# Corpus-Based Learning of Compound Noun Indexing *

**Byung-Kwan Kwak,**
**Jee-Hyub Kim,**
and **Geunbae Lee**†
NLP Lab., Dept. of CSE
Pohang University of
Science & Technology
(POSTECH)
{nerguri,gblee}@postech.ac.kr

**Jung Yun Seo**
NLP Lab.,
Dept. of Computer Science
Sogang University
seojy@ccs.sogang.ac.kr

## Abstract

In this paper, we present a corpus-based learning method that can index diverse types of compound nouns using rules automatically extracted from a large tagged corpus. We develop an efficient way of extracting the compound noun indexing rules automatically and perform extensive experiments to evaluate our indexing rules. The automatic learning method shows about the same performance compared with the manual linguistic approach but is more portable and requires no human efforts. We also evaluate the seven different filtering methods based on both the effectiveness and the efficiency, and present a new method to solve the problems of compound noun over-generation and data sparseness in statistical compound noun processing.

## 1 Introduction

Compound nouns are more specific and expressive than simple nouns, so they are more valuable as index terms and can increase the precision in search experiments. There are many definitions for the compound noun which cause ambiguities as to whether a given continuous noun sequence is a compound noun or not. We, therefore, need a clean definition of compound nouns in terms of information retrieval, so we define a compound noun as "any continuous noun sequence that appears frequently in documents." [1]

In Korean documents, compound nouns are represented in various forms (shown in Table 1), so there is a difficulty in indexing all types of compound nouns. Until now, there have been much works on compound noun indexing, but they still have limitations of covering all types of compound nouns and require much linguistic knowledge to accomplish this goal. In this paper, we propose a corpus-based learning method for compound noun indexing which can extract the rules automatically with little linguistic knowledge.

Table 1: Various types of Korean compound noun with regard to "jeong-bo geom-saeg (information retrieval)"

| |
|---|
| jeong-bo-geom-saeg (information-retrieval) |
| jeong-bo-eui geom-saeg (retrieval of information) |
| jeong-bo geom-saeg (information retrieval) |
| jeong-bo-leul geom-saeg-ha-neun (retrieving information) |
| jeong-bo-geom-saeg si-seu-tem (information-retrieval system) |

As the number of the documents is growing retrieval, efficiency also becomes as important as effectiveness. To increase the efficiency, we focus on reducing the number of indexed spurious compound nouns. We perform experiments on several filtering methods to find the algorithm that can reduce spurious compound nouns most efficiently.

* This research was supported by KOSEF special purpose basic research (1997.9 - 2000.8 #970-1020-301-3)

† Corresponding author

[1] The frequency threshold can be adjusted according to application systems.

The remainder of this paper is organized as follows. Section 2 describes previous compound noun indexing methods for Korean and compound noun filtering methods. We show overall compound noun indexing system architecture in Section 3, and explain each module of the system in Section 4 and 5 in detail. We evaluate our method with standard Korean test collections in Section 6. Finally, concluding remarks are given in Section 7.

## 2 Previous Research

### 2.1 Compound Noun Indexing

There have been two different methods for compound noun indexing: statistical and linguistic. In one statistical method, (Fagan, 1989) indexed phrases using six different parameters, including information on co-occurrence of phrase elements, relative location of phrase elements, etc., and achieved reasonable performance. However, his method couldn't reveal consistent substantial improvements on five experimental document collections in effectiveness. (Strzalkowski et al., 1996; Evans and Zhai, 1996) indexed subcompounds from complex noun phrases using noun-phrase analysis. These methods need to find the head-modifier relations from noun phrases and therefore require difficult syntactic parsing in Korean.

For Korean, in one statistical method, (Lee and Ahn, 1996) indexed general Korean nouns using n-grams without linguistic knowledge and the experiment results showed that the proposed method might be almost as effective as the linguistic noun indexing. However, this method can generate many spurious n-grams which decrease the precision in search performance. In linguistic methods, (Kim, 1994) used five manually chosen compound noun indexing rule patterns based on linguistic knowledge. However, this method cannot index the diverse types of compound nouns. (Won et al., 2000) used a full parser and increased the precision in search experiments. However, this linguistic method cannot be applied to unrestricted texts robustly.

