# Factors Affecting the Accuracy of Korean Parsing

**Tagyoung Chung**, **Matt Post** and **Daniel Gildea**
Department of Computer Science
University of Rochester
Rochester, NY 14627

## Abstract

We investigate parsing accuracy on the Korean Treebank 2.0 with a number of different grammars. Comparisons among these grammars and to their English counterparts suggest different aspects of Korean that contribute to parsing difficulty. Our results indicate that the coarseness of the Treebank's nonterminal set is a even greater problem than in the English Treebank. We also find that Korean's relatively free word order does not impact parsing results as much as one might expect, but in fact the prevalence of zero pronouns accounts for a large portion of the difference between Korean and English parsing scores.

## 1 Introduction

Korean is a head-final, agglutinative, and morphologically productive language. The language presents multiple challenges for syntactic parsing. Like some other head-final languages such as German, Japanese, and Hindi, Korean exhibits long-distance scrambling (Rambow and Lee, 1994; Kallmeyer and Yoon, 2004). Compound nouns are formed freely (Park et al., 2004), and verbs have well over 400 paradigmatic endings (Martin, 1992).

Korean Treebank 2.0 (LDC2006T09) (Han and Ryu, 2005) is a subset of a Korean newswire corpus (LDC2000T45) annotated with morphological and syntactic information. The corpus contains roughly 5K sentences, 132K words, and 14K unique morphemes. The syntactic bracketing rules are mostly the same as the previous version of the treebank (Han et al., 2001) and the phrase structure annotation schemes used are very similar to the ones used in Penn English treebank. The Korean Treebank is constructed over text that has been morphologically analyzed; not only is the text tokenized into morphemes, but all allomorphs are neutralized.

To our knowledge, there have been only a few papers focusing on syntactic parsing of Korean. Hermjakob (2000) implemented a shift-reduce parser for Korean trained on very limited (1K sentences) data, and Sarkar and Han (2002) used an earlier version of the Treebank to train a lexicalized tree adjoining grammar. In this paper, we conduct a range of experiments using the Korean Treebank 2.0 (hereafter, KTB) as our training data and provide analyses that reveal insights into parsing morphologically rich languages like Korean. We try to provide comparisons with English parsing using parsers trained on a similar amount of data wherever applicable.

## 2 Difficulties parsing Korean

There are several challenges in parsing Korean compared to languages like English. At the root of many of these challenges is the fact that it is highly inflected and morphologically productive. Effective morphological segmentation is essential to learning grammar rules that can generalize beyond the training data by limiting the number of out-of-vocabulary words. Fortunately, there are good techniques for doing so. The sentences in KTB have been segmented into basic morphological units.

Second, Korean is a pro-drop language: subjects and objects are dropped wherever they are pragmatically inferable, which is often possible given its rich morphology. Zero pronouns are a remarkably frequent phenomenon in general (Han, 2006), occuring

*Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 49–57,
Los Angeles, California, June 2010. ©2010 Association for Computational Linguistics

an average of 1.8 times per sentence in the KTB. The standard approach in parsing English is to ignore NULL elements entirely by removing them (and recursively removing unary parents of empty nodes in a bottom-up fashion). This is less of a problem in English because these empty nodes are mostly trace elements that denote constituent movement. In the KTB, these elements are removed altogether and a crucial cue to grammatical inference is often lost. Later we will show the profound effect this has on parsing accuracy.

Third, word order in Korean is relatively free. This is also partly due to the richer morphology, since morphemes (rather than word order) are used to denote semantic roles of phrases. Consider the following example:

| | | | | |
|---|---|---|---|---|
| 존이 | 매리에게 | 책을 | 주었다 | . |
| John-NOM | Mary-DAT | book-ACC | give-PAST | . |

In the example, any permutation of the first three words produces a perfectly acceptable sentence. This freedom of word order could potentially result in a large number of rules, which could complicate analysis with new ambiguities. However, formal written Korean generally conforms to a canonical word order (SOV).

## 3 Initial experiments

There has been some work on Korean morphological analysis showing that common statistical methods such as maximum entropy modeling and conditional random fields perform quite well (Lee et al., 2000; Sarkar and Han, 2002; Han and Palmer, 2004; Lee and Rim, 2005). Most claim accuracy rate over 95%. In light of this, we focus on the parsing part of the problem utilizing morphology analysis already present in the data.

