Statistical Morphological Tagging and Parsing of Korean with an LTAG Grammar

Anoop Sarkar and Chung-hye Han
University of Pennsylvania *Simon Fraser University*

1. Introduction

This paper describes a lexicalized tree adjoining grammar (LTAG) based parsing system for Korean which combines corpus-based morphological analysis and tagging with a statistical parser. Part of the challenge of statistical parsing for Korean comes from the fact that Korean has free word order and a complex morphological system. The parser uses an LTAG grammar which is automatically extracted using LexTract (Xia *et al.*, 2000) from the Penn Korean TreeBank (Han *et al.*, 2002). The morphological tagger/analyzer is also trained on the TreeBank. The tagger/analyzer obtained the correctly disambiguated morphological analysis of words with 95.78/95.39% precision/recall when tested on a test set of 3,717 previously unseen words. The parser obtained an accuracy of 75.7% when tested on the same test set (of 425 sentences). These performance results are better than an existing off-the-shelf Korean morphological analyzer and parser run on the same data.

In section 2, we introduce the Korean TreeBank and we discuss how an LTAG grammar for Korean was extracted from this TreeBank. Also, we discuss how the derivation trees extracted from the TreeBank are used in the training of the statistical parser. Section 3 presents the overall approach of the morphological tagger/analyzer that we use in the parser. A detailed discussion about the parser is presented in section 4. This section also presents the method we used to combine the morphological information into the statistical LTAG parser. We also provide the experimental evaluation of the statistical parser on unseen test data in section 4.

2. Automatically Extracted LTAG Grammar for Korean

In this section we describe the Penn Korean TreeBank and the nature of the extracted LTAG grammar from this TreeBank.

2.1. Korean TreeBank

The LTAG grammar we use in the parser is extracted using LexTract (Xia *et al.*, 2000) from the Penn Korean TreeBank. The derivation trees obtained by using LexTract on the Treebank are used to train the statistical parser. The TreeBank has 54,366 words and 5,078 sentences. The annotation consists of a phrase structure analysis for each sentence, with head/phrase level tags as well as function tags (e.g., -SBJ, -OBJ) and empty category tags for traces (*T*) and dropped arguments (*pro*). Each word is morphologically analyzed, where the lemma and the inflections are identified. The lemma is tagged with a part-of-speech (POS) tag (e.g., NNC: noun, NPN: pronoun, VV: verb, VX: auxiliary verb), and the inflections are tagged with inflectional tags (e.g., PCA: case, EAU: inflection on verbs followed by an auxiliary verb, EPF: tense, EFN: sentence type). Example TreeBank trees are given in Figure 1. The figure on the left is an example of a bracketed structure for a simple declarative with canonical subject-object-verb order. The figure on the right is an example with a displaced constituent. In this example, the object NP '권한을' appears before the subject, while its canonical position is after the subject. The sentences used to illustrate bracketing structures in Figure 1 are romanized, glossed and translated in the following examples:

(1) a.

Cey-ka	kwanchuk	sahang-ul	pokoha-yess-supnita.
I-Nom	observation	item-Acc	report-Past-Decl

'I reported the overvation items.'

b.

Kwenhan-ul	nwukwu-ka	kaci-ko	iss-ci?
authority-Acc	who-Nom	have-AuxConnective	be-Int

'Who has the authority?'

* We would like to thank Fei Xia for the use her LexTract package without which we could not have started this research. We are also indebted to Martha Palmer, Aravind Joshi and David Chiang for their helpful suggestions.

```
(S (NP-SBJ 제/NPN+가/PCA)                    (S (NP-OBJ-1 권한/NNC+을/PCA)
   (VP (NP-OBJ 관측/NNC                          (S (NP-SBJ 누구/NPN+가/PCA)
               사항/NNC+을/PCA)                     (VP (VP (NP-OBJ *T*-1)
       보고하/VV+었/EPF+습니다/EFN)                            가지/VV+고/EAU)
   ./SFN)                                                 있/VX+지/EFN))
                                                 ?/SFN)
```

