LT = t_1, UT = (t_2, t_3), RT = t_4, L_{uw} = 2 \mid t_b, t_1^{N_u}, t_e, E_{uw} = 1\Big), \tag{13}$$

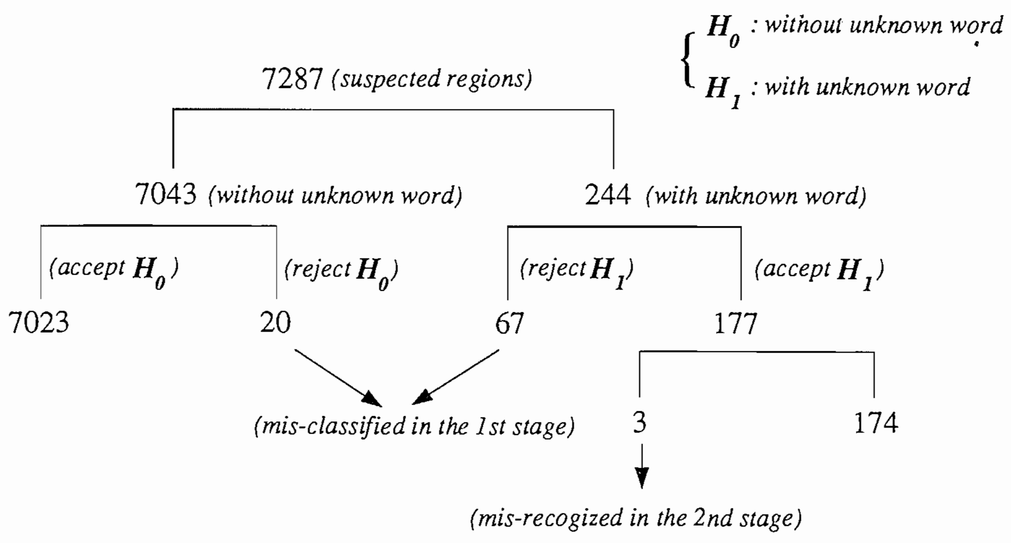
where L_{uw} is the random variable expressing the length (number of words to be combined) of the unknown word. In such a way, this model will only take into consideration the information related to the length of the unknown word, the lexical tags of its elementary word and its left and right contexts. Furthermore, Eq.(13) is derived as follows:

$$
\begin{aligned}
&P\Big(LT = t_1, UT = (t_2, t_3), RT = t_4, L_{uw} = 2 \mid t_b, t_1^{N_u}, t_e, E_{uw} = 1\Big) \\
&= P\Big(RT = t_4 \mid UT = (t_2, t_3), LT = t_1, L_{uw} = 2, t_b, t_1^{N_u}, t_e, E_{uw} = 1\Big) \\
&\quad \times P\Big(UT = (t_2, t_3) \mid LT = t_1, L_{uw} = 2, t_b, t_1^{N_u}, t_e, E_{uw} = 1\Big) \\
&\quad \times P\Big(LT = t_1 \mid L_{uw} = 2, t_b, t_1^{N_u}, t_e, E_{uw} = 1\Big) \\
&\quad \times P\Big(L_{uw} = 2 \mid t_b, t_1^{N_u}, t_e, E_{uw} = 1\Big) \\
&\approx P(RT = t_4 \mid UT = (t_2, t_3), L_{uw} = 2, E_{uw} = 1) \\
&\quad \times P(UT = (t_2, t_3) \mid LT = t_1, L_{uw} = 2, E_{uw} = 1) \\
&\quad \times P(LT = t_1 \mid L_{uw} = 2, E_{uw} = 1) \\
&\quad \times P(L_{uw} = 2 \mid E_{uw} = 1)
\end{aligned}
\tag{14}
$$

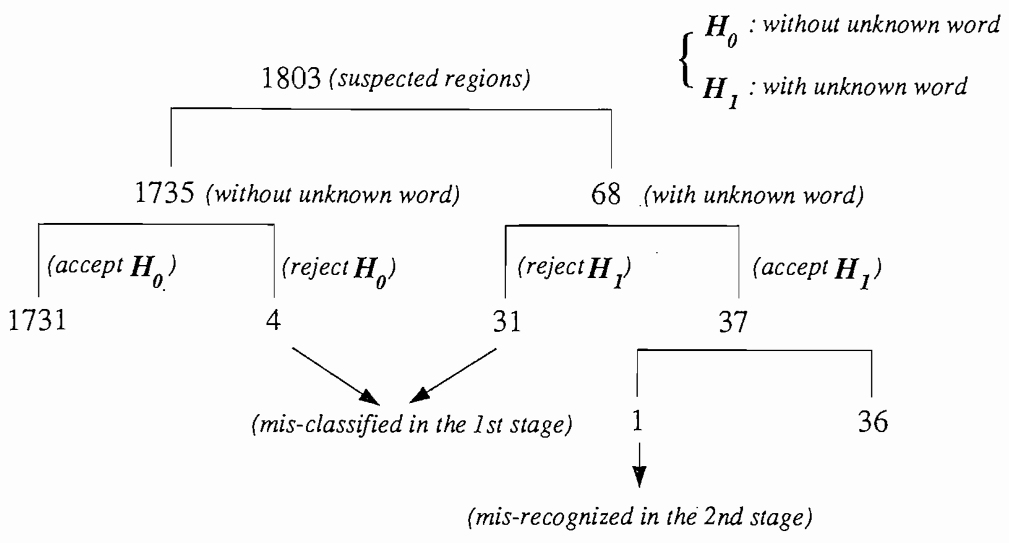
In a similar way, the scores corresponding to the other types of the unknown word in Figure 4 can be computed by analogy.

Experimental Results and Discussions

In the training corpus, there are 336 irregular unknown words, in which there are 247 double-character words, 69 tri-character words, and the rest 20 words are composed of over 3 characters. That is, at most 247 unknown words can be possibly identified through the model described above, for only the case of double-character words being considered in the simulation. Meanwhile, there are 89 irregular unknown words in the testing set, where there are 71 double-character words. The examination of the unknown word model is illustrated as follows.


H_0 : without unknown word
H_1 : with unknown word
7287 (suspected regions)
7043 (without unknown word)
244 (with unknown word)
(accept H_0)
(reject H_0)
(reject H_1)
(accept H_1)
7023
20
67
177
3
174
(mis-classified in the 1st stage)
(mis-recognized in the 2nd stage)

(a) Training Set.


H_0 : without unknown word
H_1 : with unknown word
1803 (suspected regions)
1735 (without unknown word)
68 (with unknown word)
(accept H_0)
(reject H_0)
(reject H_1)
(accept H_1)
1731
4
31
37
1
36
(mis-classified in the 1st stage)
(mis-recognized in the 2nd stage)

(b) Testing Set.

ILLUSTRATION 2
The illustration of the error types in the unknown word modeling.

In the above illustration, the null hypothesis H_0 is defined as follows:
H_0: There is no unknown word in the suspected region.
H_1: There is at least one unknown word in the suspected region.
Note that there are 247 double-character words in the training set, but only 244 single word regions containing unknown words, it implies that at most three of the suspected region include more than one unknown word. Nevertheless, it is reasonable for us to assume that there is only one unknown word in a single word region.

From the above illustration, it is observed that 87 errors arise from the first stage by using Eq.(11) to inspect the suspected region; that is, 1.19% (87/7,287) error rate is for the first stage. Aside from that, there are additional 3 identification errors imposed using Eq.(14) in the second stage; it means that there is 1.69% (3/177) error rate introduced by the second stage. Thus, there are total 90 errors, which correspond to 1.23% (90/7,287) error rate, resulting from using the current model to identify the unknown words in the training set.

With the taxonomy described above, the unknown words of class 3 and class 4 are what you are really interested in. But there are only 2 words belonging to class 4, i.e., geographical names, none for class 3 in the training set; what is more, none of unknown words in the testing set belongs to class 3 or class 4. In view of the recognition of unknown words in the training set, 174 of the total 247 unknown words, i.e., 70.44%, are identified correctly. However, the rest 70 ones are missed, and another 20 mis-combined errors are imposed through the unknown word model. In the testing set, 36 of the 71 unknown words are recognized correctly; it corresponds to 50.7% recognition rate. According to the above analysis, it is apparent that the errors are mainly introduced from the first stage. Therefore, to improve the performance of the model in the future, Eq.(11) should be modified.

The progressive results of the unknown word recognition procedure are summarized in Table 9. Compared with the baseline model, the error reduction rate of 78.34% in word and 81.87% in sentence are obtained with the unknown word recognition procedure.

		Error rate in the testing set	
	Computational Model	word (%)	sentence (%)
BS	$P(w_k \mid l_{k-1})$	7.48	44.16
BS+MA	$P(w_k \mid l_{k-1})$	1.81	10.70
BS+MA+TG	$P(w_k \mid t_k) \times P(t_k \mid t_{k-1})$	1.74	11.16
BS+MA+TG+UW	unknown word model	1.62	10.70

Note: BS: (baseline); MA: (morphological analysis); TG: (tagging); UW: (unknown word model).

TABLE 9
The progressive results in our approaches on unknown word recognition.

To examine our approaches in a more general task, we also test a corpus of newspaper (Free Times), which consists of 400 training sentences and 100 testing sentences; the results are shown in Table 10. From this table, the error reduction rate of 40.15% in word and 34.78% in sentence can be observed.

	Computational Model	Error rate in the testing set	
		word (%)	sentence (%)
BS	$P(w_k \mid l_{k-1})$	19.00	69.0
BS+MA	$P(w_k \mid l_{k-1})$	13.06	50.0
BS+MA+TG	$P(w_k \mid t_k) \times P(t_k \mid t_{k-1})$	12.21	52.0
BS+MA+TG+UW	unknown word model	11.37	45.0

Note: BS: (baseline); MA: (morphological analysis); TG: (tagging); UW: (unknown word model).

TABLE 10
The results on newspaper task.

Looking into the errors in more detail, 3 unknown words of class 3 and 22 ones of class 4 are identified correctly in the training set, where originally, there are 6 and 39 unknown words of class 3 and class 4 respectively. It means that 55.56% unknown words in these two classes are recovered. Actually, the 17 mis-recognized class 4 unknown words are all caused by the missing of the first stage. Hence, how to select more discriminative features in the first stage is a key issue to improve the model in our next work. On the other hand, 1 of 2 unknown words for class 3, and 5 of 6 unknown words for class 4 are recognized correctly in the testing set; it corresponds to 75% recognition rate for these two classes. Both these two errors are tri-character words that are not considered in the current models. Although the promising results have shown the superiority of the resolution procedure, the model proposed in this paper, however, only tackles a very restrictive form of unknown words. We will extend and modify the model to more general cases in the future.

7. Summary

Since we have shown in our previous work that the existence of unknown words is the main factor that causes the performance of word segmentation task to be unsatisfied, we, therefore, shift the focus to this issue in the paper. Unknown words are generally formed in terms of regular or irregular ways. First, in this paper, a set of 17 morphological rules are applied to recognize those regular unknown words. In addition, an unknown word model is further proposed to deal with the unknown words of irregular forms. With the unknown word resolution procedures, the error reduction rate of 78.34% in word and 81.87% in sentence are obtained in a task of technical manuals. To examine the procedures in more general task, a corpus of newspaper is also tested and the error reduction rate of 40.15% in word and 34.78% in sentence are observed.

Acknowledgement

The authors would like to express our gratitude to Miss Shu-Jun Ke (Behavior Design Corporation) for her efforts in preparing the morphological rules and corpus. Her works provide useful training and testing materials for verifying the proposed models. Special thanks are given to the Behavior Design Corporation for providing the Chinese-English Electronic Dictionary to this research project.

References

[BDC 92] Behavior Design Corporation, "The BDC Chinese-English Electronic Dictionary: Version 2," 1992.

[Chia 92a] Chiang Tung-Hui, Ming-Yu Lin and Keh-Yih Su, "Statistical Models for Word Segmentation and Unkonwn Word Resolution," *Proceedings of 1992 R.O.C. Computational Linguistics Conference (ROCLING V)*, pp. 121-146, Taipei, Taiwan, 1992.

[Chia 92b] Chiang Tung-Hui, Yi-Chung Lin and Keh-Yih Su, "Syntactic Ambiguity Resolution Using A Discrimination and Robustness Oriented Adaptive Learning Algorithm," *Proceedings of International Conference on Computational Linguistics (COLING '92)*, pp. 352-358, Nates, France, Jul. 23-28, 1992.

[Devi 82] Devijver, P. A. and J. Kittler, "Pattern Recognition: A Statistical Approach," *Prentice-Hall International Inc.*, 1982.

[Lin 93] Lin Ming-Yu, *A Study in Chinese Word Segmentation, master thesis*, National Tsing Hua University, Taiwan, R.O.C., 1993.

[Liu 93] Liu Yuan-Ling, Shih-ping Wang and Keh-Yih Su, "Corpus-based Automatic Rule Selection in Designing a Grammar Checker," to appear in *1993 R.O.C. Computational Linguistics Conference (ROCLING VI)*.

Appendix A　Morphological Rules

（1）附著在後一詞項的詞綴：共計3個。

1　總：「總　工程師」

2　主：「主　電腦板」

3　副：「副　處長」

（2）與前一詞項組合的詞綴：共計13個。

1　氏：「劉 氏」

2　師：「工程 師」

3　們：「同學 們」

4　族：「安公子 族」

5　器：「轉換 器」

6　員：「操作 員」

7　碼：「字元 碼」

8　鍵：「說明 鍵」

9　式：「階層 式」

10　極了：「美　極了」

11　多了：「好　多了」

12　的很：「好　的很」

13　得很：「快樂　得很」

（3）限定詞與數量詞的形成

例：一個個，一條條

規則表示法：$n \rightarrow q+cl+cl$

$a \rightarrow q+cl+cl$

（n：名詞；a：形容詞；q：數詞；cl：量詞）

規則說明：數詞+量詞+量詞可形成一個名詞或形容詞。

（4）日期與時間

例；八十二年，五月，十八日，六點，廿七分，卅九秒

規則表示法：n → q+ 年

n → q+ 月

n → q+ 日

n → q+ 點

n → q+ 分

n → q+ 秒

(5) 名詞的前綴（簡稱 npfx）

例：「非　產品」，「全　比例尺」

規則表示法：n → npfx+n

這一類的詞有：全、初、反、非、老，共計5個。

(6) 名詞的後綴（簡稱 nsfx）

例：「教育　性」，「叉　狀」

規則表示法：n → n+nsfx

這一類的詞有：性、家、狀，共計3個。

(7) 動詞的後綴（簡稱 vsfx）

例：「情緒　化」，「綠　化」

規則表示法：v → n+vsfx (v：動詞)

這一類的詞有：化，共計1個。

(8) 結果詞綴（簡稱 rsfx）

例：「跑　得」，「哭　得」

規則表示法：v → v+rsfx

這一類的詞有：得，共計1個。

(9) 趨向標誌（簡稱vdir）

例：「動　起來」，「哭　出來」，「吞　下去」，「拿　出」，「丟　下」

規則表示法：v → v＋vdir

這一類的詞有：上、下、回、出、進、過、起、來、去、上去、下去、下來、上來、回去、回來、過去、過來、進去、進來、出去、出來、起來，共計22個。

(10) 動相標誌（簡稱phase）

例：「喝　光」，「用　完」，「吃　掉」，「寫　好」，「叫　住」，「用　作」，「飛　向」，「走　向」

規則表示法：v → v＋phase

這一類的詞有：住、入、到、完、掉、好、過、光、開、做、作、成、爲、向，共計14個。

(11)～(13) 爲動詞的特殊句型

(11) 特殊動詞（一）

例：「丟丟　看」，「吃吃　看」，「寫寫　看」

規則表示法：v → v＋v＋看

說明：出現在「看」前的兩個動詞必須完全一樣。

(12) 特殊動詞（二）

例：「高高興興」，「歡歡喜喜」，「漂漂亮亮」，「迷迷糊糊」

規則表示法：v → va＋va＋vb＋vb

說明：va表動詞的前一成份，而vb表動詞的後一成分。

(13) 特殊動詞（三）

例：「打打　球」，「跑跑　步」，「寫寫　字」，「高高」，「瘦瘦」，「慢慢」，「黑黑」

規則表示法：v →v+v+n

v →v+v

說明：兩個動詞必須完全一樣。

(14) 動詞前綴（簡稱 vpfx）

例：「已　定義」，「反　革命」，「未　成年」

規則表示法：v →vpfx+v

這一類的詞有：反、未、已，共計3個。

(15) 形容詞前綴（簡稱 apfx）

例：「非　人」，「非　人性」

規則表示法：a →apfx+n

這一類的詞有：非，共計1個。

(16) 副詞後綴（簡稱advsfx）

例：「快樂　地」，「勉強　地」

規則表示法：adv →adv+advsfx

這一類的詞有：地，共計1個。

(17) 動詞中綴（簡稱vifx）不予合併

例：「快樂　不　快樂」，「同　不　同意」

規則表示法：v + vifx + v →v + vifx + v

這一類的詞有：不，共計1個。

國語語音辨認中
詞群雙連語言模型的解碼方法
A Word-Class Bigram Approach to Linguistic Decoding in Mandarin Speech Recognition

林頌堅[+] 簡立峰[*] 陳克健[*] 李琳山[+*]

[+]國立台灣大學資訊工程學研究所

[*]中央研究院資訊科學研究所

摘要

在國語語音辨認的語言解碼方法(linguistic decoding approach) 中，字雙連語言模型(character bigram) 和詞雙連語言模型(word bigram) 是兩種最常被使用的方法。其中，詞雙連語言模型在描述語言現象上有較字雙連語言模型強的能力；然而若詞彙量較大時，它所需要估計的參數卻遠較字雙連語言模型多。因此若考慮在大詞彙、無限文句的國語語音辨認應用上，此二者皆有其適用上的限制。本文乃提出一個詞群雙連語言模型(word-class bigram) 的語言解碼方法，這個方法是以對詞分群來大幅縮減參數量的大小，且又能接近詞雙連語言模型的辨認效果。分群方法是根據國語特殊的構詞特性來分群，比起其他西文常使用依據語法(syntactical)或語意訊息(semantical information) 的分群方法不但計算簡單，且同樣具有分群的效用，而且詞群並不必須事先訓練或做詞類標記(part-of-speech tagging) 即可決定，同時此法對新增詞及低頻率的詞也具某種程度的平滑(smoothing) 能力。此外為達成即時計算的要求，我們針對構詞及格狀詞組搜尋提出二項技術來加快它們的速度。目前這個方法已實際應用在國語語音辨認上，實驗結果證實這個方法是相當實用且有效的。

第一節　緒論

在國語語音辨認的應用中，所謂語言解碼(linguistic decoding) 的過程是指在一個由音節辨認(syllable recognition) 後所獲得的格狀詞組(word lattice) 或格狀字組(character lattice) 中找出最可能的文句。傳統上，詞雙連語言模型(word bigram) 或字雙連語言模型(character bigram) 都是常用的方法[1,2,3,4]。基本而言，這兩種方法各有優缺點[2]：詞雙連語言模型在描述語言現象上有較強的能力；然而若詞彙量較大的話，它所需要估計的參數卻遠較字雙連語言模型多。若是考慮在大詞彙、無限文句的語音辨認應用上，這兩種方法都並不十分適合。因此，一個參數量比詞雙連語言模型小很多，但卻比字雙連語言模型有效的語言模型及快速解碼方法是非常需要的。

近年來盛行於西文的將詞分群以減少模型參數量的方法似乎是一個可行的方法[5,6]。然而此類方法大多是依據詞的語法或語意訊息來分群，由於需要做詞類標記(part-of-speech tagging) [5] 或是事先需要利用語料庫來抽取其中的語法或語意訊息做為分群上的依據[6] ，所以分類的成本相當高，另外這類方法在面對新增詞大增時都可能必須重新分群訓練。因此本文根據中文的構詞特性提出一相當簡單的稱為「詞群雙連語言模型(word-class bigram)」的統計式語言模型。

這個模型的基本構想是中文具有相同起始字或結尾字的詞常有接近的語意，如一些複合詞或定量詞的中心語是在詞尾，像物理學、化學、科學，車、火車、機車，一個、十個、百個；而動補式動詞和動賓式動詞的中心語在詞頭，像打破、打壞、打爛、打棒球、打電話等。因此對所有的中文詞，我們可以考慮根據起始字和結尾字將它們分成若干群。分群的方法是對具有相同起始字的詞都歸成同一群，例如前述：打破、打壞、打爛等都歸成同一群S(打)。同樣的，對具有相同結尾字的詞，如前述：車、火車、機車等都歸成同一E(車)。因此，對於每一詞串(word sequence)，其機率估算方法，可以改以詞群的機率來計算；這種機率值的逼近方式相對於西文使用依據語法或語意訊息(syntactical or semamtical information) 的分群方法，不僅計算非常簡單，而且無須事先以語料庫訓練即可決定詞群，並且當新增詞大增時，詞群也不須重新認定。當然依

據這種方式，一些與語意差距甚大的詞也會歸於同一群，如：國中生、國語、國家等，因此在第二節裡我們提出更進一步的方法。根據我們對國語語音辨認的實驗發現，這種方法與字雙連語言模型相比較，因為它的詞群數目和中文字的數目一樣，所以它所需的參數量接近字雙連語言模型，但遠較詞雙連語言模型小。同時，此種方法在國語語音辨認上的效果比字雙連語言模型好。

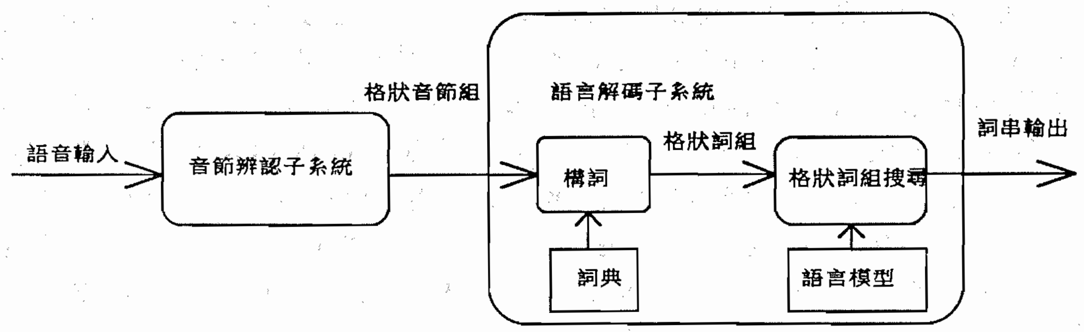
然而詞群雙連語言模型的語言解碼方法由於必須先構成格狀詞組，再從其中搜尋，構詞所花費的時間相當多。因此，我們特別建一個輔助表來及早刪除不可能成詞的音節串。如此一來，因為減少了多次不必要的詞典搜尋，使得構詞速度加快很多。另外我們從實驗觀察到，即使正確的詞不在最可能的一條詞串中被搜尋到，也很可能出現在前幾條詞串中。因此，若能使系統多搜尋幾條詞串輸出，如此有較多的詞可供選擇，一來便於人工修正；二來未來可加入語法或語意規則做為後處理，可以較詳細地過濾掉不合語法語意的文句。然而在多搜尋幾條詞串使正確的詞能出現的同時，為了不增加使用者和系統的負擔，我們也希望不正確的詞不會因此而增加太多。因此本文根據近來常用於連續語音辨認的樹狀一格狀最佳N條路徑搜尋法(tree-trellis N-best paths searching)[7] 修改成適合我們應用的一個在搜尋時間和記憶體用量都是最佳化的格狀詞組的多條路徑搜尋法。根據我們實驗的結果，運用此方法搜尋五條詞串比搜尋一條詞串可以多對了10%的字，但只需多找40%的字，並且速度仍然相當快。

這個語言解碼方法已實際應用在國語語音辨認中，並獲得了很好的結果。另外，因為辨認的結果是以詞串的形式輸出而非僅以字串的形式輸出，此法對需要以詞串做處理單位的其他更高階的自然語言處理(natural language processing)，如語音瞭解(speech understanding)、語音機器翻譯(speech machine translation) 和資訊檢索(information retrieval) 等應用有相當大的助益。

