Machine Translation (2005) 18:275–297
DOI 10.1007/s10590-004-7693-4

A Morphological Tagger for Korean: Statistical Tagging Combined with Corpus-Based Morphological Rule Application

CHUNG-HYE HAN[1] and MARTHA PALMER[2]
[1]*Department of Linguistics, Simon Fraser University, 8888 University Dr., Burnaby, BC, V5A1SC, Canada. E-mail: chunghye@sfu.ca*
[2]*Department of Computer and Information Sciences, University of Pennsylvania, 3330 Walnut St., Philadelphia, PA 19104-6389, USA. E-mail: mpalmer@linc.cis.edu*

Abstract. This paper describes a novel approach to morphological tagging for Korean, an agglutinative language with a very productive inflectional system. The tagger takes raw text as input and returns a lemmatized and morphologically disambiguated output for each word: the lemma is labeled with a part-of-speech (POS) tag and the inflections are labeled with inflectional tags. Unlike the standard approach to tagging for morphologically complex languages, in our proposed approach the tagging phase precedes the analysis phase. It comprises a trigram-based tagging component followed by a morphological rule application component, obtaining 95% precision and recall on unseen test data.

Key words: agglutinative morphology, Korean, morphological rules, morphological tagger, statistical tagging, Treebank

1. Introduction

Korean is an agglutinative language with a very productive inflectional system. Inflections include different types of case markers, postpositions, prefixes, and suffixes on nouns, as in (1a); tense, honorific and sentence type markers on verbs and adjectives, as in (2a,b); among others. Furthermore, these inflections can combine with each other to form complex compound inflections. For example, postpositions, which correspond to English prepositions, can be followed by case markers such as nominative or accusative markers, as in (1b), or auxiliary postpositions such as 도 (*to* 'also') and 만 (*man* 'only'), as in (1c), which then can be followed by a topic marker, as in (1d).[1] Honorific and tense markers on verbs can be followed by a nominalizer and then a case marker or a postposition and a topic marker, as in (2c,d). Based on the possible inflectional sequences, we estimate that the number of possible morphological variants of a word can, in principle, be in the tens of thousands.

(1) a. 학교-가
hakkyo-ka
SCHOOL-nom
b. 학교-에서-가
hakkyo-eyse-ka
SCHOOL-FROM-nom
c. 학교-에서-만
hakkyo-eyse-man
SCHOOL-FROM-ONLY
d. 학교-에서-만-은
hakkyo-eyse-man-un
SCHOOL-FROM-ONLY-topic

(2) a. 가-었-다
ka-ess-ta
GO-past-decl
b. 가-시-었-다
ka-si-ess-ta
GO-honorific-past-decl
c. 가-기-가
ka-ki-ka
GO-nominalizer-nom
d. 가-시-었-기-에-는
ka-si-ess-ki-ey-nun
GO-honorific-past-nominalizer-TO-topic

This morphological complexity implies that for any NLP application on Korean to be successful, a component that does morphological analysis is necessary. Without it, any application that makes use of a computational lexicon would call for large and unnecessary space requirements, because it would require a lexicon that lists all possible morphological variants of a word; moreover, the development of applications such as a part-of-speech (POS) tagger and a parser based on statistical models would not be feasible due to the sparse data problem caused by the multiplicity of morphological variants of the same stem in the training data. Further, to best exploit the agglutinative nature of morphological structures, a tagger for Korean needs to be morpheme-based, rather than word-based. That is, an output of a tagger should consist of a segmentation of each word into its constituent morphemes and an assignment of a tag to each of them.

In this paper, we describe an implemented morphological tagger (that also does morphological analysis) for Korean that takes raw text as an input and returns a lemmatized and morphologically disambiguated output where for each word, the base-form is labeled with a POS tag and the inflections are labeled with inflectional tags. Throughout this paper, we will use the term "word" to mean character sequences separated by white spaces in the original raw text, and the term "lemma" to mean the stem or base-form of a word stripped off of all inflectional and derivational morphemes.

In the standard approach to tagging for morphologically complex languages like Korean, a morphological analysis phase precedes a tagging phase. That is, all possible ways of morphologically segmenting a given word are generated which are then disambiguated at the tagging phase (Lim et al., 1995, 1997; Hong et al., 1996; Chan et al., 1998; Yoon et al., 1999; Lee et al., 2000).[2] The main motivation behind applying morphological analysis prior to tagging is the sparse data problem which could potentially arise due to the fact that base forms of words can appear with a wide array of morphological affixes. However, this presupposes knowledge of all possible morphological rules, requiring quite a lot of time and effort for their construction and implementation. Furthermore, while the generation of all possible morphological analyses is not an especially onerous task, resolving the resulting ambiguity in the tagging phase is quite challenging. In contrast, in our proposed approach, the tagging phase precedes the analysis phase. It will be shown that even with the sparse data problem, by applying tagging before analysis, we can achieve accuracy results that are comparable to state-of-the-art systems. This is shown to be possible because tagging is done in two stages: (i) trigram-based tagging, and (ii) updating the tags for unknown words using morphological information automatically extracted from the training data. For us, the rich inflections are not seen as a problem causing sparse data, but rather they are exploited when guessing the tags for unknown words. As we will see in Section 2.4, our approach allows for morphological analysis to be done deterministically by using the information obtained from the tagging phase. This makes our approach efficient since there is no time-consuming morphological disambiguation step.