Figure 1: Example TreeBank Trees

2.2. LTAG formalism

LTAGs are based on the Tree Adjoining Grammar formalism developed by Joshi and his colleagues (Joshi, Levy and Takahashi, 1975; Joshi and Schabes, 1997). The primitive elements of an LTAG are elementary trees which are of two types: initial trees (α trees) and auxiliary trees (β trees). Initial trees are minimal linguistic structures that contain no recursion. They include trees containing the phrasal structures of simple sentences, NPs and so forth. Auxiliary trees represent recursive structures, which are adjuncts to basic structures, such as adverbials, adjectivals and so forth. Auxiliary trees have unique leaf node, called the *foot* node ($*$), which has the same syntactic category as the root node. Each elementary tree is associated with a lexical item (anchor) and encapsulates all arguments of the anchor, possessing an extended domain of locality. Elementary trees are combined by two operations: substitution and adjunction. In the substitution operation, a node marked for substitution ($\downarrow$) in an elementary tree is replaced by another elementary tree whose root category is the same as the substitution-marked node. In an adjunction operation, an auxiliary tree is inserted into an initial tree. The root and the foot nodes of the auxiliary tree must match the node label at which the auxiliary tree adjoins. The combination of elementary trees produces two structures: derived and derivation trees. Derived trees correspond to phrase structure representation and derivation trees are a record of the history of the combination process.

2.3. Extracted LTAG grammar

We use LexTract (Xia *et al.*, 2000) to convert the phrase structure trees of the Korean TreeBank into LTAG derivation trees. Each node in these derivation trees is an elementary tree extracted from the Korean TreeBank by LexTract. The elementary trees of the LTAG Korean grammar are exactly the set of elementary trees used in the derivation trees obtained using LexTract. For example, the elementary trees extracted from the TreeBank bracketed structures in Figure 1 are given in Figure 2. The entire extracted grammar contains 632 elementary tree template types and 13,941 lexicalized elementary tree types (Xia *et al.*, 2001).

As mentioned earlier, in addition to the elementary trees, LexTract produces derived and derivation trees for each TreeBank tree. For instance, for the second TreeBank tree in Figure 1, 권한 and 누구 trees are each substituted into 가지 tree, and 있 tree is adjoined onto the VP node of 가지 tree. This produces the derived and derivation trees in Figure 3.

3. TreeBank-trained Morphological Tagger/Analyzer

Korean is an agglutinative language with a very productive inflectional system. This means that for any NLP application on Korean to be successful, some amount of morphological analysis is necessary. Without it, the development of a statistical based parser would not be feasible due to the sparse data problem bound to exist in the training data.

To avoid this problem in our parsing system, we use a morphological tagger/analyzer. This tagger/analyzer also performs statistical disambiguation and it was trained on 91% of the Korean TreeBank. The tagger/analyzer takes raw text as intput and returns a lemmatized disambiguated output in which for each word, the lemma is labeled with a POS tag and the inflections are labeled with inflectional tags. This system is based on a simple statistical model combined with a corpus-driven rule-based approach, comprising a trigram-based tagging component and a morphological rule application component.

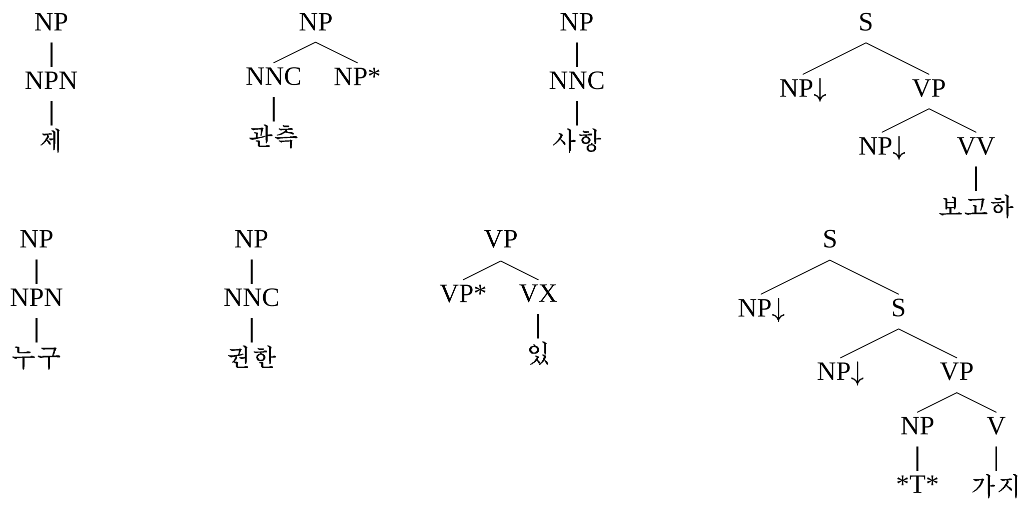
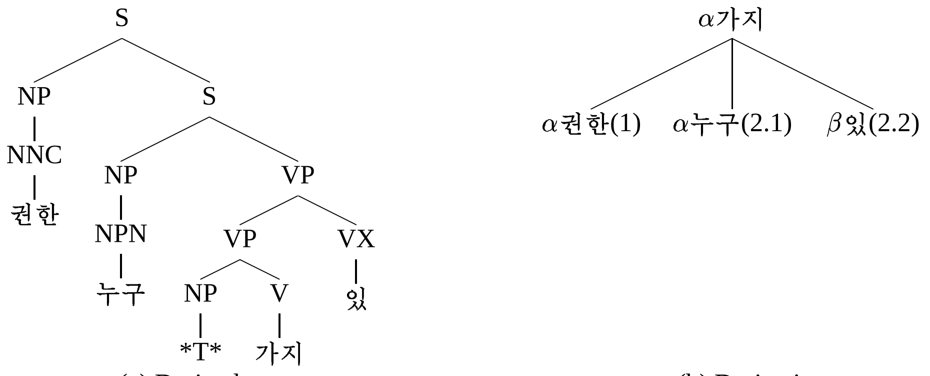


