An Analytical Study of Transformational Tagging for Chinese Text

Helen Meng* and Chun Wah Ip
Human-Computer Communications Laboratory
Department of Systems Engineering and Engineering Management
The Chinese University of Hong Kong
Email:{hmmeng, cwip}@se.cuhk.edu.hk
Fax: (852)2603-5505
*Please send all related correspondences to this author

6 Aug 1999

Topic Areas: (o), (s)

Abstract

This work is our initial attempt in using the transformation-based error-driven learning (TEL) procedure for tagging Chinese text. TEL has previously been shown to be effective in POS tagging for English [Brill 1995]. TEL provides several attractions: (i) automation for tagging, (ii) induction of interpretable rules, (iii) learning aimed at error-reduction. Our experimental corpus consist of over 70,000 words of Chinese text, divided into disjoint training and test sets of a 9:1 ratio. With an unknown word/tag proportion of 13%, we achieved overall tagging accuracies of 94.56% (training) and 86.87% (testing).

1. Introduction

Part of speech tagging is an important linguistic problem which has garnered much research interest and effort over the years. Automatic part of speech (POS) taggers are particularly attractive for providing syntactic information applicable to speech recognition and understanding, information retrieval, machine translation and other applications. A myriad of techniques have previously been used for automatic POS tagging, ranging from rule-based to data-driven approaches. The former tends to be hand-annotated by linguistic experts, while the latter includes stochastic n-grams,

HMMs, neural networks, trigger-pair predictions, genetic algorithms, etc. [Bai et al., 1992][Kupiec 1992][Lua 1996][Black 1998].[1] Rule-based approaches are linguistically well-motivated, but expert handcrafting is often an expensive and tedious process. Data-driven approaches attempts to ameliorate the tedium by capturing relevant linguistic constraints from a corpus of annotated data. However, the linguistic constraints captured are encoded in a large body probabilities and statistics, which do not lend themselves well for exploratory linguistic analysis.

Brill [Brill 1995] had previously proposed an alternative technique of transformation-based error-driven learning for automatic POS tagging in English. This approach combines the merits of rule-based and data-driven techniques in an elegant manner. The algorithm may be initialized randomly or with some linguistically-motivated specifications. Machine learning then proceeds with an annotated corpus, and with the objective of maximizing tagging accuracy. Such learning produces a compact rule set, which encodes the contextual and lexical constraints for tagging, and are easily interpretable by humans for studying the linguistic cues for POS tagging.

This work explores the use of transformation-based error-driven learning (TEL) for POS tagging (or transformational tagging) of Chinese text. The Chinese language presents a unique set of characteristics for the tagging algorithm, which include:

(i) The ideographic (character-based) nature of Chinese, in contrast to the alphabetic nature of English. Chinese text consists of strings of characters separated by punctuation marks. A Chinese word may consists of a single character, or multiple characters with no delimiters between words. Hence, Chinese text need to be *segmented* to form sequences of words. For a given string of characters, there may exist multiple legitimate segmentations. Different segmentations lead to different word sequences and hence different sequences of POS tags. In this work, our task is simplified by using a pre-segmented corpus.

[1] Informative citiations are many, those included here are by no means exhaustive.

(ii) Aside from the ambiguity caused by multiple segmentations, a given word may have multiple possible POS assignments. For example, 白/a 馬/ng and 馬/nf 步芳/npf[2], where 馬 is a common noun in the former and a last name of person in the latter.

(iii) The lexical structure of the Chinese word is very different compared to English. Inflectional forms are minimal, while morphology and word derivations abide to a different set of rules. A word may inherit the syntax and semantics of (some of) its compositional characters, for example, 紅 means *red* (a noun or an adjective), 色 means *color* (a noun), and 紅色 together means *the color red* (a noun) or simply *red* (an adjective). Alternatively, a word may take on totally different characteristics of its own, e.g.東 means *east* (a noun or an adjective), 西 means *west* (a noun or an adjective), and 東西 together means *thing* (a noun). Yet another case is where the compositional characters of a word do not form independent lexical entries in isolation, e.g. the characters in 彷彿 (a verb) do not occur individually.

This work examines the utility of transformational tagging for Chinese text. We are especially interested in the linguistic rules induced automatically by TEL for individual Chinese words, as well as across a sequence of multiple words. Chinese linguistic structures may be observed in such rules, including grammar, morphology and word derivations. TEL is applicable not only to in-vocabulary words, it is also designed to handle the occurrences of unknown words in corpora.

2. Corpus and Tags

This work is based on the pre-segmented and hand-tagged corpus from Tsinghua University [Bai et al., 1992]. This news corpus is derived from the People's Daily (Renmin Ribao) in the year 1993. Altogether there are 112 articles and 71,804 words of running text, distributed across five domains: computer, military, science, technology and general news. Unique vocabulary entries exceed 9,000. Information about the entire corpus is tabulated in Table 1, and the word count in the table refers to

[2] These are word/tag pairs extracted from our corpora

the length of running text. Table 2 displays some example sentences from each domain, which shows the word/tag pairs for each sentence. In this work, we only tackle the tagging problem – our tagger learns from pre-segmented and tagged training sets, and tests on a pre-segmented test sets.

Domain	No. of Articles	# of Words (train)	# of Words (test)
Computing	10	5,479	509
Military	23	12,243	1,787
Science	20	12,922	1,391
Technology	20	11,383	1,228
News	39	22,358	2,505

Table 1: Distribution of Training and Testing Sets from the Tsinghua news corpus.

Domain	Example sentences
Computing	我們/rn 根據/p 這/rn 一/mx 漢化/vg 策略/ng 對/p DECnet-DOS/xch 進行/vgv 了/utl 分析/ vgo 。/。
Military	反/vgn 機降/ng 成爲/vgn 戰鬥/vgo 的/usde 重要/a 內容/ng 。/。
Science	17世紀/t 英國/s 的/usde 醫學家/ng 哈維/npf ,/,
Technology	工作/ng 模式/ng 是/vy 當前/t 科技/ng 情報/ng 體制/ng 改革/nvg 中/f 的/usde 又/d 一/mx 熱點/ng 。/。
News	共同體/ng 將/va 參與/vgn 德國/s 統一/vg 問題/ng 的/usde 討論/nvg ,/,

Table 2. Example sentences from our corpus.

The original tag set found in the Tsinghua corpus consists of 108 unique labels. These were exhaustively enumerated in [Lua 1996]. Out of this set, 25 are for punctuation, and the remaining ones draw fine distinctions for Chinese parts of speech. As an example, nouns are divided into 5 types: **nf** (last name), **npf** (name of person), **npu** (name of organization), **npr** (other proper nouns) and **ng** (common noun). We added an extra tag, **nvg**, to represent words which can either be a

noun or a verb, such as 運動 (exercise) or 表示 (express/expression). The reason is as follows: in the original tagged corpus, there are words like 運動 which are tagged as general verbs, e.g.

雖然/cf 經歷/vgn 了/utl 各/rn 種/qnk 運動/vg 變化/vg ，/，

where the tags are: cf (連詞前段), vgn (帶體賓動詞), utl (連詞 "了"), rn (體詞性代詞), qnk (種類量詞) and vg (一般動詞).

In this context, however, 運動 seems to play a role more similar to a noun, which motivated the design of the **nvg**[3] because 運動 in this case is not suitable to tag as word.

One may wonder whether the full tag set is necessary for Chinese POS tagging.[4] A preliminary investigation of our entire corpus reveals that approximately 100 tags occurred, with the most frequent one being **ng** (common noun), which occurred about 25.5% of the time. The most frequent 18 tags (which include a few punctuation tags) covers 80% of our running text corpus, while the most frequent 32 tags already covers 90%. Nevertheless, we proceeded with the full set of 109.

The ambiguities found in the Tsinghua corpus is 1.88 tags per word. (Please see Figure 1) Over 40% of the vocabulary can be tagged multiple ways. Out of this, the maximum number of tags per a word is 8. Table 3 lists the 8 POS tags of the word (表示) and their contexts.

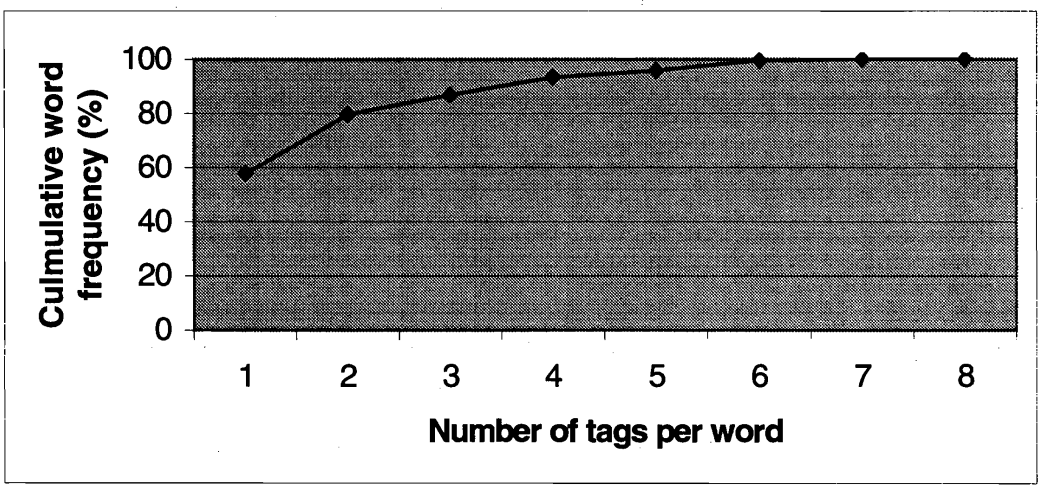

Figure 1: Cumulative distribution of words with single to multiple POS tags

[3] The idea of using nvg tags is attributed to Dr. Wenjie Li.
[4] We have found tag sets of approximately 50 entries of fewer in other literature.

Tag	Example Sentences
vg (一般動詞)	這/rn 種/qnk 表示/vg 法/ng 與/p ＪＡＥ/xch 類似/a 。/。
vgo (不帶賓動詞)	文獻/ng 和/cpw 查詢/nvg 都/d 用/vgn 一/mx 組/qnc 正交基/ng 詞/ng 向量/ng 表示/vgo ，/，
vgn (帶體賓動詞)	我們/rn 采用/vgn Ｓｅｔ０/xch 表示/vgn 單/b 字節/ng 的/usde ＡＳＣＩＩ/xch 字符/ng ，/，
vgv (帶動賓動詞)	ＤＧ/xch 人士/ng 表示/vgv 將/d 爲/p 此/rn 繼續/vgv 做/vgv 出/vc 努力/vgo 。/。
vga (帶形賓動詞)	“/“ 是/vy ”/” 可以/va 表示/vga 一樣/a 。/。
vgs (帶小句賓動詞)	無非/d 表示/vgs ：/：
ng (普通名詞)	情報/ng 檢索/vg 系統/ng 所/ng 儲存/ng 的/usde 是/vy 文獻/ng 的/usde 某/rn 種/qnk 表示/ng ，/，
nvg (動名詞)	一/mx 篇/qni 文獻/ng 的/usde 表示/nvg 中/f 所/ussu 使用/vgn 的/usde 標引詞/ng 的/usde 個數/ng …

Table 3. Example sentences of the word “表示” from our corpus.

3. Transformational Tagging

The algorithm is presented in detail in [Brill 1995]. The tagger addresses its problem at both the *lexical* and *contextual* levels. Here we will provide a procedural sketch.

3.1 Notations

For the sake of simplicity, we will adopt the following notations in describing our work:

- C^{d}_{type} , denotes a corpus *C* belonging to a specific domain *d*, and of a particular *type* - *training, testing, lexical* or *contextual*. The type is related to the transformational tagging procedure, and will be explained later.
- $T_i(C^{d}_{type})$, denotes a *tagged* corpus *C* . The variable *i* may adopt the instances *ref* (for the set of reference tags), *start* (for the tags resulting from the initialization of the tagger) or *final* (for the tags resulting from the final stage of the tagger, having applied all tagging rules). Details will be explained later. An example of a tagged sentence is:

17世紀/t 英國/s 的/usde 醫學家/ng 哈維/npf ,/,

- $U(C^d_{type})$, denotes a *untagged* corpus *C*. A procedure may be applied to strip off all the tags, resulting in 17世紀 英國 的 醫生家 哈維 ，

  from the previous example.

- R^d_{type} , denotes a set of rules *R*. Rules may be of the type *lex* (lexical rules) or *context* (contextual rules). Example rules include:[5]

  Lexical rule : 了goodleft vgn 135.820116353036
  Contextual rule : vgn vgo NEXT1OR2TAG STAART

  The associated explanation is in the following section.

- L_{type} , denotes a lexicon, which may be of type *lex* (the lexicon for training lexical rules only), or *all* (the lexicon containing all words in the training corpus).

3.2 Corpus Utilization

Figure 2 shows how the corpus is utilized. The entire corpus is first divided into a training set (90% of the size) and a test set (10%). The training set is in turn divided into two halves. One half is used to train *lexical* rules -- these are rules applied in order to predict the tag of a word based on the intra-word characteristics. The other half is used to train *contextual* rules -- these are rules applied to tag a word based on its neighboring word contexts.