### 3.1 Setup

For our experiments we used all 5,010 sentences in the Korean Treebank (KTB), which are already segmented. Due to the small size of the corpus, we used ten-fold cross validation for all of our experiments, unless otherwise noted. Sentences were assigned to folds in blocks of one (i.e., fold 1 contained sentences 1, 11, 21, and so on.). Within each fold, 80% of the data was assigned to training, 10% to development, and 10% to testing. Each fold's vocabulary was set to all words occurring more than once in its training data, with a handful of count one tokens replacing unknown words based on properties of the word's surface form (all Korean words were placed in a single bin, and English words were binned following the rules of Petrov et al. (2006)). We report scores on the development set.

| model | F1 | F1$_{\leq 40}$ | types | tokens |
|---|---|---|---|---|
| Korean | 52.78 | 56.55 | 6.6K | 194K |
| English (§02–03) | 71.06 | 72.26 | 5.5K | 96K |
| English (§02–04) | 72.20 | 73.29 | 7.5K | 147K |
| English (§02–21) | 71.61 | 72.74 | 23K | 950K |

Table 1: Parser scores for Treebank PCFGs in Korean and English. For English, we vary the size of the training data to provide a better point of comparison against Korean. Types and tokens denote vocabulary sizes (which for Korean is the mean over the folds).

We report parser accuracy scores using the standard F1 metric, which balances precision and recall of the labeled constituents recovered by the parser: $2PR/(P+R)$. Throughout the paper, all evaluation occurs against gold standard trees that contain no NULL elements or nonterminal function tags or annotations, which in some cases requires the removal of those elements from parse trees output by the parser.

### 3.2 Treebank grammars

We begin by presenting in Table 1 scores for the standard Treebank grammar, obtained by reading a standard context-free grammar from the trees in the training data and setting rule probabilities to relative frequency (Charniak, 1996). For these initial experiments, we follow standard practice in English parsing and remove all (a) nonterminal function tags and (b) NULL elements from the parse trees before learning the grammar. For comparison purposes, we present scores from parsing the Wall Street Journal portion of the English Penn Treebank (PTB), using both the standard training set and subsets of it chosen to be similar in size to the KTB. All English scores are tested on section 22.

There are two interesting results in this table. First, Korean parsing accuracy is much lower than English parsing accuracy, and second, the accuracy difference does not appear to be due to a difference in the size of the training data, since reducing the size of the English training data did not affect accuracy scores very much.

Before attempting to explain this empirically, we note that Rehbein and van Genabith (2007) demonstrate that the F1 metric is biased towards parse trees with a high ratio of nonterminals to terminals, because mistakes made by the parser have a smaller effect on the overall evaluation score.[1] They recommend that F1 not be used for comparing parsing accuracy across different annotation schemes. The nonterminal to terminal ratio in the KTB and PTB are 0.40 and 0.45, respectively. It is a good idea to keep this bias in mind, but we believe that this small ratio difference is unlikely to account for the huge gap in scores displayed in Table 1.

The gap in parsing accuracy is unsurprising in light of the basic known difficulties parsing Korean, summarized earlier in the paper. Here we observe a number of features of the KTB that contribute to this difficulty.

**Sentence length** On average, KTB sentences are much longer than PTB sentences (23 words versus 48 words, respectively). Sentence-level F1 is inversely correlated with sentence length, and the relatively larger drop in F1 score going from column 3 to 2 in Table 1 is partially accounted for by the fact that column 3 represents 33% of the KTB sentences, but 92% of the English sentences.

**Flat annotation scheme** The KTB makes relatively frequent use of very flat and ambiguous rules. For example, consider the extreme cases of rule ambiguity in which the lefthand side nonterminal is present three or more times on its righthand side. There are only three instances of such "triple+-recursive" NPs among the ∼40K trees in the training portion of the PTB, each occurring only once.

NP → NP NP NP , CC NP
NP → NP NP NP CC NP
NP → NP NP NP NP .