Figure 2: Some examples of Extracted Elementary Trees



(a) Derived tree (b) Derivation tree

Figure 3: Extracted Derived and Derivation Trees

The tagger/analyzer follows several sequential steps to label the raw text with POS and inflectional tags. After tokenization (mostly applied to punctuations), all morphological contractions in the input string are uncontracted (STRING CONVERSION). The known words are then tagged with tag sequences of the form POS + *inflectional-tag* (e.g., NNC+PCA, VV+EPF+EFN) extracted from the TreeBank, and unknown words are tagged with NNC (common noun) tag, NNC being the most frequent tag for unknown words (MORPH TAGGING). Tags for unknown words are then updated using inflectional templates extracted from the TreeBank (UPDATE TAGGING). And then using the inflection dictionary and stem dictionary extracted from the TreeBank, the lemma and the inflections are identified, splitting the inflected form of the word into its constituent stem and affixes (LEMMA/INFLECTION IDENTIFICATION), creating the final output. This process is summarized in Figure 4. The proposed approach to morphological analysis is different from other approaches in that the tagging phase precedes the morphological analysis phase. This allows morphological analysis to be done deterministically through using the information obtained from tagging. An example input to the tagger/analyzer and the final output are shown in Figure 5.

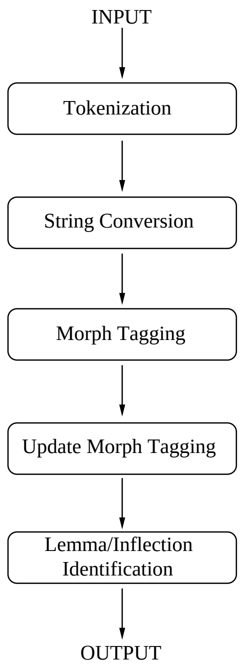


Figure 4: Overview of the Tagger/Analyzer

```
Input:
    제가 관측 사항을 보고했습니다 .

Output:
    제/NPN+가/PCA 관측/NNC 사항/NNC+을/PCA 보고하/VV+었/EPF+습니다 ./SFN
```

Figure 5: Input and output from the morphological tagging phase

The performance of the morphological analyzer/tagger has been evaluated on the 9% of the Treebank. The test set consists of 3,717 word tokens and 425 sentences. Both precision and recall were computed by comparing the morpheme/tag pairs in the test file and the gold file. The precision corresponds to the percentage of morpheme/tag pairs in the gold file that match the morpheme/tag pairs in the test file. And the recall corresponds to the percentage of morpheme/tag pairs in the test file that match the morpheme/tag pairs in the gold file. This approach yielded a precision/recall score of 95.79/95.39%.