Our approach employs techniques from POS tagging to resolve ambiguity in morphological tagging and analysis. POS tagging was introduced by Church (1988), and Ratnaparkhi (1996) currently has one of the best performances for a POS tagger that has been trained on the Penn English Treebank (Marcus et al., 1993). Since English is not as morphologically complex as Korean, Ratnaparkhi models very limited suffix information to deal with unseen words. Taggers for morphologically rich languages like Czech, an inflectional language, and Hungarian, Basque and Turkish, agglutinative languages, have also been developed. Hajič and Hladká (1998) use an exponential probabilistic model for tag disambiguation after employing morphological analysis for Czech. Hajič et al. (2001) develop a hybrid system for Czech, where a rule-based system performs partial disambiguation, followed by the application of a trigram Hidden Markov Model (HMM) tagger. Tufiş et al. (2000) propose a two-step tagging process for Hungarian in which initial tagging is done using a reduced tag set with an off-the-shelf HMM tagger, followed by a recovery of the full morphosyntactic description. Hakkani-Tür et al. (2002) propose to break up the full morphological tag sequence into smaller units called inflectional groups, treating the members of each group as subtags, and then to determine the correct sequence of tags via statistical techniques. For Basque, Ezeiza et al. (1998) suggest the application, after initial morphological analysis, of morphological rules followed by an HMM-based tagger for disambiguation. In comparison, our model is based on a generative statistical model and a rule-based approach, and applies a trigram-based tagging component followed by a morphological rule application component.

We employed a corpus-based approach for the implementation of our morphological tagger, and automatically extracted the training data and morphological rules from the Penn Korean Treebank which contains 54,366 words and 5,078 sentences (Han et al., 2002). The Treebank is composed of 33 files in total, out of which 30 files were used for training data, and three files (files 05, 20, and 30, comprising 9% of the entire corpus) were set aside for test data. The Treebank has phrase-structure annotation, with head/phrase-level tags as well as function tags. Each word is morphologically analyzed, where the lemma and the inflections are identified. The lemma is tagged with a POS tag (e.g., NNC, NPN, VV, VX), and the inflections are tagged with inflectional tags (e.g., PCA, EAU, EFN).[3] An example annotation is given in (4) for the sentence in (3).

(3) 권한을 누가 갖고 있지?
kwenhan-ul-nwu-ka-kacko-iss-ci
AUTHORITY-acc WHO-nom HAVE BE-int
'Who has the authority?'

```
(4)  (S (NP-OBJ-1 권한/NNC+을/PCA)
        (S (NP-SBJ 누구/NPN+가/PCA)
           (VP (VP (NP-OBJ *T*-1)
                     가지/VV+고/EAU)
               있/VX+지/EFN))
        ?/SFN )
```

As we did not need to spend any time on manual construction of morphological rules, the implementation of the entire system was done quite efficiently, allowing us to develop a quick and effective technique for rapidly prototyping a working system.

The rest of the paper is organized as follows. In Section 2, we discuss the general approach, the training data, and the workings of each component underlying the system. Section 3 presents the results of the performance evaluation done on the tagger.

2. Description of the Morphological Tagger

The tagger follows several sequential steps for the labeling of the raw text with POS and inflectional tags. After tokenization (mostly applied to punctuation), all morphological contractions in the input string are decomposed (spelling recovery). The known words are then tagged (morph tagging) followed by a phase that updates the tags for any unknown words (update morph tagging). Finally, the lemma and its tags are separated from the inflections with their tags, creating the final output (lemma/inflection identification). These steps are summarized in Figure 1.

2.1. TOKENIZATION

When the raw text is fed through our tagger, the first task of the tagger is to separate out the punctuation symbols. This is a simple process in which white space is inserted between the word and the associated punctuation symbol.

2.2. SPELLING RECOVERY

In many cases, in the creation of an inflected word, a syllable or a character is deleted from the stem as in (5), and/or a morphological contraction occurs between the stem and the inflection as in (6).

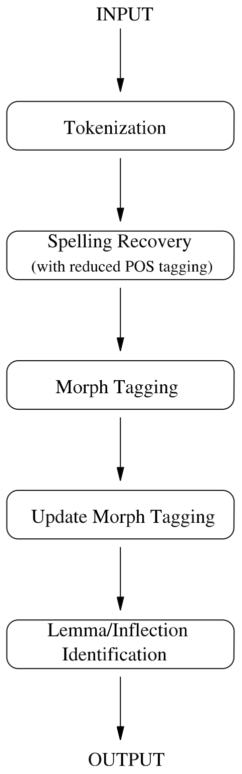

Figure 1. Overview of the tagger.

(5) a. 누 구+가 ⇒ 누 가
nwukwu+ka nwuka
WHO+nom
b. 덥+으 면 ⇒ 더 우 면
tep+umyen tewumyen
HOT+IF

(6) a. 가+았+다 ⇒ 갔 다
ka+ass+ta kassta
GO+past+decl
b. 무 엇+이+ㄴ 가 ⇒ 무 언 가
mwues+i+nka mwuenka
WHAT+co+int

In other words, there can be several allomorphs for a given morpheme, and thus an important part of the task for a morphological tagger/

analyzer is to identify the base-form of a morpheme, by recovering the materials that have been deleted, and decomposing the materials that have been contracted. We call this process "SPELLING RECOVERY."