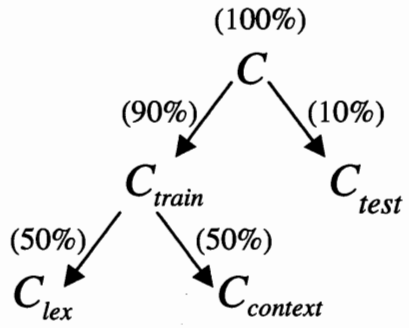

Figure 2. Corpus utilization in a particular domain.

[5] The associated explanation is in the following section.

3.3 Transformational-based Error Driven Learning

Learning takes place in two phases. Lexical rules are learnt first, and are used during the subsequent learning of contextual rules.

3.3.1 Lexical rules

These are used to tag unknown words. Learning lexical rules requires three word lists:

(i) A list of all the words occurring in the untagged training corpus $U(C_{train})$, sorted by decreasing frequency of occurrences. The word list is used to find the most common prefixes and suffixes.

(ii) A list of triplets [word tag count] derived from $T_{ref}(C_{lex})$, e.g.

是 vy 365

和 cpw 358

在 pzai 339

The words with more than one tags will get different entries in the list. Besides the triplets [和 cpw 358], the list also contains three more triplets, [和 p 13], [和 cpc 1] and [和 cpw 1] . The count of the triplet is the frequency of the word tag pair in the tagged training corpus. The tagged words are used to calculate the weights of possible tags for a given word.

(iii) A list of word bigrams found in the untagged training corpus, $U(C_{train})$, e.g.

是 利用

都 採用

心 還

The bigrams list is used to calculate the weight of the tags to the preceding/following word .

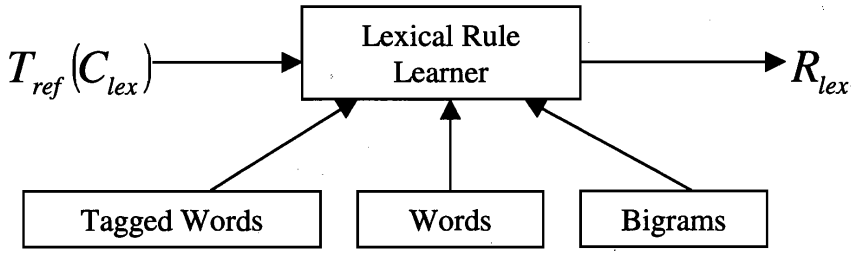

Figure 3. Learning Lexical Rules.

The learning process begins by giving the unknown word an initial tag. Such initialization can be done in a number of ways: The unknown word may be assigned **unk**, to denote its out-of-vocabulary nature. Alternatively, since unknown words are often common nouns, we may assign them with the tag **ng** upon initialization. In addition, we may utilization simple prior knowledge, e.g. assign **xch** (tag for non-chinese word) if English letters are encounted, or **mx** (tag for numbers used in measurements).

Lexical rules are learnt according to some prescribed templates, so that they can utilize prefixes, suffixes, constitutent characters and bigram relationships to infer an appropriate tag for an unknown word. Some example templates include:

- **{x w fgoodright/fgoodleft y n}**, i.e. given the word in focus **wc** currently tagged as **x**, should the word **w** occur to its right/left, change its tag from **x** to **y**. A close variant of this template is {w goodright/goodleft y n}, which does not constrain the current tag of the word in focus. **n** reflects the relative frequency of rule application in the training set. Here is the equation for calculating n.

$$n = \sum_{j=1}^{W} N\{word_j, tag_k\} - N\{word_j, tag_i\}$$

where ***W*** is the number of words in the training set, ***tag$_k$*** is target tag to be changed, ***tag$_i$*** is current tag.

$$N\{word_j, tag_k\} = \frac{word_j, tag_k}{\sum_{i=k}^{T} word_j, tag_i}$$

where ***word$_j$*** is a word in the training set, ***tag$_k$*** is a tag for the ***word$_j$***, ***T*** is the number of tags for the ***word$_j$***, ***word$_j$ tag$_k$*** is the number of frequency for the pair ***word$_j$ tag$_k$*** in the training set.

Example of rule application:

Rule: { ng 李 fgoodright npf 11}

Sentence: 年/ng 過/vgn 半百/mx 的/usde 煉鐵廠/ng 老/a 工人/ng 李/nf 傳杰/npf

Here 傳杰 is a unknown word, and the tagger assigns it with **ng** upon initialization. However, seeing the last name 李 towards its left (i.e. 李 is to the *right* of our current word) invokes the specified rule. 傳杰 is then correctly transformed as a **npf** (name of a person).

- **{x z fchar y n},** i.e. given the word in focus **wc** currently tagged as **x**, should the character **z** occur in the word, change its tag from x to **y**. A close variant is {z char y n} which does not constrain the current tag of the word in focus. Example of rule application:

  Rule: mx 年 fchar t 46

  Sentence: 1957年/t 7月/t 到/p 1958年/t 12月/t

  The unknown word 1957年will be tagged as **mx** (number for measurements) upon initialization. This invokes the specified rule to change to the correct tag **t** (tag for time).

- **{x a fhassuf/fhaspref p y n},** i.e. given the word in focus **wc** currently tagged as **x**, should it contain the **p** characters in its prefix or suffix **a**, change its tag from **x** to **y**.

A close variant of this template is {a hassuf/haspref p y n}. Example of rule application:

Rule: 委員會 hassuf 3 npu 5

Sentence: 聯合國常規軍備委員會/npu 曾/d 通過/vgn 決議/ng ,/,

The unknown word 聯合國常規軍備委員會 will be initialized as **ng**. Owing to the occurrence of suffix 委員會 its tag will be changed to **npu** (name of organization). Therefore it can be seen that the lexical rules automatically learnt during this stage offers insight as to the lexical nature of the words, interpreted with the use of prefixes, suffixes, constitutent characters as well as bigram information.

3.3.2 Contextual rules

The use of lexicons and lexical rules ensure that each and every word in the text is initialized with a tag. Contextual rules need to be learnt in order to correct any possible errors in the initialization. Hence these rules should be effective in disambiguating among the multiple tag assignments for a given word, using across-word contextual information. The learning process for contextual rules is depicted in Figure 4.

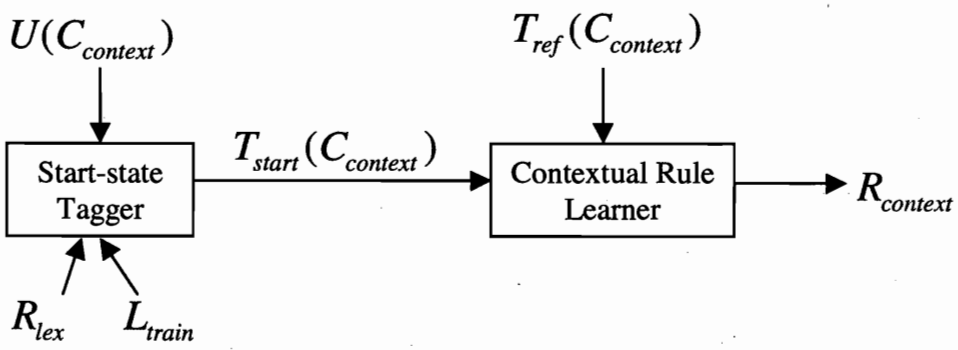

Figure 4. Flow chart showing the process of learning contextual rules

The untagged corpus for learning contextual rules is first processed by the start-state tagger. This tagger references the training lexicon, L_{train}, to assign the most frequent tag to each of the words. Unknown words are tagged by applying the lexical rules. These

procedures produce a set of start-state tags T_{start} $(C_{context})$ for the corpus. These are then compared with the reference tags, T_{ref} $(C_{context})$, in order to proceed with error-driven learning, which finally produces the set of contextual rules $R_{context}$. Error-driven learning of the contextual rules also follow a set of templates, which considers the across-word context in a seven-word window - between one to three words/tags to the left and right of the current word (word in focus). Examples of the templates include:

- **{x y next1or2tag staart},** i.e. given that the current word **wc** is tagged as **x**, change the tag to **y** if the following one or two tags is the start/end of sentence symbol (**staart**).

  Example of rule application:

  Rule: usde y next1or2tag staart

  Sentence: 全/a 過程/ng 中/f 是/vy 可/va 變/vgo 的/y 。/。

  的 is most commonly tagged as **usde**, and is initialized by the start-state tagger thusly. Application of our rule corrects the assignment from usde to y (語氣詞).

- **{x y prevwd w},** i.e. given the current word wc is tagged as **x**, change the tag to **y** if the previous word is **w**. Example application:

  Rule: vv f prevwd 年

  Sentence: 爲/vi 過去/t 15/mx 年/ng 來/f 的/usde 最/d 大/a 跌/vg 幅/ng 。/。

  The most frequent tag of 來 is **vv**, which becomes the initial assignment of the start-state tagger. However, the application of the rule corrects it to **f** (方位詞).

  During the learning process, the start-state tags are compared with the reference tags for each sentence in $C_{context}$. Rules for error correction are proposed according to the templates. The proposed rule which maximally reduces the number of errors is adopted in the *ordered* transformational rule set. The adopted transformation is then applied to the entire training corpus, from left to right, and the transformation is

invoked only after all matching contexts in the training set are identified. This constitutes one iteration in learning. Iteration continues until no proposed rules can reduce the minimum count of tagging errors. This minimum count threshold is therefore an experimental parameter.

The difference between the templates of lexical rules and contextual rules is that lexical rules only consider the lexical information of the words (such as prefix, suffix and characters in the word) and neighbouring words. For contextual rules, the considerations are contextual information (such as the previous/following tag of current word), lexical information (such as the previous/following words of current word) and combination of lexical and contextual information (such as the previous/following word and previous/following tag together).

4. Experiments

Our experiments are based on disjoint training and test sets, with a 9:1 divide. Each corpus domain is processed individually. We have also combined all the articles for all domains to form a large corpus (71,804 words). This is also divided into training and test sets of the same proportion, and used for experimentation. Figure 5 displays a couple of example sentences.

UNIX/xch	Pacific/xch	公司/ng	與/p	AT&T/xch	是/vy	什麼/rn	關係/ng ?
(UNIX)	(Pacific)	(company)	(and)	(AT & T)	(is)	(what)	(relationship)
它/rn	主要/d	是/vy	幹/vgn	什麼/rn	的/usde ?		
(It)	(mainly)	(is)	(doing)	(what)			

Figure 5. Examples from the training set, with both segmentation and tagging included. We also include a pseudo English translation in parentheses.

Since the training and test sets are disjoint, we see the occurrences of both *unknown words* as well as *unknown tags* in the test set. An "unknown tag" refers to the tagging of a (known) word in the test set, but the word/tag combination never appeared in the training set. For example, the single-character word 幹 was only seen with the tag **vgn** in the training set. However, it occurred in the test set with the tag **vgv**. Our tagger is bound to make mistakes with cases of unknown tags. The

proportion of unknown words and unknown tags range from 8.95% to 33.20% across our domains. Details are shown in Table 4.

Domain / Proportion(%)	Computing	Military	Science	Technology	News	Total
Unknown Words	29.08	13.26	22.14	7.33	15.85	10.00
Unknown Tags	4.13	4.31	3.31	1.63	3.07	2.99
Unknown Words & Tags	33.2	17.57	25.45	8.96	18.92	12.99

Table 4. Distribution of unknown words and unknown tags in the test sets across domains.

4.1 Lexical Tag Initialization

As mentioned in the previous section, there are multiple schemes for assigning the initial tag to an unknown lexical entity. We can either assign it as **unk** (unknown), **ng** (common noun, most frequently occurring tag for unknown words), or according to our *initial assignment rule*, which incorporates a small amount of prior knowledge:

If the word contains an English letter (A-Z / a-z), tag it as ***xch*** *(non-chinese word)*

Else tag as ***ng*** *(common noun).*

Results comparing the three schemes are shown in Figure 6.

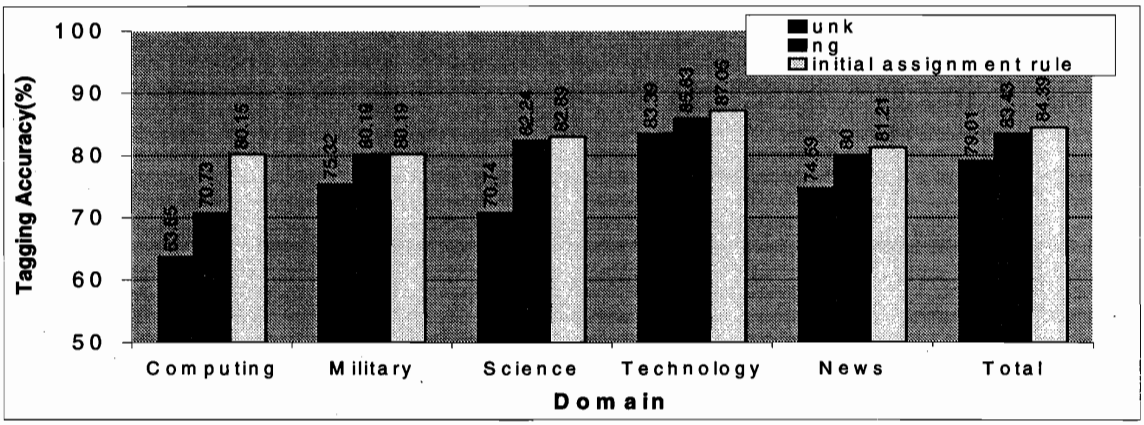

Figure 6. Test-set tagging accuracies (%) for the three different initial assignment schemes across the various domains.

