Compound Noun Decomposition Using a Markov Model

Jongwoo Lee
Department of Computer Engineering
Seoul National University. Korea
jongwoo@nova.snu.ac.kr

Byoung-Tak Zhang
Department of Computer Engineering
Seoul National University, Korea
btzhang@comp.snu.ac.kr

Yung Taek Kim
Department of Computer Engineering
Seoul National University, Korea
ytkim@comp.snu.ac.kr

Abstract

A statistical method for compound noun decomposition is presented. Previous studies on this problem showed some statistical information are helpful. But applying statistical information was not so systemic that performance depends heavily on the algorithm and some algorithms usually have many separated steps. In our work statistical information is collected from manually decomposed compound noun corpus to build a Markov model for composition. Two Markov chains representing statistical information are assumed independent: one for *the sequence of participants' lengths* and another for *the sequence of participants ' features*. Besides Markov assumptions, *least participants preference assumption* also is used. These two assumptions enable the decomposition algorithm to be a kind of *conditional dynamic programming* so that efficient and systemic computation can be performed. When applied to test data of size 5027, we obtained a precision of 98.4%.

1 Introduction

Generally, compound noun takes the form of successive appearance of elementary nouns. Decomposition of compound nouns is an important problem in natural language processing (NLP), especially in oriental languages such as Korean, Chinese, Japanese in that there can be no intervening blank between participant nouns. It is more complicated in Korean case, since most Korean words are represented by phonetic alphabets. To analyze the meaning of a Korean compound noun, its decomposition is necessary.

Only with dictionary lookup, several compositions of a compound noun are possible. Finding all possible candidate compositions [3] is of high cost since the number of compositions increases exponentially in proportion to the length of the compound noun. Although the length of many compound noun is usually manageable, some heuristic search is required in order to avoid full search, for search involves dictionary lookup. Even after all the candidates possible are found, semantically correct one must be selected by some semantic resolver. If heuristics are adopted, selection choices can be made before complete participants of a candidate are searched.

Some heuristics on this problem have been studied before. One of the simplest heuristics is *longest match first heuristics*. With this method, decomposition points are searched from left to right and the next decomposition point is the end point of longest elementary noun starting at the previous decomposition point. On the other hand, there is a different method in which the most probable decomposition point is found in the whole string, and then this decomposition is performed on the substrings recursively [2]. If algorithms like these did not have a backtracking mechanism they would frequently fall in local maxima, resulting in low accuracy. Some flexible heuristics that reduces the number of search paths and partially allows backtracking are studied [5, 2, 9]. But in these approaches models can not completely specify the decomposition process and performance depends heavily on the algorithm implemented and usually have exception handlers or multiple steps.

Many useful statistical information for selection among candidates or the next composition position were proposed as well. These information include mutual information between syllables [2], syllable pattern of com-

pound nouns [9], similarity/disimilarity between term distribution [6], word collocation and mutual information [4], lexical frequencies [3]. Mutual information between syllables is closely related to the syllable composition statistics in a lexicon but rarely represents the semantic relationship among elementary nouns in compound noun. Syllable pattern of compound nouns need some exception handler owing to the variety in the number of participants. Lexical frequencies are not specific to compound nouns. So it may not represent the statistical information of compound nouns. Except when selections among completely decomposed candidates are performed, the statistical information itself works as heuristics for reducing search space, too. All the methods mentioned above require large corpus for collection statistical information. But the sizes of corpus have wide ranges due to which statistical information should be used.
In this paper we proposes *two Markov chains* assumed independent and *least participants preference assumption* both as a heuristics and a selection measure. Each Markov chain represents the sequence tendency in the sense of participant's semantic feature and participant's length respectively. Fundamentally while full search is performed, if assumptions we proposed are satisfied, the premature determination is made so that the whole process takes the form of *conditional dynamic programming scheme*. The basic idea behind this approach is that the participants of compound noun would have some relationship with each other and when given these statistical information, the chances of other factors such as the algorithm could be excluded.