第二節　詞群雙連語言模型

通常國語語言辨認系統可分為音節辨認子系統(syllable recognition subsystem) 和語言解碼子系統(linguistic decoding subsystem) 兩部分，其架構

如圖一所示[1,3,4]。我們所提出的詞群雙連語言模型就是一種語言解碼的方法，它包括了兩個過程：構詞(word formation) 和格狀詞組搜尋(word lattice searching) 。所謂構詞是將一個由音節辨認所到格狀音節組(syllable lattice) 與詞典做匹配(matching) 而形成一個格狀詞組；而格狀詞組搜尋是根據語言模型(language model) 在所得到格狀詞組中搜尋一或多條可能的詞串做為輸出。



圖一　國語語音辨認系統架構圖

由於構詞和格狀詞組搜尋都要花費相當多的時間，尤其是當詞彙量非常大時，情形更是嚴重。因此在我們所提的語言解碼方法中特別加入一些輔助方法來加快它們的速度，這些方法在下一節中會有詳細的說明。此節中所要闡述的是前述我們所提語言模型—詞群雙連語言模型以及它的訓練方法。

機率計算

若產生一詞串，$W_1, \ldots, W_n$的機率為$Pr(W_1,\ldots,W_n)$，則

$$Pr(W_1,\ldots,W_n) = Pr(W_1) \times Pr(W_2|W_1) \times \ldots \times Pr(W_n|W_1,\ldots,W_{n-1}) \ldots (1)$$

由於上式所需的參數量過為龐大，前人的研究就作了一個馬可夫程序(Markov Process) 的假設，假設此詞串的產生是一個一階馬可夫模型(first-order Markov model) 的隨機程序(random process)，亦即每一個詞的產生只與它的前一個詞有關，因此：

$$Pr(W_1,\ldots,W_n) \cong Pr(W_1) \times \prod_{i=2}^{n} Pr(W_i|W_{i-1}) \quad \ldots(2)$$

此處$Pr(W_i \mid W_{i-1})$便是一個詞雙連語言模型的機率值。對每一個這樣的機率，如前節所說明的國語中複合詞的中心語在詞尾，因此大部分的詞，我們假設它可以用和前一詞的結尾字的條件機率(conditional probability)來逼近：

$$Pr(W_i|W_{i-1}) \cong Pr(W_i|E(W_{i-1})) \ldots(3)$$

此處$E(W_{i-1})$為具有與詞W_{i-1}相同的結尾字所有的詞所成的群。

若將每一個詞W_i視為由$S(W_i)$和$R(W_i)$兩部分組成，此處$S(W_i)$為詞W_i的起始字部分；而$R(W_i)$則是詞W_i除了起始字的部分，則由貝氏定理(Bayes' Theorem)，我們可以得到：

$$\begin{aligned} Pr(W_i|W_{i-1}) &\cong Pr(W_i|E(W_{i-1})) \\ &= Pr(S(W_i)R(W_i)|E(W_{i-1})) \\ &= Pr(S(W_i)|E(W_{i-1})) \times Pr(R(W_i)|E(W_{i-1})S(W_i)) \ldots(4) \end{aligned}$$

再假設$R(W_i)$的產生與前一詞W_{i-1}的結尾字$E(W_{i-1})$無關，而又因為事件$S(W_i)R(W_i)$產生的機率與事件W_i的機率相同，所以：

$$\begin{aligned} Pr(W_i|W_{i-1}) &\cong Pr(S(W_i)|E(W_{i-1})) \times Pr(R(W_i)|S(W_i)) \\ &= Pr(S(W_i)|E(W_{i-1})) \times \frac{Pr(S(W_i)R(W_i))}{Pr(S(W_i))} \\ &= Pr(S(W_i)|E(W_{i-1})) \times Pr(W_i|S(W_i)) \quad \ldots(5) \end{aligned}$$

式(5)便稱為詞群雙連語言模型的機率值。在此式中，前一項在具有相同結尾字的詞所成的集合出現之下，接連出現相同起始字的詞所成的集合的條件機率；而後一項為詞在具有與它相同起始字的詞所成的集合的條件機率。

參數量的評估

針對式(5)中$Pr(S(W_i) \mid E(W_{i-1}))$的計算，我們需先對訓練語料斷詞(word segmentation)，接著統計事件$E(W_{i-1})$和事件$E(W_{i-1})S(W_i)$出現的頻率：

$$Pr(S(W_i)|E(W_{i-1})) = \frac{Pr(E(W_{i-1})S(W_i))}{Pr(E(W_{i-1}))} \cong \frac{f(E(W_{i-1})S(W_i))}{f(E(W_{i-1}))} \quad ...(6)$$

同樣地，$Pr(W_i \mid S(W_i))$也是如此，但是由於事件$W_iS(W_i)$和事件W_i出現的機率相同，所以：

$$Pr(W_i|S(W_i)) = \frac{Pr(S(W_i)W_i)}{Pr(S(W_i))} = \frac{Pr(W_i)}{Pr(S(W_i))} \cong \frac{f(W_i)}{f(S(W_i))} \quad ...(7)$$

由上面兩式我們來估計詞群雙連語言模型的參數量。每一個參數由是由式(6)和式(7)兩部分組成。對式(6)而言，參數量的大小類似字雙連語言模型，但由於並不是每一個中文字都可做為詞的起始字或結尾字，因此式(6)的參數量應比字雙連語言模型稍少一些；而式(7)的參數量則與詞典大小有關，但相較於式(6)的參數量則顯得微不足道。因此，這個模型的參數量與字雙連語言模型參數量接近，但遠少於詞雙連語言模型的參數量。

其他問題的討論

以下我們針對語言模型常發生的問題來進一步討論。第一個問題是在詞雙連語言模型中由於要估計的參數量太大，所以有許多參數在語料庫中沒有出現或者是出現頻率太低，而使所估計的語言模型不準；針對這個問題由於詞群雙連語言模型所需估計的參數量比詞雙連語言模型小很多，而且分群具有平滑的效果，因此可以適度減輕這個問題。

其次這個語言模型由於是以詞的起始字和結尾字做為分群的依據，不需事先做詞類標記或從語料庫中抽取語法或語意訊息，因此不但運算簡單使訓練成本大大減低，同時若是有新增詞出現，由於可以自動分群並不需重新分群訓練。

此外這個語言模型具有比字雙連語言模型還強的語言現象描述能力，因此它的語言解碼的正確率比較高，這一點在第四節中將以實驗來佐證。但由於這個語言解碼方法必需先構詞，使得它花費的時間比較多，因此第三節我們將提出改進的方法。

第三節　構詞與格狀詞組搜尋

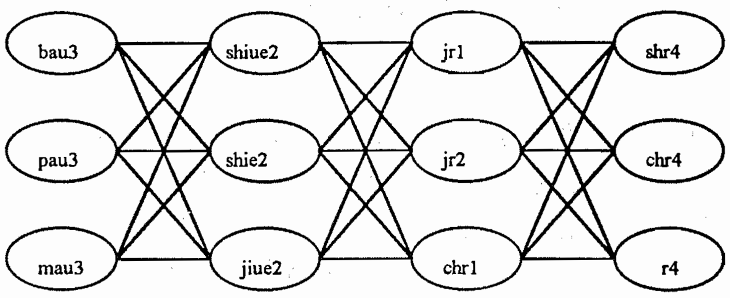
利用前節所提之語言解碼方法應用於國語語音辨認，當格狀音節組進入語言解碼子系統之後(圖一)，首先需經過構詞程序建構出一個相對應的格狀詞組，之後格狀詞組搜尋程序則在這個格狀詞組中搜尋出若干條可能的詞串輸出。由於這兩個程序都要花費相當的時間，所以針對這兩項速度的瓶頸，我們進而提出兩項技術使我們所提的語言解碼方法更為完整。

快速構詞法 (fast word formation)

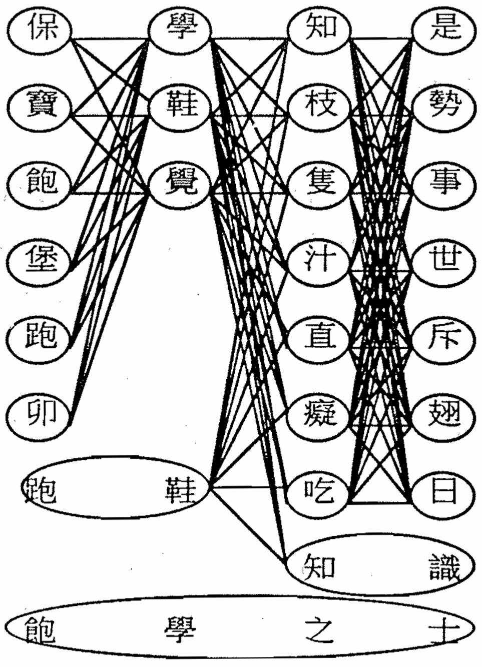
構詞程序將某一給定的格狀音節組轉換成相對應的格狀詞組，如圖二所示。方法是查尋格狀音節組中每一可能的音節串(syllable sequence) 在詞典中有無對應的詞，每一音節串是由在格狀音節組有前後關係的音節所構成的鏈結(link) 串接起來。但由於在格狀音節組中會有很多不可能成詞的音節串（尤其是當音節串愈長時，愈會有這樣的情形發生），造成很多不必要的詞典查尋。若能降低這些不必要查尋的次數，將使構詞的速度加強不少。

在這裡，我們的方法是使用一個輔助表來及早去除不可能成詞的鏈結；這個輔助表是由一組二元向量(binary vector) 組成；每一個二元向量代表每個音節在所有長度的詞中可能出現的位置：向量的第一個成分(component) 代表這個音節可否出現在單字詞，第二個成分代表這個音節可否出現在雙字詞的第一個字，第三個成分則是代表可否出現在雙字詞的第二個字，以此類推(圖三)。當要形成一個長度K的音節串S_1，...，S_K時，若在已形成長度(i-1)的鏈結S_1-S_2...-S_{i-1}串後，發現下一音節S_i在代表K字詞的第i個位置的二元向量的成分為0，即此音節S_i不能出現在K字詞的第i個字，則之後的鏈結便都不需形成。

舉例而言，如圖二(a)之格狀音節組，mau3[1]並不會出現在4字詞的第一個字，所以像mau3-shiue2-jr1-shr4， mau3-shiue2-jr1-chr4， mau3-shiue2-jr1-r4，...等鏈結便都不用形成。使用這個輔助表之後，大約減少了90%的詞典查尋，使得即時應用成為可能。



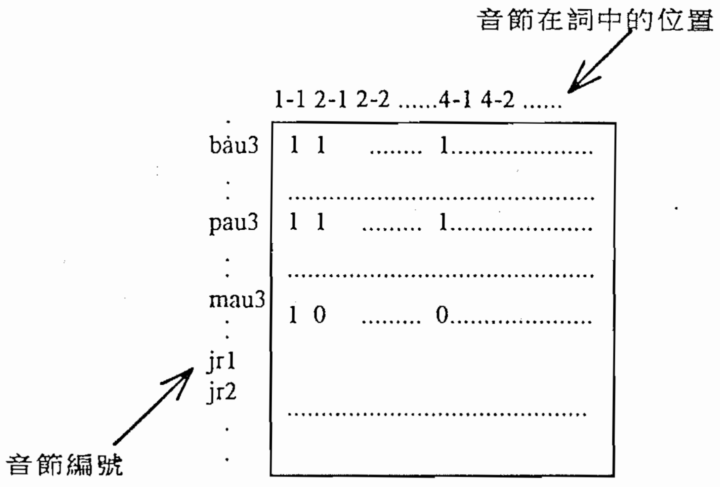
(a)



(b)

圖二　一個格狀音節組及其相對應的格狀詞組。(a)格狀音節組，(b)格狀詞組。

[1] 本文所採用之注音符號為教育部於民國七十五年頒布之國語注音符號第二式。



圖三 快速構詞法的輔助表示意圖

快速格狀詞組多路徑搜尋法(fast word lattice multiple path searching)

傳統使用維特比搜尋法(Viterbi's search) 的國語語音辨認系統的語言解碼子系統，為了節省記憶體空間和搜尋時間，通常只搜尋一條最可能的文句，作為結束；而辨認錯誤之處則由使用者自行鍵入中文字來更正。但我們認為一個對使用者友善(user-friendly) 的系統應該要減少使用者使用鍵盤更正的機會，所以需要一個能搜尋多條可能詞串的格狀詞組搜尋法。這樣一個能同時搜尋多條可能詞串的方法，除了可以提供給使用者更方便的環境外，未來這些詞串也才可以用需要較多運算的語法或語意規則來挑出最可能的詞串。同時在系統的建立時期，更可提供給設計者更多有關系統的能力和優缺點的資訊，進而加強系統的功能。

過去語言解碼系統只搜尋一條可能文句的原因主要是由於缺乏一個快速、使用記憶體少而有效的多條路徑演算法。在此，我們參考了莊、黃二氏為連續語音辨認(continuous speech recognition) 所提出的樹狀一格狀最佳N條路徑搜尋法(tree-trellis N-best search) [7]，經過一些修改後，將其運用在格狀詞組的最佳N條可能的詞串的搜尋上。這個演算法是一個兩階段式的搜尋法：第一階段是正向的經過修改的維特比搜尋法(forward modified Viterbi's search)，第二階段是一個回向的A*演算法(backward A* algorithm) ，而此回向A*演算法與其他A*演算法最大的不同就是其他A*演算法的優先值是一個預測值，而這一個回向A*演算法的優先值則是真

正經過此節點和回向未完成路徑的最佳分數，所以這個A*演算法是A*一可容許的(A*-admissible)[8]，因此它的記憶體用量和所花費的時間都是最經濟的。在我們的實驗中發現，若搜尋五條可能的詞串，可以比只搜尋一條詞串多對10%的字，而卻只要多找40%的字，這正是我們所要求的，能多辨認出一些正確的字，而多找出來的字又不造成太大的干擾。

第四節　實驗環境及初步實驗結果

為驗證所提出的語言解碼方法的效果，我們做了一系列在國語語音辨認應用的實驗，來比較它與傳統所採用的字雙連語言模型方法在辨認中文字上的效果。同時，我們觀察了一些辨認錯誤的結果，做為日後加強的參考。實驗結果顯示我們的系統在字的回叫率(recall rate) 及精確率(precision rate) 上都比傳統所使用的字雙連語言模型來得高。接下來在說明實驗結果之前，我們先介紹使用的實驗環境，包括音節辨認子系統、詞典以及訓練和測試語料。

音節辨認子系統

辨認結果	基底音節辨認	聲調辨認
前一個結果	93.87%	98.04%
前二個結果	99.51%	99.35%
前三個結果	100.00%	100.00%
前四個結果	100.00%	N.A.
前五個結果	100.00%	N.A.

表1　本實驗的音節辨認子系統的平均辨認率 N.A.代表NonAvailable

如前所述，語言解碼子系統從音節辨認子系統取得格狀音節組作為輸入，然後轉換出可能的文句輸出。實驗中，所使用的音節辨認演算法是根據金聲二號國語聽寫機的演算法[4]。此音節辨認子系統可分為兩部分：基底音節辨認(base syllable recognition) 及聲調辨認(tone recognition)。對每一個輸入音節的語音，基底音節辨認自408個基底音節中選5個最有可能的候選音節；而聲調辨認4個聲調(lexical tone)和1個輕聲調(neutral tone)中

選出3個候選聲調。然後，將這兩組結果加以組合，去除掉不可能的基底音節和聲調組合後，送至語言解碼子系統做為輸入。平均的音節辨認率如表1所示。

詞典及訓練和測試語料

本實驗中所使用的詞典是由中央研究院詞知識庫小組所研發[9]，但為配合語音辨認上的需要，我們做了一些修改，包括：1.由於太長的詞在日常生活中很少使用，所以我們將五字以上的詞刪除。2.目前專有名詞的處理是視做一串單字詞來處理；有些字並沒有出現在詞典的單字詞中，但它們有可能出現在專有名詞之中。為了這緣故，我們將所有的中文字都納入作為單字詞。經過這樣的修改之後，我們的詞典共包括八萬多個中文詞，其不同長度中文詞的個數如表2所示。

	詞數
單字詞	14052
雙字詞	48339
三字詞	11559
四字詞	10433
五字詞	583
總和	84966

表2　詞典中不同長度的中文詞數目

為了訓練我們所提出的語言模型，需要經過斷詞處理的大量中文文章做為訓練語料。此處我們的訓練語料亦為中央研究院詞知識庫小組所發展[10]；共包括三家主要的報紙，即中國時報、聯合報及自由時報，約400萬個中文字。相信這對字雙連語言模型以及詞群雙連語言模型的訓練大致是足夠的。

因為所需語料量相當大，所以若以人工來斷詞的話，不但是件繁瑣的工作，而且要花費相當多的時間。因此，勢必要採用自動斷詞系統。在我們的實驗裡，採用的是陳、劉二氏所提出並發展的斷詞系統[11]。這個斷詞系統基本上是一個字串匹配演算法(string matching algorithm)加上

六條解決當某一段字串可以被斷成一種以上的詞串的情況的經驗法則。它的成功率可達99.77%，也就是說在訓練語料中仍有少許的雜訊(noise)。

而實驗所用的測試文章則是從三種不同文體的題材中隨機挑選出來，共有四篇從報紙中選取的新聞(共2309字)、一篇選自天下雜誌的短文(1548字)，以及一篇短篇小說「明還」(970字) (選自「人子」，鹿橋著，台北市，遠景出版公司出版)，這些都是由不包括訓練語料的語料庫中隨機抽取出來的。這三種文體相信已包含了日常生活中常用的詞彙和句型，應該可以瞭解我們所提出的方法的效果。

回叫率及精確率的定義

在本實驗中，字辨認的效果是由回叫率(recall rate)及精確率(precision rate) 來評估，其定義如下：

假設 C_j為測試文句的第j個字
H_{ij}為經搜尋後第i條可能詞串的第j個字

則在搜尋i 條可能詞串後，回叫率R_i的定義為：

$$R_i = \frac{\sum\limits_{for\ all\ sentences} \sum\limits_{j=1}^{n_s} \left| \left(\bigcup\limits_{k=1}^{i} \{H_{kj}\} \right) \cap \{C_j\} \right|}{\sum\limits_{for\ all\ sentences} n_s} \times 100\%$$

而精確率P_i的定義為：

$$P_i = \frac{\sum\limits_{for\ all\ sentences} \sum\limits_{j=1}^{n_s} \left| \left(\bigcup\limits_{k=1}^{i} \{H_{kj}\} \right) \cap \{C_j\} \right|}{\sum\limits_{for\ all\ sentences} \sum\limits_{j=1}^{n_s} \left| \bigcup\limits_{k=1}^{i} \{H_{kj}\} \right|} \times 100\%$$

此處，n_S為測試文句的長度，而|A|為集合A的元素個數。

上面兩式中，分子部分是對文章中每一句的每一字從搜尋到的第一條詞串到第i 條詞串所辨認的字做聯集，然後與正確的字交集。因此，若正確的字不出現在辨認出來的字的聯集中，則為空集合，亦即集合元素個數為0，反之元素個數為1，然後再總和整篇文章。R_i的分母部份是文章的長度；P_i的分母部份是所有搜尋到的字數。所以回收率是辨認正確的字數佔整篇文章的比率；而精確率是正確的字數佔全體找出字數的比率。由於是斷開音節(isolated syllable) 的國語語音辨認，所以不會有插入(insertion) 和刪除(deletion) 字的情況發生，因此第一條搜尋出的可能詞串的字數與所輸入文句的字數相等，也就是在搜尋第一條詞串後，其回叫率R_1應該等於精確率P_1。

舉例而言，如果要辨認的句子是「維持現有名額」，而經搜尋後得到第一條和第二條詞串分別是「維持現有明德」和「維持現有名額」，則此時回叫率和精確率的計算如下：在此n_s=6，而第一條詞串辨錯兩個字，所以 $\sum_{j=1}^{n_s}\left|\left(\bigcup_{k=1}^{i}\{H_{kj}\}\right)\bigcap\{C_j\}\right| = 4$，$i = 1$，而 $\sum_{j=1}^{n_s}\left|\left(\bigcup_{k=1}^{i}\{H_{kj}\}\right)\right| = 6$，$i = 1$，因此回叫率$R_1 = 4/6*100\% = 66.67\%$，精確率$P_1 = R_1 = 66.67\%$；而第二條詞串多辨出兩個字，所以 $\sum_{j=1}^{n_s}\left|\left(\bigcup_{k=1}^{i}\{H_{kj}\}\right)\bigcap\{C_j\}\right| = 6$，$i = 2$，而 $\sum_{j=1}^{n_s}\left|\left(\bigcup_{k=1}^{i}\{H_{kj}\}\right)\right| = 8$，$i = 2$，因此回叫率$R_2 = 6/6*100\% = 100\%$，精確率$P_2 = 6/8*100\% =75\%$。

在傳統語音辨認實驗之中，通常正確率是指第一條可能詞串的回叫率。但正如第三節所指出的我們認為一個對使用者友善(user-friendly) 的系統，應該讓使用者減少鍵入的次數，除了能搜尋多條可能的詞串之外，我們也希望系統所找出的正確字數愈多愈好而不需要的字不會因此增加太多。換言之，除了回叫率之外，精確率也是一項重要的評估標準。底下的實驗結果顯示，我們所提出的方法確實符合我們的要求。

測試結果

正如前面所提到的，對每一段音節的語音輸入，音節辨認子系統找出五個候選音節及最多三個候選聲調作為語言解碼子系統的輸入。此外，對每一輸入文句的語音，語言解碼子系統針對不同的應用可以找出不同數目的可能文句輸出。因此，可以形成很多組不同的實驗參數。底

下我們說明其中一些值得探討的現象，其結果顯示在表3至表6，表中包括了兩種不同的音節辨認結果及兩種語言解碼方法辨認三種不同文體的測試文章所得到的回叫率及精確率。

	新聞		雜誌		短篇小說	
模型	回叫率	精確率	回叫率	精確率	回叫率	精確率
字雙連語言模型	83.61%	83.61%	83.20%	83.20%	77.73%	77.73%
詞群雙連語言模型	87.93%	87.93%	87.92%	87.92%	77.73%	77.73%

表3 兩種不同的語言解碼方法測試三種不同文體的平均回叫率和精確率，實驗參數為取一個正確的聲調和五個候選基底音節及只找一條可能的詞串。

	新聞		雜誌		短篇小說	
模型	回叫率	精確率	回叫率	精確率	回叫率	精確率
字雙連語言模型	90.08%	64.38%	92.25%	64.50%	86.39%	59.18%
詞群雙連語言模型	93.40%	66.85%	95.16%	67.72%	86.29%	60.65%

表4 兩種不同的語言解碼方法測試三種不同文體的平均回叫率和精確率，實驗參數為取一個正確的聲調和五個候選基底音節及找五條可能的詞串。

	新聞		雜誌		短篇小說	
模型	回叫率	精確率	回叫率	精確率	回叫率	精確率
字雙連語言模型	82.75%	82.75%	83.07%	83.07%	77.42%	77.42%
詞群雙連語言模型	87.11%	87.11%	87.73%	87.73%	77.32%	77.32%

表5 兩種不同的語言解碼方法測試三種不同文體的平均回叫率和精確率，實驗參數取三個候選聲調和五個候選基底音節及只找一條可能的詞串。

	新聞		雜誌		短篇小說	
模型	回叫率	精確率	回叫率	精確率	回叫率	精確率
字雙連語言模型	89.31%	63.82%	92.12%	64.41%	86.19%	59.08%
詞群雙連語言模型	92.67%	66.18%	95.03%	67.57%	87.01%	61.20%

表6 兩種不同的語言解碼方法測試三種不同文體的平均回叫率和精確率，實驗參數為取三個候選聲調和五個候選基底音節及找出五條可能的詞串。

從這些表中，我們可以發現所提出的詞群雙連語言模型語言解碼方法所得到的辨認結果較傳統用字雙連語言模型的方法還要好。同時，可以發現在三種不同文體之中，以「短篇小說」的辨認結果最差，相信這是由於我們用新聞作為訓練語料，而這篇短篇小說的文體和詞彙與新聞相去甚遠的緣故，若是訓練語料中能包括這種文體，結果應該不會太差。