Our initial assignment rule fares better than the straightforward **unk** or **ng** assignments. Hence we have decided to adopt it for our experiments.

4.2 Contribution of Lexical and Contextual Rules

Having acquired the initial stage assignments T_0, we proceeded with our experiments by applying first the lexical rules, and subsequently the contextual rules. At each point (T_{start} and T_{final}) we measured the tagging accuracy, in order to assess the respective contributions from the lexical and contextual rules. This procedure is illustrated in Figure 7. Experimental results on the test sets are shown in Figure 8.

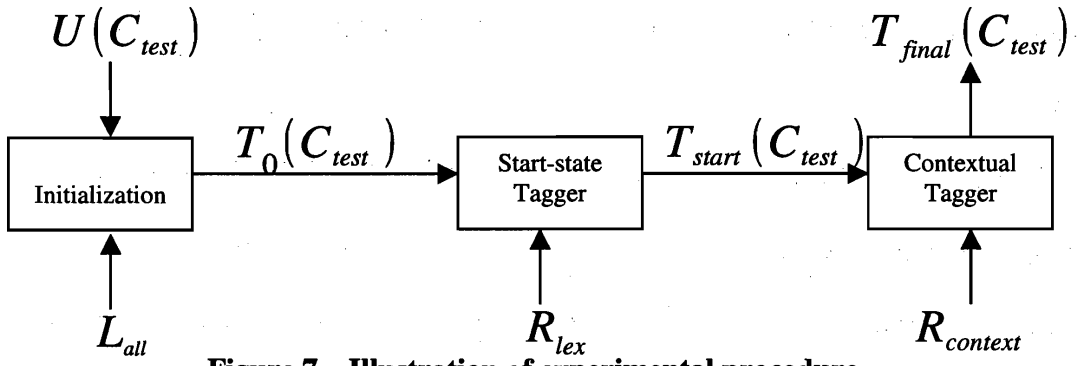

Figure 7. Illustration of experimental procedure.

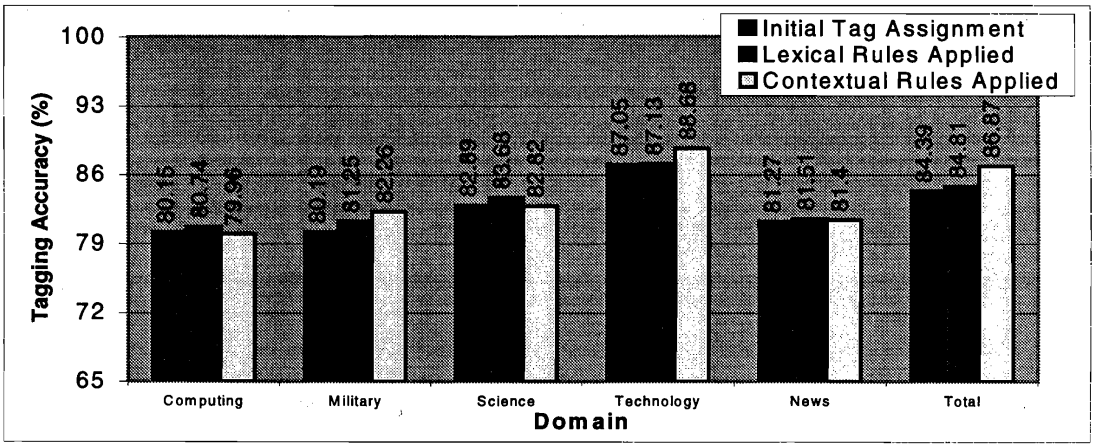

Figure 8. Tagging accuracies on the test sets.

Figure 8 shows that the lexical rules brought about a small but consistent improvement (from 0.08% to 1.06% across different domains) over the initial tag assignments across all the domains. However, the contextual rules led to a slight degradation in performance in three of

the five domains. For the "Total" category, we believe that the relatively higher improvement is due to a greater amount of training data made available from gathering together 90% of the entire corpus and the co-operation between lexical rules and contextual rules. As an illustration of the co-operation between lexical rules and contextual rules, consider the example sentence:

Untagged Sentence: 各 個 崗位 上 的 各 族 青年 朋友 致以 節日 的 祝賀 ！/！

Reference Sentence: 各/rn 個/qng 崗位/ng 上/f 的/usde 各/rn 族/ng 青年/ng 朋友/ng 致以/vgn 節日/ng 的/usde 祝賀/nvg ！/！

Since 致以 is an unknown word, which is tagged as ng by the start-state tagger. After the initial tag assignments and application of the lexical rule {以 hassuf 2 vgv}, the sentence is tagged as: 各/rn 個/qng 崗位/ng 上/f 的/usde 各/rn 族/ng 青年/ng 朋友/ng 致以/vgv 節日/ng 的/usde 祝賀/nvg ！/！

Finally, the application of the contextual rule {vgv vgn SURROUNDTAG ng ng} corrects the tag for 致以 from vgv to vgn and it's the correct tag for 致以 in the sentence.

In order to further assess the contribution of the contextual rules, we examined their effects on the training corpus. Results are shown in Figure 9. Since the training corpus does not have unknown words, we only have two sets of tagging accuracies - one from the initial tag assignments, and the other from lexical rule application.

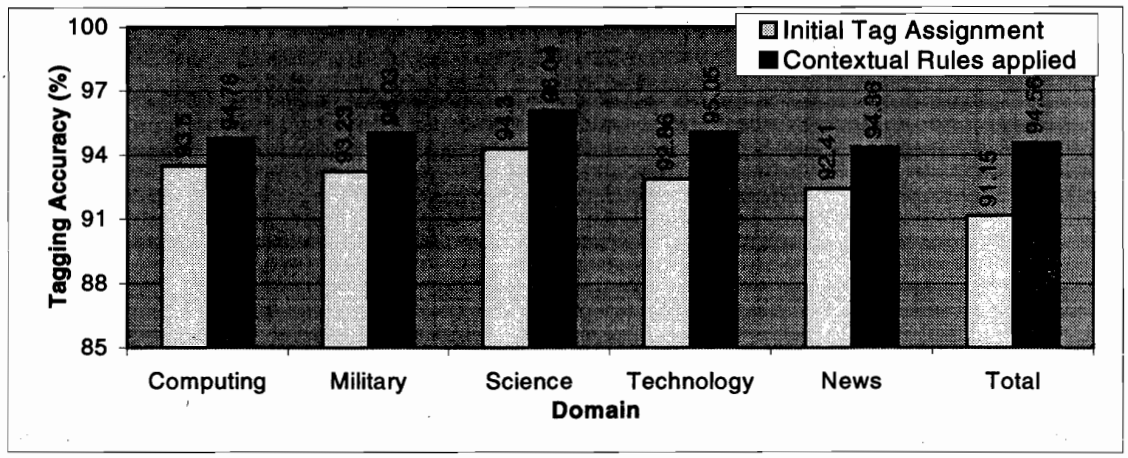

Figure 9. Tagging accuracies (%) on the training sets.

For the results in Figure 9, the initial tag assignments utilized the lexicon derived from the training set of the corresponding domain only. Compared to the test-set results, the contextual rules contributed to a more pronounced improvement, across the training sets in all the domains. The improvement did not carry over to the test sets, possibly due to over-fitting to the training sets.

4.3 Performance on Unknown Words

We have also examined our tagging performance on the unknown words and unknown tags in the test set. Performance accuracies on unknown words range between 40 to 50%, as shown in Table 5.

Test Sets	Computing	Military	Science	Technology	News	Total
Unknown word Performance	55.41	44.73	56.16	53.33	43.31	56.57

Table 5. Tagging accuracies (%) on the unknown words in the test sets.

Our experiments have also shown that the contextual rules learnt have not corrected any of the unknown tag errors in the test set. One reason is due to the propagation of errors - an errorful tag assignment to an unknown word may propagate via contextual rule applications to cause errors in subsequent tags. As an illustration of error propagation, consider the example sentence:

全 市 鄉鎮 企業 中 已 有 30 多 家 中 外 合資 合作 企業。

where 合資 is the unknown word, the tag **qni** (個體量詞) of 家 is the unknown tag. After the initial tag assignments and application of the lexical rules, the sentence is tagged as:

全/a 市/ng 鄉鎮/ng 企業/ng 中/f 已/d 有/vh 30/mx 多/mg 家/ng 中/f 外/f 合資/ng 合作/vg 企業/ng 。/。

The unknown word 合資 is tagged as **ng**.

Subsequent to this, application of the contextual rule {vg vgn prev1or2tag ng} transforms the tag for 合作 (from **vg** to **vgn**) since its left tag of word 合資 is **ng**. Therefore, the tag of 合作 is becomes an error. Now the sentence tags become:

全/a 市/ng 鄉鎮/ng 企業/ng 中/f 已/d 有/vh 30/mx 多/mg 家/ng 中/f 外/f 合資/ng 合作/vg 企業/ng 。/。

This is compared with the reference tags:

全/a 市/ng 鄉鎮/ng 企業/ng 中/f 已/d 有/vh 30/mx 多/mg 家/ng 中/f 外/f 合資/ng 合作/vg 企業/ng 。/。

We find five errors in the TEL tagging:

家/ng, 中/f, 外/f, 合資/ng, 合作/vg (hypothesized)

家/qni, 中/j, 外/j, 合資/d, 合作/vg (reference)

and among these three originated from unknown words and unknown tags (家, 合資, 合作)

4.4 A Possible Benchmark

We attempt to come up with an upper bound benchmark for our performance accuracies, by ameliorating the unknown word problem. To achieve this we included all the words in our *entire* corpus (L_{all}) for initial tag assignment. We have also used the entire training corpus for training the contextual rules (instead of divided it into the lexical and contextual portions, as mentioned previously). This experimental procedure is illustrated in Figure 10. Our experimental results suggest that possible upper bounds for tagging performance lies around 97% for training and 94% for testing in domain total. This compares with the previous performances of 94.56% in the training set (please see Figure 9 in pp.16) and 86.87% in the testing set (please see Figure 8 in pp. 16).

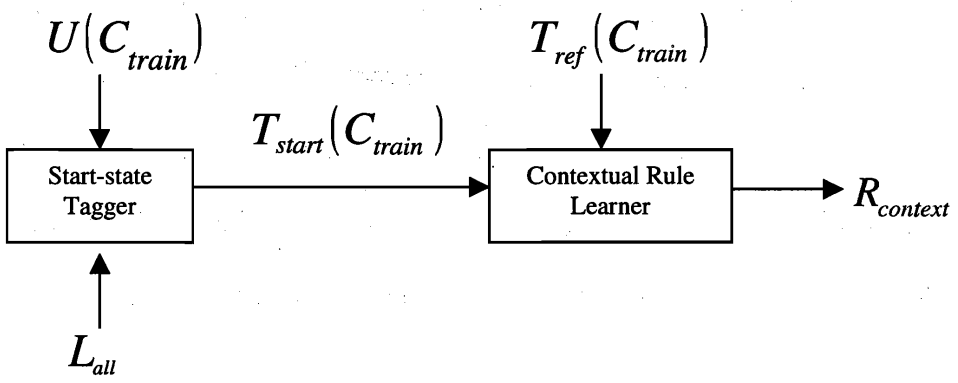

Figure 10. Training procedure which attempts to ameliorate the effect of unknown words.

Experimental results for both training and test sets are tabulated in Table 6.

Domains	Computing	Military	Science	Technology	News	Total
Training Accuracies	96.13	96.98	97.70	96.98	96.70	96.96
Testing Accuracies	94.10	92.05	94.18	92.51	92.73	93.88

Table 6. Tagging accuracies (%) for both training and test sets, under the condition with no unknown words.

4.5 Comparison between the TEL approach and the stochastic approach

We attempted to compare the TEL approach with a stochastic approach for POS tagging. Our stochastic tagger is provided by Tsinghua University. It utilizes a Markov model for POS tagging, i.e.

$$P(T's \mid W_s) = \max_{T_1 T_2 \ldots T_n} P(T_1 \mid T_2) \prod_{i=2}^{n} P(T_i \mid T_{i-1}) P(W_i \mid T_i)$$

and has been previously trained.[6] Therefore it was not straightforward for us to compare the two taggers based on identical training and testing sets. We divided each corpus into 10 partitions – 9 of them were used to train the TEL tagger and the remaining one for testing. This preserves the 9:1 divide between training and testing sets. These experiments are repeated 5 times by jackknifing the data sets, and the performance accuracies were averaged (see row 2, row 3 and column 7 of Table 7). We combined the average training and testing accuracies according to the formula:

Overall Accuracy (TEL) = 0.9 x average training accuracy + 0.1 average testing accuracy

The weights of the training and testing accuracies follow the proportion of the respective data sets. The Overall Accuracy (TEL), shown in the third row of Table 7 were compared with the corresponding values of the stochastic tagger, shown in the last row of the table. Our results suggest that the TEL and stochastic approaches produce comparable results.