To obtain training material for the spelling recovery component, the Treebank was converted to a data format containing a list of triples for each sentence: a word with morphological deletion and contraction, the corresponding string with the deletion and contraction undone, and a POS tag for the word. All the inflectional tags are stripped away, and all the POS tags are mapped onto one of the five tags: NN (noun), AN (noun modifier), VV (verb and adjective), AD (adverb and conjunction), or NC (noun with a copula inflection). This means that the tag for a word annotated with a noun POS tag and a case inflectional tag is reduced to NN, and the tag for a word annotated with a verb POS tag and a tense and sentence type inflectional tags is reduced to VV. Importantly, spelling recovery is conditioned on these five POS tags. Note that the Treebank makes finer distinctions in tagging, and has 34 tags in total. For instance, the Treebank has five noun tags: NPR for proper nouns, NNC for common nouns, NNX for dependent nouns, NPN for personal pronouns, NNU for numerals and NFW for words written in foreign characters. For the purposes of spelling recovery though, all the words belonging to the five noun categories behave the same way. The same goes for verb/adjective categories. Although the Treebank makes a three-way distinction for verb/adjective tags, all types of verbs and adjectives behave the same way with respect to spelling recovery. We thus simplified the rules involved in the spelling recovery process by collapsing 34 tags to five tags in order to condition the process.

We then automatically extracted 605 spelling recovery templates from the list of triples, by comparing the word and the corresponding string. If the two forms differ, they are listed from the point where the two forms start to differ, with the corresponding POS tag. For a small set of function words, entire words and the corresponding strings are listed. Some example spelling recovery templates are given in Table I. The first column contains "subwords" (which are the parts of words that are to be targeted for spelling recovery), the second column POS tags, and the third column substrings (which defines the strings the subwords will be converted into).[4]

The POS for each word is needed before we can attempt the spelling recovery step because this is constrained by POS tags. For instance, 동해 *tonghay* is ambiguous between a verb 'move' and a noun 'east sea', and it should have different spelling recovery outputs depending on its POS tag, as illustrated in (7).

Table I. Examples of spelling recovery templates

Subword	POS tag	Substring
누가 *nwuka*	NN	누구가 *nwukwuka*
그건 *kuken*	NN	그것은 *kukesun*
거라고 *kelako*	NC	것이라고 *kesilako*
갖고 *kacko*	VV	가지고 *kaciko*
골라 *kolla*	VV	고르어 *kolue*
쳤소 *chyessso*	VV	치었소 *chiessso*
추운 *chwuwun*	VV	춥은 *chwupun*
했습니다 *haysssupnita*	VV	하었습니다 *haesssupnita*

(7) a. 동해/NN ⇒ 동해
tonghay *tonghay*
'east sea'
b. 동해/VV ⇒ 동하어
tonghay *tonghae*
'move'

We use a statistical model to infer the most likely sequence of POS tags for the input word sequence. The training data is the word and its POS tag from the list of triples shown in Table I. This training data is used to create a probability model to infer the most likely tag sequence $\mathbf{t}^* = t_0^*, \ldots, t_n^*$ given an input word sequence $w_0, \ldots, w_n$. This probability model is used as a POS tagger which finds $\mathbf{t}^*$ by finding the tag sequence with the highest probability as in (8).

(8) $\mathbf{t}^* = \operatorname{argmax}_{t_0,\ldots,t_n} = P(t_0, \ldots, t_n \mid w_0, \ldots, w_n)$

This provides the most likely tag from the five possible POS tags for each word in a given input sentence, as in (9) (for gloss see (3)). Because the tag set at this stage contains only five tags, the sparse data problem is minimized.

(9) a. 권한을 누가 갖고 있지?
b. 권한을/NN 누가/NN 갖고/VV 있지/VV?

In our implementation, we used a standard trigram model combined with Katz backoff smoothing for unseen events. Unknown words are tagged as NN. The details of the model we use are explained in Appendix A.3. The code for the implementation of this trigram-based tagger was adapted and modified from the trigram-based SuperTagger by Srinivas (1997).[5]

In the spelling recovery phase, the input sentence is first run through the POS tagger, followed by the application of the spelling recovery templates to each word. At this point, each template is taken as an instruction, as follows:

> If the input word contains the subword and is tagged with the POS tag, then replace the part from the word matching the subword with the substring.

Some spelling recovery templates that are unambiguously verb/adjective templates are made to apply to input words tagged as NN as well as VV. This takes care of unknown verbs and adjectives that are wrongly tagged as NN.

An example input and output of spelling recovery is given in (10) (for gloss see (3)). The parts to which spelling recovery has been applied are underlined.

(10) a. Input:
권한을 누가 갖고 있지?

b. POS tagging:
권한을/NN 누가/NN 갖고/VV 있지/VV?

c. Output:
권한을 누구가 가지고 있지?