錯誤結果觀察

在這裡，我們觀察了一些詞群雙連語言模型語言解碼辨認錯的結果，這些錯誤可分為五類。第一類是由於雙連語言模型的影響。這類錯誤的主要原因是要辨認的前後詞在訓練語料中接連出現的次數較少，使得這兩個詞一起出現的機率很低。

第二類錯誤是詞在詞群的相對頻率太低而引起。如jia1 ru4會被辨認成「掐住」的原因，是因為「加入」在S(加)中的相對頻率太低。

第三類錯誤是長詞遮蓋短詞，例如：he2 li3被辨認成「合理」而不是「河　裡」，而shr2 jian1被辨認成「時間」而不是「十　間」。因為同樣字數的文句中，長詞比短詞少乘了好幾項機率，所以長詞的機率比較容易來得高。

第四類錯誤則是專有名詞和簡稱。由於每天都會有許多專有名詞產生及消失，所以我們不可能也沒有必要將所有的專有名詞收入詞典中。但由於我們的測試資料包括了新聞類的文體，所以錯誤的專有名詞和錯誤的簡稱情形相當嚴重。

最後一類錯誤是同音且語意接近的詞，例如：「他」、「她」、「它」。要校正這種錯誤，只靠語法的統計訊息是不夠的，還要加入更高層的語言訊息，如語意(semantic)和語用(pragmatic)等。有些錯誤為同音而且同義的詞，如「充分」、「充份」和「保母」、「保姆」、「褓姆」等。這種錯誤嚴格來說並不能算是錯誤，因為事實上或許根本沒有標準答案。

第五節　結論

在國語語言辨認的語言解碼方法中，字雙連語言模型和詞雙連語言模型是兩種最常被使用的方法。其中，詞雙連語言模型因較描述語言現象的能力，所以辨認效果較好，然而所使用的參數量較大；相反的，字雙連語言模型的參數量較小，而辨認效果卻較差。在大詞彙、無限文句的國語語音辨認的應用上，這兩個方法都有其適用上的困難。本論文提出一個詞群雙連語言模型的語言解碼方法。這個方法包括一個新的語言模型—詞群雙連語言模型並且提出兩項技術來克服速度上的瓶頸：以一個輔助表來及早去除不可能的成詞的音節串，使構詞的速度加快；而以一個兩階段式的格狀詞組多路徑搜尋法用最經濟的記憶量和時間找到最佳的N條可能詞串輸出。

我們所提出的語言模型是一種以對詞分群的方法，用來大幅減少模型的參數量，且仍然接近詞雙連語言模型的效果。這種分群的方法是根據國語特殊的構詞特性，用詞的起始字和結尾字做為分群的依據，這種根據中文特有語彙訊息的方法不僅具有其他分群方法在節省記憶體空間及對低頻的詞平滑化的優點，同時計算上相當方便：經實驗證明辨認結果也很理想。

實驗結果顯示，在八萬多個詞彙和400萬個字的訓練語料的系統中，詞群雙連語言模型對三種測試文體的辨認結果都比字雙連語言模型好。因此，我們所提出的方法是相當實用而有效的。此外，若將這個語言解碼子系統的輸入和構詞程序加以修改，這個語言解碼方法便可適用於其他類似的應用，如注音輸入法的音轉字方法及光學中文字形識別的後處理。

致謝

在此感謝台大資訊所楊燕珠同學、張元貞同學和楊榮荃同學幫忙撰寫所需程式及提供給我們寶貴的意見，以及中央研究院詞知識庫小組提供語料庫和詞典。

參考文獻

[1] L. S. Lee, C. Y. Tseng, H. Y. Gu, K. J. Chen, F. H. Liu, C. H. Chang, S. H. Hsieh, C. H. Chen, "A Real-time Mandarin Dictation Machine for Chinese Language with Unlimited Texts and Very Large Vocabulary," ICASSP'90, pp.65-68.

[2] H. Y. Gu, L.Y.Tseng, and L. S. Lee, "Markov Modelling of Mandarin Chinese for Decoding the Phonetic Sequence into Chinese Characters," Computer Speech and Language, Vol. 5, NO. 4, Oct. 1991, pp.363-377.

[3] L. S. Lee, C. Y. Tseng, H. Y. Gu, F. H. Liu, C. H. Chang, Y. H. Lin, Y. Lee, S. L. Tu, S. H. Hsieh, C. H. Chen, "Golden Mandarin (I) - A Real-time Mandarin Dictation Machine for Chinese Language with Very Large Vocabulary," to appear in IEEE Trans. on Speech and Audio Processing, Vol.1, NO. 2, Apr 1993.

[4] L. S. Lee, et. al., "Golden Mandarin (II) - An Improved Single-chip Real-time Mandarin Dictation Machine For Chinese Language with Very Large Vocabulary," ICASS'93, pp. 503-506.

[5] A. Derouault and B. Merialdo, "Natural Language Modeling for Phoneme-to-Text Transcription," IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-8, NO. 6, Nov. 1986, pp.742-749.

[6] P. F. Brown, V. J. Della Pietra, D. V. deSouza, J. C. Lai, and R. L. Mercer, "Class-Based n-gram Models of Natural Language," Computational Linguistics, Vol. 18, NO. 4, 1992, pp.467-479.

[7] F. K. Soong and E. Huang, "A Tree-Trellis Based Fast Search for Finding the N-Best Sentence Hypotheses in Continuous Speech Recognition," ICASSP'91, S10.5, 1991, pp.705-708.

[8] N. J. Nilsson, Principles of Artificial Intelligence. Tioga Publishing Company, California, 1980.

[9] 黃瑞珠, 詞彙檔案編輯及維護系統使用手冊. 中央研究院詞知識庫小組，技術手冊編號CKIP-93-07, 1993.

[10] C. R. Huang and K. J. Chen, "A Chinese Corpus for Linguistic Research," Proceeding COLING'92, Vol. 4, 1992, pp. 1214-1217.

[11] K. Chen and S. Liu, "Word Identification for Mandarin Chinese Sentences," Proceedings of COLING'92, Vol. 1, 1992, pp.23-28.

Corpus-based Automatic Rule Selection in Designing a Grammar Checker

[†]Yuan-Ling Liu, [†]Shih-ping Wang
[‡]Keh-Yih Su

[†] Behavior Design Corporation
2F, 28, R&D Rd II
Science-based Industrial Park
Hsinchu, Taiwan 300, R.O.C.

[‡] Department of Electrical Engineering
National Tsing-Hua University
Hsinchu, Taiwan 30043, R.O.C.

[†]wsp@bdc.com.tw

[‡]kysu@bdc.com.tw

Topic Area: Grammar Checker; Key Words: Corpus & Sequential Forward Selection (SFS)

ABSTRACT

In designing grammar-checking systems, the pattern matching algorithm, although failing to handle complex errors, is still widely adopted today. This is because when compared with the method of employing full scale parsing, pattern matching is efficient in detecting local errors with much less computer time and memory. However, the patterns used in the pattern matching approach are usually hand-tuned, and thus suffer from inadequacy in handling correlations among patterns. These error patterns may conflict or overlap with each other. Therefore, an automatic rule selection method, called ***Sequential Forward Selection (SFS)***, is proposed in this paper to tackle these problems. SFS uses objective performance measures to automatically search the suboptimal rule-set from all the possible combinations of rules. With SFS, the effectiveness of each rule can be measured, and problematic patterns can be identified systematically and efficiently for the linguist to fine-tune. Therefore, the error patterns can be revised efficiently. In our tests based on a corpus of 1956 sentences, the false rate decreases by 11.8% (from 26.4% to 14.8%) if the suboptimal rule set (81) selected by SFS is adopted, instead of the whole rule set (127). With this suboptimal rule set, the recognition rate decreases only by 3.9% (from 38.9% to 35%).

I. Introduction

The research of grammar checking has long been an attractive area in computational linguistics. Its main function is to detect grammatical mistakes. To detect grammatical errors, various algorithms have been proposed in the past decade, such as pattern matching (e.g., Kay, 1987), partial match

(Pfaltz et al., 1980), short range grammar verifications, syntactic parsing (cf. Vergne et al., 1986), etc. All these approaches can be generally classified into (1) ***pattern matching*** and (2) ***syntactic parsing***. Although the parsing method can solve the problem of long distance dependency at the syntactic level, it is time-consuming and occupies too much disk space. Moreover, parsing does not always yield the correct results. False alarms and missing error rates still exist in parsing. On the other hand, pattern matching is used to detect the grammatical errors without parsing (cf. Atwell, 1987). It can search the "desired shape" with local distance dependency. Although it usually fails to catch complex errors, many local grammatical errors still can be detected effectively. It can also save time and operating costs. Therefore, this approach is still largely adopted and currently used in our system, i.e., **B**ehavior **D**esign **C**orporation-Grammar Checker (hereafter, BDC-GC).

However, the patterns used in the pattern matching approach are usually hand-tuned, which suffer the following problems:

(1) It is not easy to manage the correlation among a bunch of rules. That is, we are not sure whether rules conflict or overlap with each other or not.

(2) As different applications might have various requirements and characteristics, the best rule sets for different applications are usually different. For example, grammatical errors made by Chinese are different from those by native speakers of English. However, there is no systematic approach to revise the patterns for various implementations.

(3) It is difficult to identify the effectiveness of each rule and to pinpoint the problematic rules systematically and effectively.

Therefore, an automatic rule selection method, i.e., ***Sequential Forward Selection (SFS)***, is proposed in this paper to tackle those problems. Given this SFS, we can (i) automatically find the suboptimal rule sets for different applications, (ii) objectively measure the effectiveness of each rule and (iii) systematically identify the problematic rules to let linguists revise them. Thus, the goal of a smaller set of patterns but better performance can be achieved. As a result, it takes 195 seconds (originally 269 seconds) to check 1956 sentences. If the suboptimal rule set (81) selected by SFS is adopted, not the whole rule set (127), the false rate decreases by 11.8% (from 26.4% to 14.8%) . However, the recognition rate decreases only by 3.9% (from 38.9% to 35%).

II. The Framework of BDC-Grammar Checker

A. The Construction of Error Patterns

Our error patterns of current version consist of 127 rules/patterns which are constructed by a linguist. They are based on common mistakes found from Chinese students' compositions and references (cf. C-L Su, 1991, Strunk et al., 1979, among others). These patterns have been encoded in terms of Arabic numerals (about 600 code numbers). An example of such a pattern is represented in the second line of Table 1.

/*2.1.2.1 Fragment*/	(error code & error type)
% (#) (,) although (!-1 % (#) (,) * [v]|be * , * [v]|be	(error pattern/condition)
/* [Advice]: This sentence needs a main clause. */	(advice)
/* [Example]: Although the weather was bad. */	(example)
/* [Correction]: Although the weather was bad, he went hunting. */	(correction)

Table 1 An Example of ERROR PATTERN

A close look at the error pattern shows that our pattern also includes the part of speech (Lin et al., 1992; cf. Church, 1988). With the aid of the part of speech, many different words can be clustered into various equivalent classes to formulate more concise rules. The number of rules/patterns, thus, can be significantly reduced to save time and space. Therefore, when compared with other systems such as RightWriter's large (±6,500) rule base (Brace, 1992), our rule size seems small. However, in our pilot test, it performs even better than several other tools (please see Section III). Additionally, it is allowable for linguists to write patterns which include some special symbols such as {#, %, *, (), |, [ ]} ('#': one token;'%': syntactic boundary; '*': zero to many tokens; '()': optional; '|': or; '[ ]': categorical brackets).

B. The Construction of Finite State Automata & the Operating Flow

To put these error patterns to real use, they must be converted into Finite State Automata (FSA, cf. Hopcroft et al., 1979). First, patterns with special symbols are converted into regular expressions which are then converted into FSA (Karttunen, L. et al., 1992 among others). This FSA includes a finite set of both states and transitions from state to state at input symbols, as shown in Figure 1a.

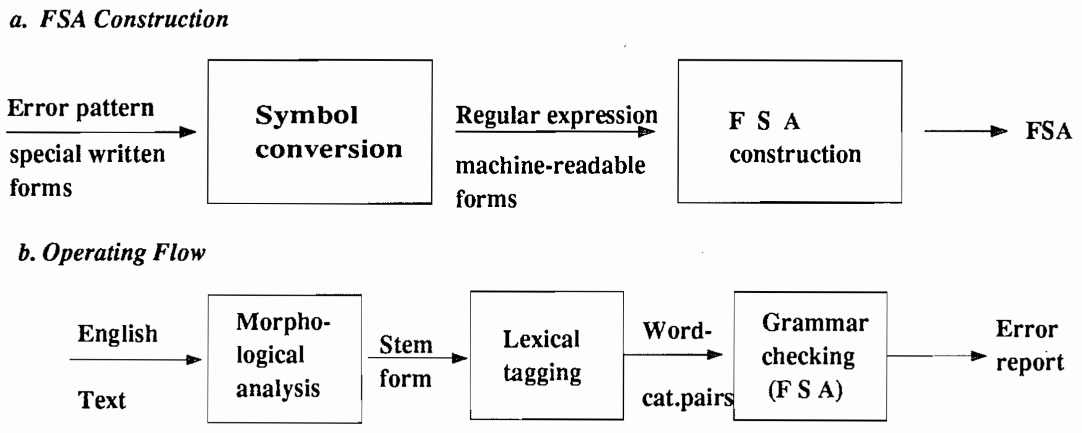


Figure 1 The flowchart of BDC-GC

A close look at the top flow in Figure 1a reveals that the special written forms such as {#, %, *, (), |, [ ]} are first converted to regular expressions. They are then transferred into machine-readable forms for the realization of FSA, which is the kernel of BDC-GC.

To illustrate the basic concept of our method, the operating flow of GC is shown in Figure 1b. Three stages of operation used to operate grammar-checking are stipulated as follows: (i) morphological analysis, (ii) lexical tagging and (iii) grammar checking. Let's take the sentence in Table 2 as an example. The surface form of a singular verb *influenced* is decomposed to the stem form '*influence*' with suffix ***ed*** through the morphological analysis. After the categorical tagging, it becomes a *word-category pair* with category ***v***. Afterwards, it turns out to be ***v/ed*** which is needed for grammar-checking.

(1)	Science has influenced our life	***Surface form***
(2)	science have influence our life	***Stem form***
(3)	n /- v /es v/ed poss/- n /-	***Category/Suffix***

Table 2 The morphological analysis & lexical tagging of BDC-GC

Currently, this system operates on Sun Sparc & IBM RS 6000. It takes about 269 seconds to check 1956 sentences (= 24069 words) on Sparc station ELC.

III. The Baseline System & Comparison with Other GC-Systems

To show the superiority of the proposed method, the original 127 rule set as a baseline system is used for comparison. Additionally, to give readers a general feeling about the performance of our baseline system, it is also compared with several other popular commercial products, e.g., Grammatik IV, RightWriter, The Writer's Toolkit, PowerEdit, etc. The comparison of performance evaluation for five different writer's tools is based on five pieces of student essays related to SCIENCE. There are 72 sentences (869 words) in total. Fifty-two errors are checked and hand-labeled by a linguist.

Table 3.1 illustrates the performance of different systems, where the recognition rate is calculated by the formula *recognition rate = (number of detected errors/number of total errors)%*; and the false rate is computed by *false rate = (number of false errors/number of total detected errors) %*. The best recognition rate is 65%, which is performed by BDC-GC. Likewise, the highest false rate (33%) also goes to our system. On the other hand, Writer's Toolkit performs at a high recognition rate (53%), but hits the lowest false rate (15%).

Unfortunately, PowerEdit, for example, does not perform as well here as it did in two previous tests listed in Table 3.2 & Table 3.3, where the recognition rate was approximately 51% (Rabinovitz, 1991) or 72% (Smith 1992).

% / Tools	BDC-GC	Writer's Toolkit	RightWriter	PowerEdit	Grammatik IV
Recognition	65%	53%	30%	38%	44%
False	33%	16%	15%	31%	24%

Table 3.1 Performance evaluation for five GC-tools (52 mistakes made by Chinese student)

The large performance variation among different tests might suggest that a large testing corpus is required for the sake of fair comparison. However, this kind of test is very time-consuming without modifying the error reporting program of other products. Besides, the great decrease of performance for PowerEdit (recognition rate: only 38%) in our test might suggest that it is not suitable for correcting essays written by Chinese students. This phenomenon also implies that different rule sets might be required for various applications.

% / Tools	RighrWriter	PowerEdit	Grammatik IV
Recognition	13%	51%	30%
False	8%	11%	3%

Table 3.2 Performance evaluation (grammar & style) based on Rabinovitz's report (150 test sentences) (Rabinovitz, 1991)

% / Tools	Writer's Toolkit	RightWriter	PowerEdit	Grammatik V
Recognition	56%	54%	72%	48%
Fa!se	2%	2%	2%	26%

Table 3.3 Performance evaluation based on Smith's report (50 errors) (Smith, 1992)

IV. How To Select Better Rules Based on Corpus

A. The Construction of Corpus Annotated with Error Patterns

To select rules automatically, an annotated corpus is required. Various archives of written tests and compositions from 2 universities and high schools were first collected. To make the students' essays easy to deal with, the raw material was first hand-labeled with the corresponding error codes, and then put together to form the corpus. For instance, the following case in Table 4 is an example of our training set. The double question mark "??" (e.g., in "**?? can not -> 7.3**") represents *error mark* (where an error is stipulated). After the question mark, the string "**can not**" indicates *error scope* and "**7.3**" illustrate *error code* (or error type).

Sentence	*"For example, people can travel around ... that can not be seen ... "*			
Hand-labled	??	can not	->	7.3
Meaning	error symbol	error scope	->	error code

Table 4 An example of hand-labeled correction

There are 30 grammatical categories employed in the system. (Examples are shown in Table 5.) Each code number represents a specific error type. The error scope as shown in the given example is indicated by a pair of brackets.

Code	**Type**	**Example**	**Explanation**
1.1	Dangling participle	*Seeing her teacher, [her face] turns red.*	Logical subjects in two clauses should be consistent.
2.1.1.1	Fragment	*[Although the weather is bad.]*	Missing main clause
18.1.2.1	Agreement	*[He get] up early every day.*	Subject-verb agreement

Table 5 Examples of 30 grammatical types to detect errors in GC-system

B. Automatic Rule Selection with Sequential Forward Selection (SFS)

To find the best rule set, it is necessary to check all the possible subsets of the original rule set. Different search algorithms (optimal & suboptimal) have been proposed (cf. Devijver et al., 1982) to do so. Among those, Sequential Forward Selection (SFS, also cf. Devijver) is adopted in our grammar checker. SFS is a simple bottom up search procedure which can be used to take care of the correlation among rules. Compared with other approaches, the SFS algorithm is faster and less complex. Thus, it is preferable in our system.

To implement the SFS algorithm, we first initialize two groups of rules (i.e., error patterns):

i). Group 1 (G1_rule) with all the rules (127); (original rule set)

ii). Group 2 (G2_rule) used to include the selected rules. (It is empty in the very beginning.)

The basic concept of SFS is to activate grammar-checking and to take the best rule of performance from G1_rule to G2_rule. Then, we choose the best one again from the remaining rules in G1_rule until it is empty. This algorithm is shown as follows.

The Sequential Forward Selection (SFS) Algorithm

```
SFS (n rules) {
     G1_rule = n rules; /*initialization*/
     G2_rule = empty;

     loop (while there are still rules in G1_rule){
            max_score = -9999; /*initialize the maximum score as -9999 here*/
            loop (for each rulei in G1_rule) do {
                build_fsa (rulei + G2_rule);  /*Combine rulei with those*/
                                /*already selected rules, then construct FSA*/
                check_grammar (input_text);
                                /*Operate grammar checker with new FSA.*/

                score = W1 * number of detected error - W2 * number of false alarm
                           /*Evaluate the performance;*/
                           /*where W1 and W2 are reference weight.*/
                if (score > max_score) then {
                   max_score = score;    /*Replace the max_score */
                                         /*with the current one.*/
                   best_rule = rulei;    /*rulei as the best rule*/
                }
            } /* end of for-loop */
            move the best_rule from G1_rule to G2_rule
        }    /* end of while-loop */
        output G2_rule;

    }
```

The program includes two loops. The first one (while-loop) implies that while there are still rules/patterns in the G1_rule, it keeps working. The second loop (for-loop) indicates that for each remaining rule in the G1_rule, it builds up FSA for the rule set which is the union of the original G2_rule and the rule just picked up. Because all the rules in G2_rule are applied jointly, not disjunctively, the correlation among rules has been considered. Afterwards, it performs the grammar-checking and evaluation in terms of the following formula:

Score = [W1 * (number of detected errors)] — [W2 * (number of false alarms)], where W1 and W2 are the weighting factors which give different degrees of preference for *(number of detected errors)* and *(number of false alarms)*. Different weighting factors may be used for different applications. In summary, SFS uses the score function as the criteria for selecting the suboptimal rule set, which is a subset of the original rule set in many cases.

If any new score is bigger than the current max_score, then this score replaces the current one and becomes the max_score. Again, the best one is chosen and put into the G2_rule until the last one is done. That is, the G1_rule becomes empty in the long run.

V. Performance Evaluation

After executing SFS and operating 127 rules on the corpus of 1965 sentences, the results in Table 6 are generated. The rules are ranked according to their performance. For example, the most powerful rule, which is ranked Rule 1 here, finds 56 errors with 13 false alarms. Thus, it scores 43 for its performance by means of the formula (i.e., *Score = (number of errors detected) — (number of false alarms)*). The total detected mistakes throughout these 127 rules are 450 errors with 163 false alarms.

Rule	Detected errors	False alarms	Accumulative score	Rule	Detected errors	False alarms	Accumulative score
1	56	13	43	115	418	80	338
2	91	13	78	116	435	98	337
:	:	:	:	:	:	:	:
81	409	71	338				
82	416	78	338	126	448	129	319
:	:	:	:	127	450	163	287

Table 6 Statistical table with number of detected errors, false alarms & score

However, Table 6 shows that the rule set including the first 81 rules performs best. To capture a clear picture, Figure 2 is provided below.

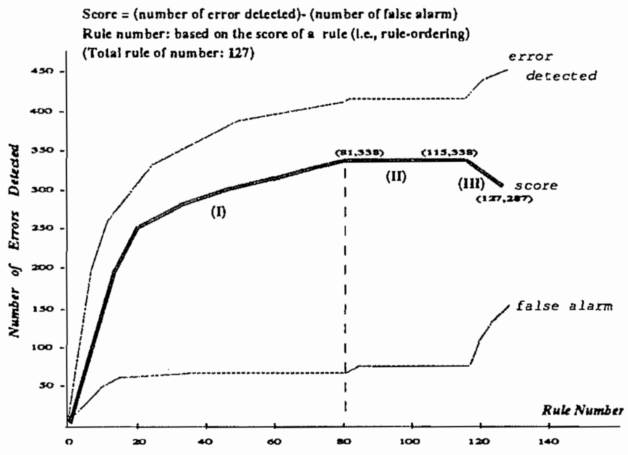


Figure 2 The Results of SFS with 127 Rules

The vertical axis shows the number of detected errors, 450 in total. The horizontal axis represents the ordered rule number (i.e., Rules 1–127) based on their performance as indicated by the given *formula.* The top curve shows the number of detected errors; the middle one illustrates the score of their performance, and the bottom one represents the number of false alarms. (The left Arabic numeral in each pair on the middle curve represents a rule number with its score at the right side of this pair, e.g., (81,338).)

The results demonstrate the following: .

(I). *The first 81 rules (scoring 338) perform better than the others.*

(II). *The performance of rules from 81 to 115 (also scoring 338) remain unchanged. This implies that these rules may be covered by the previous rules. Therefore, they can be deleted without any effect in our experiments.*

(III). *Afterwards, it goes down and finally degrades to 287. This suggests that Rules (116–127)* *should be revised or thrown away because they cause more false alarms than detected errors.*

As mentioned above, our tests are based on a corpus of 1956 sentences. After the application of SFS and the subsequent rule-revision, the false rate decreases by 11.8% (from 26.4% to 14.8%) if the suboptimal rule set (81) selected by SFS is adopted, not the whole rule set (127). However, the recognition rate decreases only by 3.9% (from 38.9% to 35%), without ruining the merit of its better performance. That is, the best 81 rules (scoring *338*) perform better than that of the total 127 rules (scoring *282*).