Experimental Runs	1	2	3	4	5	Average (over 5 runs)
TEL tagger (Training Accuracy)	95.20	95.17	95.16	95.00	95.17	95.14
TEL tagger (Testing Accuracy)	88.33	87.60	87.46	88.40	87.26	87.80
TEL tagger (Overall Accuracy)	94.50	94.35	94.33	94.39	94.41	94.38
Tsinghua tagger	91.59	91.59	91.59	91.59	91.59	91.59

Table 7. Tagging accuracies (%) for both training and test sets. Comparison between the TEL approach and stochastic approach.

5. Conclusion

This work is our initial attempt in using the transformation-based error-driven learning (TEL) procedure for tagging Chinese text. TEL has previously been shown to be effective in POS tagging for English (achieving over 96% tagging accuracies in using the Brown and WSJ corpora) [Brill 1995]. It has several attractive properties: (i) it provides an automatic procedure for tagging, (ii) the lexical and contextual rules it learns often make intuitive sense for the Chinese language, and potential provides room for the incorporation of linguistic knowledge by a human, should there be sparse training data problems, (iii) the learning procedure aims to minimize errors to obtain maximum tagging accuracies.

[6] Previous literature indicates that the training was based on 90% of the corpus.

Using a Chinese news corpus of over 70,000 words, divided into disjoint training and test sets of a 9:1 ratio, we achieved overall tagging accuracies of 94.56% (training) and 86.87% (testing). Across the different domains, the proportion of unknown words and unknown tags range between 8% to 33%, and tagging performance from 79.96% to 88.68%. In general, the higher the proportion of unknown words/tags, the lower the tagging performance. The baseline performance (without applying any rules) was 91.16% (training) and 84.39% (testing). Both the lexical and contextual rules were found to be contributive towards tagging performance. Performance accuracies are much improved upon the use of a comprehensive lexicon to ameliorate the unknown word problem, reaching 96.96% (training) and 93.88% (testing) respectively as a possible gauge of an upper bound performance for our experiment. While direct comparison with the work of others[7] is difficult due to uncertainties in training/testing data partitioning, our experimental results in comparison with a stochastic tagger suggests that TEL is equally effective and applicable for Chinese.

Acknowledgements

We thank Eric Brill for his transformational tagger, and Tsinghua University (in Beijing) for providing the corpora for our experiments.

References

Bai, S. H., Xia, Y. and Huang, C. N., "Automatic Part of Speech Tagging System for Chinese" , Technical Report, Tsinghua University, Beijing, China, 1992.

Black, E., A. Finch and H. Kashioka, "Trigger-Pair Predictors in Parsing and Tagging", Proceedings of the International Conference on Computational Linguistics, pp. 131-137, 1998.

Brill, E., "Transformation-based Error-driven Learning and Natural Language Processing: A Case Study in Part of Speech Tagging", Computational Linguistics, Vol. 21, Number 4, 1995.

Chang, C. H. and Chen, C. D., "A Study on Integrating Chinese Word Segmentation on Part-of-Speech Tagging", Communications of COLIPS, Vol. 3, No. 1, pp. 69-77, 1993.

[7] e.g. [Bai et al., 1992]

Chiang, T. H., Chang, J. S., Lin M. Y. and Su K. Y., "Statistical Word Segmentation", Journal of Chinese Linguistics, pp. 147-174, 1996.

Chen, C.J., M.H. Bai, K.J. Chen, 1997, "Category Guessing for Chinese Unknown Words" Proceedings of the Natural Language Processing Pacific Rim Symposium 1997: pp. 35-40, NLPRS1997 Thailand.

Church, K., "A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text", Proceedings of ICASSP-89, pp. 695-698, 1989.

Jelinek, F., "Self-Organized Language Modeling for Speech Recognition", Readings in Speech Recognition, A. Waibel and K. F. Lee, eds., Morgan Kaufman Publishers, 1990.

Kupiec, J., "Robust Part-of-Speech Tagging using a Hidden Markov Model", Computer Speech and Language, 6:226-242, 1992.

Lua, K. T., "Part of Speech Tagging of Chinese Sentences using Genetic Algorithm", International Conference on Chinese Computing, pp. 45-49, 1996.

Merialdo, B., "Tagging Text with a Probabilistic Model", Proceedings of ICASSP-91, pp. 809-812, 1991.

Qin A. and Wong, W. S., "ACCESS: Automatic Segmentation and Part of Speech Tagging of Chinese Text", Technical report, The Chinese University of Hong Kong, 1998.

Shing-Huan Liu, Keh-jiann Chen, Li-ping Chang, Yeh-Hao Chin, "A Practical Tagger for Chinese Corpora", Proceedings of ROCLING VII, pp.111-126

Su, K. Y., Chiang, T. H. and Chang, J.S., "An Overview of Corpus-based Statistics-Oriented (CBSO) Techniques for Natural Language Processing", Computational Liguistics and Chinese Language Processing, Vol. 1., No. 1, pp. 101-157, August 1996.

On Modeling Remote and Local Dependencies in Language

Yu-Sheng Lai and Chung-Hsien Wu
Department of Computer Science and Information Engineering,
National Cheng Kung University, Tainan, Taiwan, R.O.C.
E-mail: {laiys, chwu}@csie.ncku.edu.tw

Abstract

In this paper, a statistical language model that can model both remote and local dependencies is proposed. This model takes into account the relationship between the predicted word and its preceding words without considering the order of the preceding words. Two primary parameters, the reliability coefficient and the combination factor, are proposed to achieve a better performance of the language model. The reliability coefficients identify the reliabilities of the remote dependencies to the predicted word. The combination factor gives a weight to the combination of the local dependency and the remote dependency.

The language model was tested on the task of word clustering and compared to the traditional N-gram language model. A large corpus provided by Academia Sinica, Taiwan, containing 5 million words was used for training and testing. The experimental results show that the proposed model takes littler computation and achieves a better performance for large N compared to the traditional N-gram language model.

1. Introduction

Statistical language models have proved useful when enough data is abailable to estimate the word probabilities. The most commonly used statistical language modeling technique is to consider the word sequence $w_1 \cdots w_Q$ as a Markov process and is termed as the N-gram language model. The traditional N-gram language model estimates the word sequence probability by the following equation

$$P(w_1 \cdots w_Q) = \prod_{n=1}^{Q} P(w_n \mid w_{n-N+1}^{n-1}) \tag{1}$$

where w_{n-N+1}^{n-1} represents the word sequence $w_{n-N+1} \cdots w_{n-1}$ for short and the conditional probability $P(w_n \mid w_{n-N+1}^{n-1})$ indicates that the probability of the word w_n

can be predicated by its preceding N-1 words $w_{n-N+1} \cdots w_{n-1}$.

The N-gram language model has been shown that it can work very well on dealing with local dependency in language. But it takes heavy computation and large memory requirement for large N. For practical reasons, most systems use bigram or trigram only. That is, they estimate the conditional probabilities only for N=2 or 3. Thus computational complexity and memory requirement can be reduced efficiently. In this model, however, the remote dependencies will not be taken into account. That is, some grammatical structures like "if...then" clause will not be modeled.

Without caring about heavy computation and memory requirement, the conditional probability $P(w_n \mid w_{n-N+1} \cdots w_{n-1})$ strictly constrains that the predicted word w_n is related to the preceding word sequence $w_{n-N+1} \cdots w_{n-1}$ and their order. In practice, however, the word w_n is partially related to the word sequence $w_{n-N+1} \cdots w_{n-1}$ only. In other words, the word w_n is only related to some words in the word sequence $w_{n-N+1} \cdots w_{n-1}$ rather than the whole word sequence. For instance, considering the sentence "I went for a long long walk this morning," using the conditional probability $P("walk" \mid "go","for","a")$ to predict the word "walk" will be more appropriate than using $P("walk" \mid "go","for","a","long","long")$. The phrase "go for a walk" is a very common usage in texts but the phrase "go for a long long walk" is often used in spoken language or is an unseen event.

One of the primary difficulties encountered using the N-gram language model is the problem of sparse data. No matter how large a training corpus you have, there will always be many unseen events that will come up in testing. For this sake, many people invested in modeling unseen events [1, 2]. Smoothing methods solved the problem of sparse data only for some cases. For instance, the unseen events never appearing in real world and the unseen events resulting from incomplete collection are different, but they are viewed as the same by the smoothing methods. In our opinion, the kinds of problems should be essentially dealt with in modeling phase rather than in smoothing phase.

A different approach in language modeling was proposed by using the technologies of class mapping [3]. For an unseen word m-gram, it is still possible to map it to a corresponding class m-gram. Because the number of model parameters such as the m-gram probabilities is reduced due to the class mapping, each parameter

can be estimated more reliably. On the contrary, reducing the number of model parameters will result in a rough model with less precise prediction of the next word. It is a tradeoff between these two extremes.

In terms of linguistics, however, word equivalence class is an important concept in syntax and semantics. It is defined by linguistic experts and is called part of speech (POS). In the past years, many techniques for word clustering have been proposed [4-6]. Generally, the algorithms are based on minimum perplexity or maximum likelihood. In this paper, the most commonly used quantity, perplexity, is used to evaluate the proposed language models on the task of word clustering.

The goal of this paper is to model both remote and local dependencies in language but just requires low computation and memory requirements. We will describe the remote dependency modeling in Section 2. The proposed language model will be described in Section 3. In Section 4, we will describe how to implement word clustering efficiently by the exchange algorithm. We designed several experiments to show the performance of the language model we proposed on word clustering. We will show the experimental results in Section 5. Finally, we will make some conclusions in Section 6.

2. Remote Dependencies Modeling

The N-gram language model encounters two difficulties while estimating remote dependencies. The first one is that it takes much time in computation and requires much memory for large N. The second one is the problem of sparse data. Here, we will describe a way for modeling remote dependencies but reducing the above requirements.

2-1 Estimation of Remote Dependencies

Estimating remote dependency between two disconnected words, intuitively, can be viewed as estimating remote bigram. If there is a pair of disconnected words v and w, where v appears in front of w in the text, then computing remote bigram of v and w can be viewed as computing the conditional probability $p_R(w \mid v)$ defined as

$$D_R(v,w) \equiv p_R(w \mid v) = \frac{F_R(v,w)}{F(v)} \tag{2}$$

where $F(v)$ denotes the frequency of the word v and $F_R(v,w)$ denotes the

frequency of the disconnected word pair (v, w) in the corpus.

However, the estimation of conventional bigram is not applicable to remote bigram. For each word, it counts remote dependencies in a proper range M based on the corpus. It will happen that $\sum_{w} p_R(w \mid v) \geq 1$ due to $\sum_{w} F_R(v, w) \geq F(v)$ when the range M is greater than 2. For instance, for the word sequence $v \cdots w_1 w_2$, the summation $F_R(v, w_1) + F_R(v, w_2)$ will be greater than the frequency $F(v)$ if we increase the frequencies $F(v)$, $F_R(v, w_1)$ and $F_R(v, w_2)$ by 1 respectively. To avoid this inequality, we just increase the frequency by c rather than 1 for each remote frequency $F_R(w_{n-i}, w_n), i = 2 \cdots M-1$ and c can be computed as

$$c \equiv \max\{\frac{1}{M-2}, \frac{1}{L-2}\} \tag{3}$$

where L is the number of the words from the left boundary of the sentence to the predicted word . Thus, it will keep the equal sign of the following equation

$$\sum_{w} F_R(v, w) = F(v) \tag{4}$$

Nevertheless the above estimation will lose some dependencies from more complex grammatical structures like "prefer to ... rather than." To avoid this problem, we can increase the degree of remote dependency by using remote m-gram rather than remote bigram. In our experiments, we model remote dependencies by using remote bigram only.

2-2 Reliability Coefficients

The remote dependency $D_R(v, w)$ is defined to represent the dependency between the predicted word w and a prior word v. Since there are several dependencies in the proper range M, it is reasonable to assign a weight for each dependency. We call them reliability coefficients. They identify the reliability of the corresponding dependency to the predicted word. The more the appearance frequency is, the better the reliability is. For a remote dependency $D_R(w_{n-i}, w_n)$, therefore, the reliability coefficient $\lambda_{i,n}$ can be estimated as

$$\lambda_{i,n} = \frac{F_R(w_{n-i}, w_n)}{\sum_{j=2}^{M-1} F_R(w_{n-j}, w_n)} \tag{5}$$

3. The Proposed Model

In this section, we will describe how to combine remote dependencies into N-gram language model. In order to solve the problem of sparse data, we categorize words into word equivalence classes and estimate unseen events by using Turing-discounted probabilities [7].

3-1 Combination of Remote and Local Dependencies

The proposed model consists of two components: the N-gram language model (N-gram) and the language model with parallel remote dependencies (PRD). These two components could be defined as follows.