The spelling recovery phase was evaluated on the test sentences which were set aside from the Treebank. The results are given in Table II.

2.3. MORPHOLOGICAL TAGGING

Morphological tagging is done in two stages: first, input words are tagged with a trigram-based morphological tagger, and then the tag for unknown words is updated using what we call “inflectional templates”. Each step is described in more detail below.

Table II. Evaluation of spelling recovery

Process	Accuracy (%)
POS tagging for spelling recovery	92.03
Spelling recovery	96.33
Spelling recovery assuming 100% POS tagging	98.62

2.3.1. *Initial Morphological Tagging*

A morphological tagger based on a trigram-based model was trained on 91% of the 54k Korean Treebank.[6] The statistical model for this morphological tagger is the same as the one used for the POS tagger that was used in the spelling recovery phase. That is, given the input word sequence $w_0, \ldots, w_n$, the model is used to find the tag sequence $\mathbf{t}^* = t_0^*, \ldots, t_n^*$ with the highest probability. The only difference is that the morphological tagger uses a different set of tags. See Appendix A.3 for the details of the model. The tag set for the morphological tagger consists of possible complex tags extracted from the Treebank, altogether 165 complex tags. These complex tags are based on 14 POS tags, 10 inflectional tags, and five punctuation tags.[7] Examples are given in Table III. For example, NNC+CO+EFN+PAD is the complex tag for a common noun, inflected with a copula, a sentence type marker and a postposition. The tagger assigns an NNC (common noun) tag to unknown words, NNC being the most frequent tag for unknown words. Dealing with unknown words using inflectional information is discussed in Section 2.3.2.

Table III. Examples of complex tags

NNC
NNC+PCA
NNC+CO+EFN+PAD
NPR+PAD+PAU
NNU+PAU
VJ+EPF+EAN
VV+EPF+EFN
VV+EPF+EFN+PAU
VX+EPF+EFN+PAU

The tagging phase was evaluated on the test sentences, obtaining 81.96% accuracy. An example input and output of the morphological tagger is given in (11).

(11) a. Input:
제가 관측 사항을 보고하였습니다.
Cey-ka kwanchuk sahang-ul pokoha-ess-supnita.
I-nom OBSERVATION ITEM-acc REPORT-past-decl
'I reported the observation items.'

b. Output:
제가/NPN+PCA 관측/NNC 사항을/NNC+PCA
보고하었습니다/VV+EPF+EFN ./SFN

2.3.2. *Updating the Morphological Tags*

The results obtained from the trigram-based morphological tagger are not very accurate, the main reason being that both the number of unknown words and the number of tags in the tag set are quite large. This vindicates the discussion in Section 1 that argues for the importance of morphological analysis. Without morphological analysis, the tagging results will be quite low. To compensate for this, we added a phase that updates the morphological tags for unknown words. The update process exploits the fact that the POS tag of a given word is in general predictable from the inflections on the word. That is, a large number of inflections can only occur on verbs/adjectives, and a large number of inflections can only occur on nouns.

For this purpose, inflectional templates, which are pairs of inflection sequences and the corresponding complex tag, were extracted from the Treebank. The total number of extracted inflectional templates is 594. This comprises only 1% of the number of word tokens in the training data, showing that the size of inflectional templates is relatively quite small in comparison to the size of the training data. A few example templates are listed in Table IV. Further, a list of common noun stems was extracted from the Treebank.

The procedure for updating the morphological tags is shown in Figure 2.

An example of the tagging update process is given in (12). The main verb of the sentence 잊었습니다 *icesssupnita* ‘forgot’, which is inflected with a past-tense marker and a declarative sentence type marker, is wrongly tagged as NNC in the trigram-based tagging phase. This tag is updated as VV+EPF+EFN, VV for verb, EPF for tense, and EFN for sentence type.

Table IV. Examples of inflectional templates

었습니다 *esssupnita*	VV+EPF+EFN
었으니까 *essunikka*	VV+EPF+ECS
었으며 *essumye*	VV+EPF+ECS
었으므로 *essumulo*	VV+EPF+ECS
었음에 *essumey*	VV+EPF+ENM+PAD
까지는 *kkacinun*	NNC+PAD+PAU
대로만 *tayloman*	NNC+PAD+PAU
로부터의 *lopwuteuy*	NNC+PAD+PCA

1. For each input word that is tagged as an NNC, first check if the word is included in the common noun list:
   a) If the word is included in the common noun list, it is a known word, hence leave the tag as it is.
   b) If the word is not included in the common noun list, it is an unknown word, hence check the inflectional templates to update the tag.
      i) Match the inflection sequence with the word, starting from the right side. If there are matching inflection sequences, choose the complex tag paired with the longest matching inflection sequence, and replace the NNC tag with that complex tag. That is, adopt a right-to-left longest match requirement.
      ii) If there is no matching inflection sequence, leave the complex tag as NNC.
2. For those words that are not tagged simply as NNC, leave their tags as they are.

Figure 2. Procedure for updating the morphological tags.