VI. Conclusions

This paper stipulates that the pattern matching approach is still widely used in the area of grammar checking. The reason is that when compared with the method of employing full scale parsing, pattern matching is efficient in detecting local errors with much less computer time and memory. However, the error patterns used in the pattern matching algorithm are usually hand-tuned, and thus suffer problems such as the problem of correlation among patterns. These patterns may conflict or overlap with other patterns. The purpose of this paper, therefore, provides an automatic rule selection method, i.e., ***Sequential Forward Selection (SFS)***, to handle these problems. This algorithm uses objective performance measures and then automatically searches the suboptimal rule-set among all possible combinations. With the help of SFS, the effectiveness of each rule can be measured and the problematic patterns can be identified systematically by linguists in order to fine-tune them effectively. Moreover, the error patterns can be revised efficiently. The above tests based on a corpus of 1956 sentences display that the false rate decreases by 11.8% (from 26.4% to 14.8%) if the suboptimal rule set (81) selected by SFS is adopted, instead of the whole rule set (127). However, the recognition rate decreases only by 3.9% (from 38.9% to 35%) by using this proposed algorithm. The implementation of SFS in BDC-GC, therefore, is strongly recommended to improve the performance of grammar-checking.

ACKNOWLEDGEMENTS

This project is supported by *The Science-based Industrial Park* (Project Numbers: 81–1155–111–189 & 82–1230–111–267). We would like to thank Dr. Hsian-chin Liu (NTHU), Dr. Wen-yu Chiang (NTU) and English instructors Shu-ying Chang, Chiu-chu Li, Yung-hsian Ting, among others from Hsinchu Girl's Senior High School, who all have provided their students' essays to construct the error pattems used in the prototype of our grammar checker. We would also like to express sincere thanks to our consultants, Dr. Ting-chi Tang & Dr. Samuel H. Wang (NTHU), for their kind assistance. Many thanks also to our colleagues, Jing-shin Chang, Ting-yu Cheng, Pen-hwa Lin, Cheng-li Tien, Renee Ting, Jong-nai Wang, and Ming-wen Wu for their comments and help. Last but not least, we appreciate two anonymous reviewers' valuable suggestions.

References

Atwell, E.S. 1987. "How to Detect Grammatical Errors in a Text without Parsing it," Proc. of Third Conference of the European Chapter of the ACL: pp. 38–45, Univ. of Copenhagen, Copenhagen, Denmark.

Brace, C. 1992. "RightWriter: State of the Art?" Language Industry Monitor, No. 12 Nov.-Dec.: p.9.

Chanod, J.P., M. El-Beze & S. Guillemin-Lanne. 1992. "Coupling an Automatic Dictation System with a Grammar, " Proc. of Coling-92: pp. 940–944, Nantes, France.

Church, K. 1988. "A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text," ACL Proceedings of 2nd Conference on Applied Natural Processing: pp.136–143, Austin, Texas, USA.

Devijver, P.A. & J. Kittler. 1982. *Pattern Recognition: A Statistical Approach*, Prentice-Hall Int., Inc., London.

Hopcroft, J. & J. Ullman. 1979. *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley Publishing Company, Philippines.

Kay, M. 1987. "Nonconcatenative Finite-State Morphology," Proc. of ACL: pp. 2–10, Stanford Univ., Stanford, CA, USA.

Karttunen, L., R.M. Kaplan & A. Zaenen. 1992. "Two-Level Morphology with Composition," Proc. of Coling-92: pp. 141–148, Nantes, France.

Lin, Y.-C., T.-H. Chiang & K.-Y. Su. 1992. "Discrimination Oriented Probabilistic Tagging," ROCLING 5: pp. 87–96, Taipei.

Pfaltz, J., W. Berman & E. Cagley. 1980. "Partial-Match Retrieval Using Indexed Descriptor Files," Communications of ACM 23 (9): pp. 522–528.

Rabinovitz, R. 1991. "Write on Target: 15 Writer's Tools," PC Magzine (Sept. 24, 1991): pp. 321–369.

Smith, J. 1992. "Mark Your Words with Grammar-Checking Software," PC/Computing (Oct. 1992): pp. 243–252.

Strunk, W. & E.B. White. 1979. *The Elements of Style*, Macmillam Publishering Co., Inc., New York.

Su, C-L (eds.) 1991. *Common Mistakes in English Composition and Translation*, Bookman Co., Inc., Taipei.

Su, K-Y & J-S Chang. 1992 "Why Corpus-based Statistics-oriented Machine Translation," 4th Int. Conf. on Theoretical and Methodological Issues in Machine Translation, Proc. of TMI-92: pp. 249–262, Montreal, Canada.

Vergne, J. & I. Paris. 1986. "Synergy of Syntax and Morphology in Automatic Parsing of French Language with a Minimum of Data," Proc. of Coling '86: pp. 269–271, Univ. of Bonn, Bonn, Germany.

Vosse, T. 1992. "Detecting and Correcting Morpho-syntactic Errors in Real Texts," Proc. of Third Conference on Applied Natural Language Processing: pp. 111–118, Trento, Italy.

中文辭彙岐義之研究—斷詞與詞性標示

彭載衍　　張俊盛

國立清華大學資訊科學研究所

摘要

目前電腦應用在中文處理方面，對於斷詞已能達到相當高的正確率（95%以上），然而在中文詞性標示的基礎研究上，仍未有相當的研究及令人滿意的結果。分析其原因，不外乎中文詞性訂定尚無標準；中文句法較複雜，變化較大，想要以法則分析法來運作似乎不太容易；還有缺乏良好的含有詞性及頻率的電子詞典。

目前我們已擁有含有詞性及頻率的電子詞典，且捨棄傳統法則分析的方法，改以機率式的方法，來作詞性的標示。在這個系統裡，我們用了幾個模型，並且分析比較了它們的結果，以期達到最好的效果。此外，對於部份的未知詞，詞長為一或二的，我們也做了處理；還有中文姓名的部份，也在我們討論範圍內。

在本篇的後面部份，我們作了正確率的評估與錯誤的分析，以利我們了解什麼是發生錯誤的主因，尋求改進之道。

一、簡介

在英文或其他西方語言中，並沒有斷詞的問題，然而在中文的處理上，斷詞卻是首先面臨的問題。什麼是中文斷詞？所謂中文斷詞是指將輸入的中文句子，依據語意及文法結構，切割文句至以詞爲基本單位的工作。文句經過斷詞後，形成以詞爲基本單位的句子，再進一步的分析是將句內的每個詞標上詞性，然而詞性如何正確地標示上去，正是我們所要探討的。

中文斷詞的研究，在近幾年來已相當成熟。歸納以往所使用的方法，基本上可以分成兩大類。第一類是法則式的斷詞方法，第二類是以統計數據爲基礎的機率式作法。在法則式的作法中，有何[1]、陳[2,3]和李[4]。第二類有蔡[5]的鬆弛法、Sproat 和 Shih[6]、張[7]、江和蘇[8]等人。

中文詞性標示處於起步階段，已有陳[9]，張[10]，張簡和李[11]等人做過這方面的研究。然而詞性標示的研究在英文發展甚早，到目前爲止，可說是到達相當成功的地步，正確率已高達百分之九十六以上。所發展出來的方法相當多，基本上亦可分成兩大類。第一大類是法則式的方法，第二大類是機率式的方法。

法則式的方法發展甚早，在1971年 Greene and Rubin [12] 發展出 TAGGIT，在Brown corpus 上測試的結果，正確率達77%。一般認爲法則式的方法正確率不夠高，然而 Brill [13] 在1992年提出了一個新方法，正確率高達95%，說明了法則式的作法亦可達到高正確率。

機率式的作法一般包含兩個不同的機率值，一個是字本身所能貢獻的機率值 (lexical probability) ，另一個是前後文字所能提供的機率值 (contextual probability) ，大多數的公式都是由這兩個所組成的。Leech[14]的 CLAWS，採用詞性的二元接續 (bigram) 與字本身所擁有的詞性分配頻率。從文獻上得知 CLAWS 對測試資料達到96.7%的正確率。Church[15]採用的是詞性的三元接續表 (trigram)，方法與 Leech 差不多。此外尚有 DeRose[16] 發展的 VOLSUNG ，其用的方法與 CLAWS 的類似，但若加上片語 (idiom) 的處理，在 LOB corpus 上可達99%的正確率。林和蘇[17]等人，利用學習 (learning) 和結合 (merging) 詞性二元接續表與三元接續表的方法來處理詞性標示，正確率亦達 95% 以上。

二、斷詞與詞性標示的機率式方法

在這一節裡，首先介紹斷詞的方法，接著是詞性標示，然後是斷詞與詞性標示整合在一起的機率模式。此外，尚有未知詞與中文姓名的處理。

機率式的斷詞方法

假設輸入系統的中文字串為 $C_1C_2...C_n$，輸出為詞串 $W_1W_2...W_p$ ($p<=n$) ，如何決定輸出的詞串是倚賴一類似 0 階的馬可夫模式 (Markov Model) (2.3)，計算出各種詞串組合的機率值，找出擁有最大值的詞串，此時詞的組合為在機率模式下的最佳組合，也就是被輸出的詞串。

公式的推導如下：

$$P(W_1,W_2,...,W_p|C_1,C_2,...,C_n) \qquad -(2.1)$$
$$\approx P(W_1,W_2,...,W_p) \qquad -(2.2)$$
$$\approx P(W_1)P(W_2)...P(W_p) \qquad -(2.3)$$
$$=\prod_{i=1}^{p}P(W_i)$$

其中 $P(W_i)$ 表示句子內第 i 個詞的出現機率值

式子 (2.1) 表示在給定中文字串 $C_1C_2...C_n$ 下，詞串 $W_1W_2...W_p$ 的出現機率。當這個機率值高時，表示此狀況下的詞串出現的機會較大，通常正確的斷詞結果，即是機率模式下的最佳組合，因此我們能以機率的模式來解決斷詞的問題。

然而式子 (2.1) 中的條件機率，需要統計相當大的語料才可以得到，在實做上幾乎是不可能的，因此需要做一些假設，以得到一個較簡單的機率模式，能在電腦上快速運作。

首先，假設詞在句子內的出現機率與詞在整個大語料內的出現機率並無不同，那麼式子 (2.1) 就可以化簡爲式子 (2.2)。此外，我們再假設詞與詞間的出現並無關連，彼此間是互相獨立的，那麼式子 (2.2) 可再進一步被化簡爲式子 (2.3)。至此我們將斷詞問題轉變成找出一詞串 $W_1W_2...W_p$ 使得式子 (2.3) 擁有最大值。因此機率式的斷詞模式可被寫成一個如下公式 (2.4)。

$$Max\{\prod_{i=1}^{p}P(W_i)\} \qquad -(2.4)$$

在眾多的詞串中，滿足式子 (2.4) 的詞串即是此模式下斷詞的輸出結果。

斷詞與詞性標示的解決方法

假設斷詞完後的詞串爲 $W_1 W_2 \ldots W_k$，所要標示的對應詞性爲 $T_1, T_2, \ldots, T_k$ 。詞性標示所用的公式是使用詞性二元接續表的模式。表示成式子(2.5)。

$$\prod_{j=1}^{k} P(T_j|W_j)P(T_j|T_{j-1}) \qquad \text{-(2.5)}$$

現在我們將說明如何將斷詞與詞性標示合在一起成爲一個單獨的模式。若將斷詞視爲事件 A，發生的機率爲 P(A)；詞性標示視爲事件 B，發生的機率爲 P(B)。事件 B 是發生在事件 A 之後，因此事件A與B皆要發生的機率爲此兩機率之積P(A)P(B)。從機率的觀點來看，斷詞與詞性標示合在一起的公式爲式子 (2.3) 與式子 (2.5) 的合併，成爲下面的式子 (2.6)：

$$\begin{aligned}&\prod_{i=1}^{k} P(W_i)P(T_i|W_i)P(T_i|T_{i-1})\\ &=\prod_{i=1}^{k} P(W_i)\frac{P(W_i,T_i)}{P(W_i)}P(T_i|T_{i-1})\\ &=\prod_{i=1}^{k} P(W_i,T_i)P(T_i|T_{i-1}) \qquad \text{-(2.6)}\end{aligned}$$

其中 $P(W_i,T_i)$ 表示第 i 個詞 W_i 與其詞性爲 T_i 的出現機率。T_0 是句子開頭的標示。

機率式的斷詞詞性標示可被表示爲底下的公式：

$$Max\{\prod_{i=1}^{k} P(W_i,T_i)P(T_i|T_{i-1})\} \qquad \text{-(2.7)}$$

在詞性標示的處理上，句子結束這個訊息亦有助於標示的正確性，因此我們加入句子末尾的標示（記爲 $T_{\&}$ ）於公式 (2.7) 內 。將式子 (2.7) 改爲式子 (2.8) 。

$$Max\{(\prod_{i=1}^{k} P(W_i,T_i)P(T_i|T_{i-1}))P(T_{\&}|T_k)\} \qquad -(2.8)$$

未知詞的解決模式

在斷詞與詞性標示的系統裡，詞典所收錄的詞受到記憶體大小限制和執行速度的要求，不可能將所有的詞納入。即使在系統設備允許的狀況下，想要將所有詞納入，亦需花費相當大的成本。此外，詞是具有時間性的特徵，古代所使用的詞有許多在今日已不被使用，當代依需求、流行或其他因素，亦創造出許多新的詞彙。還有姓名、譯名、專有名詞等，這一類名詞數量相當多，須要常常做更新以符合需求。因此，詞典是無法收錄所有的詞，僅能依據使用者的需求和系統的要求，適當地涵蓋所需求的辭彙。

在這裡我們所稱的未知詞是指詞典未收錄的詞。既然無法要求詞典擁有所有的詞，而未知詞會影響系統的斷詞與詞性標示的正確性，因此，未知詞的處理是必須的。未知詞的種類很多，如：譯名、姓名、專有名詞等。它們被使用的程度依文章的性質而異，所使用的單字在頻率方面也有差異，因此，這些是需要分開討論的。因爲所得到的資料有限，我們只將姓名的部份在下一節特別討論，其餘的未知詞不再細分類，將在底下討論。

根據統計，文章中出現的詞長度爲一或二的佔了絕大部分，所以我們將詞長爲二的未知詞作爲首先解決的問題，詞長超過二的未知詞暫不討論。

假設未知詞 W 分別由左邊的字 M_1 與右邊的字 M_2 所組成，表示爲 $W = M_1M_2$，它的詞性爲 T。因爲一個詞所擁有的詞性不只一個，對未知詞而言，每種詞性似乎都有可能，但是以名詞、動詞、形容詞居多。所以我們只針對這三類詞性做估計，估計名詞、動詞、形容詞出現的可能機率。

估計的公式如下：

$$\begin{aligned} P(T|M_1,M_2) &\approx P(T|LM = M_1 \& RM = M_2) \\ &\approx P(T|LM = M_1) \times P(T|RM = M_2) \end{aligned} \qquad \text{-(2.9)}$$

其中 LM 表示左邊的字，RM 表示右邊的字。

$$P(T|LM = M_1) = \frac{P(LM = M_1 \& POS = T)}{P(LM = M_1)}$$
$$P(T|RM = M_2) = \frac{P(RM = M_2 \& POS = T)}{P(RM = M_2)}$$

其中 POS 表示詞 W 的詞性標示，

$P(LM = M_1 \& POS = T)$ ， $P(RM = M_2 \& POS = T)$,
$P(LM = M_1)$ ， $P(RM = M_2)$

分別可由對詞典的統計而來。

整個計算未知詞 W 的公式可被表示爲：

$$
\begin{aligned}
P(W,T) &= P(M_1M_2,T) \\
&\approx P(UNW)P(M_1M_2|UNW)P(T|M_1M_2 \cap UNW) \\
&= P(UNW)P(M_1M_2|UNW)P(T|M_1M_2) \qquad \text{-(2.10)}
\end{aligned}
$$

其中 UNW 表示未知詞；P(UNW) 爲未知詞的出現頻率，可以從測試的語料估計得到；$P(M_1M_2|UNW)$ 是未知詞爲 M_1M_2 的機率，可從底下的估計法求得；$P(T|M_1M_2)$ 是未知詞 M_1M_2 詞性爲 T 的機率，可從式子 (2.10) 估計求得。

$P(M_1M_2|UNW)$ 的估計方法。

$$P(M_1M_2|UNW) \approx P(LM = M_1|D)P(RM = M_2|D)$$

其中 D 表示詞典。

以式子 (2.10) 代入上一節的斷詞詞性標示系統，實驗顯示的確能改善詞性標示中未知詞所引起的問題。

中文姓名的處理

完整的中文姓名是由姓的部份與名的部份所共同組成，如：張無忌。張是姓，無忌是名。因爲在文章中姓與名不一定要成對出現，如：張先生，只出現姓的部份：無忌孩兒，只出現名的部份。因此我們分別給予姓和名不同的詞性，'fn' 和 'gn' 來反應上述的情況。

中文的姓可分成單姓和複姓，名可分爲單名和複名，這些區別在機率的模式底下所使用的的公式差異不大，不同的地方在所使

用的統計資料不同，底下所介紹的公式，是參考張 [18] 所得到的。

單姓的情況：

假設 W 是一單字，其爲單姓的機率爲：

$P(W,fn) = P(FN1)P(W|FN1)$ -(2.11)

其中 $P(FN1)$ 是指語料中出現單姓的機率，可由統計語料得到。$P(W|FN1)$ 是指詞 W 的標示爲 fn，在單姓底下的條件機率。

複姓的情況：

$W = C_1C_2$，其爲複姓的機率爲：

$P(W,fn) = P(FN2)P(W|FN2)$ -(2.12)

其中 $P(FN2)$ 是指語料中出現複姓的機率，可由統計語料得到。$P(W|FN2)$ 是指詞 W 的標示爲 fn，在複姓底下的條件機率。

接著將介紹名字的機率公式。

單名的機率公式爲：

$P(W,gn) = P(GN1)P(W|GN1)$ -(2.13)

其中 $P(GN1)$ 是指語料中出現單名的機率。$W = C_1$

複名的機率公式爲：

$$\begin{aligned}P(W,gn) &= P(GN2)P(W|GN2)\\ &= P(GN2)P(C_1C_2|GN2)\\ &\approx P(GN2)P(LG=C_1|LG)P(RG=C_2|RG)\end{aligned} \qquad -(2.14)$$

其中 $P(GN2)$ 是指語料中出現複名的機率。$W=C_1C_2$ ：LG 表示複名的第一字， RG 表示複名的第二字。

從式子 (2.11) 到式子 (2.14) ，是以一個機率的方法來解決中文姓名部份，與前面的斷詞詞性標示是一致的，能合併成爲一個系統。

三、實驗結果與錯誤分析

一個模型的好壞，除了理論的嚴密性外，還需要實驗的驗證。因此，在這一節中，我們將列出實驗的結果，並且分析錯誤的成因，進而對模型做一些修正，以期得到較好的結果。

基線模式

在利用詞性二元接續表斷詞與詞性標示之前，我們先介紹一個基線 (base line) 的斷詞與詞性標示模式，作爲詞性接續表模式的比較對象，使得實驗結果較爲客觀。這個基線模式是這樣的：斷詞與詞性標示所用的機率公式爲

$$MAX\{\prod_{i=1}^{I=P} P(W_I,T_I)\} \qquad -(3.1)$$

其中 P 爲句內的總詞數

基線模式是在詞典內挑選詞 W 的詞性 T ，使得 P(W,T) 是在詞 W 下的最大頻率值。這樣的作法，並不考慮詞性與詞性間的接連關係，與斷詞的作法相類似。

斷詞與詞性標示結果及分析

從實驗的結果可看出，基線模式的正確率爲 70.8% ，召回率爲 71.4% ，這樣的結果並不令人滿意。剛才已提過，這個基線模式並未充分利用有助於結果正確的訊息—詞性間的接連關係。例如'然後按 Enter 鍵' 在基線模式底下標示爲：

```
|然後|按|Enter|鍵|
|adv |p | np   |nc|
```

若考慮詞性間接連的關係，標示的結果成爲：

```
|然後|按|Enter|鍵|
|adv |v | np   |nc|
```

'按'的介詞機率最高，因此在基線模式中被標示爲介詞。在考慮詞性接連關係的模式中，標示結果變成爲動詞。然而標示爲介詞是錯誤的，動詞才是正確，所以使用詞性間接續關係的機率模式應會有許多改進。

本實驗所用的是詞性二元接續表的模式，僅考慮與前一個詞的接連關係，其他的關係並不考慮。實驗的結果列於表一，其中檔案一至檔案八爲訓練資料，檔案九爲測試資料，明顯地看出檔案九並不因未參與詞性二元接續表的建立而結果表現較差，其正確率與召回率仍有不錯的表現。因此，詞性二元接續表在類似的文章中是具有好的轉移性，也能在它們中使用同樣的接續表。

因爲接續表有好的轉移性，所以在統計與分析方面，我們並不將檔案九分開，而是全部合在一起，表中所列的平均值部份亦包含檔案九。

比較基線模式與詞性二元接續表模式的結果，在平均表現上，後者正確率較前者多了 5.0% ，召回率多了 5.4% ，這點說明了二元接續表模式確實有助於詞性標示，大約能提升百分之五左右。

	總字數 A	總詞數 B	標示後總詞數 C	標示正確詞數 D	標示正確字數 E	詞的正確率 D/C %	詞的召回率 D/B %	字的正確率 E/A %
檔案1	3275	2137	2161	1792	2818	82.9	83.9	86.0
檔案2	3177	2076	2124	1551	2365	73.0	74.7	74.5
檔案3	3412	2177	2199	1564	2465	71.1	71.8	72.2
檔案4	3340	2118	2137	1651	2623	77.3	78.0	78.5
檔案5	2667	1721	1753	1299	2020	74.1	75.5	75.8
檔案6	4701	3044	3049	2418	3777	79.3	79.4	80.3
檔案7	4605	3021	3063	2292	3571	74.8	75.9	77.5
檔案8	3210	2150	2196	1571	2381	75.1	73.1	74.2
檔案9	3091	2018	2056	1583	2445	77.0	78.4	79.1
	31478	20462	20738	15721	24465	75.8	76.8	77.7

表一 ： 使用詞性二元接續表的標示結果

錯誤分析

以下是從測試資料中摘選出一些錯誤的例子，例句中底線表示詞性標示的主要錯誤。

```
(1) |檔案|名稱|後面|加上|兩|個|字|元|
    |nc  |nc  |nc  |v   |q |cl|nc|nc|
(2) |包含| 1-2-3 |的 |群|組|視|窗|
    |v   | np    |ctm|cl|nc|v |nc|
(3) |您    |可以|按照|姓|氏|或|識別|碼|來|排序|
    |pron|aux |v   |nc|nc|cj|v   |cl|v |nc  |
(4) |本|章|將  |說明|如何|使用|資料庫|函數|
    |cl|nc|aux|v   |adv |v   |nc    |nc  |
(5) |使|用於| 1-2-3 |舊|版本|
    |v |v   |  np   |a |nc  |
(6) |然後|再  |加上|工作|表|表|名|
    |adv |adv|v   |nc  |v |v |nc|
```

在錯誤分析方面，除了機率模式本身標示錯誤外 (例句 (5)(6))，還有其他非模式本身所產生的錯誤，我們分析的結果可歸納成兩類。第一類是詞典內未含有的詞所產生的錯誤，這類的詞我們稱爲未知詞 (例句 (1) (2))；第二類是詞典內雖含有詞但並不含有正確標示的詞性，我們稱這類詞性爲未知詞性 (例句 (3)(4))。這兩項因素對於標示的正確性有絕對的影響，機率式的斷詞詞性標示模式，若未加上特殊的處理，並無法對未知詞和未知詞性做正確的標示，因此它們將造成絕對性的錯誤，影響甚巨。平均而言，未知詞有 4.9% ，未知詞性有 4.3% ，兩者合起來共佔9.2%。系統標示的錯誤率爲 24.2% ，因未知詞與未知詞性所產生的錯誤爲 9.2% ，因此，系統本身所產生的錯誤爲
24.2% - 9.2% = 15.0% 。