In summary, the previous methods, whether they are statistical or linguistic, have their own shortcomings. Statistical methods require significant amounts of co-occurrence information for reasonable performance and can not index the diverse types of compound nouns. Linguistic methods need compound noun indexing rules described by human and sometimes result in meaningless compound nouns, which decreases the performance of information retrieval systems. They cannot also cover the various types of compound nouns because of the limitation of human linguistic knowledge.

In this paper, we present a hybrid method that uses linguistic rules but these rules are automatically acquired from a large corpus through statistical learning. Our method generates more diverse compound noun indexing rule patterns than the previous standard methods (Kim, 1994; Lee et al., 1997), because previous methods use only most general rule patterns (shown in Table 2) and are based solely on human linguistic knowledge.

Table 2: Typical hand-written compound noun indexing rule patterns for Korean

| |
|---|
| Noun without case makers / Noun |
| Noun with a genitive case maker / Noun |
| Noun with a nominal case maker or<br>an accusative case maker /<br>Verbal common noun or adjectival common noun |
| Noun with an adnominal ending / Noun |
| Noun within predicate particle phrase / Noun |

(The two nouns before and after a slash in the pattern can form a single compound noun.)

### 2.2 Compound Noun Filtering

Compound noun indexing methods, whether they are statistical or linguistic, tend to generate spurious compound nouns when they are actually applied. Since an information retrieval system can be evaluated by its effectiveness and also by its efficiency (van Rijsbergen, 1979), the spurious compound nouns should be efficiently filtered. (Kando et al., 1998) insisted that, for Japanese, the smaller the number of index terms is, the better the performance of the information retrieval system should be.

For Korean, (Won et al., 2000) showed that segmentation of compound nouns is more efficient than compound noun synthesis in search performance. There have been many works on compound noun filtering methods; (Kim, 1994) used mutual information only, and (Yun et al., 1997) used mutual information and relative frequency of POS (Part-Of-Speech) pairs together. (Lee et al., 1997) used stop word dictionaries which were constructed manually. Most of the previous methods for compound noun filtering utilized only one consistent method for generated compound nouns irrespective of the different origin of compound noun indexing rules, and the methods cause many problems due to data sparseness in dictionary and training data. Our approach solves the data sparseness problem by using co-occurrence information on automatically extracted compound noun elements together with a statistical precision measure which fits best to each rule.

## 3 Overall System Architecture

The compound noun indexing system proposed in this paper consists of two major modules: one for automatically extracting compound noun indexing rules (in Figure 1) and the other for indexing documents, filtering the automatically generated compound nouns, and weighting the indexed compound nouns (in Figure 2).

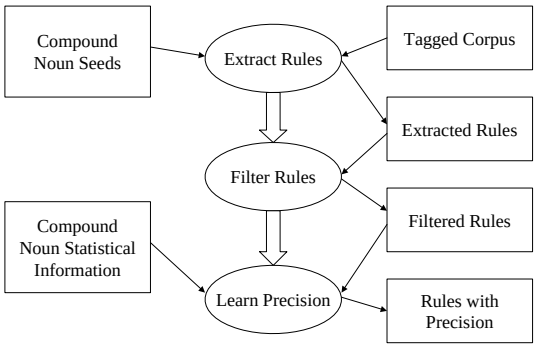


Figure 1: Compound noun indexing-rule extraction module (control flow ⇒, data flow →)

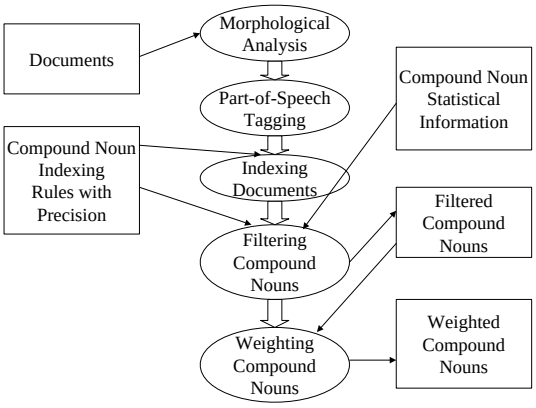


Figure 2: Compound noun indexing, filtering, and weighting module (control flow ⇒, data flow →)