(12) a. Input:
제가/NPN+PCA 관측/NNC 사항을/NNC+PCA
잊었습니다/NNC
Cey-ka kwanchuk sahang-ul ic-ess-supnita.
I-nom OBSERVATION ITEM-acc FORGET-past-decl
'I forgot the observation items.'
b. Output:
제가/NPN+PCA 관측/NNC 사항을/NNC+PCA
잊었습니다/VV+EPF+EFN ./SFN

After the tags for unknown words were updated, the morphological tagging accuracy on the test set increased to 93.86%, a 12-point increase from the trigram-based morphological tagging. If the shortest match is employed to the tagging update process, the accuracy reduces to 90.29%.

2.4. LEMMA/INFLECTION IDENTIFICATION

Lemma/inflection identification produces the final output of our morphological tagger. Given a pair of a word and its complex tag, the complex tag is decomposed and each constituent tag is paired up with the corresponding morpheme in the word. That is, the lemma and its POS tag are paired up, and then each inflection is paired up with the corresponding inflectional tag. This is illustrated in (13), where *a*, *b*, *c* and *d* stand for substrings of a word, *p* stands for a POS tag, and *x*, *y* and *z* stand for inflectional tags.

Table V. Example entries from inflection and stem dictionaries

Inflection dictionary		Stem dictionary	
ㅂ시다 (*psita*)	EFN	간격 (*kankyek*)	NN
ㅂ시오 (*psio*)	EFN	간단하 (*kantanha*)	VJ
가 (*ka*)	PCA	간략히 (*kanlyakhi*)	ADV
겠 (*keyss*)	EPF	간선 (*kansen*)	NNC
다 (*ta*)	EFN	간섭하 (*kansepha*)	VV
보다 (*pota*)	PAD	간섭 (*kansep*)	NNC
았 (*ass*)	EPF	간주하 (*kancwuha*)	VV
은 (*un*)	PAU	갈 (*kal*)	VV

(13) $abcd/p+x+y+z$

$$\Downarrow$$

$a/p+b/x+c/y+d/z$

The complex tag in conjunction with the inflection and stem dictionary (extracted from the Treebank) look-up are used to determine the lemma/inflection identification process. The inflection dictionary contains 230 entries in total, and the stem dictionary contains 3,880 entries in total. Example entries from the inflection dictionary and stem dictionary are given in Table V. The list in the stem dictionary is small because the size of the Treebank itself is small. It is possible to include a much larger stem dictionary, using resources from outside the Treebank, to enhance the overall performance of the tagger. We however did not do so for present purposes in order to be faithful to the corpus-based approach we have adopted; the data used for the implementation and subsequently the evaluation of the tagger is restricted to the Treebank. An example input and output of the lemma/inflection identification is given in (14) (for gloss see (11)).

(14) a. Input:
제가/NPN+PCA 관측/NNC 사항을/NNC+PCA
보고하었습니다/VV+EPF+EFN ./SFN

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

The procedures for lemma/inflection identification are shown in Figure 3.

Given a pair of a word and its complex tag ($abcd$, $p+x+y+z$):[8]

1. Separate the substring from the input word that matches the longest inflection in the inflection dictionary associated with the last inflectional tag in the input complex tag, and label it with the inflectional tag (e.g., d/z);
   If there is no matching inflection in the inflection dictionary, separate the substring that matches the longest inflection in the inflection dictionary associated with the second-to-last inflectional tag, and label it with the inflectional tag (e.g., d/y);
2. Repeat the above procedure until the POS tag (i.e., the first tag) is reached. The remaining substring is labeled with the POS tag (e.g., a/p).

Figure 3. Procedures for lemma/inflection identification.

In this section we showed that the combination of corpus-based rules and the simple algorithm given above applied to morphologically tagged data is sufficient for the usually complex task of lemma/inflection identification.[9] Hoping to improve the performance, we have also implemented a variation of this technique using corpus-particular knowledge such as special cases for POS tags like NNC, VV, and VJ based on our intuitions. The performance however did not improve much. The performance results are given in Table VI.

3. Evaluation and Discussion

3.1. EVALUATION OF THE MORPHOLOGICAL TAGGER

The performance of the morphological tagger trained on the Treebank has been evaluated on 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 gold file contains hand-corrected annotations that are used to evaluate the accuracy of the tagger output. The precision corresponds to the percentage of morpheme/tag pairs in the gold file that match the morpheme/tag pairs in the test file. The recall corresponds to the percentage of morpheme/tag pairs in the test file that match the morpheme/tag pairs in the gold file. Our approach yielded 95.43% precision and 95.04% recall for the Treebank-trained tagger. Further, using the corpus-specific lemma/inflection

Table VI. Evaluation of the morphological tagger

Method	Precision (%)	Recall (%)
Treebank-trained	95.43	95.04
Treebank-trained^{+}	95.78	95.39

Table VII. Comparison with other morphological taggers

Reference	Accuracy (%)
Chan et al. (1998)	97.0
Yoon et al. (1999)	94.7
Lim et al. (1997)	94.8

identification procedure in the Treebank-trained$^+$ tagger, the performance increased only very slightly, yielding precision and recall scores of 95.78% and 95.39%, respectively.