An off-the-shelf morphological analyzer/tagger (Yoon *et al.*, 1999) was tested on the same test set. This system is reported to have obtained 94.7% tagging accuracy on a test set drawn from the same corpus as it was trained on. For the sake of fair comparison, the output of the off-the-shelf tagger was converted to look as close as possible

	precision/recall (%)
Treebank trained	95.78/95.39
Off-the-Shelf	29.42/31.25

Table 1: Evaluation of the Morphological Analyzer/Tagger

to the Treebank trained analyzer/tagger output, including the tagset. However, not all tokenization mismatches in the off-the-shelf system could be resolved. The results (in Table 1) show, not surprisingly, that better performance on test data from a particular domain is obtained by training on annotated data from the same domain. Even so, the results from another system on the same data provide at least a baseline performance to compare against our results.

4. Statistical LTAG Parser

The use of lexical information plays a prominent role in statistical parsing models for English. In this section, we discuss how to extend a statistical parser that relies on bigrams of lexical dependencies to a morphologically complex language like Korean. While these types of parsers have to deal with sparse data problems, this problem is exacerbated in the case of Korean due to the fact that several base-forms of words can appear with a wide array of morphological affixes. This problem is addressed by incorporating the morphological tagger/analyzer described above, which significantly improves performance.

Apart from the use of a specialized morphological tagger/analyzer for Korean, our methods are language independent and have been tested in previous work on the WSJ Penn English TreeBank (Marcus, Santorini and Marcinkiewicz, 1993). As described in above, we use Lextract to convert the TreeBank (the same method is used for both the English and the Korean TreeBanks) into a parser derivation tree for each sentence. The statistical parsing model is then trained using these derivation trees.

4.1. Probability Models

The statistical parser uses three probabilistic models: one model for picking a tree as the start of a derivation; and two models for the probability of one tree substituting or adjoining into another tree. Each of these models can be trained directly using maximum likelihood estimation from the Lextract output. The probabilistic models of substitution and adjunction provide a natural domain to describe the dependencies between pairs of words in a sentence.

(Resnik, 1992) provided some early motivation for a stochastic version of Tree Adjoining Grammars and gave a formal definition of stochastic TAG. Simultaneously, (Schabes, 1992) also provided an identical stochastic version of TAG and also extended the Inside-Outside algorithm for CFGs (Lari and Young, 1990) to stochastic TAGs. (Schabes, 1992) also performed experiments to show that a stochastic TAG can be learnt from the ATIS corpus.

A stochastic LTAG derivation proceeds as follows (Schabes, 1992; Resnik, 1992). An initial tree is selected with probability P_i and subsequent substitutions are performed with probability P_s and adjunctions are performed with probability P_a.

For each τ that can be valid start of a derivation:

$$\sum_{\tau} P_i(\tau) = 1$$

Each subsequent substitution or adjunction occurs independently. For possible substitutions defined by the grammar:

$$\sum_{\alpha} P_s(\tau, \eta \rightarrow \alpha) = 1$$

where, α is substituting into node η in tree τ. For possible adjunctions in the grammar there is an additional factor which is required for the probability to be well-formed:

$$P_a(\tau, \eta \rightarrow \textsc{na}) + \sum_{\beta} P_a(\tau, \eta \rightarrow \beta) = 1$$

where, β is adjoining into node η in tree τ, and $P_a(\tau, \eta \rightarrow \textsc{na})$ is the probability that there is no adjunction (NA) at node η in τ.

Each LTAG derivation $\mathcal{D}$ is built starting from some initial tree α. Let us consider the probability of a derivation $\mathcal{D}$ which was constructed using p substitutions and q adjunctions and r internal nodes which had no adjunction. If we assume that each elementary tree is lexicalized (*anchored*) by exactly one word, then the length of the sentence $n = p + q + 1$. In fact, in the experiments we report on in this paper, each elementary tree has exactly one lexical item as an anchor.

$$\begin{aligned} &Pr(\mathcal{D}, S = w_0 \ldots w_n) = \\ &\quad P_i(\alpha, w_i) \times \prod_p P_s(\tau, \eta, w \rightarrow \tau', w') \times \\ &\quad \prod_q P_a(\tau, \eta, w \rightarrow \tau', w') \times \\ &\quad \prod_r P_a(\tau, \eta, w \rightarrow \textsc{na}) \end{aligned} \tag{2}$$

This derivation $\mathcal{D}$ can be drawn graphically as a tree where each node in this derivation tree is an elementary tree in the original LTAG (this is the standard notion of a TAG derivation tree).