## 4 Automatic Extraction of Compound Noun Indexing Rules

There are three major steps in automatically extracting compound noun indexing rules. The first step is to collect compound noun statistical information, and the second step is to extract the rules from a large tagged corpus using the collected statistical information. The final step is to learn each rule's precision.

### 4.1 Collecting Compound Noun Statistics

We collect initial compound noun seeds which were gathered from various types of well-balanced documents such as ETRI Kemong encyclopaedia[2] and many dictionaries on the Internet, and we collected 10,368 seeds, as shown in Table 3. The small number of seeds are bootstrapped to extract the compound noun indexing rules for various corpora.

Table 3: Collected compound noun seeds

| No. of component elements | 2 | 3 | Total |
|---|---|---|---|
| ETRI Kemong encyclomedia | 5,100 | 2,088 | 7,188 |
| Internet dictionaries | 2,071 | 1,109 | 3,180 |

To collect more practical statistics on the compound nouns, we made a 1,000,000 eojeol(Korean spacing unit which corresponds

[2] Courteously provided by ETRI, Korea.

to an English word or phrase) tagged corpus for a compound noun indexing experiment from a large document set (Korean Information Base). We collected complete compound nouns (a continuous noun sequence composed of at least two nouns on the condition that both the preceding and the following POS of the sequence are not nouns (Yoon et al., 1998)) composed of 1 - 3 nouns from the tagged training corpus (Table 4).

Table 4: Statistics for complete compound nouns

| No. of component elements | 1 | 2 | 3 |
|---|---|---|---|
| Vocabulary | 264,359 | 200,455 | 63,790 |

### 4.2 Extracting Indexing Rules

We define a template (in Table 5) to extract the compound noun indexing rules from a POS tagged corpus.

The template means that if a front-condition-tag, a rear-condition-tag, and sub-string-tags are coincident with input sentence tags, the lexical item in the synthesis position of the sentence can be indexed as a compound noun as "x / y (for 3-noun compounds, x / y / z)". The tags used in the template are POS (Part-Of-Speech) tags and we use the POSTAG set (Table 17).

The following is an algorithm to extract compound noun indexing rules from a large tagged corpus using the two-noun compound seeds and the template defined above. The rule extraction scope is limited to the end of a sentence or, if there is a conjunctive ending (eCC) in the sentence, only to the conjunctive ending of the sentence. A rule extraction example is shown in Figure 3.

Table 5: The template to extract the compound noun indexing rules

| |
|---|
| front-condition-tag \| |
| sub-string-tags (tag 1 tag 2 ... tag n-1 tag n) \| |
| rear-condition-tag \| |
| synthesis locations (x y) |
| → |
| lexicon x / lexicon y |
| (for 3-noun compounds, |
| synthesis locations (x, y, z) |
| → |
| lexicon x / lexicon y / lexicon z) |

Algorithm 1: Extracting compound noun indexing rules (for 2-noun compounds)

```
Read Template
Read Seed
  (Consist of Constituent 1 / Constituent 2)
Tokenize Seed into Constituents
Put Constituent 1 into Key1 and Constituent 2
  into Key2
While (Not(End of Documents))
{
  Read Initial Tag of Sentence
  While (Not(End of Sentence or eCC))
  {
    Read Next Tag of Sentence
    If (Read Tag == Key1)
      {
        While (Not(End of Sentence or eCC))
        {
          Read Next Tag of Sentence
          If (Current Tag == Key2)
            Write Rule according
              to the Template
        }
      }
  }
}
```

The next step is to refine the extracted rules to select the proper ones. We used a rule filtering algorithm (Algorithm 2) using the frequency together with the heuristics that the rules with negative lexical items (shown in Table 6) will make spurious compound nouns.