The performance obtained is comparable to the results reported for state-of-the-art systems. Table VII shows accuracy scores reported for instance by systems such as those described in Chan et al. (1998), Yoon et al. (1999), and Lim et al. (1997), when tested on the test set drawn from the same corpus each was trained on. Ideally, for a fair comparison, it would be desirable to retrain other systems on the Penn Korean Treebank and compare their performances with the proposed tagger on the same test set, or retrain the proposed tagger on the same domain as the other taggers and compare the results on the same test set. Unfortunately, both options are not realistic given that neither the training data nor the source code for the other taggers are publicly available.

3.2. ERROR ANALYSIS

The sources for errors in our system can be divided into six major types: tagging, spelling recovery and ambiguity resolution errors, and missing entries from the common noun list, the inflectional template list and the inflection dictionary. The most frequent type of error was a tagging error, comprising 63% of total errors. In examples with tagging errors, the lemma or the inflection is mistagged. The second most frequent type of error was caused by incorrect spelling recovery, comprising 21% of total errors. If all morphological contractions in the input string are not correctly decomposed, the input will be misanalyzed and mistagged. Ambiguity resolution errors are mostly caused by our longest match strategy. Given a pair of a word and a complex tag (abc, $x + y$), if the inflection dictionary contains an entry where bc is associated with y and another entry where c is associated with y, then the possible analyses for abc are $ab/x + c/y$ and $a/x + bc/y$. The lemma/inflection identification procedure will always select the analysis where the longest substring is identified as an inflection, selecting the second analysis. Thus, an error will result if the correct analysis is the first one. This type of error covered 16% of total errors. Further, in a small number of cases, missing entries from the common noun list, the inflectional template list and the inflection dictionary resulted in errors. The number and the percentage of the types of errors from the test data is summarized in Table VIII, along with examples.

4. Conclusion

In this paper, we describe an implemented morphological tagger (that also does morphological analysis) for Korean, a language with a rich inflectional system. The training data for the trigram-based tagging component, and morphological rules, inflection and stem dictionaries for morphological tagging and analysis are automatically extracted from an annotated corpora, the Penn Korean Treebank. This corpus-based approach enabled us to implement the entire system in a short period of time (roughly eight

Table VIII. Summary of error analysis

Type of error	Number	(%)	Gold	Test
Tagging	174	63	정직하/VJ+지/ECS *cengcikha-ci* HONEST-connective ‘be honest’	정직하/VV+지/ECS *cengikha-ci*
Spelling recovery	57	21	피우/VV+ㄹ/EAN *piwu-l* EMINATE-adnominal ‘which will eminate’	피울/NNC *piwul*
Ambiguity	16	6	부대/NNC+로/PAD *pwutay-lo* SQUAD-TO ‘to the squad’	부/NNC+대로/PAD *pwu-taylo*
Missing NNC	15	5	학도/NNC *hakto* CADET ‘cadet’	학/NNC+도/PAU *hak-to*
Missing template	11	4	훈련/NNC+이/CO+군/EFN *hwunlyun-i-kwun* DRILL-cop-decl ‘be a drill’	훈련이군/NNC *hwunlyunikwun*
Missing inflection	4	1	소대급/NNC+에까지/PAD *sotaykup-eykkaci* PLATOON LEVEL-TOWARDS ‘towards the platoon level’	소대급에/NNC+까지/PAD *sotaykupe-kkaci*

weeks), with resulting performance in the range of 95% precision/recall on the test set from the same domain as the training data. The treebank-trained morphological tagger performed as well as state-of-the-art taggers whose training data are from different domains, even though, in contrast to other taggers, in our system the analysis phase follows the tagging phase. This was possible because of the two-stage morphological tagging process we employed. In particular, in the second tagging stage, the morphological tags for the unknown words are updated, exploiting the inflections by matching these against information stored in inflectional templates. Our tagger has been successfully incorporated into a Korean statistical parser (Sarkar and Han, 2002), and a Korean–English machine translation system as part of the source language analysis component (Palmer et al., 2002). Our approach is applicable to any language, as long as a corpus annotated with morphological information is available. Although the training data and the morphological rules will be corpus-specific, and hence language-specific, the general approach and the procedures that we have proposed for spelling recovery, updating morphological tags, and lemma/inflection identification as well as trigram-based tagging are language-neutral, and can be applied to any agglutinative language. In the future, we would like to test the portability of our approach by first retraining our tagger on a larger corpus, and second by applying our approach to morphological tagger development for other languages with rich morphology.

Appendix A. A List of Tags used in the Penn Korean Treebank

A.1. POS TAGS

NPR: proper noun: 한국 (*hankwuk* ‘Korea’), 클린톤 (*kullinthon* ‘Clinton’)

NNC: common noun: 학교 (*hakkyo* ‘school’), 컴퓨터 (*kemphyute* ‘computer’)

NNX: dependent noun: 것 (*kes* ‘thing’), 등 (*tung* ‘etc’), 년 (*nyen* ‘year’)

NPN: personal pronoun, demonstratives: 그 (*ku* ‘he’), 이것 (*ikes* ‘this’)

NNU: ordinal, cardinal, numeral: `1`, 이것 (*hana* ‘one’), 첫째 (*chesccay* ‘first’)

NFW: words written in foreign characters: `Clinton`, `computer`

VV: verb: 가 (*ka* ‘go’), 먹 (*mek* ‘eat’)