在未知詞分析上，我們對它做了更進一步的分析，分成單字詞、雙字詞、三字詞、四字詞四類，將各種所佔的詞數與比例求出，列於表二。從表二中看出未知詞的種類絕大部分是屬於單字詞或是雙字詞，兩者合起來共佔了 92% 。從這個訊息得知，未知詞的處理首要在單字詞與雙字詞上。

	缺少詞数A	單字詞数B	雙字詞数C	三字詞数D	四字詞数E	單字詞百分比%	雙字詞百分比%	三字詞百分比%	四字詞百分比%
總計	993	505	410	75	3	50.9	41.3	7.6	0.3

表二 ： 未知詞的統計

未知詞處理結果及分析

前面已說明過未知詞佔了測試資料的 4.9% ，這一部份使得系統無法對它處理，除了本身詞與詞性的標示錯誤外，對於接連的詞與詞性亦造成影響，因此所發生的錯誤將大於其所佔的比例。對未知詞的處理可分爲兩部份，第一部份是未知詞中比例最高的單字詞，第二部份是雙字未知詞。首先介紹單字未知詞的處理。

單字未知詞的處理結果

在原始系統處理過程中，單字未知詞無法與鄰近的字結合成詞，因此會被標示爲單字詞，但並無給予詞性，所以我們要做的工作是給未知單字詞一個詞性。

詞性標示的過程中必須用到二元接續表，因此我們利用它來協助標示單字未知詞的詞性。只要給未知詞每種詞性一個相同頻率值，那麼系統所決定的詞性將僅由其前後的詞性與後面的詞所決定，這個詞性滿足機率公式的最大值。說明如下：

若 W_u 爲單字未知詞，爲句内第 i 個詞

設定 $P(W_u,T_i)=C$ $n(T_i)=s$

其中 C 爲常數，$n(T_i)$表示詞性 T_i的個數，s 爲所有詞性總數。

實驗的結果顯示，加上單字未知詞的處理後，正確率變爲 77.2%，召回率變爲 78.1%，與先前結果比較，正確率增加了 1.4%，召回率增加了 1.3%。

雙字未知詞的處理結果

這個部份是用先前所提的雙字未知詞方法，再加上單字未知詞的處理。平均而言，正確率爲 78.6%，召回率爲 76.7%，與僅單字未知詞的處理結果比較，加上雙字未知詞後，正確率增加了 1.4%，然而召回率卻下降了 1.4%，這樣的結果並不令人滿意。

召回率的提高必須靠標示正確的詞才能提升；然而正確率的增加卻並不一定代表著標示正確的詞增加。召回率的下降代表著有過多的詞被錯誤地合成雙字詞。因此，在修正的作法裡，我們設法減少二字詞的合成，減少犯錯的機會。從實驗的經驗得到單字詞中詞性爲介詞、方位詞、連接詞、量詞等詞性都較少有二字合成詞的現象，所以我們將具有這類詞性的單字詞排除在二字詞的合成範圍內，以期減少錯誤的詞合成。

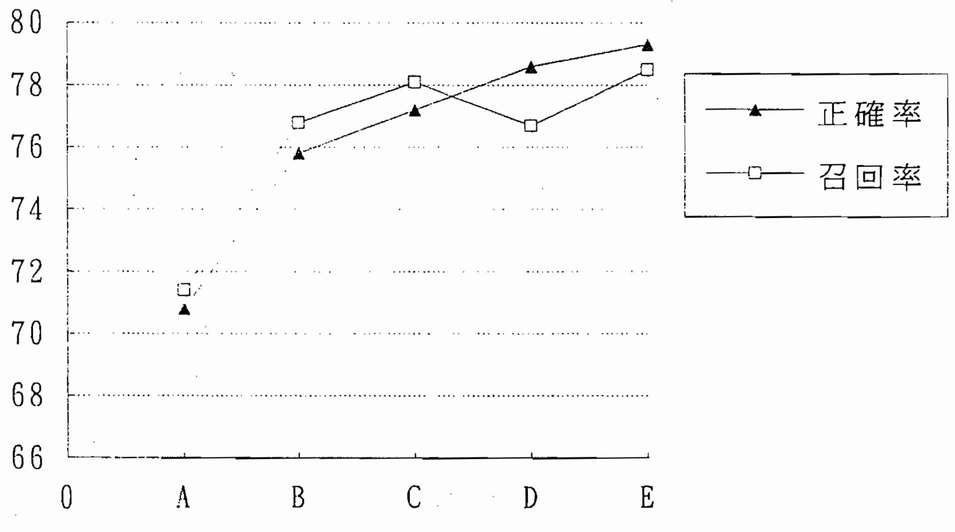
實驗顯示，正確率較未修正前多了 0.7%，召回率多了 1.8%，這點說明了改進的方法確實有效。與沒有未知詞處理的二元接續表模式相比較，得到整個未知詞的處理，在正確率上增加了 3.5%，召回率上增加了 1.7%。

綜合前面實驗的結果，繪成圖一，以方便觀察正確率與召回率的變化。經過一連串的改進後，正確率較基線模式增加了8.5%，召回率增加了7.1%。

表三所列的是測試資料的分析表，有未知詞、未知詞性及詞所含有的詞性數目等項的百分比。

未知詞	未知詞性	一個詞性	二個詞性	三個詞性	四個詞性	五個詞性	七個詞性
4.9%	4.3%	51.9%	26.7%	8.0%	1.6%	2.6%	0.03%

表三：測試資料分析表



	基線模式 A	二元接續表 B	接續表+單字未知詞 C	接續表+單字未知詞+雙字未知詞 D	接續表+單字未知詞+雙字未知詞(修改) E
正確率	70.8	75.8	77.2	78.6	79.3
召回率	71.4	76.8	78.1	76.7	78.5

圖一：正確率與召回率一覽圖

中文姓名處理結果及分析

姓名的測試資料是新聞類的語料，共有三個檔案，大約三千五百個詞。實驗結果將姓與名分開來，各別計算其正確率與召回率。

表四所列的是姓、名標示正確與錯誤的數目。表五是根據表九內的數據，所求出姓與名的正確率和召回率。姓的正確率達到86.7% 召回率更高達 93.3%，這數據說明姓的處理是相當不錯的。名的正確率是 78.3% 召回率是85.5%，對於名的處理也有不錯的結果。表六顯示姓與名在測試資料中分別所佔的比例，平均上姓佔 2.4%、名佔 2.2%，共是 4.6%。除了從處理姓名的觀點來分析結果外，我們亦須從整體的標示系統來看姓名的處理效果如何？在表七中顯示當系統加上姓名的處理後，姓的部份能增加 2.0%，名的部份能增加 1.4%，總共是 3.4%。

	姓總數 A	名總數 B	標示正確姓數 C	標示錯誤姓數 D	標示正確名數 E	標示錯誤名數 F	標示總合姓數 G=C+D	標示總合名數 H=E+F
檔案一	27	21	26	1	17	3	27	20
檔案二	27	26	27	5	22	5	32	27
檔案三	30	29	25	6	26	10	31	36
	84	76	78	12	65	18	90	83

表四：姓名標示正確錯誤數統計表

	姓正確率 A/G	姓召回率 A/B	名正確率 C/H	名召回率 C/B
檔案一	96.3	96.3	85.0	81.0
檔案二	84.4	100.0	81.5	84.6
檔案三	80.6	83.3	72.2	89.7
	86.7	93.3	78.3	85.5

表五：姓與名的正確率與召回率

	總詞數 I	姓所佔的比例 A/I %	名所佔的比例 B/I %
檔案一	777	3.5	2.7
檔案二	1390	1.9	1.8
檔案三	1285	2.3	2.3
	3452	2.4	2.2

表六 ： 姓與名在測試資料中所佔的比例

	姓標示正確所佔的比率 C/I %	姓標示錯誤所佔的比率 D/I %	名標示正確所佔的比率 E/I %	名標示錯誤所佔的比率 F/I %	姓標示淨正確比率M= C/I-D/I%	名標示淨正確比率 N= E/I-F/I%	姓名標示淨增加比率 M+N %
檔案一	3.3	0.1	2.2	0.4	3.2	1.8	5.0
檔案二	1.9	0.1	1.6	0.4	1.8	1.2	3.0
檔案三	1.9	0.5	2.0	0.8	1.4	1.2	2.6
	2.3	0.3	1.9	0.5	2.0	1.4	3.4

表七 ： 姓名標示對測試資料正確姓的改進

四、結論

由於機率式的中文詞性標示研究還處於剛開始的階段，在這方面並沒有許多的論文可以參考，所以只能從英文在這方面的研究得到一些想法；然而中文與英文間存有很大的差異，在英文處理不錯的詞性標示方法，運用到中文卻未有相同的效果。從實驗的結果看出，利用詞性二元接續表機率式的標示模式，較基線模式改進了百分之五。雖然結果並非理想，但是這卻也告訴我們，在中文詞性標示的領域尚有許多研究的空間，值得努力改進的地方還有很多。

中文的未知詞與英文的未知詞不同。在英文方面，英文詞的分界明顯，因此不會造成詞與詞間的混淆，未知的程度僅在詞性以上的階層。中文未知詞會使得系統無法辨識這個詞，所以中文未知詞的處理比英文的還要困難。本篇對單字與雙字未知詞做了些簡單的處理，結果顯示正確率增加了 3.5% ，召回率增加了 1.7% ，使得系統的正確率與召回率接近百分之八十。

另外，我們利用一些姓名的頻率資料，加入標示系統內。結果顯示姓的召回率達到 93.3% ，名的召回率達 85.5% ，這點證明以機率模式方法來處理中文姓名是可行的，且有很好的效果。

參考資料

[1] 何文雄，1983，中文斷詞的研究，國立台灣工業研究技術學院（碩士論文）。

[2] 陳克健，陳正佳，林隆基，1986，中文語句分析的研究—斷詞語構詞，TR-86-004，Nankang : Academia Sinica (技術報告)

[3] Chen, K .J. et al. Word Identification for Mandarin Chinese Sentences , Proceedings of COLING-92 ,14th Int. Conference on Computational Linguistics , pp. 101-107, July 23-28 ,1992.

[4] C. K. Fan and W. H. Tsaai ,1987, Automatic Word Identification in Chinese Sentence by the Relaxation Technique ,Proc. of National Computer Symposium, pp. 423-431,National Taiwan University,Taipei,Taiwan.

[5] Lee, H. J. et al. Rule-Based Word Identification for Mandarin Chinese Sentences - A unification Approach . Computer Processing of Chinese and Oriental Languages . Vol 5 , no 2, pp. 97-118 , March 1991.

[6] Richard Sproat and Chilin Shih , 1990 , A Statisstical Method for Finding Word Boundaries in Chinese Text, Computer Processing of Chinese & Oriental Languages, Vol. 4 , March 1990.

[7] Jyun-Sheng Chang, Chi-Dah Chen ,and Shun-Der Chen, Chinese Word Segmenttation through Constraint Statisfication and Statistical Optimization , In Proceedings of ROC Computational Linguistics Conference, pp. 147-165, 1991.

[8] T. H. Chiang,T. S. Chang,M.Y. Lin,and K. Y. Su,1992, Statistival Models for Word Segmentation and Unknown Word Resolution , ROCLING V , pp.147-175.

[9] 陳志達， 991，中文斷詞與詞性標定，國立清華大學資訊科學研究所，碩士論文。

[10] J.S. Chang, T.Y. Tseng, Y. Cheng, H.C. Chen, S.D. Cheng, S.J. Ker, and J.S. Liu, A Corpus-Based Statistical Approach to Book Indexing , Applied Natural Language Processing ,1992.

[11] 張簡哲輝，1992，馬可夫語言模式于手寫中文辨識之應用，國立交通大學資訊工程研究所，碩士論文。

[12] Greene and Rubin ,1971, Automatic Grammatical Tagging of English Technique report , Department of Linguistics , Brown University, Providence,Rhode Island.

[13] Brill ,1992, A Simple Rule-Based Part of Speech Tagger , ACL , pp .152-155.

[14] G. Leech,R. Garside, and E. Atwell ,1983 ," The Automatic Grammatical Tagging of the LOB corpus", ICAME News 7, pp .13-33.

[15] Church K. W. , 1988 , A Stochastic Parts Program and Noun Phrase Paser for Unrestricted Text , Second Conference on Applied Natural Language Processing, pp . 136-143.

[16] DeRose , S. J. , 1988 , Grammatical Category Disambiguation by Satistical Optimization, Computational Lingustics 14, 31-39.

[17] Y.C. Lin, T.H.Chiang, and K.Y. Su , Discrimination Oriented Probabilistic Tagging , R.O.C. Computational Linguistics Conference ,pp.85-96,1992.

[18] Chang, J.S. et al. A multiple-corpus Approach to Identification of Chinese Surname-Names , Journal of Computer Processing of Chinese and Oriental Languages , Feb .1993.

從中文語料庫中自動選取連續國語語音特性平衡句的方法

王 新 民[1]　張 元 貞[2]　李 琳 山[1,2,3]

[1] 國立台灣大學電機工程學研究所
[2] 國立台灣大學資訊工程學研究所
[3] 中央研究院資訊科學研究所

摘　要：

本文提出一個能從中文語料庫中自動選取連續國語語音特性平衡句的方法，解決了以往必須由人工選造特性平衡句時所遭遇的費時費力的困難，這個方法除了可以有效的找到包含所有辨識單元（例如單音節）、足供訓練用的特性平衡句外，也可以自動加選句子使所選取的特性平衡句與來源語料庫擁有近似的辨識單元的統計分佈，故以它來對語音處理系統進行測試時，所得的結果更能夠反映出系統實際的效能。由於它是一個自動的系統，當應用領域或辨識單元改變時（例如改成聲母、韻母），在不更動原本架構下，只要重新定義新的辨識單元或語料庫，即可輕易的調適到新的應用領域(Domain)。在我們以國語1333個帶聲調的音節為辨識單元的實驗中，這套方法得到相當不錯的成果。

一、緒論：

中文連續語音特性平衡句是一組中文文句，它的用途在於能夠提供各種語音處理研究一套有效而完整的連續語音的訓練及測試語料。故所謂的連續語音特性平衡句就是指一組合乎文法和語意的條件，由語音特性差異很大的辨識單元所構成的相當少量的句子，卻能包含所有不同的辨識單元並有合理的統計分佈。由於希望句子的數量儘量減少，故應避免在句子中相同的辨識單元有太多重覆出現的現象〔2〕。

在過去，語音特性平衡句的產生，主要是由人工來選造〔1〕〔2〕，也就是經由專家來選句或造句。人工選造語音特性平衡句所遭遇的問題是成本太高，且往往會造出一些較繞口的句子，加上難以兼顧其中辨識單元的統計分佈情形，因此模擬測試的結果並不能反映真實的情況。此外，只要應用領域（Domain）或辨識單元有所改變時（例如音節改成聲母韻母），往往必須重新選造一套適用的語音特性平衡句，此時將再度面臨成本太高的問題。以目前國內電信研究所已發展完成的一套由人工製訂的語音特性平衡句為例〔2〕，雖然這一套根據國語基本音節（407 個不考慮聲調的音節）來選造的語音特性平衡句對於國內語音相關研究有相當的價值並已使用多年，然而它的應用範圍基本上是有所限制的，例如在聲調也必須一併考慮的應用上（407個變成一千三百多個音節），這套語音特性平衡句就不再適用了。基於對連續語音特性平衡句的需求及人工選造的缺點，且近年來國內在語料庫語言學的發展日漸成熟，因此，我們嘗試發展一套能從語料庫中自動選取連續語音特性平衡句的方法。這個方法基本上是一個兩階段式的架構，每個階段所選取的結果可以滿足不同的需求。其中第一階段處理的目的是希望找到一組總數量儘可能小而仍能包含語料庫中所有辨識單元的連續語音特性平衡句的集合。因此，第一階段選取的連續語音特性平衡句足以充當語音處理的訓練語料之用。至於第二階段處理的目的則是以第一階段的選取結果為基礎，從來源語料庫加選句子來使選取的連續語音特性平衡句能與來源語料庫有相近的辨識單元的統計分佈。因此，第二階段選取的連續語音特性平衡句將是極佳的測試語料，所得的模擬測試結果將能夠真實地反映出語音處理系統的實際效能。

這套方法主要的特點在於它具有普遍性(Generality)，不但適用於不同的應用領域(Domain)，而且因為無論將辨識單元定義成音節（帶聲調抑或不帶聲調）、次音節單位（聲母、韻母）、乃至考慮前後文相關(Context dependence)，只要提供語料庫，同樣的一套方法即可滿足

不同的需求。為使本文以下的內容說明簡單清楚，以下將以自動選取以國語1333個帶聲調的音節為辨識單元所需之連續語音特性平衡句為例，來詳細說明我們的方法和實驗的結果,當然這方法也可自動延伸到其他的辨識單元上。

以下本論文的第二、三、四節將詳細介紹這套連續語音特性平衡句的選句方法，第五節則提出我們的實驗結果及相關討論，第六節是結論。

二、兩階段式的自動選句方法：

本文所提出的方法是一個兩階段式的選句架構。我們以國語的1333個帶聲調的音節為辨識單元為例，說明其基本構想 。首先第一階段是找出一組數量儘可能最小的連續語音特性平衡句集合，它能夠包含來源語料庫中所有的辨識單元（在這例子中也就是音節），所以這些句子唸出來的語音部分當做連續國語語音處理的訓練語料。但由於第一階段所選取的訓練語料它的辨識單元分佈情況與來源語料庫可能相差極大，所以未必適合做測試語料，因此第二階段的作用即是用來改進第一階段的選句結果，透過從來源語料庫中加選句子來增加連續語音特性平衡句集合與來源語料庫各辨識單元（音節）分佈情形的相似程度。當第二階段的選取結果達到我們要求的相似度標準時，所得到的連續語音特性平衡句集合唸出來的語音便可以做為連續國語語音處理的測試語料。

以下是此方法的進一步介紹，首先是根據我們對連續語音特性平衡句的需求，而有下列的選句原則：

1. 整套連續語音特性平衡句集合必須要涵蓋來源語料庫中所出現的所有辨識單元（也就是音節）。
2. 基於日常使用習慣及方便，長度為6至12字的句子優先考慮。
3. 在同一句子內的相異辨識單元（音節）數目愈多愈好，如此可以包含更多的不同辨識單元間的關連特性。
4. 在同一句子內相異的聲母、韻母數目也愈多愈好。
5. 第一階段中優先選取來源語料庫中低頻率辨識單元（音節）所屬的句子；而第二階段則優先選取來源語料庫中高頻率音節較多的句子，這樣才可達到上述的目標，其詳細原因及作法以下會進一步說明。

此兩階段式的選句方法，其流程如圖一所示，並詳述於以下兩節。

三、第一階段選句方法：

這一階段的輸入是來源語料庫，而輸出是一組連續語音特性平衡句集合，可以作為訓練語料之用。其詳細方法如下：

步驟1：統計來源語料庫的音節分佈，對每個音節給定一初始加權值，來代表其重要性，其加權值給法如下：

$$S[\mathrm{i}] = N_T / n_T[\mathrm{i}] \quad \text{for i} = 1 \text{ to } \mathrm{N_s}$$

$S[\mathrm{i}]$：第 i 個音節的加權值。
N_T：來源語料庫中音節總數。
$n_T[\mathrm{i}]$：來源語料庫中第 i 個音節的總數。
$\mathrm{N_s}$：國語帶聲調的音節個數(1333)。

也就是某個音節的加權值等於來源語料庫的音節總數除以這個音節在來源語料庫中出現的次數，意即出現頻率愈小的音節會有愈高的分數，這個特性有助於選句處理，以下會有進一步說明。此外，上述這些音節加權值將隨著平衡句逐一選出後而有異動。

步驟2：計算來源語料庫中所有尚未被選取句子的加權值，計算公式如下：

$$\mathrm{S_S} = \frac{1}{L}\sum_{i=1}^{\mathrm{L}} \mathrm{S[syl[i]]} * \mathrm{W_{hs}} * \mathrm{W_{hif}} * \mathrm{W_L}$$

$\mathrm{W_{hs}} = 1 - 0.9*$(句中同音字數目/句長(L))
$\mathrm{W_{hif}} = 1 - 0.9*$(句中相同聲韻母數/句中聲韻母總數)
$\mathrm{S_S}$：某一候選句的加權值。
L：某一候選句的音節總數。
syl[i]：某一候選句的第 i 個音節。
$\mathrm{W_{hs}}$：某一候選句中同音字(Homonym)的加權係數。
$\mathrm{W_L}$：某一候選句之句長加權係數。如果句長介於6至12之間，則$\mathrm{W_L}$為1；否則為0.5。

它的作法是把候選句中每個字所對應的音節加權值總和，然後除以句子長度，所得的結果來當做候選句的加值；由於候選句中可能會出現同音字的情況，所以我們必須要乘上候選句中同音字的加權係數W_{hs}，W_{hs}的基本想法是要把具有同音字的候選句的重要性降低；但是如果某個句子全為同音字，而在來源語料庫中這個音節只出現在這個句子中，則依據我們的算法$W_{hs}=0.1$，最後還是會把這句選入平衡句集合中，但是如果在計算公式中少掉0.9這因子，則$W_{hs}=0$，使得該候選句完全沒有機會被選入，則此時不能夠滿足第一項選句原則：平衡句集合必須包含來源語料庫中所有辨識單元的條件。同理可得W_{hif}。至於W_L的使用則是為了滿足第二項選句原則。

步驟3：選取加權值最高的句子(Stc)加入連續語音特性平衡句集合中。

步驟4：檢查平衡句集合中是否已包含來源語料庫的所有音節，如果是，則結束第一階段選句過程，進入第二階段選句過程，否則繼續執行步驟5。

步驟5：將Stc內每個字所對應的音節加權歸零。

```
for i = 1 to L of Stc
   S[syl[i]] = 0;
syl[i]：Stc的第i個音節。
```

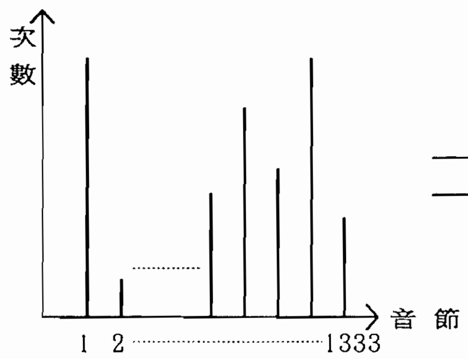
重覆步驟2到步驟5直到連續語音特性平衡句集合中包含來源語料庫中所有出現的音節為止。

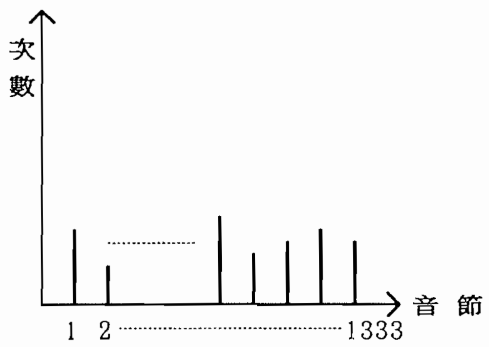
在這個階段的初始加權值給法是來源語料庫中的音節總數除以這個音節在來源語料庫中出現的個數。意即在來源語料庫中出現愈少的音節，其