2 Compound Nouns

In our framwork, compound nouns may comprise somewhat involved patterns. It is a string which has no blank character in it. Its components are to be nouns, prefixes, suffixes which can be found in dictionary, and recursively compound nouns. Furthermore, in our definition, compound nouns can be followed by a postposition, which cases show up frequently and are hard to be distinguished from the other cases.
Representing formally in syntactic form, compound noun can be of the following form.

$$\begin{aligned} &\textit{elementary Cnoun} := (\textit{prefix})^*(\textit{noun})^+(\textit{suffix})^* \\ &\textit{Cnoun} := (\textit{elementary Cnoun})^+(\textit{postposition})^* \end{aligned}$$

where every element of the sets *prefix, noun, suffix, postposition* is in dictionary and $(L)^*$ denotes "zero or more concatenations of" L and $(L)^+$ denotes "one or more concatenations of " L. The problem of compound noun decomposition is to select a syntactically and semantically correct sequence of elements in $prefix \bigcup suffix \bigcup noun \bigcup postposition$ given a string $\in Cnoun$ defined above. We will refer the components of a compound noun to its *participants* in the rest of this paper.
For example, if we're given a *Cnoun* string "공무원재임용법안", then after decomposition, "공무원(official)"+"재임용(reappointment)"+"법안(bill)" sequence should be found and selected even though "공무원(official)"+"재임(reappointment)"+"용법(usage)"+"안(inside)" is also a syntactically correct candidate, since the latter is not semantically correct one.

3 Markov Chain Models for Compound Nouns

We suppose that compound nouns have statistical tendencies among their participants. The statistical tendencies considered are *the sequence of participants' semantic features* and *the sequence of participants' lengths*. Once the tendencies have been chosen to be incorporated in Markov chain, some Markov assumptions should be made. We adopted *trigram* assumptions considering the size of training data and the average number of participants in Korean compound nouns. When more than one sequence tendencies are used (in our case, two sequence tendencies), one Markov chain which covers multiple tendencies would have so many states that immense size of learning data is required. Moreover if each tendency has a low dependency on the others, it may cause waste of states in the Markov chain, resulting in lack of consistency. In such a case, multiple Markov chains are considered *independent* and probability can be calculated accommodating this assumption.

Assumption 1 (Trigram Assumption) *For given a participant in a sequence of participants in a compound noun, any other participant but the two left-nearest ones can not have any statistical tendency with that given participant.*

We will define some notations which will be used. S^l is the set of states in a Markov chain for the sequence of participant's length. s_i^l or $s_{(u,v)}^l$ is a state in S^l. If s_i^l is used, l means the participant's sequence index, and if $s_{(u,v)}^l$ is used, u is the previous participant's length and v is the current participant's length, since trigram Markov model is assumed. In the same way, S^f, s_i^f and $s_{(u,v)}^f$ can be defined for the Markov chain of the sequence of participant's semantic feature except that u means the previous participants' semantic features and v means the current participant's semantic feature. In addition, S is defined as $S \equiv S^l \otimes S^f$ and s_i is a state in S.
Given a candidate decomposition of compound noun string $w_1 \cdot w_2 \cdot \cdots w_n$ where w_i is ith participant, there are two most likely state paths $s_1^l, \cdots, s_n^l$ from S^l and $s_1^f, \cdots, s_n^f$ from S^f. The calculation of the probability $P(w_1 \cdot w_2 \cdot \cdots w_n)$ for the selection among candidates decompositions can be, $P(s_1, \cdots, s_n) =$