VJ: adjective: 예쁘 (*yeppu* ‘pretty’), 다르 (*talu* ‘different’)

VX: auxiliary predicate: 있 (*iss* ‘present progressive’), 하 (*ha* ‘must’)

ADV: constituent and clausal adverb: 매우 (*maywu* ‘very’), 조용히 (*coyonghi* ‘quietly’), 만일 (*manil* ‘if’)

ADC: conjunctive adverb: 그리고 (*kuliko* ‘and’), 그러나 (*kulena* ‘but, however’)

DAN: configurative, demonstrative: 새 (*say* ‘new’), 헌 (*hen* ‘old’), 그 (*ku* ‘that’)

IJ: exclamation: 아 (*a* ‘ah’)

LST: list marker: `a`, `(b)`, `1`, `2.3.1`, 가 (*ka*), 나. (*na.*)

A.2. INFLECTIONAL TAGS

PCA: case marker: 가/이 (*ka/i* nominative), 을/를 (*ul/lul* accusative), 의 (*uy* possessive), 야 (*ya* vocative)
PAD: adverbial postposition: 에서 (*yese* 'from'), 로 (*lo* 'to')
PCJ: conjunctive postposition: 와/과 (*wa/kwa* 'and'), 하고 (*hako* 'and')
PAU: auxiliary postposition: 만 (*man* 'only'), 도 (*to* 'also'), 는 (*nun* topic)
CO: copula: 이 (*i* 'be')
EFN: sentence type marker: 는다/ㄴ다 (*nunta/nta* declarative), 어라/라 (*ela/la* imperative), 니/는가/는지 (*ni/nunka/nunci* interrogative), 자 (*ca* propositive)
ECS: coordinate, subordinate, adverbial, complementizer: 고 (*ko* 'and'), 므로 (*mulo* 'because'), 게 (*key* attaches to adjectives to derive adverbs), 다고 (*tako* 'that')
EAU: auxiliary, on verbs or adjectives that immediately precede auxiliary predicates: 아 (*a*), 게 (*key*), 지 (*ci*), 고 (*ko*)
EAN: adnominal, on main verbs or adjectives in relative clauses or complement clauses of a complex NP: 는/ㄴ (*nun/n*)
ENM: nominal, on nominalized verb: 기 (*ki*), 음 (*um*)
EPF: pre-final ending: 었 (*ess* past), 시 (*si* honorific)
XSF: suffix: 님 (*nim*), 들 (*tul*), 적 (*cek*)
XPF: prefix: 제 (*cey*), 각 (*kak*), 매 (*may*)
XSV: verbalization suffix: 하 (*ha*), 되 (*toy*), 시키 (*siki*)
XSJ: adjectivization suffix: 스럽 (*sulep*), 답 (*tap*), 하 (*ha*)

A.3. PUNCTUATION TAGS

SCM: comma: ,
SFN: sentence ending markers: . ? !
SLQ: left quotation mark: ` (" {
SRQ: right quotation mark: ′) ″ }
SSY: other symbols: ... ; : -

Appendix B. Description of the Statistical Model for Tagging

In this appendix, we provide details of the statistical model we used in two different steps in our overall morphological tagging algorithm. The only difference in the two uses of the model was the change in the tag set used. In Section 2.2, we used a tag set of five tags in order to perform the task of spelling recovery, while in Section 2.3.1, we used a tag set of 165 tags in order to perform the full task of tagging each morpheme.

Let the input sentence (word sequence) be $\mathbf{w} = w_0, w_1, \ldots, w_n$. Let the most likely tag sequence be $\mathbf{t}^* = t_0^*, t_1^*, \ldots, t_n^*$. Given an input word sequence $\mathbf{w}$ we want to find the most likely tag sequence, which we do by constructing a probability model which provides a method for comparing all possible tag sequences given the input word sequence, $P(t_0, t_1, \ldots, t_n \mid w_0, w_1, \ldots, w_n)$.
The best (or most likely) tag sequence is (1).

$$\mathbf{t}^* = \text{argmax}_{t_0,\ldots,t_n} P(t_0, \ldots, t_n \mid w_0, \ldots, w_n) \tag{1}$$

In order to estimate this from our training data, we use Bayes rule to convert our conditional probability into a generative model (2). We are comparing tag sequences for the same input word sequence, and so we can ignore the denominator $P(w_0, \ldots, w_n)$ (3) since it does not contribute to the distinction between different tag sequences.

$$\begin{aligned} &\text{argmax}_{t_0,\ldots,t_n} P(t_0, \ldots, t_n \mid w_0, \ldots, w_n) \\ &= \text{argmax}_{t_0,\ldots,t_n} \frac{P(w_0, \ldots, w_n \mid t_0, \ldots, t_n) \times P(t_0, \ldots, t_n)}{P(w_0, \ldots, w_n)} && (2) \\ &= \text{argmax}_{t_0,\ldots,t_n} P(w_0, \ldots, w_n \mid t_0, \ldots, t_n) \times P(t_0, \ldots, t_n) && (3) \end{aligned}$$

The two terms in this equation cannot be reasonably estimated due to sparse data problems. We apply the Markov assumption to simplify each term. For the first term that involves lexical probabilities, the assumption made is that the probability of generating a word is dependent only on its POS tag (4).