Algorithm 2: Filtering extracted rules using frequency and heuristics

```
1. For each compound noun seed, select
   the rules whose frequency is greater than 2.
2. Among rules selected by step 1, select
   only rules that are extracted
   at least by 2 seeds.
3. Discard rules which contain
   negative lexical items.
```

Table 6: Negative lexical item examples

| negative items (tags) | example phrases |
|---|---|
| je-oe(MC) (exclude) | no-jo-leul je-oe-han hoe-eui (meeting excluding union) |
| eobs(E) (not exist) | sa-gwa-ga eobs-neun na-mu (tree without apple) |
| mos-ha(D) (can not) | dog-lib-eul mos-han gug-ga (country that cannot be liberated) |

We automatically extracted and filtered out

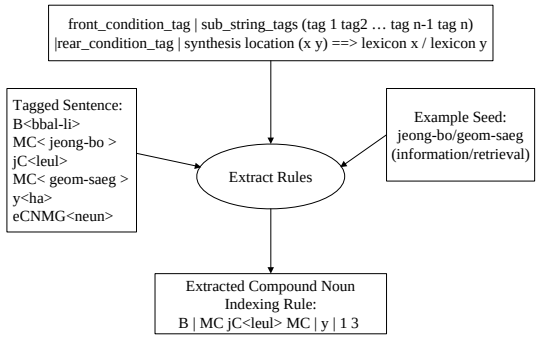


Figure 3: Rule Extraction Process Example

2,036 rules from the large tagged corpus (Korean Information Base, 1,000,000 eojeol) using the above Algorithm 2. Among the filtered rules, there are 19 rules with negative lexical items and we finally selected 2,017 rules. Table 7 shows a distribution of the final rules according to the number of elements in their sub-string-tags.

Table 7: Distribution of extracted rules by number of elements in sub-string-tags

| No. | Distribution | Example |
|---|---|---|
| 2 tags | 79.6 % | MC MC |
| 3 tags | 12.6 % | MC jO⟨eui⟩ MC |
| 4 tags | 4.7 % | MC y eCNMG MC |
| 5 tags | 1.5 % | MC MC jO⟨e⟩ DI⟨sog-ha-neun⟩ MC |
| over 6 tags | 1.6 % | |

The automatically extracted rules have more rule patterns and lexical items than human-made rules so they can cover more diverse types of compound nouns (Table 8). When checking the overlap between the two rule collections, we found that the manual linguistic rules are a subset of our automatically generated statistical rules. Table 9 shows some of the example rules newly generated from our extraction algorithm, which were originally missing in the manual rule patterns.

### 4.3 Learning the Precision of Extracted Rules

In the proposed method, we use the precision of rules to solve the compound noun overgeneration and the data sparseness problems. The precision of a rule can be defined by

Table 8: Comparison between the automatically extracted rules and the manual rules

| Method | No. of general rule patterns | No. of lexical terms used in rule patterns |
|---|---|---|
| Manual linguistic method | 5 | 16 |
| Our method | 23 | 78 |

Table 9: Examples of newly added rule patterns

| Rule |
|---|
| Noun + bound noun / Noun |
| Noun + suffix / Noun |
| Noun + suffix + assignment verb + adnominal ending / Noun |

counting how many indexed compound noun candidates generated by the rule are actual compound nouns:

$$Prec(rule) = \frac{N_{actual}}{N_{candidate}}$$

where $Prec(rule)$ is the precision of a rule, $N_{actual}$ is the number of actual compound nouns, and $N_{candidate}$ is the number of compound noun candidates generated by the automatic indexing rules.

To calculate the precision, we need a defining measurement for compound noun identification. (Su et al., 1994) showed that the average mutual information of a compound noun tends to be higher than that of a non-compound noun, so we try to use the mutual information as the measure for identifying the compound nouns. If the mutual information of the compound noun candidate is higher than the average mutual information of the compound noun seeds, we decide that it is a compound noun. For mutual information (MI), we use two different equations: one for two-element compound nouns (Church and Hanks, 1990) and the other for three-element compound nouns (Su et al., 1994). The equation for two-element compound nouns is as follow:

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

where $x$ and $y$ are two words in the corpus, and $I(x;y)$ is the mutual information of these two words (in this order). Table 10 shows the average MI value of the two and three elements.