- *N-gram Language Model (N-gram)*

$$P_{N-gram}(w_n \mid w_{n-N+1}^{n-1}) = \frac{F(w_{n-N+1}^{n})}{F(w_{n-N+1}^{n-1})} \tag{6}$$

- *Language Model with Parallel Remote Dependencies (PRD)*

$$P_{PRD}(w_n \mid w_{n-M+1}^{n-1}) = \prod_{i=2}^{M-1} D_R(w_{n-i}, w_n)^{\lambda_{i,n}} \tag{7}$$

Since the N-gram model considers the local dependency only, it is enough for N=2 or 3 in the combination model. The combination model named Language Model with M-Remote and N-Local Dependencies (MRNLD) consists of N-gram with small N and the language model with M parallel remote dependencies. Fig.1 shows the relationship between the predicted word and the remote and local dependencies. The language model can be defined as

$$P_{MRNLD}(w_n \mid w_{n-M+1}^{n-1}) \equiv P_{N-gram}(w_n \mid w_{n-N+1}^{n-1})^{\alpha(w_n)} \cdot P_{PRD}(w_n \mid w_{n-M+1}^{n-N})^{1-\alpha(w_n)} \tag{8}$$

where $\alpha(w_n)$ is the combination factor. It weights the N-gram language model and the language model with M parallel remote dependencies for each word w_n. We model its behavior by using a sigmoid function that can be computed as

$$\alpha(w_n) = \frac{1}{1+e^{-(l(w_n)-r(w_n))}}, \tag{9}$$

where $l(w_n)$ and $r(w_n)$ represent the local and remote log likelihood functions for the word w_n respectively. They are defined as follows

$$l(w_n) = \log \prod_{w \in W} p_L(w_n \mid w)^{F_L(w,w_n)} = \sum_{w \in W} (\log F_L(w,w_n) - \log F(w)) F_L(w,w_n) \tag{10}$$

$$r(w_n) = \log \prod_{w \in W} p_R(w_n \mid w)^{F_R(w,w_n)} = \sum_{w \in W} (\log F_R(w,w_n) - \log F(w)) F_R(w,w_n) \tag{11}$$

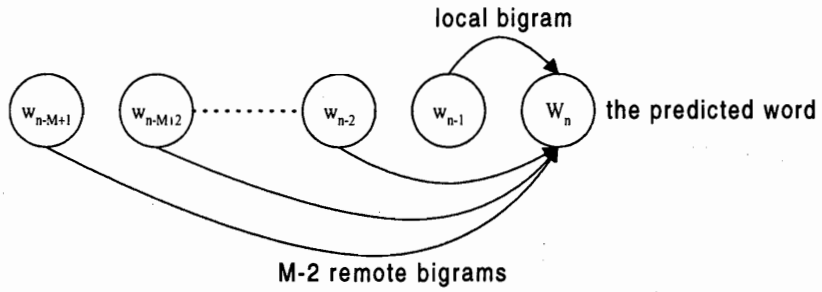

Fig.1 the relationship between the predicted word and the remote and local dependencies

3-2 Word Equivalence Class Mapping

In word clustering, we assumed that each word belongs to only one class. By this assumption, a mapping C from vocabulary W to classes G can be represented as

$$C : W \rightarrow G \tag{12}$$

and by this mapping, the bigram probability [1] can be defined as

$$P(w \mid v) \equiv P(w \mid C(w)) \cdot P(C(w) \mid C(v)) \tag{13}$$

where $P(w \mid C(w))$ denotes the membership probability of the word w and $P(C(w) \mid C(v))$ denotes the transition probability from class $C(v)$ to class $C(w)$. Then Eq.8 can be recomputed as

$$\begin{aligned} & P_{MRNLD}(w_n \mid w_{n-M+1}^{n-1}) \\ & = P(w_n \mid C(w_n)) \\ & \times P_{N-gram}(C(w_n) \mid C(w_{n-N+1}) \cdots C(w_{n-1}))^{\alpha(C(w_n))} \\ & \times P_{PRD}(C(w_n) \mid C(w_{n-M+1}) \cdots C(w_{n-N}))^{1-\alpha(C(w_n))} \end{aligned} \tag{14}$$

where the combination factor $\alpha(C(w_n))$ and the related tokens are well defined as follows:

$$\alpha(g) = \frac{1}{1 + e^{-(l(g)-r(g))}} \tag{15}$$

$$l(g) = \sum_{h \in G} (\log F_L(h,g) - \log F(h)) F_L(h,g) \tag{16}$$

$$r(g) = \sum_{h \in G} (\log F_R(h,g) - \log F(h)) F_R(h,g) \tag{17}$$

After the word clustering process, the number of unseen evens can be greatly reduced. For the remaining unseen events, the Turing-discounted probabilities [7] are adopted for further smoothing.

4. Implementation of Word Clustering

4-1 Clustering Algorithm

We use the exchange algorithm [4] in this word clustering process. The main idea of the algorithm is to find a class mapping $C : W \rightarrow G$ such that the perplexity of the language model is minimized over the training corpus, where an observation word may be exchanged from a class to another class in order to improve the criterion. In the case of language modeling, the optimization criterion is the entropy described in next subsection. The initialization method is to assign the most frequent $|G|-1$ words into their own word equivalence classes, where $|G|$ is the number of classes, and the remaining words into an additional word equivalence class.

4-2 Performance Measure

Having constructed a language model, we need to show how well the proposed language model performs in a task. It is necessary to have a method for measuring the performance. We use the perplexity to measure the performance of the MRNLD on word clustering. The formal perplexity PP is defined as [8]

$$PP \equiv P(w_1 w_2 \cdots w_Q)^{-\frac{1}{Q}} \tag{18}$$

For the MRNLD, the estimation of well-defined entropy can be decomposed in terms of frequencies as follows

$$H_P = \log PP \tag{19}$$

$$= -\frac{1}{Q} \log P(w_1 w_2 \cdots w_Q) \tag{20}$$

$$= -\frac{1}{Q} \log \prod_{n=1}^{Q} P_{MRNLD}(w_n \mid w_{n-M+1}^{n-1}) \tag{21}$$

$$= -\frac{1}{Q}\sum_{n=1}^{Q}\log P_{MRNLD}(w_n \mid w_{n-M+1}^{n-1}) \tag{22}$$

$$\begin{aligned} = -\frac{1}{Q}\{&\sum_{n=1}^{Q}\log p(w_n \mid C(w_n)) \\ &+ \sum_{n=1}^{Q}\alpha(C(w_n))\log p_L(C(w_n)\mid C(w_{n-N+1})\cdots C(w_{n-1})) \\ &+ \sum_{n=1}^{Q}[(1-\alpha(C(w_n)))\sum_{i=N}^{M-1}\lambda_{i,n}\log p_R(C(w_n)\mid C(w_{n-i}))]\} \end{aligned} \tag{23}$$

$$\begin{aligned} = -\frac{1}{Q}\{&\sum_{w\in W}F(w)\log\frac{F(w)}{F(C(w))} \\ &+ \sum_{g\in G,H\in G^{N-1}}\alpha(g)F_L(H,g)\log\frac{F_L(H,g)}{F(H)} \\ &+ \sum_{n=1}^{Q}((1-\alpha(C(w_n)))\sum_{i=N}^{M-1}\lambda_{i,n}\log\frac{F_R(C(w_{n-i}),C(w_n))}{F(C(w_{n-i}))})\} \end{aligned} \tag{24}$$

$$\begin{aligned} = -\frac{1}{Q}\{&\sum_{w\in W}F(w)\log F(w) - \sum_{g\in G}F(g)\log F(g) \\ &+ \sum_{g\in G}\alpha(g)\sum_{H\in G^{N-1}}F_L(H,g)(\log F_L(H,g)-\log F(H)) \\ &+ \sum_{n=1}^{Q}((1-\alpha(C(w_n)))\sum_{i=N}^{M-1}\lambda_{i,n}(\log F_R(C(w_{n-i}),C(w_n))-\log F(C(w_{n-i}))))\} \end{aligned} \tag{25}$$

$$\begin{aligned} = -\frac{1}{Q}\{&\sum_{w\in W}F(w)\log F(w) - \sum_{g\in G}F(g)\log F(g) \\ &+ \sum_{g\in G}\alpha(g)\sum_{H\in G^{N-1}}F_L(H,g)(\log F_L(H,g)-\log F(H)) \\ &+ \sum_{g\in G}(1-\alpha(g))\sum_{n\ni C(w_n)=g}\frac{\sum_{i=N,h_i=C(w_{n-i})}^{M-1}F_R(h_i,g)(\log F_R(h_i,g)-\log F(h_i))}{\sum_{i=N,h_i=C(w_{n-i})}^{M-1}F_R(h_i,g)}\} \end{aligned} \tag{26}$$

By Eq.26, it takes much time on computing remote dependencies due to dynamic reliability coefficients. In order to reduce the computational complexity, $\lambda_{i,n}$ is chosen as a constant, $\frac{1}{M-N}$. It means that the reliabilities for all remote dependencies are equal. Then Eq.26 can be rewritten as

$$H_p = -\frac{1}{Q}\{\sum_{w\in W} F(w)\log F(w) - \sum_{g\in G} F(g)\log F(g) + \sum_{g\in G}\alpha(g)\sum_{H\in G^{N-1}} F_L(H,g)(\log F_L(H,g) - \log F(H)) + \sum_{g\in G}(1-\alpha(g))\sum_{h\in G} F_R(h,g)(\log F_R(h,g) - \log F(h))\} \quad (27)$$

5. Experimental Results

In this section, we will show the experimental results for the word clustering process. The test corpora, ASBC (Academia Sinica Balanced Corpora), were provided by Academia Sinica, Taiwan. We tested on four aspects: The first one is model testing. It tests on three models: the traditional N-gram language model, the language model with M parallel remote dependencies, and the proposed model MRNLD. The second one is the testing for CPU time. It compares the CPU time in word clustering by using different language models: the class trigram language model and the MRNLD. The third one is parameter testing. It tests the reliability coefficient $\lambda_{i,n}$ and the combination factor α. The fourth one is corpus test including inside test and outside test. All of these tests evaluate the performance by perplexities.

5-1 Corpora

ASBC consists of several corpora that were collected and tagged by Institute of Information Science, Academia Sinica. It contains 5 million words and a vocabulary of 130,000 words including common words, proper nouns and compound words. In our experiments, we chose about 27,000 most frequent words as the vocabulary.

In the word clustering process, we predefined 6 classes. The first two classes consist of one word respectively. The first two classes are "iou3" (有) and "shz4" (是) and their grammar behaviors are very complex, so we pre-clustered them into 2 classes respectively. The third class consists of 4 words: "de" (的), "jr" (之), "de" (得), and "de" (地) due to their special functions. The fourth class collects all borrowed words from foreign languages in the corpora. The fifth class collects those out-of-vocabulary words. The sentence boundary was viewed as a word and pre-clustered into the sixth class.

5-2 Word Clustering Experiments

In the experimental results, the traditional trigram language model is abbreviated

to trigram, the language model with 3 parallel remote dependencies is abbreviated to 3-PRD, and the language model with 3 remote and 2 local dependencies is abbreviated to 3-R-2-LD. Additionally, 3-PRD is defined as 3-R-2-LD with the local degree (N) being 1.

5-2-1 Model Test

Table 1 shows perplexities of trigram, 3-PRD, and 3-R-2-LD. In this experiment, we tested on remote degree of 3, dynamic combination factors, and static reliability coefficients. We used the whole corpus of 5 million words in testing. However, since trigram needs large computation, it was just tested on cluster numbers of 50, 100, and 200. The results show the language model with 3 remote and 2 local dependencies is better than the traditional trigram language model in word clustering.

Table 1. Perplexities for different models with different numbers of word equivalence classes

No. of Classes / L. M.	50	100	200	500	1000	2000
Trigram	247.63	212.58	182.38	-	-	-
3-PRD	215.23	195.46	160.78	136.92	108.44	95.45
3-R-2-LD	201.39	173.85	135.26	112.45	89.86	78.93

Table 2 shows perplexities of PRD and MRNLD with different remote degrees (M) from 3 to 8 and a fixed local degree (N) being 2. In this experiment, we clustered the whole corpus of 5 million words into 50 classes by using dynamic combination factors and static reliability coefficients. The results show that the perplexities of both two models decrease as the remote degrees increase and MRNLD performs better than PRD.

Table 2. Effect of remote degree (M) for different models

M / L. M.	3	4	5	6	7	8
PRD	215.23	208.91	199.73	193.28	205.63	211.37
MRNLD (N=2)	201.39	196.54	188.49	185.10	190.57	196.25

5-2-2 CPU Time Test

Table 3 shows the CPU time per iteration by using the 3-R-2-LD model and the trigram model on word clustering and the result shows that the 3-R-2-LD model is more efficient than the trigram model. This experiment is tested on the corpus of 5 million words. Due to large computations of trigram, we tested only on cluster numbers of 50, 100, and 200.

Table 3. CPU time (minutes per iteration) for clustering algorithm on different models

No. of Classes / L. M.	50	100	200	500	1000	2000
Trigram	172	340	1035	-	-	-
3-R-2-LD	115	230	621	2016	5138	13740

5-2-3 Parameter Test

To reduce the computational complexity, we simplified the dynamic reliability coefficients to be static ones. We want to know the simplification effect in this experiment. Additionally, due to the large computation in testing on the dynamic reliability coefficients, we used a small corpus that is only part of the ASBC and it is also clustered into 50, 100, and 200 classes. The downsized corpus consists of 1 million words. Table 4 shows the experimental results. The static reliability coefficients are better than the dynamic ones. This seemly contradicts to our expectation. A reasonable explanation is the problem of data sparseness.