加權值會愈高。所以含有這些音節的候選句子，它的加權值會相對地較其它的候選句子為高，因此會先被選進連續語音特性平衡句集合中。其原因可以下例進一步說明：

(1)	我	讀	一	本	書
	ㄨㄛˇ	ㄉㄨˊ	ㄧˋ	ㄅㄣˇ	ㄕㄨ
(2)	我	看	一	本	書
	ㄨㄛˇ	ㄎㄢˋ	ㄧˋ	ㄅㄣˇ	ㄕㄨ

假設(1)(2)兩句是來源語料庫中的兩個句子，如果ㄉㄨˊ這個音節在來源語料庫中只在句(1)中出現，而ㄎㄢˋ這個音節有較高的出現頻率，從加權值來看，會先選取句(1)到連續語音特性平衡句集合中，因此，後續的選句過程就無須將句(2)再選入，但是如果我們先選取句(2)時，因為ㄉㄨˊ只在句(1)中出現，所以我們必須將句(1)也選入連續語音特性平衡句集合中，這樣的話ㄨㄛˇ、ㄧˋ、ㄅㄣˇ、ㄕㄨ就多選一次，不符合我們希望以較少量字數來句含語料庫中所有音節的要求。此外，在我們把某一候選句選入連續語音特性平衡句集合之後，則將這個句子中所出現音節的加權值歸零，在重覆選句的過程中，只要某一句子的加權值不為0，就表示它含有連續語音特性平衡句集合目前所沒有的音節。因此，當最高分的候選句的加權值是0時，就表示已經找到所有的音節。而這個階段希望達到的理想狀態如下圖所示：





來源語料庫
音節分佈情況

連續語音特性平衡句
音節分佈情況

四、第二階段選句方法：

第二階段的選句目的是要趨近來源語料庫的音節統計分佈情形，因此在衡量相似程度方面，採用下面的作法，我們把來源語料庫和連續語音特性平衡句集合的音節分佈情況視為兩個向量，分別是：

$$\vec{V}_1 = (n_T[1], n_T[2], \ldots n_T[i] \ldots, n_T[N_S])$$

代表來源語料庫音節分佈情形
其中，$n_T[i]$表示來源語料庫中第i個音節的總數

$$\vec{V}_2 = (n_b[1], n_b[2], \ldots n_b[i] \ldots, n_b[N_S])$$

代表連續語音特性平衡句音節分佈情形
其中，$n_b[i]$表示連續語音特性平衡句集合中第i個音節的總數

因此，我們可將二者的音節統計分佈相似度定義如下：

$$\text{統計分佈相似度} = \frac{\vec{V}_1 \bullet \vec{V}_2}{|\vec{V}_1||\vec{V}_2|} = \cos\theta$$

亦即以此兩向量的正規化(Normalized)內積值作為統計分佈的相似度，這個值也相當於兩個向量夾角的餘弦值。顯然的，當$\vec{V}_1 = k\vec{V}_2$時，cos θ =1，亦即二者完全相同。

這一階段的輸入是來源語料庫，及第一階段所選出的語音特性平衡句，而輸出是具備較理想統計分佈的連續語音特性平衡句集合（測試語料）。

步驟1：根據來源語料庫及第一階段選入之連續語音特性平衡句集合的音節分佈，對每個音節給定一初始加權值，來代表其重要性，方法如下：

```
for i = 1 to $N_S$
{
    $S[i]$ = Constant;
    $S_d[i] = S[i] / n_T[i]$;
    $S[i] = S[i] - S_d[i] * n_b[i]$;
}
```

N_s	:國語帶聲調的音節個數(1333)。
$n_T[i]$	:來源語料庫中第i個音節的總數。
$n_b[i]$	:連續語音特性平衡句集合中第i個音節的總數。
$S_d[i]$	:第i個音節的減分加權值。

起初，所有的音節都給定同樣的加權值，$S_d[i]$則表示某個音節在每次被選入連續語音特性平衡句所要減掉的分數。 比如說對每個音節給定的初始加權值都是1000，而某個音節在來源語料庫中出現10次，則其減分加權值為1000/10=100分。如果這個音節在第一階段中已經被選入3次，則這個音節真正的初使加權值為1000−100×3=700分。接下來解釋為何這樣的加權值給法可以反映出音節在來源語料庫的分佈情況，比如說另一個音節在來源語料庫中出現50次，已經選入5次，則其初始加權值為1000− (1000/50)×5=900分，因此從音節的加權值來看，我們便可以發現這個音節在這個階段的選句過程中較前一音節重要。透過這樣的加權值給法，即可達到第二階段的選句目的。

步驟2：計算來源語料庫中所有尚未被選取句子的加權值。細節同第一階段的步驟2。

步驟3：選取加權值最高的句子(Stc)，看它在加入平衡句集合後是否能夠改進統計分佈相似度，如果不行則將目前加權值最高的句子，其加權值歸零，重覆步驟3。如果選入的句子(Stc)可以改進統計分佈相似度，則將Stc加入連續語音特性平衡句集合中，並進入步驟4。

步驟4：更新新選入句所包含音節之加權值。

```
for i = 1 to L of Stc
S[syl[i]] = S[syl[i]]- $S_d$[syl[i]];
```

重覆步驟2至步驟4直到達成我們要求的語料相似度標準為止。

綜上所述，我們歸納出這個方法的一些特點。兩個階段的選句結果都各有其用途，而我們只要更換來源語料庫或重新定義辨識單元，就可以將它轉換到不同的應用領域(Domain)上。再者，因為使用的運算並不複雜，所以在執行速度上是可以接受的。此外，因為資料來源是語料庫，所以選出的語音特性平衡句有一定的品質。

五、實驗結果

在初步的舉例實驗中，我們選用1333個帶聲調的音節為辨識單元，而來源語料庫是中國時報一個月份的報紙，它含有107419個句子、822829個字元。實驗結果如表1所示，其中效益的定義如下：

$$效益 = \frac{目前累積音節數}{語料庫音節總數} \times 100\%$$

句　　數	角　　度	餘　弦　值	累積音節數	效　　益
366	24.990	0.9064	2790	0.34%
400	19.777	0.9410	3022	0.37%
450	14.515	0.9681	3377	0.41%
500	11.433	0.9802	3723	0.45%
550	9.295	0.9869	4067	0.49%
600	7.808	0.9907	4405	0.54%
650	6.742	0.9931	4744	0.58%
700	5.817	0.9949	5093	0.62%
750	5.171	0.9959	5477	0.67%

表 1 ：選句結果統計表

表 1 中的第一列數據為第一階段的選句結果,其餘部份則為第二階段的選句結果,從上表可以明顯地看出只須選到366句(2790音節)，即可涵蓋來源語料庫中所有出現的音節。附錄A提供部份平衡句以供參

考。平均而言，每個音節只出現大約2.5次。此外，僅就第一階段選句的結果而言，已經和來源語料庫有相當的相似程度，因為常用的音節總是會在選每一音節時被夾帶選入，故總數一定較多。當我們進行第二階段的選句時，隨著句數的增加，相似度也隨著迅速提高，當選到750句時，來源語料庫及連續語音特性平衡句集合分佈向量的夾角約等於5度，而其句數遠小於來源語料庫中的句子數目，我們認為這是一個相當不錯的結果。

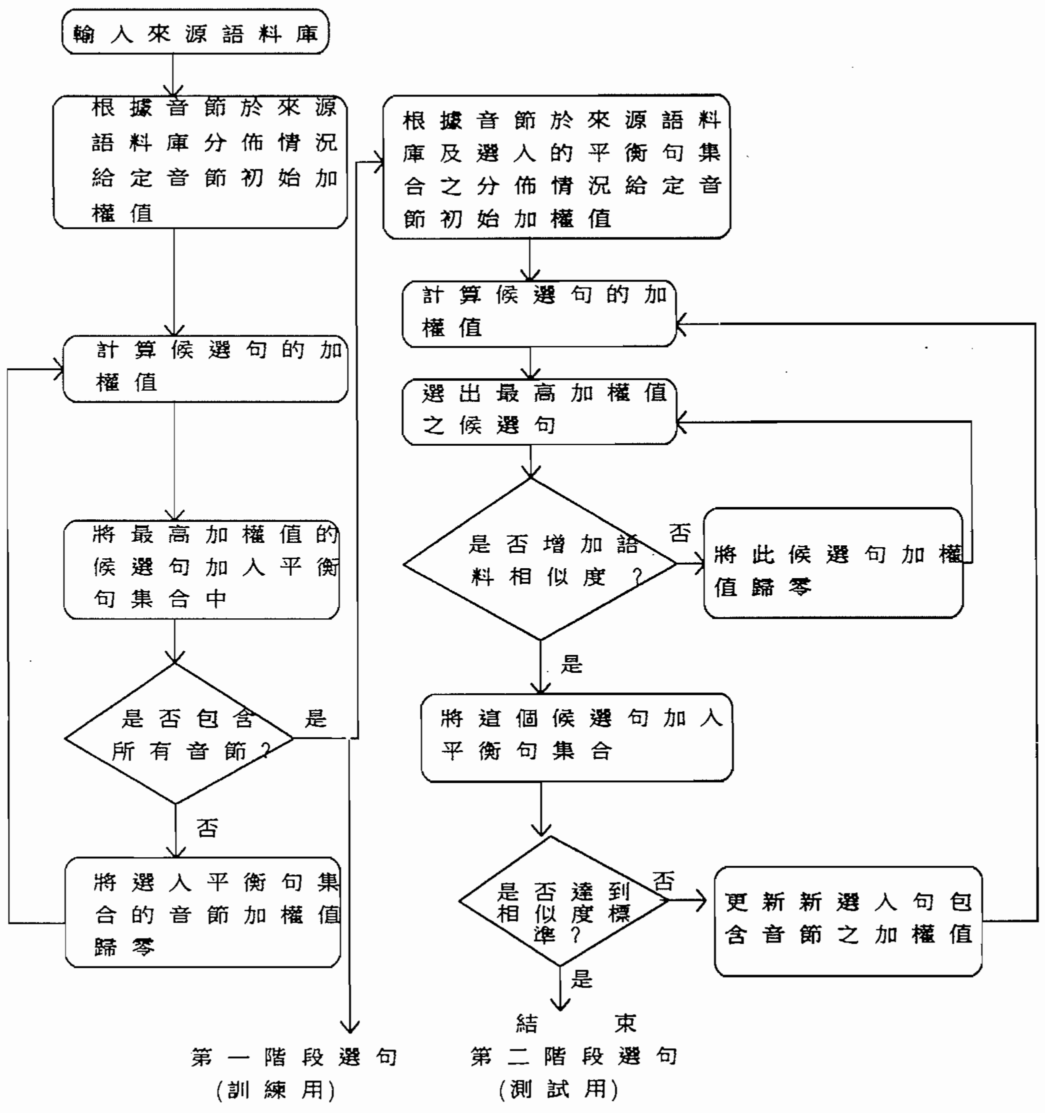
六、結論：

在本篇論文中，我們提出了一個能從語料庫中自動選取連續語音特性平衡句的方法，它不僅能夠幫助我們產生訓練語料，而且也可以產生測試語料。就方法的層次而言，我們提供了相當合理的說明，其實用性及正確性也從實際執行所得的結果得到證實。

參考文獻：

〔1〕 "IEEE Recommended Practice for Speech Measurement", IEEE Transactions on Audio and Electroacoustics, Vol. AU-17, No. 3, pp. 225-246, Sep, 1969.

〔2〕 余秀敏、劉繼謚，"國語語音特性平衡句的建立"，電信研究季刊，第19卷，第1期，民國78年3月。



輸入來源語料庫
根據音節於來源語料庫分佈情況給定音節初始加權值
計算候選句的加權值
將最高加權值的候選句加入平衡句集合中
是否包含所有音節？
是
否
將選入平衡句集合的音節加權值歸零
第一階段選句
(訓練用)
根據音節於來源語料庫及選入的平衡句集合之分佈情況給定音節初始加權值
計算候選句的加權值
選出最高加權值之候選句
是否增加語料相似度？
否
將此候選句加權值歸零
是
將這個候選句加入平衡句集合
是否達到相似度標準？
否
更新新選入句包含音節之加權值
是
結　束
第二階段選句
(測試用)

圖一：兩階段式選句法

附　錄　A：連續國語語音特性平衡句之部份例句

1. 必須撙節開支
2. 希望別再出摟子
3. 小傢伙靦腆地答
4. 鞋襪應大小合適而且通風
5. 雨天則泥濘不堪
6. 終日在院內踱步
7. 為什麼非得提前授課不可
8. 涉嫌拐誘無知少年離家出走
9. 令人有被捉弄的感覺
10. 捐款救助非洲娃娃
11. 咱們等著瞧
12. 拿著臉盆裝水
13. 所謂覆巢之下無完卵啊
14. 無不目眩神搖
15. 儘管粥少僧多
16. 讓歹徒屢試不爽的得逞
17. 藉以表達慰勞之意
18. 嘉南大圳換上了新面貌
19. 不啻是一個奢侈的夢想
20. 債券可能成為未來投資新寵
21. 引起各界揣測不已
22. 這個理由簡直是荒謬可笑
23. 實在令人納悶
24. 肥胖不見得是福
25. 沒有人能忍受這種窩囊氣
26. 怎麼做都不免挨罵
27. 歡迎共襄盛舉
28. 難道醜人真要作怪嗎
29. 民眾若想趁機撈一筆
30. 醞釀杯葛行動
31. 旗幟飄飄十分美麗
32. 我敢拍胸脯保證
33. 慈濟不講深奧的佛理
34. 時報再領風騷
35. 台灣居民生活富裕

Corpus-based Automatic Compound Extraction with Mutual Information and Relative Frequency Count

Ming-Wen Wu[†]
Keh-Yih Su[‡]

[†]Behavior Design Corporation
2F, 28, R&D Road II
Science-based Industrial Park
Hsinchu, Taiwan 300, R.O.C.
mingwen@bdc.com.tw

[‡]Department of Electrical Engineering
National Tsing-Hua University
Hsinchu, Taiwan 300, R.O.C.
kysu@bdc.com.tw

ABSTRACT

In machine translation systems, a computer-translated manual is usually concurrently processed by several posteditors; thus, to maintain the consistency of translated terminologies between different posteditors is very important. If all the terminologies used in the manual can be entered into the dictionary before machine translation, the consistency can be automatically maintained, which is a big advantage of machine translation over human translation. However, since new compounds are created from day to day, it is impossible to list them exhaustively in the dictionary being prepared long time ago. To guarantee subsequent parsing and translation to be correct, new compounds must be extracted from the text every time a new manual is to be translated and then entered into the dictionary. However, it is too costly and time-consuming to let the human inspect the entire text to search for the compounds. Therefore, to extract compounds automatically from the manual is an important problem. Traditional systems are to encode some sets of rules to extract compounds from the corpus. However, the problem with the rule-based approach is that not every compound obtained is desirable since it does not assign preferences to the candidates. It is not clear whether one candidate is more likely to be a compound than the other. The human effort required is still high because the lexicographer has to search for all the compound candidate list to find the preferred compounds. A new method is thus proposed in this paper to automatically extract compounds using the features of *mutual information* and *relative frequency count*. This method tests every n-gram (n is equal to 2 or 3 in this paper) formed in the manual to see whether it is a compound by checking those features. Those n-grams that pass the test are then listed in the order of significance to let the lexicographers to build into the dictionary. A significant cutdown in postediting time has been observed in our test.

1. Introduction

In technical manuals, technical compounds [Levi 78] are very common. Therefore, quality of their translations greatly affects the performance of machine translation. If a compound is not built into the dictionary before machine translation, in many cases, it would be translated incorrectly. One of the reasons is that many compounds are not *compositional*, which means that the translation of a compound is not the composite of the respective translations of the individual words [Chen 88]. For example, the translation of *green house* into Chinese is not the composite of the Chinese translations of *green* and *house*, so is *paper document*. Another advantage of building compounds into the dictionary before translation is to reduce number of parsing ambiguities. If a compound is not listed in the dictionary, it will be regarded as a group of single words. Since more than half of English words are of multi-categories, a group of words would cause more ambiguities, which will, in turn, reduce the accuracy rate of disambiguation and also increase translation time. In addition, as a manual is usually processed by several posteditors simultaneously, the translation consistency of terminologies is very important. If the compound is not present in the dictionary before machine translation, the posteditors have to spend a lot of time retrieving correct translation of the compound and checking the consistency between different posteditors. Therefore, if all the compounds can be built into the dictionary, quality of translation will be greatly improved, and lots of postediting time can be saved.

To solve the above problems, one might propose to build a huge dictionary which contains all compounds. However, compounds are rather productive, particularly in rapidly updating fields, such as information processing. New compounds are created from day to day. Hence, it is impossible to build a huge dictionary to store all compounds. Another approach is to let the human inspect the manual before machine translation to search for the compounds. Unfortunately, it is too costly and time-consuming because he has to spend a lot of time inspecting the whole manual. Once the compounds are selected, he has to check if the selected compounds are already in the dictionary. Moreover, he is not sure if it deserves the effort to enter the relevant information of the compound into the dictionary if it only appears a few times.

For these reasons, it is important that the compounds be found and entered into the dictionary before translation without much human effort. Hence, a tool to extract compounds automatically from the corpus using some quantitative criteria is seriously required. Several approaches have been proposed to extract compounds from corpus in the past [Bour 92, Calz 90]. Traditional rule-based systems are to encode some sets of rules to find the likely candidates. In LEXTER, a corpus of language texts on any subject is fed in, and the system proceeds in two stages (*analysis* and *parsing*) to produce a list of *likely terminological units* to be submitted to an expert to be validated [Bour 92]. The advantage is that since the analysis and parsing rules are simple and surface grammatical analysis instead of complete syntactic analysis is performed, it is easy to perform very frequent tests. However, since the process is done on the syntactic level without incorporating semantic information and domain knowledge, it might extract many noun phrases which are not desirable terminologies, and thus causes high false alarm. Also, it is not clear whether the terminology is a commonly used one. Since there is no performance evaluation reported, it is not clear how this approach works. Another approach is to adopt statistical measures as the selection criteria. In [Calz 90], the *association ratio* of a word pair and the *dispersion* of the second word in the word pair are used to decide if it is a fixed phrase (a compound). The drawback is that it does not take the number of occurrences of the word pair into account; therefore, it is not known if the word pair is commonly or rarely used. Also, no performance evaluation is given for this method.

In this paper, a statistical approach to solve the compound finding problem is proposed. This method extracts compounds using *mutual information* and *relative frequency count* as the features for selection. The *likelihood ratio test* is then used to check whether an n-gram is a compound. As simulation results of the initial run show, the corpus-based approach works well except that the precision rate is too low. The reason is that there are many compound candidates, though passing the likelihood ratio test, are not suitable to be regarded as compounds, such as a preposition followed by an article (such as "in the") or an auxiliary preceded by a pronoun (such as "you can"). The performance can be improved by augmenting contents of the *exception table*, which stores scores of those entries. After involving the exception table, a significant cutdown of postediting time has been observed in our test, and quality of translation is greatly improved.

2. How to Form the Candidate List for Compounds

The first step to extract compounds is to find the candidate list for compounds. According to our operational experience on machine translation, most compounds are of length 2 or 3. Hence, only bigrams and trigrams in the corpus are of interest to us in compound extraction.

To prepare the raw compound list, a corpus is first fed into the morphological analyzer so that every word in the corpus is transformed into its stem form. The reason for storing the stem form (instead of surface form) of the word is to save memory space. Then, the manual is scanned to find the possible compound candidates. Each sentence is scanned from left to right with the window size 2 and 3. Each group of words within the window of size 2 is put into the bigram list, and each group of words within the window of size 3 is placed in the trigram list. Then, the mutual information and relative frequency count for each entry are computed.

3. Compound Extraction Procedure

3.1. Feature Selection

To find compounds from the file of bigrams and trigrams, we manage to choose some features which can discriminate compounds and non-compounds. Two quantitative features are adopted as selection features for classification, namely *mutual information* and *relative frequency count.* These two features will be discussed in more detail in the following subsections.

3.1.1 Mutual Information

Mutual information is a statistic measure of word associations. It compares the probability of a group of words to occur together (joint probability) to their probabilities of occurring independently. The mutual information in the bigram is computed by the formula [Chur 90]:

$$I(x;y) \equiv \log_2 \frac{P(x,y)}{P(x) \times P(y)},$$

where x and y are two words in the corpus, and $I(x;y)$ is the mutual information of these two words x and y (in this order). $P(x)$ is evaluated as the relative frequency of the number of occurrences of x with respect to the number of total instances of singletons. If there is a genuine association between x and y, i.e. x and y are likely to form a compound, then the joint probability $P(x,y)$ will be much larger than $P(x) \times P(y)$, and consequently $I(x,y) >> 0$. If there is no interesting relationship between x and y, i.e. x and y are not very likely to form a compound, then $P(x,y) \approx P(x) \times P(y)$, and thus $I(x,y) \approx 0$.

The mutual information of trigram is defined as follows [Su 91]:

$$I(x,y,z) \equiv \log_2 \frac{P_D(x,y,z)}{P_I(x,y,z)},$$

where $P_D(x,y,z)$ is defined as the probability for x, y and z to occur jointly ('D'ependently), and $P_I(x,y,z)$ is defined as the probability for x, y and z to occur by chance ('I'ndependently). That is:

$$\begin{aligned} P_D(x,y,z) &\equiv P(x,y,z) \\ P_I(x,y,z) &\equiv P(x) \times P(y) \times P(z) \\ &\quad + P(x) \times P(y,z) + P(x,y) \times P(z). \end{aligned}$$

3.1.2 Relative Frequency Count

The relative frequency count r_i for the *i-th* bigram (trigram) is defined as:

$$r_i = \frac{f_i}{K},$$

where f_i is the total number of occurrences of the *i-th* bigram (trigram), which is the number of occurrences of the entry in the manual, and K is the average number of occurrences of all the entries. In other words, f_i is normalized with respect to K to get the relative frequency.

As the more often a group of words appear together in the corpus, the more likely it will be a compound, the relative frequency count is used as a feature for selecting compounds. Since the cost of entering the relevant information of a compound into the dictionary is not low, it may not worth to enter a compound into the dictionary if it occurs only a few times. Moreover, for there is no inconsistency problem if a compound occurs only once, there is no need to build this kind of compounds into the dictionary.

The reason of using both the mutual information and relative frequency count as the features for selection is that using either of these two features alone can not provide enough information for compound finding. The problem with using relative frequency count alone is that it is likely to choose the bigram (trigram) with high relative frequency count but low mutual information among the words comprising the compound. For example, let the relative frequency of word x be $P(x)$, and the relative frequency of word y be $P(y)$. If $P(x)$ and $P(y)$ are very large, which may cause a large $P(x,y)$ even they are not related. However, $\frac{P(x,y)}{P(x) \times P(y)}$ would be small for this case.

On the other hand, the problem with using mutual information alone is that it is highly unreliable if $P(x)$ and $P(y)$ are too small. The chosen compound has high mutual information not because the words within it are highly correlated but due to a large estimation error. Furthermore, it may not worth the cost of entering the compound into the dictionary if it occurs very few times. Actually, the relative frequency count and mutual information supplement each other. A group of words of both high relative frequency count and mutual information is most likely to be composed of words which are highly correlated, and very commonly used. Hence, it is a preferred compound candidate.

3.2. Establishing Statistics of Training Corpus

The corpus which has been processed before and checked by the human can be used as the knowledge source, because all the real compounds in the corpus have already been built into the dictionary. The corpus is divided into two parts, one as the training corpus, and the other the testing set. Every word in the corpus is first converted into its stem form through morphological analysis. In this paper, the number of words in the training corpus is 74,404.

The bigrams and trigrams in the training corpus are divided into two clusters. The compound cluster comprises the bigrams and trigrams already in the dictionary, and non-compound cluster is composed of the bigrams and trigrams which are not in the dictionary. After the distribution statistics of two clusters are first estimated, we calculate the mean and standard deviation of mutual information and relative frequency count. The entries with outlier values (outside the range of 3 standard deviations of the mean) are discarded for the *robustness* of estimating statistic parameters. And, the entries of frequency count 1 are deleted for it is of little importance because there is no inconsistency problem with the term which occurs only once. Then, the statistics are estimated once again. The means and variances of mutual information and relative frequency count in both clusters are then estimated using the following formulae [Papo 90]:

$$\hat{\mu}_m = \frac{1}{n}\sum_{i=1}^{n} m_i,\quad \hat{\sigma^2_m} = \frac{1}{n-1}\sum_{i=1}^{n}(m_i - \hat{\mu}_m)^2$$

$$\hat{\mu}_r = \frac{1}{n}\sum_{i=1}^{n} r_i,\quad \hat{\sigma^2_r} = \frac{1}{n-1}\sum_{i=1}^{n}(r_i - \hat{\mu}_r)^2$$

where m_i is the mutual information of the *i-th* bigram (trigram), r_i is the relative frequency count of the *i-th* bigram (trigram), $\hat{\mu}_m$ is the estimated mean of mutual information, $\hat{\mu}_r$ is the estimated mean of relative frequency count, $\hat{\sigma^2_m}$ is the estimated variance of mutual information, $\hat{\sigma^2_r}$ is the estimated variance of relative frequency count, and *n* is the number of bigrams (trigrams). Since we regard the bigram and trigram models as different models, the distribution statistics are estimated separately.