Table 10: Average value of the mutual information (MI) of compound noun seeds

| Number of elements | 2 | 3 |
|---|---|---|
| Average MI | 3.56 | 3.62 |

The MI was calculated from the statistics of the complete compound nouns collected from the tagged training corpus (see Section 4.1). However, complete compound nouns are continuous noun sequences and cause the data sparseness problem. Therefore, we need to expand the statistics. Figure 4 shows the architecture of the precision learning module by expanding the statistics of the complete compound nouns along with an algorithmic explanation (Algorithm 3) of the process. Table 11 shows the improvement in the average precision during the repetitive execution of this learning process.

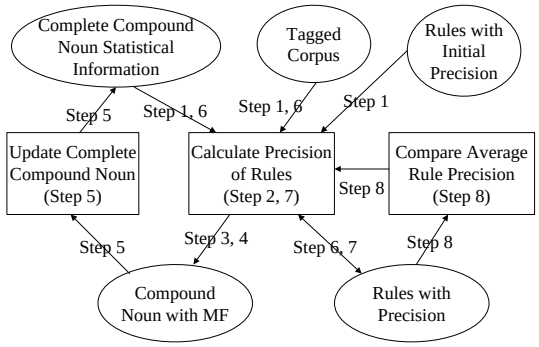


Figure 4: Learning the precision of the compound noun indexing rules (The steps are shown in Algorithm 3)

Algorithm 3:

1. Calculate all rules' initial precision using initial complete compound noun statistical information.
2. Calculate the average precision of the rules.
3. Multiply a rule's precision by the frequency of the compound noun made by the rule. We call this value the modified frequency (MF).
4. Collect the same compound nouns, and sum all the modified frequencies for each compound noun.
5. If the summed modified frequency is greater than a threshold, add this compound noun to the complete compound noun statistical information.
6. Calculate all rules' precision again using the changed complete compound noun statistical information.
7. Calculate the average precision of the rules.
8. If the average precision of the rules is equal to the previous average precision, stop. Otherwise, go to step 2.

Table 11: Improvement in the average precision of rules

| Learning cycles | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Avg. prec. of rules | 0.19 | 0.23 | 0.39 | 0.44 | 0.45 | 0.45 |

## 5 Compound Noun Indexing, Filtering, and Weighting

In this section, we explain how to use the automatically extracted rules to actually index the compound nouns, and describe how to filter and weight the indexed compound nouns.

### 5.1 Compound Noun Indexing

To index compound nouns from documents, we use a natural language processing engine, SKOPE (Standard KOrean Processing Engine) (Cha et al., 1998), which processes documents by analysing words into morphemes and tagging part-of-speeches. The tagging results are compared with the automatically learned compound noun indexing rules and, if they are coincident with each other, we index them as compound nouns. Figure 5 shows a process of the compound noun indexing with an example.

### 5.2 Compound Noun Filtering

Among the indexed compound nouns above, still there can be meaningless compound nouns, which increases the number of index terms and the search time. To solve compound noun over-generation problem, we experiment with seven different filtering methods (shown in Table 12) by analyzing their