Table 4. Perplexities for dynamic and static reliability coefficients (λ)

No. of Classes / λ	50	100	200	500	1000	2000
Dynamic	245.86	197.41	158.03	-	-	-
Static	223.07	185.15	149.79	116.93	95.23	87.74

The combination factor α is dynamic and defined by a sigmoid function. The MRNLD is the combination of the N-gram and PRD, the combination factor determines whether the N-gram model is more important than PRD or not. From Table 5, we know that sometimes N-gram is more important than PRD but sometimes

not. It depends on classes. The corpus used in this experiment consists of 5 million words.

Table 5. Effect of combination factor (α) on the number of classes

α \ No. of Classes	50	100	200	500	1000	2000
0.25	283.97	273.34	254.37	245.18	223.64	209.84
0.5	269.51	250.49	212.49	204.31	179.57	168.35
0.75	254.66	225.04	197.43	164.25	144.59	123.88
Dynamic	201.39	173.85	135.26	112.45	89.86	78.93

5-2-4 Corpora Test

A successful language model should be applied to any other corpora. So we divided the corpora into two groups of 1 and 4 million words. Let the small one be the training corpus and the big one be the test corpus. Table 6 and 7 show the experimental results. The same as our expectation, the results of the outside test are somewhat worse than the inside test. Besides, both of these two tests show that the language model with the remote degree of 6 has the best performance.

Table 6. Perplexities on inside test

M \ No. of Classes	50	100	200	500	1000	2000
3	223.07	185.15	149.79	116.93	95.23	87.74
4	218.21	179.48	144.83	113.46	93.57	82.06
5	209.32	176.25	137.68	105.30	90.37	80.64
6	207.58	172.79	136.51	102.22	88.24	79.62
7	212.03	188.16	145.22	107.97	92.75	85.34
8	220.57	190.62	146.13	112.68	93.06	88.29

Table 7. Perplexities on outside test

M \ No. of Classes	50	100	200	500	1000	2000
3	252.12	228.06	197.00	160.94	133.76	125.63
4	244.56	216.83	186.29	157.20	130.94	120.39
5	243.67	215.98	175.64	147.26	126.70	116.52
6	237.06	208.34	174.17	139.56	117.53	110.02
7	252.78	222.03	185.63	153.64	125.64	118.08
8	269.74	224.76	190.49	157.89	134.80	120.24

6. Conclusions

In this paper, we proposed a word equivalence class based language model that can model both remote and local dependencies. This model takes into account the relationship between the predicted word and its preceding words without considering the order of the preceding words. Although this model considers the remote dependency and the local dependency simultaneously, it requires littler computation than the traditional class-based N-gram language model on word clustering task and achieves a better performance for large N.

Two primary parameters, the reliability coefficient and the combination factor, are proposed to achieve a better performance of the language model. According to the experimental results, the language model achieves the best performance on static reliability coefficients and dynamic combination factors.

References

[1] S. M. Katz, "Estimation of Probabilities from Sparse Data for The Language Model Component of A Speech Recognizer," IEEE Trans. on Acoustics, Speech, and Signal Processing, vol. 35, no. 3, March 1987, pp. 400-401.

[2] F. Jelinek and R. L. Mercer, "Interpolated Estimation of Markov Source Parameters from Sparse Data," Pattern Recognition in Practice, North Holland, 1980, pp. 381-397.

[3] S. Martin, J. Liermann, and H. Ney, "Algorithms for bigram and trigram word clustering," Speech Communication, 1998, pp. 19-37.

[4] R. Kneser and H. Ney, "Improved Clustering Techniques for Class Based

Statistical Language Modeling," Proc. 3rd European Conference on Speech Communication and Technology, 1993, Berlin, pp. 973-976.

[5] P. F. Brown, V. J. Della Pietra, P. V. de Souza, J. C. Lai, R. L. Mercer, "Class Based N-gram Models of Natural Language," Computational Linguistics 18 (4), 1992, pp. 467-479.

[6] M. Jardino, G. Adda, "Automatic Word equivalence classification Using Simulated Annealing," Proc. 3rd European Conference On Speech communication and Technology, 1993, Berlin, pp. 1191-1194.

[7] I. J. Good, "The Population Frequencies of Species and The Estimation of Population Parameters," Biometrika 40, December 1953, pp. 237-264.

[8] Lawrence Rabiner and B. H. Juang, "Fundamentals of Speech Recognition," Prentice Hall, pp. 449-450.

Term Selection with Distributional Clustering for Chinese Text Categorization using N-grams

Jyh-Jong Tsay and Jing-Doo Wang
Department of Computer Science and Information Engineering
National Chung Cheng University
Chiayi, Taiwan 62107, ROC.
{tsay, jdwang}@cs.ccu.edu.tw
TEL:886-5-2720411.EXT.6207, FAX:886-5-2720859

Abstract

In this paper we propose an SB-tree approach to extract significant patterns efficiently by scanning the leaves of the SB-tree to decide the boundary of significant patterns for term extraction, and reduce the dimension of term space to an practical level by a combination of term selection and term clustering. Our current experiment uses CNA one year news as training data, which consists of 73,420 articles and is far more than previous related research. In the experiment, we compare the performance four term selection methods, odds ratio, mutual information, information gain and χ^2 statistic, when they are combined with distributional clustering method. Our experiment shows that χ^2 statistic and information gain achieve performance better than odd ratio and mutual information when they are combined with distributional clustering. With the combination of term selection and term clustering, the dimension of term space can be greatly reduced from 60000 to 120 while maintaining similar classification accuracy.

Keywords: Text Categorization, Term Selection, Term Clustering, Naive Bayes Classifier, Information Retrieval.

1 Introduction

Text classification(categorization) is the problem of automatically assigning predefined classes to free text documents, and is gaining more and more importance as the amount of text data available on World Wide Web grows dramatically. A well classified text database will be very helpful for a user to identify interesting data from the huge collection of texts. There are many studies about the text classification as well as web-page classification [17, 1, 9, 10, 27, 32, 33, 23, 24, 7, 38, 15]. While there are a great number of researches on automatic text classification for English texts, text classification for Asian languages such as Chinese, Japanese, Korean and Thai has not been studied seriously until recently [36, 21, 37, 3, 28, 31, 29].

Because text segmentation is not straightforward in Asian languages, 1-grams, 2-grams and n-grams have been used as indexing terms to represent documents. It is reasonable that n-gram is more meaningful and brings more concept than 1-gram or 2-gram. The main obstacle to apply n-grams to Chinese text classification is the huge number of possible n-grams. Notice that many of them are meaningless and non-informative for text categorization. The major challenge is to develop an approach that can reduce the dimension of term space to an acceptable level while maintains similar classification accuracy. There was a related study about term selection in Chinese text classification[29]. A practical problem there is that a news may contain very few or even non of the selected terms, and thus is classified to the default class which is the largest class. On the other hand, a large number of selected terms make Chinese text classification computationally impractical. To overcome the problems, we study the combination of the term(feature) selection and term clustering in this paper. We first use term selection to select a set of significant terms, and then use term clustering to cluster the selected terms into a small number of groups. Our experiment on one year CNA news shows that the dimension of term space can be greatly reduced while maintaining similar classification accuracy.

The remainder of this paper is organized as follows. Section 2 describes the process to remove meaningless and non-informative substrings. Section 3 gives the scoring functions of four term selection methods, and reviews distributional clustering. Section 4 introduces the

naive Bayes classifier. Section 5 gives our experimental results. Section 6 gives conclusion. Throughout this paper, we assume $2 \leq n \leq 20$ when n-gram is mentioned.

2 Term Extraction

There are several research[30, 5, 25] on the extraction of meaningful terms from Chinese texts. In [30] Tseng proposed *a multi-linear term-phrasing* technique in which adjacent character sequences are merged pairwisely to form longer character sequences if they satisfy the criteria of the merging rules. This approach is simple but can not run incrementally when new news are added. In [5] Chien proposed *PAT-tree* method to extract keyword. PAT-tree is an incremental method but does not handle the I/O problem when the amount of memory is not large enough to store the whole tree. In this paper, we propose an approach based on SB-trees [13] which use B^+tree to store all the suffix strings[14] of the training documents. Note that SB-tree can grow incrementally, is I/O efficient and is scalable to store large amount of data.

We construct two SB-trees to locate the left and right boundary of terms respectively, and compute the statistics information of extracted term by scanning the leaves of SB-tree. We use *SB-trees* [13, 29] to store all suffix strings [14] of every sentences in the training corpus, and then search for all the repeated strings which appear more than once. To eliminate redundant strings, we gather only the repeated patterns that have, at least, two different kinds of successor Chinese characters. For example, in Figure 1, there are partial sorted suffix strings listed in the SB-tree. The "傳統", "傳統工業" and "傳統工業技術升級"are considered as candidate patterns. Notice that the "傳統工業技", "傳統工業技術" and "傳統工業技術升" are not considered as candidate patterns because they have only one successor Chinese character "術","升" and "級" respectively. This process determines the right boundary of terms.

To determine the left boundary of terms, we construct another SB-tree, called *Reverse-SB-tree*, with all suffix strings that come from each reversed sentences in the training corpus. For example, in Figure 2, there are candidate repeated patterns "級升", "級升術技業工" and "級升術技業工統傳". Similarly, the "級升術", "級升術技" and "級升術技業" are

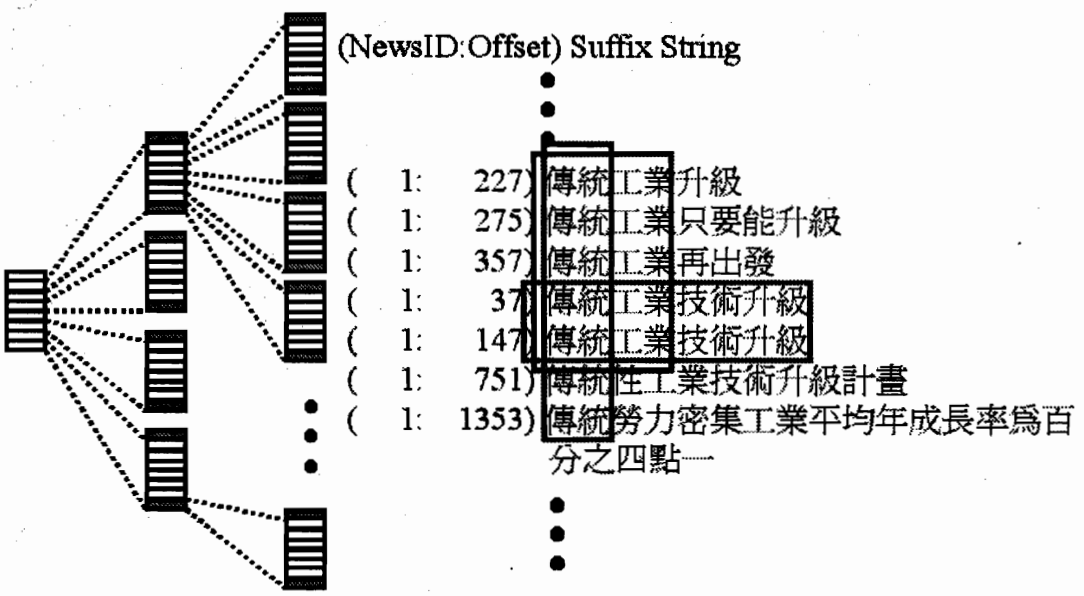

Figure 1: SB-tree

not considered as candidate patterns because they have only one successor Chinese character "技","業" and "エ" respectively. This process determines the left boundary of terms. Terms identified in above process form an initial set of terms which are used for term selection.

3 Term Selection

After extracting terms from the training corpus as described in section 2, we apply term selection algorithms to select the most representative terms for each class. All terms are given scores by the term selection method, and are choosed according to the scores. There are four term selection methods evaluated individually in this paper. These four term selection methods are odds ratio(OR), information gain(IG), mutual information(MI) and χ^2 statistic(CHI). For a term t and a class c, let A denote the number of times t and c co-occur, B is the number of times t occurs without c, C is the number of times c occurs without t, and N is the total number of documents. The following reviews the term selection methods evaluated in this paper.