The KTB is an eighth of the size of this, but has fifteen instances of such NPs (listed here with their frequencies):

NP → NP NP NP NP (6)
NP → NP NP NP NP NP (3)
NP → NP NP NP NP NP NP (2)
NP → NP NP NP NP NP NP NP (2)
NP → SLQ NP NP NP SRQ PAD (1)
NP → SLQ NP NP NP NP SRQ PAN (1)

Similar rules are common for other nonterminals as well. Generally, flatter rules are easier to parse with because they contribute to parse trees with fewer nodes (and thus fewer independent decision points). However, the presence of a single nonterminal on both the left and righthand side of a rule means that the annotation scheme is failing to capture distributional differences which must be present.

**Nonterminal granularity** This brings us to a final point about the granularity of the nonterminals in the KTB. After removing function tags, there are only 43 nonterminal symbols in the KTB (33 of them preterminals), versus 72 English nonterminals (44 of them preterminals). Nonterminal granularity is a well-studied problem in English parsing, and there is a long, successful history of automatically refining English nonterminals to discover distributional differences. In light of this success, we speculate that the disparity in parsing performance might be explained by this disparity in the number of nonterminals. In the next section, we provide evidence that this is indeed the case.

## 4 Nonterminal granularity

There are many ways to refine the set of nonterminals in a Treebank. A simple approach suggested by Johnson (1998) is to simply annotate each node with its parent's label. The effect of this is to refine the distribution of each nonterminal over sequences of children according to its position in the sentence; for example, a VP beneath an SBAR node will have a different distribution over children than a VP beneath an S node. This simple technique alone produces a large improvement in English Treebank parsing. Klein and Manning (2003) expanded this idea with a series of experiments wherein they manually refined nonterminals to different degrees, which resulted in parsing accuracy rivaling that of bilexicalized parsing models of the time. More recently, Petrov et al. (2006) refined techniques originally proposed by Matsuzaki et al. (2005) and Prescher

[1] We thank one of our anonymous reviewers for bringing this to our attention.

| | |
|---|---|
| SBJ | subject with nominative case marker |
| OBJ | complement with accusative case marker |
| COMP | complement with adverbial postposition |
| ADV | NP that function as adverbial phrase |
| VOC | noun with vocative case maker |
| LV | NP coupled with "light" verb construction |

Table 2: Function tags in the Korean treebank

| **model** | **F1** | **F1$_{\leq 40}$** |
|---|---|---|
| Korean | | |
| coarse | 52.78 | 56.55 |
| w/ function tags | **56.18** | **60.21** |
| English (small) | | |
| coarse | **72.20** | **73.29** |
| w/ function tags | 70.50 | 71.78 |
| English (standard) | | |
| coarse | 71.61 | 72.74 |
| w/ function tags | **72.82** | **74.05** |

Table 3: Parser scores for Treebank PCFGs in Korean and English with and without function tags. The small English results were produced by training on §02–04.

(2005) for automatically learning latent annotations, resulting in state of the art parsing performance with cubic-time parsing algorithms.

We begin this section by conducting some simple experiments with the existing function tags, and then apply the latent annotation learning procedures of Petrov et al. (2006) to the KTB.

### 4.1 Function tags

The KTB has function tags that mark grammatical functions of NP and S nodes (Han et al., 2001), which we list all of them in Table 2. These function tags are principally grammatical markers. As mentioned above, the parsing scores for both English and Korean presented in Table 1 were produced with grammars stripped of their function tags. This is standard practice in English, where the existing tags are known not to help very much. Table 3 presents results of parsing with grammars with nonterminals that retain these function tags (we include results from Section 3 for comparison). Note that evaluation is done against the unannotated gold standard parse trees by removing the function tags after parsing with them.

The results for Korean are quite pronounced: we see a nearly seven-point improvement when retaining the existing tags. This very strongly suggests that the KTB nonterminals are too coarse when stripped of their function tags, and raises the question of whether further improvement might be gained from latent annotations.

The English scores allow us to make another point. Retaining the provided function tags results in a solid performance increase with the standard training corpus, but actually hurts performance when training on the small dataset. Note clearly that this does *not* suggest that parsing performance with the grammar from the small English data could not be improved with latent annotations (indeed, we will show that they can), but only that the given annotations do not help improve parsing accuracy. Taking the Korean and English accuracy results from this table together provides another piece of evidence that the Korean nonterminal set is too coarse.