$P(s_1^l, \cdots, s_k^l)P(s_1^f, \cdots, s_n^f)$ since S^l and S^f are independent each other.
When probability is given this way, the selection among multiple candidate decompositions chooses one having a sequence $w_1, \cdots, w_n$ that maximizes

$$P(w_1, \cdots, w_n) = \begin{array}{l} P(s_1^l)\prod_{i=2}^{n} P(s_i^l|s_{i-1}^l) \\ \cdot P(s_1^f)\prod_{i=2}^{n} P(s_i^f|s_{i-1}^f) \end{array} \quad (1)$$

The semantic feature of a participant can be looked up in dictionary and may be multiple while the length is uniquely determine. If statistics on lexical probability and semantic feature probability on all lexicons are provided, Eq (1) can be multiplied by additional terms:

$$P'(w_1, \cdots, w_n) = P(w_1, \cdots, w_n)\prod_{k=1}^{n} P(w_k) \cdot \prod_{k=1}^{n} P(feature\ of\ w_k = v\ in\ s_{(u,v)}^f) \quad (2)$$

Since we decided the length of participant to be constricted to one of $\{1, 2, 3, 4\ or\ more\}$ and 0 is used for *start* state, the Markov chain for participant's length is composed of ${}_5P_2$ states, namely $\{s_{(u,v)}^l | u, v \in \{0, 1, 2, 3, 4\ or\ more\}\}$. It can be easily understood that the legal directional edges(which can have positive probability) can connect $s_{(i,j)}^l$ to $s_{(j,k)}^l$. In a similar way, the Markov chain for semantic feature have the size of ${}_{12}P_2$, for we use 11 semantic features. When $feature(w_k)$ have multiple values, the third term in Eq (2) should not be ignored.

When training Markov chains, compound nouns to be trained must be correctly decomposed and have only one semantic feature for w_k. All edges between states have counters initially set to zero. While decomposed compound nouns are passed through the Markov chain, the counters are updated in the specific way, and after all the training data are utilized, edge probabilities are calculated satisfying Makov chain constraints. For example, given training example "공무원(HUM)"+"재 임용(ACT)"+"법 안(ABS)", state path in S^l is $s_{(0,0)}^l, s_{(0,3)}^l, s_{(3,3)}^l, s_{(3,2)}^l$ and state path in S^f is $s_{(NO,NO)}^f, s_{(NO,HUM)}^f, s_{(HUM,ACT)}^f, s_{(ACT,ABS)}^f$ where NO, HUM, ACT, ABS are semantic features.

After probability matrix of Markov chain have been got, it can be used in finding the most likely path by *Viterbi algorithm* [10].
One assumption believed to be more influential than statistics on participant's sequence is *least participants preference* assumption. Only with this assumption, the precision of decomposition ranges over 99% while recall is low.

Assumption 2 (Least Participants Preference) *When multiple candidate decompositions are found in a prefix of a compound noun, any candidate of that compound noun which has the least number of participants is preferred to the others.*

With this assumption, for example, given a compound noun "대학생선교회모임", any candidate which have the prefix decomposed as "대학생"+"선교회" is preferred to all candidate which have the prefix decomposed as "대학"+"생선"+"교회".
While decomposition is performed, so long as candidate prefixes decomposed have the same total syllable length, all the candidates prefixes other than least word participants prefix need not be reserved any longer for the decomposition of the remaining suffix. Markov assumption as well enables the composition process to ignore prematurely all the candidates which are not searched with yet but prefixed with lower probabilities provided candidate prefixes have the same total syllable length and reach at the same state in the Markov chain.

Generally speaking, If some conditions could be satisfied, dynamic programming method can be applied to the problem in exponential search space with two assumptions mentioned. These kinds of assumptions make the search method systemic and efficient.

4 Decomposition Algorithm

While decomposition is performed from left to right, the left substring of the last composition point will be referred to the *prefix* of that partially decomposed candidate. Given a noun string N_l of length l, the leftmost participants N_ks of N_l are looked up in dictionary and the semantic features of them are also got. N_ks are inserted in the sorted list or *heap* by their length. And then, the candidate which has the shortest prefix is picked up and the next decomposition point is sought and inserted in the sorted list like the first process. But except the first process, insertion must be entailed by the verification of the assumptions mentioned above, namely some prefixes which have no possibility to be the best candidate are cannot inserted in the candidate pool. This process is repeated till all the prefixes have the length l. The candidates remained at this point are compared using Markov chain and the best one is returned. Overall Decomposition algorithm is described below.