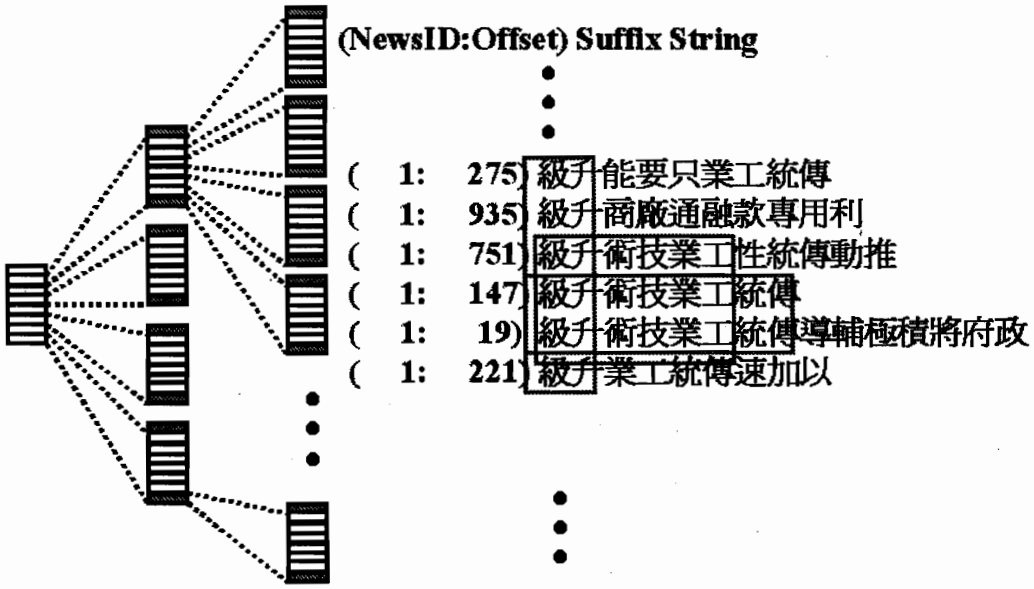

Figure 2: Reverse SB-tree

	c	$\bar{c}$
t	A	B
$\bar{t}$	C	D

Table 1: Two-Way contingency table of a term t and a class c

3.1 Odds Ratio(OR)

The odds ratio value of term t for each class (category) is different. For each term t, the value of odds ratio to class C_k is defined as follows[15].

$$\begin{aligned} OddsRatio(t, C_k) &= \log \frac{Odds(t|C_k)}{Odds(t|C_{neg})} \\ &= \log \frac{P(t|C_k)(1 - P(t|C_{neg}))}{(1 - P(t|C_k)P(t|C_{neg}))}, \end{aligned}$$

where $P(t|C_k)$ is the conditional probability of term t_j occurring given the class value k, $P(t|C_{neg})$ is the conditional probability of term t occurring given the class value $\neq k$. The odds function of X_i is defined as follows.

$$Odds(X_i) = \begin{cases} \frac{\frac{1}{N^2}}{1-\frac{1}{N^2}} & P(X_i) = 0 \\ \frac{1-\frac{1}{N^2}}{\frac{1}{N^2}} & P(X_i) = 1 \\ \frac{P(X_i)}{1-P(X_i)} & P(X_i) \neq 0 \wedge P(X_i) \neq 1 \end{cases}$$

Notice that the value of odds ratio of a term which appears in only one class will be very large even its term frequency is low. It happens that the term selection via the score of odds ratio method might suffer from low hit frequency of selected term when apply to testing documents. This indicates that it is highly possible for a new document to contain very few or even no terms selected by odds ratio method.

3.2 Mutual Information(MI)

The difference between the information uncertainty before adding t and after adding t measures the gain in information due to the Class c. This information is called *mutual information*[35] and is defined as follows.

$$\begin{aligned} MI(t, c) &= \log[\frac{1}{P(c)}] - \log[\frac{1}{P(c|t)}] \\ &= \log[\frac{P(c|t)}{P(c)}] \end{aligned}$$

$$= \log[\frac{P(t,c)}{P(t)P(c)}]$$
$$= MI(c,t)$$

If the two probabilities $P(t)$ and $P(t|c)$ are the same, then no information is gained and the mutual information is zero. In practice, the score of $MI(t,c)$ is strongly influenced by the marginal probabilities of terms. For terms with an equal conditional probability $P(t|c)$, the term with low term frequency will have a higher score than common terms. The MI can be estimated using

$$MI(t,c) \approx \log \frac{A \times N}{(A+C) \times (A+B)}$$

3.3 Information Gain(IG)

Information Gain is frequently employed as a method of feature scoring in the field of machine learning [26]. Let $|c|$ denote the number of classes. The information gain of term t is defined as follows.

$$\begin{aligned} IG(t,C) = E(C) - E(C|t) = & - \sum_{k=1}^{|c|} P(C_k) \log P(C_k) \\ & + P(t=1) \sum_{k=1}^{|c|} P(C_k|t=1) \log P(C_k|t=1) \\ & + P(t=0) \sum_{k=1}^{|c|} P(C_k|t=0) \log P(C_k|t=0) \end{aligned}$$

IG is equivalent to the weighted average of the mutual information and is called *average mutual information.* IG makes use of information about term absence, while MI ignores such information. Furthermore, IG normalizes the mutual information scores using the joint probabilities while MI uses the non-normalized scores [35].

3.4 χ^2 statistic (CHI)

The χ^2 statistic measures the lack of independence between t and c, and can be compared to the χ^2 distribution with one degree of freedom to judge extremeness. The χ^2 statistic measure is defined in [20] as follows.

$$\chi^2(t,c) = \frac{N \times (AD - CB)^2}{(A+C) \times (B+D) \times (A+B) \times (C+D)}$$

3.5 Distributional Clustering

One of the practical problems in term selection is that a document may contain very few or even non of the selected terms(n-grams) if only a small number of significant terms are selected. However, a large number of selected terms will make automatic classification computationally impractical. To overcome the problems, we combine term(feature) selection with term clustering. Notice that term clustering is hard to implement without term selection because the number of extracted terms as described in section 2 is still very large. In this paper we used the distributional clustering[2] to cluster the selected terms. In the following we give a brief description of distributional clustering.

Term clustering algorithms define a similarity measure between terms, and group similar terms into single events that no longer distinguish among their constituent terms. In [2] Baker proposed a weighted average of the parameters of its constituent terms and let, for example, the random variable over classes, C, and its distribution given a particular term, t_i. When term t_i and t_j are clustered together, the new distribution is the weighted average of the individual distributions is as following:

$$\begin{aligned} P(C|t_i \vee t_j) &= \frac{P(t_i)}{P(t_i)+P(t_j)} P(C|t_i) \\ &+ \frac{P(t_j)}{P(t_i)+P(t_j)} P(C|t_j) \end{aligned}$$

The core intuition behind distributional clustering for document classification : the class distributions, $P(C|t_i)$, express how individual terms contribute to classification, and the clustering did preserve the shape of these distributions. Term clustering methods create new, reduced-size event spaces by joining similar terms into groups. The measure of the difference between two probability distributions adapted by [2] is Kullback-Leibler divergence, which is an information-theoretic measure. The KL divergence between the class distributions induced by t_i and t_j is written $D(P(C|t_i)||P(C|t_j))$, and is defined

$$-\sum_{k=1}^{|C|} P(C_k|t_i) \log \frac{P(C_k|t_i)}{P(C_k|t_j)}$$

To avoid the odd properties of KL divergence, such as not symmetric, and it is infinite when an event with non-zero probability in the first distribution has zero probability in the second

distribution, they modify the above formula as average KL divergence.

$$\frac{P(t_i)}{P(t_i \vee t_j)} \cdot D(P(C|t_i)||P(C|t_i \vee t_j)) + \frac{P(t_j)}{P(t_i \vee t_j)} \cdot D(P(C|t_j)||P(C|t_i \vee t_j))$$

Instead of comparing the similarity of all possible pairs terms ($O(|V|^2)$ operation), Baker create clusters using a simple, greedy agglomerative approach that consider all pairs of a much smaller subset, of size M, where M is the final number of clusters desired. The clusters are initialized with M terms that have highest score, using information gain(IG) in [2]. The most similar two clusters are joined, the next term is added as a singleton cluster to bring the total number of clusters back up to M. Notice that the number of score for each term measured by IG is just one. Therefore, the M terms as initial cluster may prefer some classes such that result in a biased estimate of term probability distribution to begin with. To avoid a biased estimate of term probability distribution to begin with, we have equal number of selected terms from each class as initial seeds of clusters. Experiment results show that our modification did improve the classification accuracy and smooth the variation of accuracy between each class.

4 Naive Bayes Classifier

There are several well known text classification methods[34] in machine learning or image processing field, such as decision tree method, Neural network method[11], k-nearest-neighbors(KNN)[22], Rocchio algorithm [24] and Naive Bayes classifier [26, 19]. In this research, we implement the naive Bayes classifier for its simplicity and scalability. We are ready to implement other classifiers and measure their performance when they are combined with various term selection methods. The Naive Bayes classifier is one highly practical learning method and is based on the simplifying assumption that the probabilities of terms occurrences are conditionally independent of each other given the class value [26], though this is often not the case. The naive Bayes approach classifys a new document Doc to the most probable class, C_{NB} defined below.

$$C_{NB} = argmax_{C_k \in C} P(C_k|Doc)$$

By Bayes' theorem [18], the $P(C_k|Doc)$ can be represented as

$$P(C_k|Doc) = \frac{P(Doc|C_k)P(C_k)}{\sum_{C_i \in C} P(Doc|C_i)P(C_i)}$$

Where $P(C_k) = |C_k| / \sum_{C_i \in C} |C_i|$ is the probability of the class C_k, and $|C_k|$ is the number of training documents in class C_k.

To estimate $P(Doc|C_k)$ is difficult since it is impossible to collect a sufficiently large number of training examples to estimate this probability without prior knowledge or further assumptions. However, the estimation become possible due to the assumption that a word's(term) occurrence is dependent on the class the document comes from, but that it occurs independently of the other words(terms) in the document. Therefore, the $P(Doc|C_k)$ can be written as follows [19]:

$$P(Doc|C_k) = \prod_{j=1}^{|Doc|} P(t_j|C_k)$$

where $|Doc|$ is the number of words (terms) in document Doc, and $P(t_j|C_k)$ is the conditional probability of t_j given Class C_k. Given the term $T = (t_1, t_2, \ldots, t_n)$ that describe the document Doc, the estimation of $P(Doc|C_k)$ is reduce to estimating each $P(t_j|C_k)$ independently. Notice above equation works well when every term appears in every document; otherwise, the product becomes 0 when some terms do not appear in that document. We use the following to approximate $P(t_j|C_k)$ to avoid the possibility that the product becomes 0, and still keeps the meaning of the equation.

$$P(t_j|C_k) = \frac{1 + TF(t_j, C_k)}{|T| + \sum_j^{|T|} TF(t_j, C_k)}$$

where $TF(t_j, C_k)$ is the frequency of term t_j in documents having class value k, $|T|$ is the number of all distinct terms used in the domain of document representation. The formula used to predict probability of class value C_k for a given document Doc is as the following :

$$P(C_k|Doc) = \frac{P(C_k) \prod_{t_j \in Doc} P(t_j|C_k)^{TF(t_j, Doc)}}{\sum_i P(C_i) \prod_{t_j \in Doc} P(t_j|C_i)^{TF(t_j, Doc)}}$$

5 Experimental Results

Our experiment use one year news, 1991/1/1 to 1991/12/31, which consists of 73,420 news articles, with 23,680,756 characters as training data . We use news from 1992/1/1 to 1992/1/7

	Training : 1991/1/1-1991/12/31 (12 months)		
	Testing : 1992/1/1-/1/7 (7 days)		
		#Train	#Test
	CNA News Group	1/1-12/31	1/1-1/7
1.政治	cna.politics.*	23516	422
2.經濟	cna.economics.*	10160	219
3.交通	cna.transport.*	3423	70
4.文教	cna.edu.*	6064	94
5.體育	cna.l*	4929	73
6.社會	cna.judiciary.*	5679	107
7.股市	cna.stock.*	3313	42
8.軍事	cna.military.*	4646	79
9.農業	cna.argriculture.*	3217	54
10.宗教	cna.religion.*	1315	22
11.財政	cna.finance.*	3622	59
12.社福	cna.health-n-welfare.*	3536	66
	Total	**73420**	1307
23680756 Characters => 322.5 Characters/per News			

Table 2: CNA News : Training&Testing

as testing data. Table 2 summarizes the training and testing data.

We first compare four methods, *OR, IG, CHI and MI* [15, 35] without combining distributional clustering. All methods compute scores to all terms and terms are selected according to their scores. Let the *top k measure* denote the percentage of the correct class is in the first k classes when all the classes are sorted according to their probabilities computed by the naive Bayes classifier. Namely, the top 1 measure denotes the percentage that the news are assigned to their pre-defined classes. Notice that the top k measure will be very meaningful in a semi-automatic system when the number of classes is large as it can quickly identify the most possible k classes. Let the *HitAvg* denote the average number of the selected terms been found in testing news and use to see the popularity of selected terms. Let the *Macro Accuracy* denote the average of the accuracy of each class, and the *Variance of Accuracy* denote the variance of the accuracy of each class. Notice that Macro Accuracy and Variance of Accuracy are used to inspect the variation of accuracy between each class. The less value of Variance of Accuracy is, the less difference of classification accuracy between each class

is.

Table 3 shows that the accuracy of top 1 measure of the CHI method changes from 69.17% to 77.35% as the number of selected terms from each class increases from 100 to 5000. The performance of the IG method is similar to the performance of the CHI method. The HitAvg of IG and CHI are 39.02 and 25.35 respectively when the number of selected terms from each class is 1000. This indicates that IG prefers terms with high term frequency. Notice that the accuracy of top 2 measure of CHI is about 90% and is very meaningful in a semi-automatic system. In Table 3 CHI performs the best and achieves 77.35% accuracy in top 1 measure when the number of selected terms from each class is 5000. Both the performance of OR and MI are worse than CHI because both of them prefer to select terms whose term frequencies are low. This can be observed from their low HitAvg, and is consistent with previous theoretic assumption in section 3.1 and 3.2.