### 4.2 Latent annotations

We applied the latent annotation learning procedures of Petrov et al.[2] to refine the nonterminals in the KTB. The trainer learns refinements over the coarse version of the KTB (with function tags removed). In this experiment, rather than doing 10-fold cross validation, we split the KTB into training, development, and test sets that roughly match the 80/10/10 splits of the folds:

| section | file IDs |
|---|---|
| training | 302000 to 316999 |
| development | 317000 to 317999 |
| testing | 320000 to 320999 |

This procedure results in grammars which can then be used to parse new sentences. Table 4 displays the parsing accuracy results for parsing with the grammar (after smoothing) at the end of each split-merge-smooth cycle.[3] The scores in this table show that, just as with the PTB, nonterminal refinement makes a huge difference in parser performance.

Again with the caveat that direct comparison of parsing scores across annotation schemes must be taken loosely, we note that the KTB parsing accuracy is still about 10 points lower than the best ac-

[2] `http://code.google.com/p/berkeleyparser/`

[3] As described in Petrov et al. (2006), to score a parse tree produced with a refined grammar, we can either take the Viterbi derivation or approximate a sum over derivations before projecting back to the coarse tree for scoring.

|  | Viterbi |  | max-sum |  |
|---|---|---|---|---|
| **cycle** | **F1** | **$F1_{\leq 40}$** | **F1** | **$F1_{\leq 40}$** |
| 1 | 56.93 | 61.11 | 61.04 | 64.23 |
| 2 | 63.82 | 67.94 | 66.31 | 68.90 |
| 3 | 69.86 | 72.83 | 72.85 | 75.63 |
| 4 | 74.36 | 77.15 | 77.18 | 78.18 |
| 5 | 78.07 | 80.09 | 79.93 | 82.04 |
| 6 | 78.91 | 81.55 | 80.85 | 82.75 |

Table 4: Parsing accuracy on Korean test data from the grammars output by the Berkeley state-splitting grammar trainer. For comparison, parsing all sentences of §22 in the PTB with the same trainer scored 89.58 (max-sum parsing with five cycles) with the standard training corpus and 85.21 when trained on §2–4.

curacy scores produced in parsing the PTB which, in our experiments, were 89.58 (using max-sum to parse all sentences with the grammar obtained after five cycles of training).

An obvious suspect for the difference in parsing accuracy with latent grammars between English and Korean is the difference in training set sizes. This turns out not to be the case. We learned latent annotations on sections 2–4 of the PTB and again tested on section 22. The accuracy scores on the test set peak at 85.21 (max-sum, all sentences, five cycles of training). This is about five points lower than the English grammar trained on sections 2–21, but is still over four points higher than the KTB results.

In the next section, we turn to one of the theoretical difficulties with Korean parsing with which we began the paper.

| **model** | **F1** | **$F1_{\leq 40}$** | **tokens** |
|---|---|---|---|
| English (standard training corpus) |  |  |  |
| coarse | 71.61 | 72.74 | 950K |
| w/ function tags | 72.82 | 74.05 | 950K |
| w/ NULLs | **73.29** | **74.38** | 1,014K |
| Korean |  |  |  |
| w/ verb ellipses | 52.85 | 56.52 | 3,200 |
| w/ traces | 55.88 | 59.42 | 3,868 |
| w/ r.c. markers | 56.74 | 59.87 | 3,794 |
| w/ zero pronouns | 57.56 | 61.17 | 4,101 |
| latent (5) w/ NULLs | 89.56 | 91.03 | 22,437 |

Table 5: Parser scores for Treebank PCFGs in English and Korean with NULL elements. *Tokens* denotes the number of words in the test data. The latent grammar was trained for five iterations.

## 5 NULL elements

Both the PTB and KTB include many NULL elements. For English, these elements are traces denoting constituent movement. In the KTB, there are many more kinds of NULL elements, in including trace markers, zero pronouns, relative clause reduction, verb deletions, verb ellipsis, and other unknown categories. Standard practice in English parsing is to remove NULL elements in order to avoid the complexity of parsing with $\epsilon$-productions. However, another approach to parsing that avoids such productions is to retain the NULL elements when reading the grammar; at test time, the parser is given sentences that contain markers denoting the empty elements. To evaluate, we remove these elements from the resulting parse trees output by the parser and compare against the stripped-down gold standard used in previous sections, in order to provide a fair point of comparison.