P_i and P_s can be written as the following lexicalized conditional probabilities which can be estimated from the training data.

$$\begin{aligned} P_i(\tau) &= P_i(\alpha, w \mid TOP) \\ P_s(\tau, \eta \rightarrow \alpha) &= P_s(\alpha, w' \mid \tau, \eta, w) \\ P_a(\tau, \eta \rightarrow \beta) &= P_a(\beta, w' \mid \tau, \eta, w) \end{aligned}$$

Events for each of these probability models can be directly read from the LexTract output. Using maximum likelihood estimation we convert these events into the above parameter values in our statistical parser.

For further details about decomposing these probabilities further to generalize over particular lexical items and to make parsing and decoding easier see (Chiang, 2000). (Chiang, 2000) also has details about the standard use of prior probabilities in statistical parsing for pruning which we use in our implementation.

The probability of a sentence S computed using this model is the sum of all the possible derivations of the sentence.

$$P(S) = \sum_D Pr(D, S)$$

A generative model can be defined instead of a conditional probability to obtain the best derivation $\mathcal{D}_{\text{BEST}}$ given a sentence S. The value for (3) is computed using the Equation 2.

$$\begin{aligned} \mathcal{D}_{\text{BEST}} &= \underset{\mathcal{D}}{\text{arg max}} \ \Pr(\mathcal{D} \mid S) \\ &= \underset{\mathcal{D}}{\text{arg max}} \ \frac{\Pr(\mathcal{D}, S)}{\Pr(S)} \\ &= \underset{\mathcal{D}}{\text{arg max}} \ \Pr(\mathcal{D}, S) \end{aligned} \tag{3}$$

The particular definition of a stochastic TAG is by no means the only way of defining a probabilistic grammar formalism with TAG. There have been some variants from the standard model that have been published since the original stochastic TAG papers.

For example, the restriction of one adjunction per node could be dropped and a new variant of standard TAG can be defined which permits arbitrary number of modifications per node. This variant was first introduced by (Schabes and Shieber, 1992; Schabes and Shieber, 1994). Tree Insertion Grammar (Schabes and Waters, 1995) is a variant of TAG where the adjoining operation is restricted in a certain way and this restricted operation is named insertion. TIGs are weakly equivalent to CFGs but they can produce structural descriptions that are not obtainable by any CFG.

A stochastic version of insertion (Schabes and Waters, 1996) was defined in the context of Tree Insertion Grammar (TIG). In this model, multiple trees can be adjoined to the left and to the right of each node with the following probabilities:

$$P_{la}(\tau, \eta \rightarrow \text{NA}_l) + \sum_{\tau'} P_{la}(\tau, \eta \rightarrow \tau') = 1$$

$$P_{ra}(\tau, \eta \rightarrow \text{NA}_r) + \sum_{\tau'} P_{ra}(\tau, \eta \rightarrow \tau') = 1$$

In our parser, we allow multiple adjunctions at a node and also we exploit TIG style probabilities P_{la} and P_{ra}. This was done so that the output easily convertible to the earlier dependency style parser that was used in the project (with which we compare performance in our evaluation).

There are many other probability measures that can be used with TAG and its variants. One can easily go beyond the bi-lexical probabilities that have been the main focus in this chapter to probabilities that invoke greater amounts of structural or lexical context. (Carroll and Weir, 1997), for example, gives some additional probability models one might consider useful when using TAGs.

An example output from the statistical parser is shown in Figure 6. In the parser (and in the Lextract output), each elementary tree is anchored by exactly one lexical item. The gloss and translation for the example in Figure 6 is given in the following example:

(4)						
Motun	hochwul	tayho-nun	mayil	24	si-ey	pakkwui-key
every	call	sign	everyday	24	hour-at	switch-AuxConnect
toy-ciyo.						
be-Decl						

‘Every call sign is switched at midnight everyday.’

Index	Word	POS tag (morph)	Elem Tree	Anchor Label	Node Address	Subst/Adjoin into (Index)
0	모든	DAN	βNP*=1	anchor	root	2
1	호출	NNC	βNP*=1	anchor	root	2
2	대호+는	NNC+PAU	αNP=0	anchor	0	6
3	매일	ADV	βVP*=25	anchor	1	6
4	24	NNU	βNP*=1	anchor	0	5
5	시+에	NNX+PAD	βVP*=17	anchor	1	6
6	바뀌+게	VV+ECS	αS-NPs=23	anchor	-	TOP
7	되+지요	VX+EFN	βVP*=13	anchor	1	6
8	.	SFN	-	-	-	-

Figure 6: Example derivation of a sentence reported by the statistical parser

4.2. Incorporating the Morphological Information into the Parser

In this section we describe how the morphological tagger (described earlier) is incorporated into the smoothing of the probability models that are used in the statistical parser.