Term clustering can reduce the dimension of term space by clustering similar terms into the same group. In addition, redundant substrings and their original strings will be clustered into the same group. This compensates the weakness of term extraction methods which do not remove all redundant substrings. In Table 4, substrings "二屆國", "二屆國代" and "二屆國代選舉" are clustered into group 12; "交易所", "券交易所" and "證券交易所" are clustered into group 300. Furthermore, performance may be increased by clustering when training data is sparse because averaging statistics for similar words together can result in more robust estimates. In Table 4, similar terms, "旅行業"(a travel agent) and "旅行協會"(travel agency association) are clustered together into group 100;" 交響樂團"(a philharmonic orchestra), "巡迴演出"(a show on tour) and "演奏"(to perform) are clustered in group 207;" 犯案"(to commit a crime), "刑事警察"(penal police), "看守所"(a jailer's room) and 槍枝(firearms) are clustered into group 225.

Table 5 shows the difference among different number of selected terms when the number of term groups is fixed at 120. In Table 5, the accuracy of top 1 measure increases as the number of selected terms increases for all term selection methods. When the number of

			Micro Accuracy					
The number of selected terms from each class	The number of total selected terms	Feature Selection Method	Top1	Top2	Top3	HitAvg	Macro Accuracy	Variance of Accuracy
100	1200	OR	50.73	64.50	70.08	1.02	39.21	718.23
100	1200	IG	67.64	87.45	92.81	13.01	68.82	346.01
100	1195	CHI	69.17	86.92	91.58	9.49	68.09	329.17
100	1200	MI	37.49	54.25	61.29	0.24	18.60	616.76
500	6000	OR	62.43	74.75	79.57	2.76	56.73	470.21
500	6000	IG	72.53	89.21	94.41	28.74	74.15	214.42
500	5939	CHI	74.22	91.58	95.10	18.97	73.52	231.13
500	6000	MI	47.28	66.11	72.07	1.13	38.61	432.53
1000	12000	OR	66.03	77.43	82.17	4.04	61.23	370.12
1000	12000	IG	74.22	89.82	94.19	39.02	74.89	207.25
1000	11821	CHI	74.45	91.20	95.26	25.35	75.13	170.24
1000	12000	MI	57.23	74.98	80.49	2.30	54.89	443.64
2000	24000	OR	69.01	79.72	85.77	6.32	66.04	253.29
2000	24000	IG	73.83	90.13	95.26	49.43	75.44	163.70
2000	23513	CHI	75.82	91.51	95.26	32.31	76.81	126.21
2000	24000	MI	64.04	79.19	84.77	4.38	64.39	313.37
5000	59921	OR	74.60	86.46	91.66	16.23	74.44	166.95
5000	60000	IG	75.06	90.36	94.95	62.73	76.10	130.04
5000	57482	CHI	77.35	91.43	95.10	44.01	77.74	123.11
5000	59914	MI	73.53	85.54	91.58	14.59	73.57	214.06

Table 3: Feature Selection Comparison : Testing News(1992/1/1-1992/1/7)

	Group ID				
	12	100	207	225	300
1	二屆國	公路和	交響	犯案	今天在東京
2	二屆國代	在交通	交響樂團	刑事警察	交易所
3	二屆國代選舉	的快樂	巡迴演出	在逃	券交易所
4	的候選人	的班	的音樂	收押	證券交易所
5	候選人	旅行業	的舞	判處死刑	
6	候選人的	旅客的	委會	官認為	
7	國大代表	旅遊協會	國立藝	押回	
8	國代候選人	泰航	國樂	前科	
9	國代選舉	機票	演委	看守所	
10			演委會	書指出	
11			舞蹈	處死刑	
12			樂家	被告	
13			樂團	槍枝	
14			鋼琴	辦案	
15			藝術學	警方在	

Table 4: Term clustering Examples

terms selected from each class is 5000, the accuracy of top 1 measure of IG and CHI are 77.51% and 76.89% respectively. Compared with the accuracy of top 1 measure in Table 3, we find that we can reduce the dimension of term space from 60000 to 120 while the loss of accuracy is less than 1%.

Table 6 shows the difference among different number of term groups when the number of the selected terms from each class is fixed at 1000. The accuracy of top 1 measure of CHI ranges from 74.06% to 75.29% when the number of term groups changes from 60 to 1200. From this observation, we believe that the accuracy is not influenced significantly by the dimension of term space unless the number of term groups is very small(say,12).

				Micro Accuracy					
The number of selected terms from each class	The number of total selected terms	The number of groups	Feature Selection Method	Top1	Top2	Top3	HitAvg	Macro Accuracy	Variance of Accuracy
100	1200	120	OR	50.73	64.04	70.01	1.02	39.21	718.23
100	1200	120	IG	66.41	87.22	92.12	13.01	68.48	377.49
100	1195	120	CHI	69.55	86.76	91.43	9.49	67.68	351.75
100	1200	120	MI	37.34	54.25	61.51	0.24	18.67	603.88
500	6000	120	OR	62.36	73.60	78.96	2.76	56.61	471.53
500	6000	120	IG	72.07	88.60	92.96	28.79	74.46	183.69
500	5939	120	CHI	74.22	90.51	94.03	18.97	73.31	225.94
500	6000	120	MI	46.67	65.42	71.92	1.13	38.64	419.26
1000	12000	120	OR	66.64	77.35	82.25	4.04	61.52	354.31
1000	12000	120	IG	73.64	89.36	93.19	39.02	75.18	149.71
1000	11821	120	CHI	74.22	90.51	94.57	25.35	74.54	186.58
1000	12000	120	MI	56.47	74.52	80.72	2.30	54.47	435.64
2000	24000	120	OR	68.78	80.59	85.77	6.32	65.49	261.91
2000	24000	120	IG	75.06	89.98	94.19	49.43	76.45	124.64
2000	23513	120	CHI	75.44	91.35	95.26	32.31	75.81	129.89
2000	24000	120	MI	64.19	78.50	84.24	4.38	65.31	269.52
5000	59921	120	OR	74.98	88.14	92.12	16.23	71.02	314.07
5000	60000	120	IG	77.51	90.82	94.72	62.73	76.47	132.35
5000	57482	120	CHI	76.89	91.43	94.95	44.01	76.43	126.65
5000	59914	120	MI	66.72	81.71	89.82	14.59	72.14	130.21

Table 5: Term clustering comparison : 120 groups

			Micro Accuracy					
The number of total selected terms	The number of groups	Feature Selection Method	Top1	Top2	Top3	HitAvg	Macro Accuracy	Variance of Accuracy
12000	12	OR	62.51	75.36	81.41	4.04	58.48	506.62
12000	60	OR	66.41	77.43	82.25	4.04	61.40	352.57
12000	120	OR	66.64	77.35	82.25	4.04	61.52	354.31
12000	600	OR	66.49	77.20	81.94	4.04	61.30	358.29
12000	1200	OR	66.11	77.28	81.87	4.04	61.22	363.81
12000	12	IG	70.39	85.00	91.20	39.02	69.99	267.81
12000	60	IG	71.46	88.60	93.27	39.02	73.64	146.79
12000	120	IG	73.64	89.36	93.19	39.02	75.18	149.71
12000	600	IG	73.91	89.82	93.88	39.02	74.89	172.34
12000	1200	IG	74.37	89.90	94.03	39.02	74.44	181.35
11821	12	CHI	70.54	87.15	92.58	25.35	69.53	374.38
11821	60	CHI	74.06	89.90	94.34	25.35	74.00	164.21
11821	120	CHI	74.22	90.51	94.57	25.35	74.54	186.58
11821	600	CHI	74.06	91.20	95.03	25.35	74.38	191.07
11821	1200	CHI	75.29	91.20	95.64	25.35	75.72	166.63
12000	12	MI	53.25	68.86	75.98	2.30	49.15	713.99
12000	60	MI	56.54	73.68	80.18	2.30	55.24	423.26
12000	120	MI	56.47	74.52	80.72	2.30	54.47	435.64
12000	600	MI	56.31	74.45	80.57	2.30	54.29	446.19
12000	1200	MI	56.08	74.29	80.49	2.30	54.16	453.86

Table 6: Term clustering comparison : 1000 Terms selected from each class

6 Conclusions

In this paper, we sketch an implementation of approaches that can handle large amount of training data such as several years of news articles, and automatically assign predefined class to Chinese free text documents. We implement a SB-tree-based approach to extract terms from the original text data, and develop a simple approach to remove redundant subtrings. We also compare four term selection methods combined with distributional clustering and use the naive Bayes classifier to evaluate their performance. In our experiments Information Gain(IG) and χ^2 statistic(CHI) achieved better performance than Odd Ratio(OR) and Mutual Information(MI). With proper term selection and clustering methods, the dimension of term space can be reduced from 60000 to 120 while the loss of classification accuracy is less than 1%.

Acknowledgment. We would like to thank Dr.Chien Lee-Feng and Prof.Tseng Yuen-Hsien for many valuable discussions and comments during this research, and Mr. Lee Min-Jer for kindly help to gather the CNA news.

References

[1] Chidanand Apte, Fred Damerau, and Sholom M. Weiss. Towards language independent automated learning of text categorization methods. In *SIGIR 94*, 1994.

[2] L.Douglas Baker and Andrew Kachites McCallum. Distributional clustering of words for text classification. In *SIGIR 98*, 1998.

[3] Aitao Chen, Jianzhang He, and Liangjie Xu. Chinese text retrieval without using a dictionary. In *SIGIR 97*, 1997.

[4] Chun-Liang Chen, Bo-Ren Bai, Lee-Feng Chien, and Lin-Shan Lee. Cpat-tree-based language models with an application for text verification in chinese. In *Research on Computational Linguistics Conference(ROCLING XI)*, 1998.

[5] Lee-Feng Chien. Pat-tree-based keyword extraction for chinese information retrieval. In *SIGIR 97*, 1997.

[6] Lee-Feng Chien, Min-Jer Lee, and Hsiao-Tieh Pu. Improvements of natural language modeling approaches with information retrieval techniques and internet resources. In *Information Retrieval with Asian Languages(IRAL 1997)*, 1997.

[7] Willian W. Cohen and Yoram Singer. Context-sensitive learning methods for text categorization. In *SIGIR 96*, 1996.

[8] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms.* McGraw-Hill Book Company, 1990.

[9] David D.Lewis and William A. Gale. A sequential algorithm for training text classifier. In *SIGIR 94*, 1994.

[10] David D.Lewis and Marc Ringuette. A comparison of two learning algorithm for text categorization. In *3rd Annual Symosium on Document Analysis and Information Retrieval*, 1994.

[11] Pedro Domingos and Michael Pazzani. On the optimality of the simple bayesain classifier under zero-one loss. In *Machine Learning*, 1997.

[12] Brian S Everitt. *Cluster Analysis.* Halsted Press, New York, third edition edition, 1993.

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

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

[15] Marko Grobelink and Dunja Mladenic. Feature selection for classification based on text hierarchy. In *Conference on Automated Learning and Discovery CONALD-98*, 1998.

[16] Dan Gusfield. *Algorithms on Strings, Trees, and Sequences : computer science and computational biology.* Cambridge University Press, 1997.

[17] Rainer Hoch. Using IR techniques for text classification in document analysis. In *SIGIR 94*, 1994.

[18] M. James. *Classification Algorithms.* Wiley, 1985.

[19] Thorsten Joachims. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML'97)*, 1997.

[20] Leonard Kaufman and Peter J. Rousseeuw. *Finding Groups in Data Analysis : An Introduction to Cluster Analysis.* John Wiley and Sons,Inc., New York, 1990.

[21] K.L Kwok. Comparing representations in chinese information retrieval. In *SIGIR 97*, 1997.

[22] Wai Lam. Using a generalized instance set for automatic text categorization. In *SIGIR 98*, 1998.

[23] Leah S. Larkey and W. Bruce Croft. Combining classifiers in text categorization. In *SIGIR 96*, 1996.

[24] David D. Lewis, Robert E. Schapire, James P. Callan, and Ron Papka. Training algorithms for linear text classifiers. In *SIGIR 96*, 1996.

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

[26] Tom M. Mitchell. *Machine Learning.* The McGraw-Hill Companies, Inc, 1997.

[27] Mehran Sahami, Marti Hearst, and Eric Saund. Applying the multiple cause mixture model to text categorization. In *Machine Learning: Proc. of the 13th International Conference*, 1996.

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

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

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

[31] Shih-Hung Wu, Pey-Ching Yang, and Von-Wun Soo. An assessment of character-based chinese news filtering using latent semantic indexing. *Computational Linguistics and Chinese Language Processing*, 3(2):61–78, 1998.

[32] Yiming Yang. Effective and efficient learning from human decisions in text categorization and retrieval. In *SIGIR 94*, 1994.

[33] Yiming Yang. Noise reduction in a statistical approach to text categorization. In *SIGIR 95*, 1995.

[34] Yiming Yang and Xin Liu. A re-examination of text categorization methods. In *SIGIR 99*, 1999.

[35] Yiming Yang and Jan O. Pedersen. A comparative study on feature in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML'97)*, 1998.

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

[37] Ogawa Yasushi and Matsuda Toru. Overlapping statistical word indexing : A new indexing method for japaness text. In *SIGIR 97*, 1997.

[38] Bo-Hyun Yun, Min-Jeung Cho, and Hae-Chang Rim. Korean information retrieval model based on the principle of word formation. In *Information Retrieval with Asian Languages(IRAL 1997)*, 1997.