Algorithm 1 Decompose(N_l)
1 *Pool* $\leftarrow \phi$ ▷ *Pool is a list always sorted by prefix length*
2 Insert prefixes of N_l in *Pool* ▷ *prefixes must be registered in dictionary*
3 **while** $k < l$, for all $N_k \in$ *Pool* **do** ▷ *subscript means prefix length*
4 *Prefix* $\leftarrow$ *Pool*.remove() ▷ *pick up the shortest prefix*
5 *Str* $\leftarrow N_l$ - *Prefix* ▷ *Str contains suffix of* N_l *without Prefix*
6 Look up prefixes of *Str* in dictionary.
7 Generate all concatenations of *Prefix* and prefixes found at 6
8 Insert concatenations generated at 7 in *Pool*.
9 return candidates in *Pool* maximizing Eq(1)

In the algorithm presented above, "insert in *Pool*" deserves to be revisited. When insertion into *Pool* are performed, Markov assumption and least word participants assumption are applied and candidates not appropriate are not inserted or are removed from *Pool*. *Pool* are always sorted by prefix length so that candidates having the same prefix length may be applied to the assumptions easily when insertion is performed. If conditional insertion and removal didn't take place, our algorithm would find all the candidate decompositions.
In comparison with other algorithms, our algorithm cannot miss the best decomposition unless the least word participants preference assumption fail.

5 Experimental Results

Experiments have been performed in two ways and will be commented. One is learning of Markov chains and the other is compound noun decomposition using learned Markov chain matrices. Markov chains were trained on a set of 20564 compound nouns which are decomposed by humans. These nouns were extracted from HTML documents in home pages of several newspaper corporatioins. Some nouns may have postpositions as a suffix. The distribution of training data are given in Tables (1)–(3).

Table 1: Distribution of features in the training data

Semantic feature	Usage frequency	Percentage (%)
Abstract	14684	26.4
Action	10075	18.2
Postposition	9787	17.7
Concrete	4535	8.2
Location	4499	8.1
Human	4297	7.8
Suffix	3804	6.9
Proper Noun	1716	3.1
Prefix	1023	1.8
Temporary	668	1.2
Number	338	0.6

Now let $L(x)$ and $F(y)$ be random variables. $L(x)$ corresponds to the distribution of the length of participant syllables and $F(y)$ to the distribution of the semantic feature of the participant. Let $ML(x)$ be a random variable representing the distribution of the transition from $s^l_{(i,x)}$ to $s^l_{(x,j)}$ for all i, j and $MF(y)$ be a random variable representing the distribution of the trasition from $s^f_{(i,x)}$ to $s^f_{(x,j)}$ for all i, j.
If Markov chains trained by training data are to have any usefulness and semantic feature set is to be well organized, $L(x)$ distribution and $F(y)$ distribution is

Table 2: Training data distribution in participant's syllable length

Syllable length	Usage frequency	Percentage (%)
1	14074	25.4
2	36882	66.5
3	3998	7.2
4 or more	472	0.9