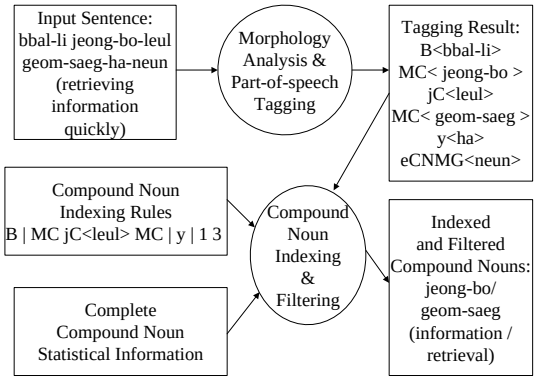


Figure 5: Compound noun indexing process

relative effectiveness and efficiency, as shown in Table 16. These methods can be divided into three categories: first one using MI, second one using the frequency of the compound nouns (FC), and the last one using the frequency of the compound noun elements (FE). MI (Mutual Information) is a measure of word association, and used under the assumption that a highly associated word n-gram is more likely to be a compound noun. FC is used under the assumption that a frequently encountered word n-gram is more likely to be a compound than a rarely encountered n-gram. FE is used under the assumption that a word n-gram with a frequently encountered specific element is more likely to be a compound. In the method of C, D, E, and F, each threshold was decided by calculating the average number of compound nouns of each method.

Table 12: Seven different filtering methods

| | |
|---|---|
| (MI) | A. Mutual information of compound noun elements (0) |
| (MI) | B. Mutual information of compound noun elements (average of MI of compound noun seeds) |
| (FC) | C. Frequency of compound nouns in the training corpus (4) |
| (FC) | D. Frequency of compound nouns in the test corpus (2) |
| (FE) | E. Frequency of compound noun heads in the training corpus (5) |
| (FE) | F. Frequency of compound noun modifiers in the training corpus (5) |
| | G. No filtering |

(The value in parantheses is a threshold.)

Among these methods, method B generated the smallest number of compound nouns best efficiency and showed the reasonable effectiveness (Table 16). On the basis of this filtering method, we develop a smoothing method by combining the precision of rules with the mutual information of the compound noun elements, and propose our final filtering method (H) as follows:

$$T(x,y) = \log_2 \frac{P(x,y)}{P(x) \times P(y)} + \alpha \times Precision$$

where $\alpha$ is a weighting coefficient and *Precision* is the applied rules learned in Section 4.3.

For the three-element compound nouns, the MI part is replaced with the three-element MI equation[3] (Su et al., 1994).

## 6 Experiment Results

To calculate the similarity between a document and a query, we use the p-norm retrieval model (Fox, 1983) and use 2.0 as the p-value. We also use the component nouns in a compound as the indexing terms. We follow the standard TREC evaluation schemes (Salton and Buckley, 1991). For single index terms, we use the weighting method atn.ntc (Lee, 1995).

### 6.1 Compound Noun Indexing Experiments

This experiment shows how well the proposed method can index diverse types of compound nouns than the previous popular methods which use human-generated compound noun indexing rules (Kim, 1994; Lee et al., 1997). For simplicity, we filtered the generated compound nouns using the mutual information of the compound noun elements with a threshold of zero (method A in Table 12).

Table 13 shows that the terms indexed by previous linguistic approach are a subset of the ones made by our statistical approach. This means that the proposed method can cover more diverse compound nouns than the

[3] $$I(x;y;z) = \log_2 \frac{P_D(x,y,z)}{P_I(x,y,z)}$$

Table 13: Compound noun indexing coverage experiment (With a 200,000 eojeol Korean Information Base)

| | Manual linguistic rule patterns | Our automatic rule patterns |
|---|---|---|
| No. of generated actual compound nouns | 22,276 | 30,168 (+35.4 %) |
| No. of generated actual compound nouns without overlap | 0 | 7,892 |

manual linguistic rule method. We perform a retrieval experiment to evaluate the automatically extracted rules. Table 14[4] and table 15[5] show that our method has slightly better recall and 11-point average precision than the manual linguistic rule method.

Table 14: Compound noun indexing effectiveness experiment I

| | Manual linguistic rule patterns | Our automatic rule patterns |
|---|---|---|
| Avg. recall | 82.66 | 83.62 (+1.16 %) |
| 11-pt. avg. precision | 42.24 | 42.33 (+0.21 %) |
| No. of index terms | 504,040 | 515,801 (+2.33 %) |

Table 15: Compound noun indexing effectiveness experiment II

| | Manual linguistic rule patterns | Our automatic rule patterns |
|---|---|---|
| Avg. recall | 86.32 | 87.50 (+1.35 %) |
| 11-pt. avg. precision | 34.33 | 34.54 (+0.61 %) |
| No. of index terms | 1,242,458 | 1,282,818 (+3.15 %) |