Parsing in this manner helps us to answer the question of how much easier or more difficult parsing would be if the NULL elements were present. In this section, we present results from a variety of experiments parsing will NULL tokens in this manner. These results can be seen in Table 5. The first observation from this table is that in English, retaining NULL elements makes a few points difference.

The first four rows of the KTB portion of Table 5 contains results with retaining different classes of NULL elements, one at a time, according to the manner described above. Restoring deleted pronouns and relative clause markers has the largest effect, suggesting that the absence of these optional elements removes key cues needed for parsing.

In order to provide a more complete picture of the effect of empty elements, we train the Berkeley latent annotation system on a version of the KTB in which all empty elements are retained. The final row of Table 5 contains the score obtained when evaluating parse trees produced from parsing with the grammar after the fifth iteration (after which performance began to fall). With the empty elements, we have achieved accuracy scores that are on par with the best accuracy scores obtained parsing the English Treebank.

## 6 Tree substitution grammars

We have shown that coarse labels and the prevalence of NULL elements in Korean both contribute to parsing difficulty. We now turn to grammar formalisms that allow us to work with larger fragments of parse trees than the height-one rules of standard context-free grammars. Tree substitution grammars (TSGs) have been shown to improve upon the standard English Treebank grammar (Bod, 2001) in parser accuracy, and more recently, techniques for inferring TSG subtrees in a Bayesian framework have enabled learning more efficiently representable grammars, permitting some interesting analysis (O'Donnell et al., 2009; Cohn et al., 2009; Post and Gildea, 2009). In this section, we try parsing the KTB with TSGs. We experiment with different methods of learning TSGs to see whether they can reveal any insights into the difficulties parsing Korean.

### 6.1 Head rules

TSGs present some difficulties in learning and representation, but a simple extraction heuristic called a *spinal grammar* has been shown to be very useful (Chiang, 2000; Sangati and Zuidema, 2009; Post and Gildea, 2009). Spinal subtrees are extracted from a parse tree by using a set of head rules to maximally project each lexical item (a word or morpheme). Each node in the parse tree having a different head from its parent becomes the root of a new subtree, which induces a spinal TSG derivation in the parse tree (see Figure 1). A probabilistic grammar is derived by taking counts from these trees, smoothing them with counts of all depth-one rules from the same training set, and setting rule probabilities to relative frequency.

This heuristic requires a set of head rules, which we present in Table 6. As an evaluation of our rules, we list in Table 7 the accuracy results for parsing with spinal grammars extracted using the head rules we developed as well as with two head rule heuristics (head-left and head-right). As a point of comparison, we provide the same results for English, using the standard Magerman/Collins head rules for English (Magerman, 1995; Collins, 1997). Function tags were retained for Korean but not for English.

We observe a number of things from Table 7. First, the relative performance of the head-left and head-right spinal grammars between English and Korean capture the linguistic fact that English is predominantly head-first and Korean is predominantly head-final. In fact, head-finalness in Korean was so strong that our head rules consist of only a handful of exceptions to it. The default rule makes heads of postpositions (case and information clitics) such as dative case marker and topic marker. It is these words that often have dependencies with words in the rest of the sentence. The exceptions concern predicates that occur in the sentence-final position. As an example, predicates in Korean are composed of several morphemes, the final one of which indicates the mood of the sentence. However, this morpheme often does not require any inflection to reflect long-distance agreement with the rest of the sentence. Therefore, we choose the morpheme that would be considered the root of the phrase, which in Korean is the verbalization/adjectivization suffix, verb, adjective, auxiliary predicate, and copula (XSV, XSJ, VV, VJ, VX, CO). These items often include the information about valency of the predicate.

| NT | RC | rule |
|---|---|---|
| S | SFN | second rightmost child |
| VV | EFN | rightmost XSV |
| VX | EFN | rightmost VJ or CO |
| ADJP | EFN | rightmost VJ |
| CV | EFN | rightmost VV |
| LV | EFN | rightmost VV |
| NP | EFN | rightmost CO |
| VJ | EFN | rightmost XSV or XSJ |
| VP | EFN | rightmost VX, XSV, or VV |
| ⋆ | ⋆ | rightmost child |

Table 6: Head rules for the Korean Treebank. NT is the nonterminal whose head is being determined, RC identifies the label of its rightmost child. The default is to take the rightmost child as the head.