The covariance μ_{mr} and correlation coefficient r_{mr} of the two clusters can be estimated as follows [Papo 90]:

$$\hat{\mu}_{mr} = \frac{1}{n-1}\sum_{i=1}^{n}(m_i - \hat{\mu}_m)(r_i - \hat{\mu}_r)$$

$$\hat{r}_{mr} = \frac{\hat{\mu}_{mr}}{\sigma_m \sigma_r}$$

The distribution statistics of the training corpus is shown in Table 1 and 2. (MI: mutual information, RFC: relative frequency count, cc: correlation coefficient, sd: standard deviation)

	number	mean of MI	sd of MI	mean of RFC	sd of RFC	covariance	cc
bigrams	659	6.799	3.011	1.674	1.726	-0.139	-0.027
trigrams	169	6.955	2.635	2.950	2.747	-0.930	-0.128

Table 1: Distribution statistics of compounds

	number	mean of MI	sd of MI	mean of RFC	sd of RFC	covariance	cc
bigrams	8151	4.116	3.271	1.436	1.473	-0.670	-0.139
trigrams	9392	4.859	2.763	1.627	0.706	-0.279	-0.143

Table 2: Distribution statistics of non-compounds

From Table 1 and 2, we can see that the means of mutual information and relative frequency count of compound cluster are larger than those in non-compound cluster. And, mutual information and relative frequency count are almost uncorrelated in both clusters since the correlation coefficients are close to 0.

Let M and R be the random variables which denote the mutual information and relative frequency count, respectively. Assume M and R are of Gaussian distribution. Let μ_m be the mean of mutual information of compound cluster, and $\mu_m^{'}$ of non-compound cluster, μ_r be the mean of relative frequency count of compound cluster, and $\mu_r^{'}$ of non-compound cluster, σ_m be the standard deviation of mutual information of compound cluster, and $\sigma_m^{'}$ of non-compound cluster, σ_r be the standard deviation of relative frequency count of compound cluster, and $\sigma_r^{'}$ of non-compound cluster, r be the correlation coefficient of mutual information and relative frequency count in compound cluster, and r' in non-compound cluster. The bivariate probability density function of the compound and non-compound clusters can be expressed as [Papo 90]:

$$f(M,R \mid Compound) = \frac{1}{2\pi\sigma_m\sigma_r\sqrt{1-r^2}} exp\left\{-\frac{1}{2(1-r^2)}\left(\frac{(M-\mu_m)^2}{\sigma_m^2} - 2r\frac{(M-\mu_m)(R-\mu_r)}{\sigma_m\sigma_r} + \frac{(R-\mu_r)^2}{\sigma_r^2}\right)\right\}$$

$$f(M,R \mid Non-Compound) = \frac{1}{2\pi\sigma_m^{'}\sigma_r^{'}\sqrt{1-r^{'2}}} exp\left\{-\frac{1}{2(1-r^{'2})}\left(\frac{(M-\mu_m^{'})^2}{\sigma_m^{'2}} - 2r'\frac{(M-\mu_m^{'})(R-\mu_r^{'})}{\sigma_m^{'}\sigma_r^{'}} + \frac{(R-\mu_r^{'})^2}{\sigma_r^{'2}}\right)\right\}$$

3.3. Two Cluster Classification

Given the joint distribution of mutual information and relative frequency count, the compound extraction problem can be formulated as a *two cluster classification problem*; that is, to assign a group of words x into either one of two clusters: compound cluster or non-compound cluster. If we form the ratio [Papo 90]

$$\lambda = \frac{f\left(X \mid it\ is\ \ a\ compound\right)}{f\left(X \mid it\ is\ not\ a\ compound\right)},$$

then a test based on the statistic λ is called the *likelihood ratio test.* If $\lambda > 1$, it is more likely that x belongs to the compound cluster. Otherwise, it is assigned to the non-compound cluster. Alternatively,

$$\begin{aligned} If\quad \lambda &= \frac{f\left(X \mid it\ is\ a\ compound\right)}{f\left(X \mid it\ is\ not\ a\ compound\right)}, \\ \text{then}\quad \ln\lambda &= -\frac{1}{2\left(1-r^2\right)}\left(\frac{\left(M-\mu_m\right)^2}{\sigma_m^2} - 2r\frac{\left(M-\mu_m\right)\left(R-\mu_r\right)}{\sigma_m\sigma_r} + \frac{\left(R-\mu_r\right)^2}{\sigma_r^2}\right) \\ &+\frac{1}{2\left(1-r'^2\right)}\left(\frac{\left(M-\mu_m'\right)^2}{\sigma_m'^2} - 2r'\frac{\left(M-\mu_m'\right)\left(R-\mu_r'\right)}{\sigma_m'\sigma_r'} + \frac{\left(R-\mu_r'\right)^2}{\sigma_r'^2}\right) \\ &+\ln\left(2\pi\sigma_m'\sigma_r'\sqrt{1-r'^2}\right) \\ &-\ln\left(2\pi\sigma_m\sigma_r\sqrt{1-r^2}\right). \end{aligned}$$

Therefore, if $\ln\lambda > 0$, there are more chances that x is a compound than it is not. To take the a priori probabilities of $P\left(x\ is\ a\ compound\right)$ and $P\left(x\ is\ not\ a\ compound\right)$ into consideration, the above equation can be changed to if $\ln\lambda > \beta$, then x is a compound, where β is a function of $P\left(x\ is\ a\ compound\right)$ and $P\left(x\ is\ not\ a\ compound\right)$. In our test, β is set to 0.

3.4. Extraction Procedure

After testing the above formula, we have found that there are some bigrams (trigrams) which have a large λ (greater than 1), but are not suitable to be regarded as compounds. For example, a preposition followed by an article (like "in the") has a very large λ, but it is not reasonable to regard it as a compound. Therefore, we use the *exception table* to store those entries. If a bigram (trigram) is found to be in the exception table, it will no longer be considered as a compound candidate.

To put it briefly, each bigram and trigram in the testing corpus is put into the following algorithm to see if it is a compound.

```
For each bigram (trigram) $x$
    if $(\lambda_x < 1)$ then
        $x$ is not a compound; exit
    else
        if $x$ is in the dictionary then
            ignore
        else
            if $x$ is in the exception table then
                $x$ is not a compound; exit
            else
                place $x$ into the compound list file
End
```

The entries in the compound list file are listed in the order of significance (in the descending order of λ) to be examined by lexicographers.

4. Simulation Results

The following experiment is conducted to investigate the performance of the compound extracting method for the training corpus and the testing set. Each bigram (trigram) is put into the above algorithm for testing. If it passes the test (i.e. $\lambda > 1$ and not in the exception table), it will be recognized as a compound. Otherwise, it will be regarded as a non-compound.

There are totally 6014 bigrams and 8620 trigrams in the testing set. The performance of compound extraction for bigrams and trigrams is shown in Table 3 and 4. The simulation results are quite satisfactory

	training corpus	testing set
recall rate	68.736	60.218
precision rate	66.985	55.380

Table 3: Performance for bigrams (%)

	training corpus	testing set
recall rate	68.853	63.830
precision rate	62.687	39.474

Table 4: Performance for trigrams (%)

Table 5 shows the first five bigrams and trigrams with the largest λ and not in the exception table for the testing set. Among them, four out of five bigrams (except *select text*) and three out of five trigrams (except *dialog box display* and *mouse pointer assume*) are plausible compounds. Also, we can see if the part of speech can be adopted as another feature in modeling, the recall rate and precision rate can be greatly improved simultaneously.

bigram	trigram
paragraph style	dialog box display
insertion point	paragraph style set
dialog box	mouse pointer assume
select text	unit of measurement
style sheet	main document text

Table 5: The first five bigrams and trigrams with the largest λ and not in the exception table for the testing set

5. Conclusion

In machine translation systems, information of the words of source language should be available before any translation process can begin. The new simple words can be found by spelling check, and they are not as productive as compounds, so that the relevant information of simple words can be entered into the dictionary before translation. However, the handling of compounds is more difficult. Since compounds are very productive, new compounds are created from day to day in every domain. Obviously, it is impossible to build a huge dictionary to contain all compounds. To guarantee correct parsing and translation, new compounds must be extracted from the text to be translated and entered into the dictionary. However, it is too costly and time-consuming for the human to inspect the entire text to find the compounds. Therefore, a method to extract compounds from corpus automatically is required.

The method proposed in this paper uses mutual information and relative frequency count as two features for selection to discriminate compounds and non-compounds. The compound extracting problem is formulated as a two cluster classification problem in which a bigram (trigram) is assigned to one of those two clusters. If a bigram (trigram) is assigned to the compound cluster, it will be put into the list of potential compound candidates. Otherwise, it is discarded. The entry already in the dictionary or in the exception table will be discarded, too. With this method, the time for posteditors to retrieve the correct translation of missed compounds can be greatly reduced, and the consistency between different posteditors is easier to maintain.

The recall rate and precision rate can be further improved if the part of speech of each word can be used as a feature in modeling. Therefore, in future research, the part of speech will be adopted as another feature for selection besides mutual information and relative frequency count.

References

[Bour 92] Bourigault, D., 1992. "Surface Grammar Analysis for the Extraction of Terminological Noun Phrases," *Proceedings of COLING-92*, vol. 4, pp. 977–981, 14th International Conference on Computational Linguistics, Nantes, France, Aug. 23–28, 1992.

[Calz 90] Calzolari, N. and R. Bindi, 1990. "Acquisition of Lexical Information from a Large Textual Italian Corpus," *Proceedings of COLING-90*, vol. 3, pp. 54–59, 13th International Conference on Computational Linguistics, Helsinki, Finland, Aug. 20–25, 1990.

[Chen 88] Chen, S.-C. and K.-Y. Su, 1988. "The Processing of English Compound and Complex Words in an English-Chinese Machine Translation System," *Proceedings of ROCLING I*, Nantou, Taiwan, pp. 87–98, Oct. 21–23, 1988.

[Chur 90] Church, K.-W. and P. Hanks, 1990. "Word Association Norms, Mutual Information, and Lexicography," *Computational Linguistics*, pp. 22–29, vol. 16, Mar. 1990.

[Levi 78] Levi, J.-N., "The Syntax and Semantics of Complex Nominals," *Academic Press, Inc.*, New York, NY, USA, 1978.

[Papo 90] Papoulis, A., "Probability & Statistics," *Prentice Hall, Inc.*, Englewood Cliffs, NJ, USA, 1990.

[Su 91] Su, K.-Y., Y.-L. Hsu and C. Saillard, 1991. "Constructing a Phrase Structure Grammar by Incorporating Linguistic Knowledge and Statistical Log-Likelihood Ratio," *Proceedings of ROCLING IV*, Kenting, Taiwan, pp. 257–275, Aug. 18–20, 1991.

中文文件自動分類之研究

A Study of Document Auto-Classification in Mandarin Chinese

†楊允言 ‡謝清俊 ⋆陳淑美 ‡陳克健

† 中央研究院資訊科學研究所研究助理

‡ 中央研究院資訊科學研究所研究員

⋆ 國立臺灣大學圖書館館員

摘　　要

本論文中,我們提出利用雙連字串(Bigram)替代關鍵詞的方法,來做中文文件自動分類的實驗。其目的,是要讓電腦來幫忙做中文文件分類,減輕人的負擔。

我們從工商時報民國80年7月到81年1月間取樣出來的2306篇財經類新聞報導,包括產業、企業、機械、電機、資訊五大類,共24小類,先以人工將之分類,並分爲訓練資料(2095篇)及測試資料(211篇)兩部分,根據次數、集中度、廣度三項條件,從訓練資料得到具有分類價值的關鍵詞,以向量模式、機率模式,和不同的分類比重方式來做自動分類實驗,並比較其結果。實驗結果,測試資料有67%左右的正確率(召回率),若取前三名有80%的正確率;至於訓練資料則有97%的正確率。

在文中,我們探討了關鍵詞的篩選以及文件自動分類的方法,採用向量模式時,並討論了標準化的方法;同時,我們針對電腦與人工在做分類以及相似性排序時的不同點提出簡單的比較與討論,讓我們了解之間的差異。

1　緒言

根據統計,自從1971年開始,平均每2.3年,線上資料庫的數量就增加一倍,而這些線上資料庫內的資料,則以更快的速度增加中[Smi89,p1]。如果沒有適當地存放這些資料,以後當我們要找尋所需的資料時,可能會遇到類似海底撈針的窘境。文件自動分類,就是將文件做某種方式的排列,使性質相近的文件被放在相同或靠近的地方,以便當人們要從眾多文件中查詢到其所需時,能有效率且迅

速地得到。

傳統的分類工作需要利用大量的人力,這樣不僅要耗去很多時間,並且,人工分類一直存在一個問題,即不同的人會做出不同的結果,不管是建索引或是判定文件的相似性等等,其一致性並不高[Sal86]。既然如此,如果我們利用電腦來幫人們做這件事,上述的兩個問題,就可以得到解決。

利用電腦來做文件自動分類的實驗,從1960年代就已開始[Mar61] [BoBe63],並陸陸續續有相關論文發表,如[Kwo75][HaZa80],最近這一、兩年則又引起了較多的關注,如[Lar92][BHMP92][Jac92][Jac93][Lew92]···等等。一般而言,利用電腦來做文件自動分類,不論哪種方式,大致包括以下的步驟:

1. 選定文件(Document)集合,並選定文件替代品(Profile),例如只採用題目、摘要等等,做為訓練及測試資料。並選定類別,即欲將文件分為哪些類。

2. 從文件替代品中,根據訂定的篩選規則,找出所要的關鍵詞。

3. 類別的指定,可以採用向量模式(Vector Space Model)或是機率模式(Probability Model)來計算。

向量模式的做法是,假設關鍵詞有$T_1, T_2, \cdots, T_m$共m個,文件分為$C_1, C_2, \cdots, C_n$共n類,我們以向量$X_j = (x_{1j}, x_{2j}, \ldots, x_{mj})^T$表示類別$C_j$,對一文件D,我們以向量$Y = (y_1, y_2, \ldots, y_m)^T$表示,關鍵詞$T_i$若出現在類別$C_j$中,則$x_{ij} > 0$,否則$x_{ij} = 0$,關鍵詞$T_i$若出現在文件D中,則$y_i > 0$,否則$y_i = 0$,將向量Y分別與向量$X_j$做內積運算,若其值最大,則表示文件D屬於此類。

,如何決定x_{ij}值是一個研究的主題,x_{ij}為何,有何意義、x_{ij}如何選取,x_{ij}的值是否要加權(Weight),x_{ij}值要不要做正規化(Normalization)、y_i的值是以二元關係(也就是說非0即1)或是以文件D的替代品(Profile)中詞彙k_i出現的次數為準,···等等不一而足。 若將向量模式的方法視為分數(Scores)的相加,則機率模式的方法也可以簡單地視為是分數的相乘。假設一篇文件D的替代品(Profile)中出現r個關鍵詞k_{i1}、k_{i2}、···、k_{ir},則此文件屬於類別C_j的機率為

$$P(C_j | k_{i1}, k_{i2}, \ldots, k_{ir})$$

根據貝氏定理,上式可化簡為

$$\begin{aligned} & \frac{P(C_j) \times P(k_{i1}, k_{i2}, \ldots, k_{ir} | C_j)}{P(k_{i1}, k_{i2}, \ldots, k_{ir})} \\ = \quad & r \times P(C_j) \times P(k_{i1}, k_{i2}, \ldots, k_{ir} | C_j) \\ = \quad & r \times P(C_j) \times P(k_{i1} | C_j) \times P(k_{i2} | C_j, k_{i1}) \times \cdots \times P(k_{ir} | C_j, k_{i1}, k_{i2}, \ldots, k_{i,r-1}) \end{aligned}$$

假設k_{i1}、k_{i2}、…、k_{ir}兩兩互相獨立(Independent),則上式可變成

$$r \times P(C_j) \times P(k_{i1}|C_j) \times P(k_{i2}|C_j) \times \cdots \times P(k_{ir}|C_j)$$

其中$r = 1/P(k_{i1}, k_{i2}, \ldots, k_{ir})$爲一常數[Mar61][HaZa80]。

如果將這些機率值取對數(Logarithm)之後再做運算,相乘變成相加,則我們又可將機率模式與向量模式看做是同一回事。

以上所提到的,都是針對英文及法文所做的,而我們清楚,中文與西方語言有相當大的差異。[Che92]開始處理中文的文件自動分類,但是其中的一個困難處是關鍵詞必須由人工所選取的。因此我們採用中文的雙連字串(Bigram)來取代「關鍵詞」,並且發現,利用雙連字串可以得到相同的結果。

本篇論文的研究方法與步驟,簡單敘述如下:

1. 選定資料(新聞報導)的範圍及分類系統。我們採用日本產經新聞所定的分類法,共挑出五大類、24小類。

2. 以工商時報爲對象,取民國80年7月至81年1月,每8天取樣一天,將當天合於選定範圍的新聞報導做爲樣本,共有2306篇,將之劃分爲訓練資料(2095篇)及測試資料(211篇)。

3. 根據次數、集中度以及廣度三條件,從訓練資料中找出雙連字串,並指定此雙連字串在各類的分類比重。

4. 分別利用向量模式及機率模式,不同的分類比重給定方法來做自動分類,並針對錯誤部分做進一步的探討。

2 實驗方法

2.1 資料選取及類別選定

本實驗所採用的文件是工商時報80年7月到81年1月的新聞報導,每8天取樣1天,共抽樣22天爲樣本。將前六個月,共20天的新聞報導(文件)爲訓練資料,最後一個月,共2天的新聞報導當做測試資料。這些文件(新聞報導)是存放在電腦內,以BIG-5碼儲存。

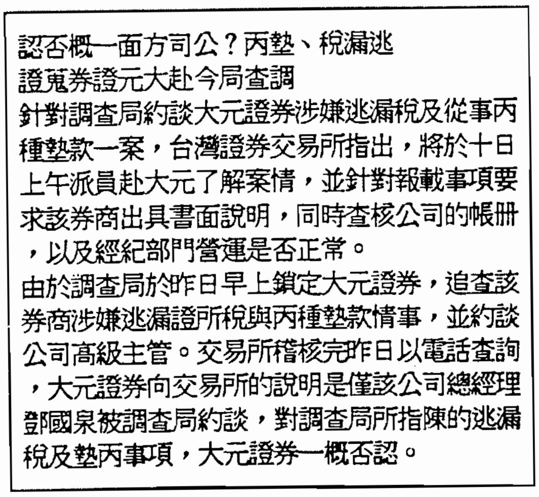

認否概一面方司公？丙墊、稅漏逃
證蒐券證元大赴今局查調
針對調查局約談大元證券涉嫌逃漏稅及從事丙
種墊款一案，台灣證券交易所指出，將於十日
上午派員赴大元了解案情，並針對報載事項要
求該券商出具書面說明，同時查核公司的帳冊
，以及經紀部門營運是否正常。
由於調查局於昨日早上鎖定大元證券，追查該
券商涉嫌逃漏證所稅與丙種墊款情事，並約談
公司高級主管。交易所稽核完昨日以電話查詢
，大元證券向交易所的說明是僅該公司總經理
鄧國泉被調查局約談，對調查局所指陳的逃漏
稅及墊丙事項，大元證券一概否認。

圖2-1 新聞報導的例子

此外,這些資料中,也會有一些雜訊(noise),這些雜訊主要來自兩方面:第一是標題,標題有右至左以及左至右;第二則是錯字。這些雜訊,自然多少會影響到後來關鍵詞的選取工作。至於存放電腦中的新聞報導,是20字一行,經過處理後,平均一篇報導有31.13行;此外,2095篇訓練資料中,一共有943961個中文字(3569個不同的中文字),852387個中文Bigram(40085個不同的中文Bigram)。

至於類別,我們採用日本產經新聞的分類方式,採用的原因請參閱[Che92, p30]。該分類法共分22大類、150小類。我們取其中五大類來做實驗,而這五大類中,又共分為45小類,我們將新聞報導數小於10篇的類別刪去,因為數量太少時,很難去做些有意義的統計或計算;這樣子剩下24小類。如此,本實驗所選定的類別即為此24類,訓練資料共有2095篇屬於此24類的新聞報導,測試資料則有211篇新聞報導。

2.2 關鍵詞(雙連字串)的選取

篩選雙連字串(Bigram),我們設了三個條件:(1)次數(2)集中度(3)廣度。

我們採用雙連字串,而沒有加入斷詞系統,主要是基於下列幾點考量:

1. 用雙連字串代價比較低;

2. 針對專有名詞以及縮寫詞,目前的斷詞系統並不能有效地解決;

3. 根據[Che92],我們發現,利用雙連字串當做關鍵詞與人工挑選關鍵詞所得的實驗結果,正確率(召回率)相去不遠。

與人工選取關鍵詞相比較,人工挑選出來的關鍵詞可能都比較具有意義(就人的觀點而言),然而,人工挑選永遠會遇到一個問題,就是不同的人會做出不同的結果,可能比較不客觀;另外,人覺得有分類意義的詞彙,不見得都是很有分類價值的詞彙。我們希望能夠做到完全自動化。

就分類系統而言,一個具有分類價值的關鍵詞,應該滿足下列三條件:

1. 次數要夠:雙連字串並非都是一個有意義的詞,任意相鄰的兩個中文字即可形成一個雙連字串;通常,一個不具意義的雙連字串出現的次數不會多,如果定一個界限值,出現次數低於此界限值者就去掉,則那些無意義的雙連字串大部分都會被摒除在外了。

2. 集中度:一個有分類價值的雙連字串,應該要集中出現在某一類或某幾類中,而不是平均分佈在各類中。

3. 廣度:在某一類頻頻出現的雙連字串,如果它出現在這類中許多篇文件裡,則它愈具有分類的價值,相反地,若此雙連字串,雖然出現次數夠多,但是卻只集中在某幾篇文件中,這種雙連字串的出現,原因可能為某一突發事件,或是撰稿者特殊的寫作風格所致,而這種雙連字串,其分類的價值相對上就小多了。

接下來,便根據這三個標準,逐一來篩選此分類系統所需要的雙連字串。

關鍵字串的篩選方法,第一步,以出現次數為篩選的標準。

在[XCYC92]的實驗中,在做五大類的分類實驗,採用1,2,3,5,10,15,20等不同界限值,而界限值定為5時,得到較好的結果,因此以5次為篩選基準。在40085個雙連字串中,共有17825個雙連字串符合此條件而留下來。

第二步,利用Entropy公式來做篩選,以符合集中度的要求。對於一個雙連字串T_i, T_i的Entropy值為:

$$H_i = -\sum_{j=1}^{24} p_{ij} \log \frac{1}{p_{ij}}$$

其中,$p_{ij} = \frac{d_{ij}}{\sum_{j=1}^{24} d_{ij}}$, d_{ij}為類別C_j中出現T_i的文件數。

H_i的值介於0(最集中)與$\log 24$(最分散)之間,如果T_i只出現在某一類中,則H_i的值為0,若平均分散在各類,即$p_{ij} = \frac{1}{24}$, H_i的值為$\log 24$。我們定的Entropy

界限值為 $\log 2\ (= -\frac{1}{2}\log\frac{1}{2} - \frac{1}{2}\log\frac{1}{2})$,大於界限值之雙連字串則予以捨棄。$\log 2$ 的意思是:一個詞彙平均分佈在兩類中,只要比這種情形還不平均的,就會被留下來。

第三步,要訂一個公式來篩選雙連字串,以符合廣度的要求。對一個雙連字串 T_i,T_i 的廣度定為:

$$Value(T_i) = \max_j(\frac{d_{ij}}{t_{ij}} \times \frac{d_{ij}}{\sum_{j=1}^{24} d_{ij}})$$