Table 3: Training data distribution in the number of participants

The number of participants	Usage frequency	Percentage (%)
2	8247	40.1
3	10467	50.9
4	1675	8.1
5 or more	169	0.8

believed to be different from $ML(x)$ and $MF(y)$ respectively. As a measure of these, here, *relative entropy* or *Kullback-Beibler divergence* is proposed and measured. The relative entropy is given by [8]

$$D(p\|q) = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)}$$

We calculated $D(ML(x)|L(x))$ and $D(MF(y)|L(y))$ and we got

$$D(ML(x)|L(x)) = 0.64$$
$$D(MF(y)|F(y)) = 2.1$$

This result above means that the Markov chain for the sequence of participants' semantic features is more informative than that of the Markov chain for the sequence of participants' lengths. Relative entropy can be compared and used to select which tendency to utilize. We suggest that feature set might be organized through the comparison of relative entropy.
To evaluate the performance of our method, 5027 compound nouns not related with training data are extracted from web documents and experimented with. No running data has any proper noun not registered in dictionary for evaluating maximum performance. The results are shown in Table 4.

Given the same test data, the longest match first method achieved precision of 90.2% and recall of 100%. The best accuracy of the other methods announced ranks up to 99%. The comparison among other methods shows that our method is not less accurate than the best one proposed before, considering that our test data may have postpositions, suffixes or prefixes.

Table 4: Recall and precision for the distribution of 5027 compound noun decomposition

	Markov chain not used	**Markov chain used**
Recall (%)	91.5	100
Precision (%)	99.7	98.4

6 Conclusion

We presented a method for the decomposition of compound nouns that uses the statistics of syllable length and semantic feature of their participants. We used two assumptions in compound noun decomposition. The first is *least participants preference assumption* and *trigram assumption* of the state sequences in word lengths and semantic features. The first assumption alone could achieved a good accuracy but could not cover some training data when ambiguity occurs. The combined method covered the rest and increased recall up to 100% and overall accuracy rate to 98.4 for the data set of 5027 compound nouns extracted from web documents.

Future work can be done in two directions.

One is to define a good feature set since it is believed that accuracy rate depends strongly on feature set organization. We have proposed that relative entropy could be a measure for finding a good set of semantic features. Another direction for future work is to deal with the compound noun which contains unregistered nouns. Unregistered nouns frequently shows up in compound nouns. The probability and patterns for unregistered nouns should be defined and learned in some way and the algorithm for this should be studied to find unregistered nouns.

References

[1] E. Charniak (1993), "*Statistical Language Learning.*" The MIT Press.

[2] K. S. Shim (1997), "A Compound Noun Segmentation using Composite Mutual Information," *Journal of KISS*, Vol. 24, No. 11, pp. 1307–1317. (*in Korean*)

[3] H. R. Park and J. H. Shin (1997), "Analysis of Korean Compound Nouns using Viterbi Training Algorithm," *Proceedings of KISS Fall Conference*, Vol. 24, No. 2, pp. 219–221. (*in Korean*)

[4] K. Yosiyuki. T. Takenobu and T. Hozumi (1994), "Analysis of Japanese Compound Nouns using Collocational Information." *Proceedings of Coling 94*, pp. 865–869.

[5] S. Maosong. S. Dayang. H. Changning (1997), "A Practical Word Segmenter and POS Tagger for Chinese Texts." *Proceedings of Fifth Conference on Applied Natural Language Processing*, pp. 119–126.

[6] H. R. Park, Y. S. Han and K. H. Lee (1996), "A Probabilistic Approach to Compound NOun Indexing in Korean Texts." *Proceedings of Coling 96*, pp. 514-518.

[7] P. Wong, C. Chan (1996), "Chinese Word Segmentation based on Maximum Matching and Word Binding Force," *Proceedings of Coling 96*, pp. 200-203.

[8] Cover, T. M. and Thomas, J. A. (1991), "Elements of information theory,", *Wiley series in telecommunications.*

[9] B. H. Yun, H. S. Yim, H. C. Yi (1995), "A Method of Korean Compound Noun Analysis using Statistical Informatioin," *Proceedings of KISS Spring Conference*, Vol. 22, No. 1, pp. 925–928.

[10] Viterbi, A.J. (1967), "Error bounds for convolutional codes and an asymptotically optimal decoding algorithm," *IEEE transactions on Information Theory 13* pp. 260–269.