Second, in both languages, finer-grained specification of head rules results in performance above that of the heuristics (and in particular, the head-left heuristic for English and head-right heuristic for Korean). The relative improvements in the two languages are in line with each other: significant, but not nearly as large as the difference between the head-left and head-right heuristics.

Finally, we note that the test results together suggest that parsing with spinal grammars may be a

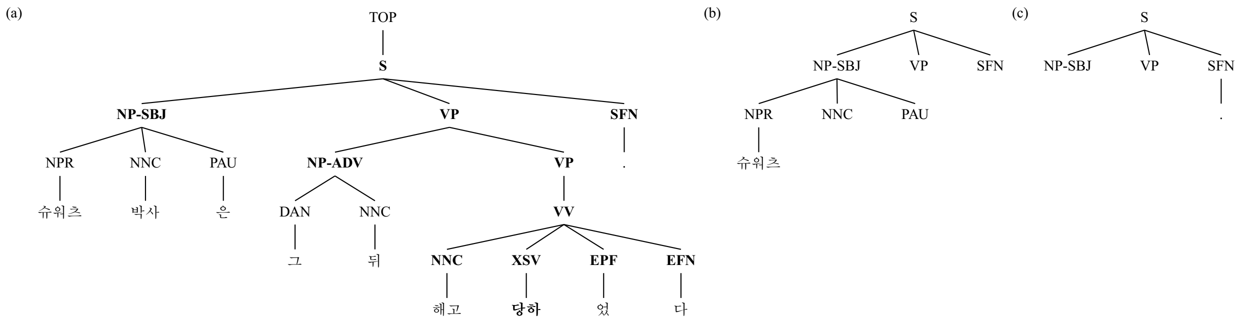

Figure 1: (a) A KTB parse tree; the bold nodes denote the top-level spinal subtree using our head selection rules. (b) The top-level spinal subtree using the head-left and (c) head-right extraction heuristics. A gloss of the sentence is *Doctor Schwartz was fired afterward.*

| **model** | **F1** | **F1$_{\leq 40}$** | **size** |
|---|---|---|---|
| Korean | | | |
| spinal (head left) | 59.49 | 63.33 | 49K |
| spinal (head right) | 66.05 | 69.96 | 29K |
| spinal (head rules) | 66.28 | 70.61 | 29K |
| English | | | |
| spinal (head left) | 77.92 | 78.94 | 158K |
| spinal (head right) | 72.73 | 74.09 | 172K |
| spinal (head rules) | 78.82 | 79.79 | 189K |

Table 7: Spinal grammar scores on the KTB and on PTB section 22.

good evaluation of a set of head selection rules.

## 6.2 Induced tree substitution grammars

Recent work in applying nonparametric machine learning techniques to TSG induction has shown that the resulting grammars improve upon standard English treebank grammars (O'Donnell et al., 2009; Cohn et al., 2009; Post and Gildea, 2009). These techniques use a Dirichlet Process prior over the subtree rewrites of each nonterminal (Ferguson, 1973); this defines a model of subtree generation that produces new subtrees in proportion to the number of times they have previously been generated. Inference under this model takes a treebank and uses Gibbs sampling to determine how to deconstruct a parse tree into a single TSG derivation. In this section, we apply these techniques to Korean.

This TSG induction requires one to specify a base measure, which assigns probabilities to subtrees being generated for the first time in the model. One base measure employed in previous work scored a subtree by multiplying together the probabilities of the height-one rules inside the subtree with a geometric distribution on the number of such rules. Since Korean is considered to be a free word-order language, we modified this base measure to treat the children of a height-one rule as a multiset (instead of a sequence). This has the effect of producing equivalence classes among the sets of children of each nonterminal, concentrating the mass on these classes instead of spreading it across their different instantiations.