其中,d_{ij} 為類別 C_j 出現 T_i 的文件數,t_{ij} 為 T_i 出現在類別 C_j 的次數。此界限值定為0.2 ,小於界限值0.2則予以捨棄。

經過這三個步驟的篩選,共得到了5579個雙連字串符合分類系統的要求,這5579個雙連字串,我們將之視為關鍵詞。

2.3 由雙連字串找到多字串

我們另外做了一個實驗,從雙連字串中找出多字串(N-gram)。簡而言之,這裡所找的多字串 $A_1A_2\ldots A_N$,其中 A_1A_2 及 $A_{N-1}A_N$ 必定是原先的關鍵詞,而多字串中間的雙連字串則不一定是關鍵詞。而這個N-gram,也許只是一個M字詞($M \geq N$)的子字串。尋找多字串的目的,在於刪減多餘的雙連字串。以機率模式的角度來看,在機率模式中,公式是假設各關鍵詞互相獨立(Independent)而推導出來的,假設一個詞彙ABCD被切成AB、BC、CD三個關鍵詞,則這三個關鍵詞可能是相依的(Dependent)。用多字串可使理論更周延[Yan93]。

我們根據前面找出來的雙連字串關鍵詞,逐步接成三字串(Trigram),四字串(4gram),...,一直到十四字串,把可能成為關鍵詞的N-gram都挑出來,接著再以長詞優先的原則,經過次數、集中度、廣度三條件的篩選,最後得到4711個多字串($2 \leq N \leq 14$)[YaZC93]。

五字串候選者(部分)			八字串候選者
大盤成交量	正所得稅法	股上週股價	各規格雞種每台斤
及合作金庫	交量無法放	股及金融股	例每千股無償配發
引市場實戶	交量萎縮至	股無償配發	券副總經理陳文鋒
引投資性買	各規格雞種	股價平均下	益證券公司自營部
日均線反壓	向交易所申	股價平均上	商時報股價平均下
加權指數上	在投資性買	股價平均則	商時報股價平均上
可逢低承接	自辦融資融	活期儲蓄存	商時報股價平均則
可無償配發	呈多頭排列	苯乙烯單體	資證券及其結匯辦
司六月份營	呈相對強勢	時報股價平	僑及外國人投資證
失望性賣壓	例每千股無	破十日均線	證證券公司自營部

表2-1 部分的五字串候選者及八字串候選者

結果請參看表2-2,表2-2列出14字串到雙連字串候選者數量以及最後成爲關鍵詞的數量。這樣子做,從原來5579個雙連字串關鍵詞,下降到有4711個多字串關鍵詞。

N-gram	候選者數量	關鍵詞數量	N-gram	候選者數量	關鍵詞數量
十四字串	1	1	七字串	11	6
十三字串	1	1	六字串	25	15
十二字串	2	0	五字串	50	30
十一字串	2	1	四字串	216	152
十字串	2	1	三字串	711	443
九字串	4	1	雙連字串	5579	4053
八字串	10	7			

表2-2 N-gram候選者及關鍵詞數量

2.4 類別的向量表示法

在資訊檢索過程中,一篇文件通常以此一文件所包含的關鍵詞的存在向量(Term Vector)來代表此一文件,因此代表某一類的向量,也可以用此類中所有文件的關鍵詞的存在與否來決定,唯其分量值,不應考慮以0表示不存在1表示存在此一關鍵詞。

在機率模式中,只採用原始分類比重;至於向量模式,除了原始分類比重以外,還採用兩種標準化的方式。

1.原始分類比重: 原始分類比重的給定,基本上是根據此關鍵詞在各類的分佈情形。假設關鍵詞T_i在類別C_j中的分類比重爲x_{ij},則

$$x_{ij} = \frac{\frac{t_{ij}}{len_j}}{\sum_{j=1}^{24} \frac{t_{ij}}{len_j}}$$

其中,t_{ij}爲關鍵詞T_i在類別C_j中出現的次數, len_j爲訓練資料中屬於類別C_j的新聞報導總數。

2.第一種標準化: 在原始分類比重的方式中,以向量X_j來表示類別C_j,在第一種標準化中,以向量$U_j = (u_{1j}, u_{2j}, \ldots, u_{mj})^T$來表示類別$C_j$,其中,$\|U_j\| = 1.00$,即$U_j$爲單位向量,且$u_{ij} = \dfrac{x_{ij}}{\|X_j\|}$。
此種標準化的想法是,將代表類別的向量變爲單位向量,以使每一類能夠「公平競爭」。

3.第二種標準化: 在第二種標準化中,以向量$V_j = (v_{1j}, v_{2j}, \ldots, v_{mj})^T$來表示類別$C_j$。假設$\|V_j\| = \sqrt{DN_j}$,其中,$DN_j$爲訓練資料中屬於類別$C_j$的新聞報導數。
檢視第一種標準化的方法,我們發現到有一個很嚴重的問題,亦即每一類的文件數目並不一樣,基於此,如果我們硬將代表每一類別的向量都弄成一樣長度,對文件數量多的類別並不公平。代表此類的向量長度會因文件數的增加而遞增,我們不知道確切的函數關係,但是這關係不會是線性關係,於是利用根號函數來逼近。

我們將利用這三種給定分類比重的方式來做文件自動分類實驗。

3 自動分類實驗結果

在本節中,我們將列出,分別採用向量模式和機率模式,其分類的實驗結果。

3.1 向量模式與機率模式實驗結果

類別	訓練資料召回率				測試資料召回率			
	文件數	原始分類比重	第一種標準化	第二種標準化	文件數	原始分類比重	第一種標準化	第二種標準化
J0201	56	92.45%	92.45%	92.45%	6	50.00%	66.67%	50.00%
J0202	11	100.00%	100.00%	100.00%	2	0.00%	0.00%	0.00%
J0203	64	100.00%	100.00%	100.00%	8	62.50%	87.50%	75.00%
J0204	59	96.61%	98.31%	96.61%	5	40.00%	60.00%	40.00%
J0205	40	100.00%	100.00%	100.00%	5	40.00%	40.00%	40.00%
J0206	285	93.68%	84.91%	95.44%	24	83.33%	58.33%	87.50%
J0207	699	96.71%	82.69%	95.42%	61	90.16%	70.49%	86.89%
J0208	33	100.00%	100.00%	100.00%	8	100.00%	100.00%	100.00%
J0209	24	100.00%	100.00%	100.00%	0	?.??%	?.??%	?.??%
J0211	116	87.83%	87.83%	89.57%	14	21.43%	28.57%	21.43%
J0301	112	84.91%	82.08%	93.40%	6	16.67%	16.67%	16.67%
J0302	32	93.10%	96.55%	93.10%	2	0.00%	50.00%	0.00%
J0303	211	94.29%	94.76%	94.76%	14	71.43%	85.71%	71.43%
J0305	36	97.06%	97.06%	94.12%	25	60.00%	64.00%	76.00%
J1008	40	100.00%	94.74%	100.00%	6	33.33%	33.33%	33.33%
J1009	24	90.00%	90.00%	85.00%	0	?.??%	?.??%	?.??%
J1012	44	100.00%	97.73%	97.73%	5	80.00%	80.00%	80.00%
J1103	20	90.00%	95.00%	95.00%	1	0.00%	0.00%	0.00%
J1105	12	100.00%	100.00%	100.00%	0	?.??%	?.??%	?.??%
J1201	30	93.33%	96.67%	93.33%	5	60.00%	60.00%	60.00%
J1202	58	85.96%	84.21%	87.72%	4	25.00%	75.00%	50.00%
J1203	29	91.67%	91.67%	91.67%	4	0.00%	0.00%	33.33%
J1204	29	88.46%	88.46%	88.46%	4	0.00%	0.00%	0.00%
J1205	31	96.55%	96.55%	93.10%	2	50.00%	50.00%	50.00%
總計	2095	94.57%	88.50%	94.86%	211	64.29%	60.95%	67.14%

表3-1 5579個雙連字串在向量模式中各種方法的分類實驗結果

	訓練資料召回率			測試資料召回率		
	原始分類比重	第一種標準化	第二種標準化	原始分類比重	第一種標準化	第二種標準化
5579個雙連字串	94.57%	88.50%	94.86%	64.29%	60.95%	67.14%
4711個N-gram	94.53%	86.73%	94.58%	59.90%	56.04%	61.35%

表3-2 5579個雙連字串與4711個N-gram在向量模式中實驗結果的比較

	訓練資料召回率			測試資料召回率		
	第一名	前二名	前三名	第一名	前二名	前三名
向量模式	94.86%	99.56%	99.95%	67.14%	76.67%	82.86%
機率模式	97.23%	99.90%	99.95%	59.52%	74.76%	79.52%

表3-3 向量模式與機率模式實驗結果的比較

請參看表3-1 、表3-2及表3-3 。簡單說,根據實驗的結果,我們可以發現,利用第二種標準化方式,測試資料的召回率可以達到67.14%,如果取前三名,則更可以達到82.86%;至於採用4711個N-gram關鍵詞,則沒有得到較好的結果,無論在向量模式或是機率模式的情形下;此外,就測試資料而言,我們利用機率模式所得的結果比向量模式稍差一些。

3.2 其它相關的實驗結果

1. 用每篇新聞報導的前面幾行當做文件替代品

通常一篇報導,重要訊息應該會傾向集中在報導的前面幾行;事實上,人工在做分類的工作時,通常也都是只看前面數行就決定了此篇新聞報導的類別。如果是這樣,在做分類時,是不是可以只取每篇新聞報導的前面數行當做文件替代品(Profile),在不影響分類正確率的前題下,減少電腦記憶儲存的負荷及計算時間?

現在,我們做一個實驗,只取前20行、前15行以及前10行來做看看,比較正確率有甚麼改變,請參看表3-4 。

所取用的行數	關鍵詞數量	訓練資料召回率			測試資料召回率		
		第一名	前二名	前三名	第一名	前二名	前三名
整篇報導	5579	94.86%	99.56%	99.95%	67.14%	76.67%	82.86%
前20行	4008	92.46%	98.49%	99.32%	62.65%	72.37%	77.82%
前15行	3451	85.34%	93.90%	95.57%	62.55%	76.83%	80.69%
前10行	2445	82.72%	91.81%	94.13%	60.54%	73.56%	79.69%

表3-4 只取前幾行來做分類所得的實驗結果(向量模式、第二種標準化)

實驗結果,就測試資料而言,整篇報導與前20行比較起來,還有將近五個百分點的差距。或許這樣的結果表現出了一個事實,即我們收集到的這些新聞報導,其訊息可能平均地分佈在報導中,並沒有特別集中在前幾行。

2. 減少訓練資料數量所得實驗結果

我們認為訓練資料嚴重不足，這部分的實驗，就是想要試著來估計大概要多少篇新聞報導才達到飽和；如果沒有辦法估計出來，也想試著以具體的數據來說明目前的訓練資料確實不足。

目前所用的訓練資料，時間是從80年7月到80年12月；現在，先扣除7月份的訓練資料然後來做實驗，再扣除8月份的訓練資料來做實驗，一直扣除到只剩下12月一個月份的訓練資料來做實驗，讓我們來比較其正確率的變化，至於測試資料則同樣是用81年1月。請參看表3-5的實驗結果，同樣是採用向量模式、第二種標準化的方法。

所用訓練	關鍵詞	訓練資料召回率			測試資料召回率		
資料月份	數量	第一名	前二名	前三名	第一名	前二名	前三名
7～12月	5579	94.86%	99.56%	99.95%	67.14%	76.67%	82.86%
8～12月	5085	94.67%	99.41%	99.85%	61.98%	74.14%	79.09%
9～12月	4344	96.09%	99.74%	99.87%	61.69%	77.39%	82.76%
10～12月	3379	97.09%	99.90%	100.00%	59.77%	71.26%	78.93%
11～12月	2379	98.16%	99.54%	100.00%	53.82%	67.18%	74.81%
12月	1297	99.62%	100.0%	100.00%	53.64%	67.43%	71.65%

表3-5 減少訓練資料數量所得的實驗結果（向量模式、第二種標準化）

就訓練資料而言，當訓練資料量愈少，正確率（召回率）會愈高是很合理的，因為資料量愈少，一個關鍵詞跨多類的機會相對減少。做此實驗，是希望能找到一個點，到了這個點以後，就算再增加訓練資料的數量，也不會提高測試資料的正確率，顯然我們並沒有找到這個點，所以訓練資料應該確實是不足夠的。

3. 將出現關鍵詞的文件數定界限值

在選取關鍵詞時，一共定了三個標準，分別是詞彙次數要夠、分佈特別集中於某些類以及在同一類中要儘量分散在各篇報導中。其中最後一項定0.2為界限值，雖然有因此而去除掉一些詞彙，但是可能標準定得太鬆了一些，導致實際上有許多出現5次且集中在一篇新聞報導的詞彙、剛好在界限值的邊緣而被「抓」進來成為關鍵詞。

不管界限值0.2是不是定得太鬆，其實還可以用此詞彙出現的文件數來定界限值，如果定了界限值，又能不影響到分類結果的正確率，則可以減少關鍵詞的數量。

我們將文件數的界限值分別定爲1(就是沒有設限)、2、3及5,實驗結果請參看表3-6。

文件數	關鍵詞	訓練資料召回率			測試資料召回率		
界限值	數量	第一名	前二名	前三名	第一名	前二名	前三名
1	5579	94.86%	99.56%	99.95%	67.14%	76.67%	82.86%
2	5432	94.47%	99.47%	99.90%	66.67%	77.14%	83.33%
3	4843	93.53%	99.17%	99.85%	64.59%	75.12%	82.30%
5	3375	91.06%	98.62%	99.44%	62.44%	73.17%	78.05%

表3-6 定文件數爲界限值所得的實驗結果(向量模式、第二種標準化)

從上面的實驗結果告訴我們,當在挑選關鍵詞,其實我們還可以定文件數的界限值,出現在兩篇或兩篇以上的新聞報導中才要,只出現在一篇新聞報導中的可以捨棄,正確率幾乎沒有甚麼差別。

4 錯誤分析

實際來檢視電腦分錯的例子,大致上,我們可以歸納爲兩點:(1)類別相近或是報導本身模稜兩可;(2)此篇報導的關鍵詞數量很少。

關於第一點,若我們直接看類別名稱,就可以發現一些較相近的類別,例如J0206金融、J0207證券;例如J1103電子材料、J1203硬體···等等。

現在我們將類別C_j以向量$X_j = (x_{1j}, x_{2j}, \ldots, x_{mj})^T$表示,如果關鍵詞$T_i$在類別$C_j$的分類比重不是0,則$x_{ij}$=1,否則$x_{ij}$=0,類別$C_i$與$C_j$的相似性,我們用$X_i$與$X_j$的餘絃(Cosine)來表示,即$\dfrac{X_i \cdot X_j}{\|X_i\|\,\|X_j\|}$。我們將相似性超過15.0%的類別列出:

J0207(證券)	J0303(業績財務)	0.39
J0206(金融)	J0207(證券)	0.36
J0204(貨幣流通)	J0205(物價)	0.20
J1202(電腦)	J1203(硬體)	0.19
J0207(證券)	J0209(商品行情)	0.18
J1103(電子材料)	J1203(硬體)	0.16
J0207(證券)	J0301(經營總論)	0.15

對於某些類別相近這樣的問題,若要增進自動分類的正確率,大致有幾種方法:

1. 重新調整類別，將較相似的類別合併。

2. 採用重覆分類，如果一篇新聞報導，經過電腦計算後，有兩類的分數都很高，則電腦將此篇報導同時指定在這兩類。當然，如果這樣的話，人工在指定類別時也要考慮將某些報導同時分爲兩類甚至三類。

另外，讓我們來考慮，電腦與人工在做分類及挑選關鍵詞時，兩者之間的差異。挑選關鍵詞時，人所挑出的關鍵詞，大致上具有分類的意義，而電腦所挑出的關鍵詞，大致上具有分類的價值，但是不一定都具有意義（就人的觀點而言）。一個有分類意義的詞彙，可能因爲次數太少而使其分類的價值減低，也可能因爲出現在太多不同類別，而失去分類的價值。反過來說，有分類價值的詞彙，都特別集中在某一類或某幾類的詞彙，在本實驗中，因爲是用雙連字串，找出來的雙連字串，可能是詞彙的一部分，但也可能因爲包含功能詞(Function Word)或是附著語(Bound Word)而看起來不具甚麼意義。若要改進這些詞彙的「品質」，利用斷詞系統以及詞性標示等技巧來過濾掉這些「不好」的雙連字串是可能的解決方式。

至於分類的做法方面，電腦在做分類時，是以關鍵詞當做線索，而人在做分類時，可能是以某一句話或是某幾句話當做線索，這樣說來，電腦做分類的結果，可能沒有辦法達到和人一模一樣，因爲重點的這幾句話如果沒有出現關鍵詞，電腦就可能會分錯。基本上，人在做分類時，利用的是「概念」，而比較不是因爲看到了某些關鍵詞而決定要將之分到哪一類。在大部分的情形下，這段概念包含一些關鍵詞，但是有些情形下，概念中沒有任何的關鍵詞。另外一點，一篇新聞報導可能會提到很多東西，人很容易看出甚麼是主題甚麼是次要的部分，但是我們顯然還沒有賦予電腦這個能力。通常，主題會寫在一篇報導較前面的部分，然而也可能整篇都在談主題。

當然，電腦分類的最大好處是每次做的結果都會一樣，有一致性，人工做分類的話，不同的人會做出不同的結果，甚至同一人在不同時間做也可能會有不同的結果。

5 結論及未來方向

針對這些新聞報導，實驗結果顯示，在我們的做法中，向量模式比機率模式的結果好一些，第二種標準化（代表此類別的向量長度正比於此類別訓練資料數量開根號）方式比原始分類比重結果好一些。

另外，利用5579個雙連字串關鍵詞，會比採用4471個N-gram關鍵詞所得的結果還要好些；但是就人的觀點而言，我們會覺得4711個N-gram關鍵詞比較有意義，且就理論層面而言也較健全些。

就測試資料而言，召回率大約在67%，平均三題對兩題，結果並不盡理想，究其主要原因，應是受限於訓練資料的不足，另外，有些問題出在人所指定的類別本身，以及有些報導本身確有模稜兩可的情形發生。 67%的召回率（正確率），尚不足以完全取代人工，然而電腦分出來的結果有絕對的一致性，不像人工分類的結果常因人因時而會有所差異。 若取分數最高的前三名，召回率有83%左右。如果電腦先找出前三名，交給人來看，再由人來做分類的工作，這樣的話，應該多少還是能夠減輕人工的負擔。

未來，我們希望能夠再做下列的工作：

1. 增加訓練資料數量：在前面曾經提到過，有些錯誤發生的原因應該歸咎於訓練資料數量的不足，實際上，有些類別只有十幾篇，要用這十幾篇新聞報導來代表這一類別，並不是很妥當。我們希望再繼續做些取樣，訓練資料增加，應該可以得到更好的結果。

2. 加入自然語言的知識：基本上，到現在為止我們的做法，主要是用統計的方法，並沒有善用電腦在自然語言處理上所獲致的成果，例如斷詞、自動詞性標示(Automatic Tagging)，例如Mutual Information ···等等。
為了提高關鍵詞的品質，加入斷詞應該是有必要的，但是得克服諸如專有名詞、縮寫詞···等等的問題。另外，規則運算式似乎也可以考慮加入，例如「昨日升值」、「前日升值」兩個詞彙，就分類的意義應該是相同的，那麼，可以考慮以「？日升值」的方式來儲存此詞彙。

3. 利用類神經網路的技巧來給定分類比重：我們想引進類神經網路的技巧，經由訓練來決定分類比重，看看能不能得到不同的結果。

4. 重覆分類：現實生活中，很容易發現一些東西難以歸類，本實驗中，有些新聞報導確實也有類似的情形，因此重覆分類可能較符合實際需要。

5. 類別選定：我們的目的，在於利用電腦來取代人工，以節省人力。若要取代人工，則得先設法使電腦分類結果與人工分類結果的一致性提高，但是經由分析，有些錯誤應該是出在類別選定的問題，如果根據電腦分類的結果，再回頭檢視人所規定的類別，做個折衝，在人覺得有意義的條件下，重新調整類別，召回率提高，才更有可能達到目標。

誌　　謝

本論文的完成，要感謝中國時報社所提供的工商時報新聞報導資料；另外，中央研究院資訊所助研究員簡立峰博士以及清華大學資訊所張俊盛教授提供了不少寶貴的意見，在取一併誌謝。

附　　錄

五大類及24小類類別名稱：

類別			訓練資料篇數
J02	經濟、產業		
	J0201	產業總論	56
	J0202	財政	11
	J0203	稅制	64
	J0204	貨幣流通	59
	J0205	物價	40
	J0206	金融	285
	J0207	證券	699
	J0208	保險	33
	J0209	商品行情	24
	J0211	業者動態	116
J03	經營、企業		
	J0301	經營總論	112
	J0302	企業組織	32
	J0303	業績財務	211
	J0305	商品流通	36
J10	機械、器具、設備		
	J1008	產業機械、半導體	40
	J1009	事務機械	24
	J1012	四輪配備	44
J11	電子、電機		
	J1103	電子材料	20
	J1105	家庭電器	12
J12	情報、通信		
	J1201	資訊總論	30
	J1202	電腦	58
	J1203	硬體	29

J1204	系統軟體	29
J1205	通訊設備	31

參考文獻

[Che92] 陳淑美.財經新聞自動分類之研究.台大圖書館學研究所碩士論文,台北.1992年12月.

[XCYC92] 謝清俊,陳淑美,楊允言,陳克健.Autoclassification of Texts.如何利用大型語料庫作研究研討會.計算語言學會,台北.1992年9月.

[Yan93] 楊允言,張俊盛,陳克健.文件自動分類及其相似性排序.清華大學資訊科學研究所碩士論文,新竹.1993年6月.

[BHMP92] Blosseville M.J., Hebrail G., Monteil M.G. and Penot N. " Automatic Document Classification : Natural Language Processing, Statistical Analysis, and Expert System Techniques Used Together ". in SIGIR '92 : Proc. of the 15th Ann. International ACM *SIGIR Conf. on R. and D. in Inform. Retr.*. pp51-57. Denmark : Copenhagen. Jun.21-24 1992.

[BoBe63] Borko, Harold and Bernick, Myrna. " Automatic Document Classification ". *J. of the ACM* 10(1) pp151-162. 1963.

[HaZa80] Hamill, Karen A. and Zamora Antonio. " The Use of Titles for Automatic Document Classification ". *JASIS* 31(6) pp396-402. Nov.1980.

[Jac92] Jacobs, Paul S. " Joining Statistics with NLP for Text Categorization ". in *Third Conference on Applied Natural Language Processing — Association for Computational Linguistics* pp178-185. Italy : Trento. 31Mar.-3Apr. 1992.

[Jac93] Jacobs, Paul S. " Using Statistical Methods to Improve Knowledge-Based News Categorization ". *IEEE Expert* 8(2) pp13-23. Apr. 1993.

[Kwo75] Kwok, K.L. "The Use of Title and Cited Titles as Document Representation for Automatic Classification ". *Inform. Proc. and Manag.* 11 pp201-206. 1975.

[Lar92] Larson, Ray R. " Experiments in Automatic Library of Congress Classification ". *JASIS* 43(2) pp130-148. 1992.

[Lew92] Lewis, David D. " An Evaluation of Phrasal and Clustered Representations on a Text Categorization Task ". in *SIGIR '92 : Proc. of the 15th Ann. International ACM SIGIR Conf. on R. and D. in Inform. Retr.*. pp37-50. Denmark : Copenhagen. Jun.21-24 1992.

[Mar61] Maron, M.E. " Automatic Indexing : an Experimental Inquiry ". *J. of the ACM* 8 pp404-417. 1961.

[Sal86] Salton, Gerard. " Another Look at Automatic Text-Retrieval Systems". *Comm. of the ACM* 29(7) pp648-652. Jul.1986.

[Smi89] Smith, Stephen Ray. *An Advanced Full-Text Information Retrieval System.* PhD thesis, Computer Science Dept.; Univ. of Alabama in Huntsville. 1989.