[4] With KTSET2.0 test collections (Courteously provided by KT, Korea. (4,410 documents and 50 queries))

[5] With KRIST2.0 test collection (Courteously provided by KORDIC, Korea. (13,514 documents and 30 queries))

### 6.2 Retrieval Experiments Using Various Filtering Methods

In this experiment, we compare the seven filtering methods to find out which one is the best in terms of effectiveness and efficiency. For this experiment, we used our automatic rules for the compound noun indexing, and the test collection KTSET2.0. To check the effectiveness, we used recall and 11-point average precision. To check the efficiency, we used the number of index terms. Table 16 shows the results of the various filtering experiments.

From Table 16, the methods using mutual information reduce the number of index terms, whereas they have lower precision. The reason of this lower precision is that MI has a bias, i.e., scoring in favor of rare terms over common terms, so MI seems to have a problem in its sensitivity to probability estimation error (Yang and Pedersen, 1997). In this experiment[6], we see that method B generates the smallest number of compound nouns (best efficiency) and our final proposing method H has the best recall and precision (effectiveness) with the reasonable number of compound nouns (efficiency). We can conclude that the filtering method H is the best, considering the effectiveness and the efficiency at the same time.

## 7 Conclusion

In this paper, we presented a method to extract the compound noun indexing rules automatically from a large tagged corpus, and showed that this method can index compound nouns appearing in diverse types of documents.

In the view of effectiveness, this method is slightly better than the previous linguistic approaches but requires no human effort.

The proposed method also uses no parser and no rules described by humans, therefore, it can be applied to unrestricted texts very robustly and has high domain porta-

[6] Our Korean NLQ (Natural Language Querying) demo system (located in 'http://nlp.postech.ac.kr/Resarch/POSNLQ/') can be tested.

Table 16: Retrieval experiment results of various filtering methods

| | A | B | C | D | E | F | G | **H** |
|---|---|---|---|---|---|---|---|---|
| Average recall | 83.62 | 83.62 (+0.00) | 83.62 (+0.00) | 83.62 (+0.00) | 83.62 (+0.00) | 83.62 (+0.00) | 84.32 (+0.84) | **84.32 (+0.84)** |
| 11-pt. avg. precision | 42.45 | 42.42 (-0.07) | 42.49 (+0.09) | 42.55 (+0.24) | 42.72 (+0.64) | 42.48 (+0.07) | 42.48 (+0.07) | **42.75 (+0.71)** |
| Precision at 10 Docs. | 52.11 | 52.44 | 52.07 | 52.80 | 52.26 | 51.89 | 52.81 | **52.98** |
| No. of index terms | 515,80 | 508,20 (-1.47) | 514,54 (-0.24) | 547,27 (+6.10) | 572,36 (+10.97) | 574,04 (+11.29) | 705,98 (+36.87) | **509,90 (-1.14)** |

bility. We also presented a filtering method to solve the compound noun over-generation problem. Our proposed filtering method (H) shows good retrieval performance both in the view of the effectiveness and the efficiency.

In the future, we need to perform some experiments on much larger commercial databases to test the practicality of our method.

Finally, our method doesn't require language dependent knowledge, so it needs to be verified whether it can be easily applied to other languages.

## References


Jeongwon Cha, Geunbae Lee, and Jong-Hyeok Lee. 1998. Generalized unknown morpheme guessing for hybrid pos tagging of korean. In *Proceedings of SIXTH WORKSHOP ON VERY LARGE CORPORA in Coling-ACL 98.*

K. W. Church and P. Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29.

David A. Evans and Chengxiang Zhai. 1996. Noun-phrase analysis in unrestricted text for information retrieval. In *Proceedingof the 34th Annual Meetinof the Association for Computational Linguistics, Santa Cruz, CA*, pages 17–24.

Joel L. Fagan. 1989. The effectiveness of a nonsyntactic approach to automatic phrase indexing for document retrieval. *JASIS*, 40(2):115–132.