To build the sampled grammars, we initialized the samplers from the best spinal grammar derivations and ran them for 100 iterations (once again, function tags were retained). We then took the state of the training data at every tenth iteration, smoothed together with the height-one rules from the standard Treebank. The best score on the development data for a sampled grammar was 68.93 (all sentences) and 73.29 (sentences with forty or fewer words): well above the standard Treebank scores from earlier sections and above the spinal heuristics, but well below the scores produced by the latent annotation learning procedures (a result that is consistent with English).

This performance increase reflects the results for English demonstrated in the above works. We see a large performance increase above the baseline Treebank grammar, and a few points above the best spinal grammar. One nice feature of these induced TSGs is that the rules learned lend themselves to analysis, which we turn to next.

## 6.3 Word order

In addition to the base measure mentioned above, we also experimented with the standard base mea-

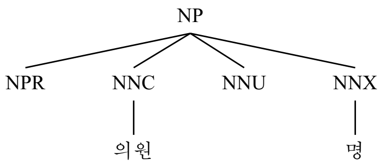

Figure 2: Example of a long distance dependency learned by TSG induction.

sure proposed by Cohn et al. and Post & Gildea, that treats the children of a nonterminal as a sequence. The grammars produced sampling under a model with this base measure were not substantively different from those of the unordered base measure. A partial explanation for this is that although Korean does permit a significant amount of reordering relative to English, the sentences in the KTB come from written newswire text, where word order is more standardized. Korean sentences are characterized as having a subject-object-verb (SOV) word order. There is some flexibility; OSV, in particular, is common in spoken Korean. In formal writing, though, SOV word order is overwhelmingly preferred. We see this reflected in the KTB, where SOV sentences are 63.5 times more numerous that OSV among sentences that have explicitly marked both the subject and the object. However, word order is not completely fixed even in the formal writing. NP-ADV is most likely to occur right before the VP it modifies, but can be moved earlier. For example,

S → NP-SBJ NP-ADV VP

is 2.4 times more frequent than the alternative with the order of the NPs reversed.

Furthermore, the notion of free(er) word order does not apply to all constituents. An example is nonterminals directly above preterminals. A Korean verb may have up to seven affixes; however, they always agglutinate in a fixed order.

### 6.4 Long distance dependencies

The TSG inference procedure can be thought of as discovering structural collocations in parse trees. The model prefers subtrees that are common in the data set and that comprise highly probable height-one rules. The parsing accuracy of these grammars is well below state of the art, but the grammars are smaller, and the subtrees learned can help us analyze the parse structure of the Treebank. One particular class of subtree is one that includes multiple lexical items with intervening nonterminals, which represent long distance dependencies that commonly co-occur. In Korean, a certain class of nouns must accompany a particular class of measure word (a morpheme) when counting the noun. In the example shown in Figure 2, (NNC 의원) (*members of assembly*) is followed by NNU, which expands to indicate ordinal, cardinal, and numeral nouns; NNU is in turn followed by (NNX 명), the politeness neutral measure word for counting people.

## 7 Summary & future work

In this paper, we addressed several difficult aspects of parsing Korean and showed that good parsing accuracy for Korean can be achieved despite the small size of the corpus.

Analysis of different parsing results from different grammatical formalisms yielded a number of useful observations. We found, for example, that the set of nonterminals in the KTB is not differentiated enough for accurate parsing; however, parsing accuracy improves substantially from latent annotations and state-splitting techniques that have been developed with English as a testbed. We found that freer word order may not be as important as might have been thought from basic a priori linguistic knowledge of Korean.

The prevalence of NULL elements in Korean is perhaps the most interesting difficulty in developing good parsing approaches for Korean; this is a key difference from English parsing that to our knowledge is not addressed by any available techniques. One potential approach is a special annotation of parents with deleted nodes in order to avoid conflating rewrite distributions. For example, S → VP is the most common rule in the Korean treebank after stripping away empty elements; however, this is a result of condensing the rule S → (NP-SBJ *pro*) VP and S → VP, which presumably have different distributions. Another approach would be to attempt automatic recovery of empty elements as a pre-processing step.

**Acknowledgments** We thank the anonymous reviewers for their helpful comments. This work was supported by NSF grants IIS-0546554 and ITR-0428020.

## References

Rens Bod. 2001. What is the minimal set of fragments that achieves maximal parse accuracy? In *Proc. ACL*, Toulouse, France, July.