The smoothing using the morphological tagger is handled as follows. The statistical parser has various probability models associated with it. One of the most crucial models is the one that decides on parser actions by using statistics collected on pairs of words. For example, P_a is the probability of combination of two trees anchored by

	On training data	On test data
Current Work	97.58	75.7
(Yoon, Kim and Song, 1997)	–	52.29/51.95 P/R

Table 2: Parser evaluation results

words w and w' via adjunction of those trees. Here w and w' are the inflected forms of the word, while p and p' are selected elements of the disambiguated morphological analysis of the inflected forms taken from the output of the morphological tagger described above. Based on the part of speech, we might want to select different components of the morphological analysis. For example, for nouns we might want to pick the final element which usually corresponds to the case marking, while for verbs we might want to pick the stem which is the first element of the analysis. We have the flexibility in the parser to choose which element should be picked. The best results were obtained when we chose the stem. That is, for the results reported here we always pick the first element of the morphological analysis for each inflected form.

$$Pr(t', p', w' \mid \eta, t, w, p)$$

We decompose this conditional probability into the following components:

$$\begin{aligned} Pr(t', p', w' \mid \eta, t, w, p) = \\ Pr(t' \mid \eta, t, w, p) \times \\ Pr(p' \mid t', \eta, t, w, p) \times \\ Pr(w' \mid p', t', \eta, t, w, p) \end{aligned}$$

For each of the equations above, we use a backoff model which is used to handle sparse data problems. We compute a backoff model as follows. Let e_1 stand for the original lexicalized model and e_2 be the backoff level which only uses the output of the morphological tagger described above:

$$\begin{aligned} Pr_{e_1} &= Pr(t' \mid \eta, t, w, p) \\ Pr_{e_2} &= Pr(t' \mid \eta, t, p) \end{aligned}$$

The backoff model is computed as follows:

$$\lambda(c) \times Pr_{e_1} + (1 - \lambda(c)) \times Pr_{e_2}$$

where $\lambda(c) = \frac{c}{(c+D)}$. $c = \mathit{count}(r)$ is the total count for each conditional probability model P, where $P(l \mid r)$. D is the *diversity* of Pr_{e_1}. *diversity* is defined as the number of distinct counts for Pr_{e_1}. Note that this backoff model is implemented for each lexicalized model used in the statistical parser.

4.3. Experimental Results

In order to evaluate the statistical parser (combined with the tagger as described above), our parser was trained on the derivations and corresponding LTAG grammar extracted from 91% of the TreeBank (4653 sentences). The parser was then tested on 9% of the TreeBank which was reserved as unseen data (425 sentences). Note that the parser did not have access to any grammar information from LexTract or lexicon information which was taken from the unseen data.

For comparison, another parser (Yoon, Kim and Song, 1997) was tested on the same test set. For the sake of fair comparison, the output of this parser was converted to look as close as possible to our output. Even so, the number of node labels did not match, due to the difference in tokenization schemes for certain lexical elements such as copulas and auxiliary verbs. We thus report precision/recall in the comparison. We report word-to-word dependency accuracy compared with the gold standard for our parser. The evaluation results are summarized in Table 2. Not surprisingly, the results show that better performance on test data from a particular domain is obtained by training on annotated data from the same domain. The results from another parser on the same data provide a baseline performance to compare against our results. Also, the performance achieved on our test set is competetive with the state-of-the-art English statistical parsers when trained on similar amounts of data.

5. Conclusion

Our work is significant in that it is the first LTAG-based parsing system for Korean. We have shown that LTAG-based statistical parsing is feasible for a language with free word order and complex morphology. Our system has been successfully incorporated into a Korean/English machine translation system as source language analysis component (Han *et al.*, 2000; Palmer *et al.*, 2002). The LTAG parser produces a single-best analysis of the input Korean sentence. We showed that the tagger/analyzer described in this paper obtained the correctly disambiguated morphological analysis of words with 95.78/95.39% precision/recall when tested on a test set of 3,717 previously unseen words. The statistical parser described in this paper obtained an accuracy of 75.7% when tested on the same test set (of 425 sentences). These performance results are better than an existing off-the-shelf Korean morphological analyzer and parser run on the same data.