E. A. Fox. 1983. *Extending the Boolean and Vector Space Models of Information Retrieval with P-norm Queries and Multiple Concept Types.* Ph.D. thesis, Cornell Univ.

Noriko Kando, Kyo Kageura, Masaharu Yoshoka, and Keizo Oyama. 1998. Phrase processing methods for japanase text retrieval. *SIGIR forum*, 32(2):23–28.

Pan Koo Kim. 1994. The automatic indexing of compound words from korean text based on mutual information. *Journal of KISS (in Korean)*, 21(7):1333–1340.

Joon Ho Lee and Jeong Soo Ahn. 1996. Using n-grams for korean text retrieval. In *SIGIR'96*, pages 216–224.

Hyun-A Lee, Jong-Hyeok Lee, and Geunbae Lee. 1997. Noun phrase indexing using clausal segmentation. *Journal of KISS (in Korean)*, 24(3):302–311.

Joon Ho Lee. 1995. Combining multiple evidence from different properties of weighting schemes. In *SIGIR'95*, pages 180–188.

Gerard Salton and Chris Buckley. 1991. Text retrieval conferences evaluation program. In *ftp://ftp.cs.cornell.edu/pub/smart/*, trec_eval.7.0beta.tar.gz.

Tomek Strzalkowski, Louise Guthrie, Jussi Karlgren, Jum Leistensnider, Fang Lin, Jose Perez-Carballo, Troy Straszheim, Jin Wang, and Jon Wilding. 1996. Natural language information retrieval: Trec-5 report. In *The Fifth Text REtrieval conference (TREC-5), NIST Special publication*, pages 500–238.

Keh-Yih Su, Ming-Wen Wu, and Jing-Shin Chang. 1994. A corpus-based approach to automatic compound extraction. In *Proceedings of ACL 94*, pages 242–247.

C. J. van Rijsbergen. 1979. *Information Retrieval.* University of Computing Science, Lodon.

Hyungsuk Won, Mihwa Park, and Geunbae Lee. 2000. Integrated multi-level indexing method for compound noun processing. In *Journal of KISS, 27(1) (in Korean)*, pages 84–95.

Table 17: The POS (Part-Of-Speech) set of POSTAG

| Tag | Description | Tag | Description | Tag | Description |
|---|---|---|---|---|---|
| MC | common noun | MP | proper noun | MD | bound noun |
| T | pronoun | G | adnoun | S | numeral |
| B | adverb | K | interjection | DR | regular verb |
| DI | irregular verb | HR | regular adjective | HI | irregular adjective |
| I | assignment verb | E | existential predicate | jC | case particle |
| jS | auxiliary particle | jO | other particle | eGE | final ending |
| eGS | prefinal ending | eCNDI | aux conj ending | eCNDC | quote conj ending |
| eCNMM | nominal ending | eCNMG | adnominal ending | eCNB | adverbial ending |
| eCC | conjunctive ending | y | predicative particle | b | auxiliary verb |
| + | prefix | - | suffix | su | unit symbol |
| so | other symbol | s‘ | left parenthesis | s’ | right parenthesis |
| s. | sentence closer | s- | sentence connection | s, | sentence comma |
| sf | foreign word | sh | Chinese character | | |


Yiming Yang and Jan O. Pedersen. 1997. A comparative study on feature selection in text categorization. In Douglas H. Fisher, editor, *Proceedings of ICML-97, 14th International Conference on Machine Learning*, pages 412–420, Nashville, US. Morgan Kaufmann Publishers, San Francisco, US.

Jun-Tae Yoon, Eui-Seok Jong, and Mansuk Song. 1998. Analysis of korean compound noun indexing using lexical information between nouns. *Journal of KISS (in Korean)*, 25(11):1716–1725.

Bo-Hyun Yun, Yong-Jae Kwak, and Hae-Chang Rim. 1997. A korean information retrieval model alleviating syntactic term mismatches. In *Proceedings of the Natural Language Processing Pacific Rim Symposium*, pages 107–112.