Eugene Charniak. 1996. Tree-bank grammars. In *Proc. of the National Conference on Artificial Intelligence*, pages 1031–1036.

David Chiang. 2000. Statistical parsing with an automatically-extracted tree adjoining grammar. In *Proc. ACL*, Hong Kong.

Trevor Cohn, Sharon Goldwater, and Phil Blunsom. 2009. Inducing compact but accurate tree-substitution grammars. In *Proc. NAACL*.

Michael Collins. 1997. Three penerative, lexicalised models for statistical parsing. In *Proc. ACL/EACL*.

Thomas S. Ferguson. 1973. A Bayesian analysis of some nonparametric problems. *Annals of Mathematical Statistics*, 1(2):209–230.

Chung-Hye Han and Martha Palmer. 2004. A morphological tagger for Korean: Statistical tagging combined with corpus-based morphological rule application. *Machine Translation*, 18(4):275–297.

Na-Rae Han and Shijong Ryu. 2005. Guidelines for Penn Korean Treebank version 2.0. Technical report, IRCS, University of Pennsylvania.

Chung-hye Han, Na-Rae Han, and Eon-Suk Ko. 2001. Bracketing guidelines for Penn Korean Treebank. Technical report, IRCS, University of Pennsylvania.

Na-Rae Han. 2006. *Korean zero pronouns: analysis and resolution*. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA, USA.

Ulf Hermjakob. 2000. Rapid parser development: a machine learning approach for Korean. In *Proc. NAACL*, pages 118–123, May.

Mark Johnson. 1998. PCFG models of linguistic tree representations. *Computational Linguistics*, 24(4):613–632.

Laura Kallmeyer and SinWon Yoon. 2004. Tree-local MCTAG with shared nodes: Word order variation in German and Korean. In *Proc. TAG+7*, Vancouver, May.

Dan Klein and Chris Manning. 2003. Accurate unlexicalized parsing. In *Proc. ACL*.

Do-Gil Lee and Hae-Chang Rim. 2005. Probabilistic models for Korean morphological analysis. In *Companion to the Proceedings of the International Joint Conference on Natural Language Processing*, pages 197–202.

Sang-zoo Lee, Jun-ichi Tsujii, and Hae-Chang Rim. 2000. Hidden markov model-based Korean part-of-speech tagging considering high agglutinativity, word-spacing, and lexical correlativity. In *Proc. ACL*.

David M. Magerman. 1995. Statistical decision-tree models for parsing. In *Proc. ACL*.

Samuel E. Martin. 1992. *Reference Grammar of Korean: A Complete Guide to the Grammar and History of the Korean Language*. Tuttle Publishing.

Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2005. Probabilistic CFG with latent annotations. In *Proc. ACL*, Ann Arbor, Michigan.

Timothy J. O'Donnell, Noah D. Goodman, and Joshua B. Tenenbaum. 2009. Fragment grammar: Exploring reuse in hierarchical generative processes. Technical report, MIT.

Seong-Bae Park, Jeong-Ho Chang, and Byoung-Tak Zhang. 2004. Korean compound noun decomposition using syllabic information only. In *Computational Linguistics and Intelligent Text Processing (CICLing)*, pages 146–157.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proc. COLING/ACL*, Sydney, Australia, July.

Matt Post and Daniel Gildea. 2009. Bayesian learning of a tree substitution grammar. In *Proc. ACL*, Singapore, Singapore, August.

Detlef Prescher. 2005. Inducing head-driven PCFGs with latent heads: Refining a tree-bank grammar for parsing. *Machine Learning: ECML 2005*, pages 292–304.

Owen Rambow and Young-Suk Lee. 1994. Word order variation and tree-adjoining grammar. *Computational Intelligence*, 10:386–400.

Ines Rehbein and Josef van Genabith. 2007. Evaluating evaluation measures. In *Proceedings of the 16th Nordic Conference of Computational Linguistics (NODALIDA)*.

Federico Sangati and Willem Zuidema. 2009. Unsupervised methods for head assignments. In *Proc. EACL*.

Anoop Sarkar and Chung-hye Han. 2002. Statistical morphological tagging and parsing of Korean with an LTAG grammar. In *Proc. TAG+6*.