References

Carroll, J. and D. Weir. 1997. Encoding Frequency Information in Lexicalized Grammars. In *Proc. 5th Int'l Workshop on Parsing Technologies IWPT-97*, Cambridge, Mass.

Chiang, David. 2000. Statistical Parsing with an Automatically-Extracted Tree Adjoining Grammar. In *Proc. of ACL-2000*.

Han, Chung-hye, Na-Rae Han, Eon-Suk Ko, Heejong Yi and Martha Palmer. 2002. Penn Korean Treebank: Development and Evaluation. In *Proceedings of the 16th Pacific Asian Conference on Language and Computation*. Korean Society for Language and Information.

Han, Chung-hye, Benoit Lavoie, Martha Palmer, Owen Rambow, Richard Kittredge, Tanya Korelsky, Nari Kim and Myunghee Kim. 2000. Handling Structural Divergences and Recovering Dropped Arguments in a Korean/English Machine Translation System. In John S. White, editor, *Envisioning Machine Translation in the Information Future*. Springer-Verlag, pages 40–53.

Joshi, Aravind and Yves Schabes. 1997. Tree Adjoining Grammars. In A. Salomma and G. Rosenberg, editors, *Handbook of Formal Languages and Automata*. Springer-Verlag, Heidelberg.

Joshi, Aravind K., L. Levy and M. Takahashi. 1975. Tree Adjunct Grammars. *Journal of Computer and System Sciences*.

Lari, K. and S. J. Young. 1990. The estimation of stochastic context-free grammars using the Inside-Outside algorithm. *Computer Speech and Language*, 4:35–56.

Marcus, Mitch, Beatrice Santorini and M. Marcinkiewicz. 1993. Building a large annotated corpus of English. *Computational Linguistics*, 19(2):313–330.

Palmer, Martha, Chung-hye Han, Anoop Sarkar and Ann Bies. 2002. Integrating Korean analysis components in a modular Korean/English machine translation system. Ms. University of Pennsylvania and Simon Fraser University.

Resnik, P. 1992. Probabilistic Tree-Adjoining Grammars as a framework for statistical natural language processing. In *Proc. of COLING '92*, pages 418–424, Nantes, France.

Schabes, Y. 1992. Stochastic Lexicalized Tree-Adjoining Grammars. In *Proc. of COLING '92*, pages 426–432, Nantes, France.

Schabes, Y. and S. Shieber. 1992. An Alternative Conception of Tree-Adjoining Derivation. In *Proceedings of the* 20^{th} *Meeting of the Association for Computational Linguistics*.

Schabes, Y. and R. Waters. 1996. Stochastic Lexcalized Tree-Insertion Grammar. In H. Bunt and M. Tomita, editors, *Recent Advances in Parsing Technology*. Kluwer, pages 281–294.

Schabes, Yves and Stuart Shieber. 1994. An Alternative Conception of Tree-Adjoining Derivation. *Computational Linguistics*, 20(1):91–124.

Schabes, Yves and Richard Waters. 1995. Tree Insertion Grammar: A Cubic-Time, Parsable Formalism that Lexicalizes Context-Free Grammar without Changing the Trees Produced. *Computational Linguistics*, 21(4):479–513.

Xia, Fei, Chung-hye Han, Martha Palmer and Aravind Joshi. 2001. Comparing Lexicalized Treebank Grammars Extracted from Chinese, Korean, and English Corpora. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*.

Xia, Fei, Martha Palmer, and Aravind Joshi. 2000. A Uniform Method of Grammar Extraction and Its Applications. In *Proceedings of the EMNLP 2000*.

Yoon, J., S. Kim and M. Song. 1997. New Parsing Method using a Global Association Table. In *Proceedings of IWPT-97*.

Yoon, Juntae, C. Lee, S. Kim and M. Song. 1999. Morphological Analyzer of Yonsei Univ., Morany: Morphological Analysis based on Large Lexical Database Extracted from Corpus. In *Proceedings of Korean Language Information Processing*. Written in Korean.