$$\begin{aligned} &P(w_0, \ldots, w_n \mid t_0, \ldots, t_n) \\ &= P(w_0 \mid t_0) \times P(w_1 \mid t_1) \times \cdots \times P(w_n \mid t_n) \\ &= \prod_{i=0}^{n} P(w_i \mid t_i) && (4) \end{aligned}$$

The second term is the model that produces POS tag sequences. We use the Markov assumption to generate tag sequences using a trigram model, where each tag is generated based on the previous two tags (5).

$$\begin{aligned} &P(t_0,\ldots,t_n)\\ &= P(t_0)\times P(t_1\,|\,t_0)\times P(t_2\,|\,t_0,t_1)\times\cdots\times P(t_n\,|\,t_{n-2},t_{n-1}) \qquad (5)\\ &= P(t_0)\times P(t_1\,|\,t_0)\times\prod_{i=2}^{n} P(t_i\,|\,t_{i-2},t_{i-1}) \end{aligned}$$

Putting these two terms back into the original equation we get (6):

$$\begin{aligned} &\text{argmax}_{t_0,\ldots,t_n} P(t_0,\ldots,t_n\,|\,w_0,\ldots,w_n)\\ &= \text{argmax}_{t_0,\ldots,t_n} P(w_0,\ldots,w_n\,|\,t_0,\ldots,t_n)\times P(t_0,\ldots,t_n)\\ &= \text{argmax}_{t_0,\ldots,t_n}\left(\prod_{i=0}^{n} P(w_i\,|\,t_i)\right)\times\\ &\quad\left(P(t_0)\times P(t_1\,|\,t_0)\times\prod_{i=2}^{n} P(t_i\,|\,t_{i-2},t_{i-1})\right) \qquad (6)\\ &= \text{argmax}_{t_0,\ldots,t_n}\prod_{i=0}^{n} P(w_i\,|\,t_i)\times P(t_i\,|\,t_{i-2},t_{i-1}) \qquad (7) \end{aligned}$$

In order to simplify our Equation from (6) to (7), we add special tokens of type <bos> to the beginning of each sentence in order to condition the first word on these tokens. We also add tokens of type <eos> to the end of the sentence.

Now that we have the model in (7), all we need to do to find the most likely tag sequence is to *train* the following two probability models, $P(w_i\,|\,t_i)$ and $P(t_i\,|\,t_{i-2},t_{i-1})$.

This is done using our training data which consists of word and tag sequences. To find the most likely tag sequence, we use the same algorithm used to find the best tag sequence in HMM: the Viterbi algorithm.

The model however cannot handle unseen data, such as unseen words w_i; unseen word–tag combinations (w_i, t_i); and unseen trigrams (t_{i-2}, t_{i-1}, t_i). For unseen words, we rename those words with singleton counts in our training data as a special <unseen> token. Due to this model, as stated in Section 2.3.1, the trigram tagger for morphological tagging assigns the NNC (common noun) tag to unknown words by default, NNC being the most frequent tag for unknown words. The update tagging step (see Section 2.3.2) does further disambiguation for the unknown words. We also use suffix and prefix features from the words and use it in the original model as was done in Weischedel et al. (1993). For unseen tri-tag sequences in the trigram model of tag sequences, we use Katz backoff smoothing (Katz, 1987) which uses bigrams of tag sequences when the trigram sequence was unobserved in our training data. The discount factor for the bigram sequence is computed using the Good–Turing estimate (Good, 1953) which is the standard practice in Katz backoff smoothing.

Acknowledgements

We thank Anoop Sarkar and Aravind Joshi for valuable discussions on many occasions. We are also grateful to the anonymous reviewer whose critical comments helped to reshape and improve this paper. All errors are ours. This work was supported by IRCS, and contract DAAD 17-99-C-0008 awarded by the Army Research Lab to CoGenTex, Inc., with the University of Pennsylvania as a subcontractor. In addition, the first author was supported by SSHRC #410-2003-0544.

Notes

[1] Hangul characters are romanized following the Yale Romanization Convention throughout the paper. In this and subsequent examples, the following abbreviations are used: acc(usative case marker), co(pula inflection), decl(arative), int(errogative), nom(inative case marker).

[2] Further relevant works, all in Korean, which the present authors were unfortunately unable to locate, and for which incomplete references are provided, include a presentation by E. C. Lee and J. H. Lee at the 4th Conference of Hangul and Korean Information Processing, a journal paper by J. H. Choi and S. J. Lee in the 1993 *Journal of Korea Information Science Society*, and Masters theses by S. Y. Kim, from KAIST in 1987, and by H. S. Lim from Korea University in 1994.

[3] NNC is a tag for common nouns, NPN for pronouns, VV for verbs, and VX for auxiliary verbs. PCA is a tag for case markers, EAU for inflections on a verb followed by an auxiliary verb, and EFN for sentence-type inflections. The tags used in the Penn Korean Treebank are listed in Appendix A. See Han and Han (2001) for further explanations of these tags.

[4] The spelling recovery templates are similar to the morphographemic rewrite rules found in traditional morphological analyzers. The difference however is that while traditional morphographemic rewrite rules are conditioned on morphological notions such as morpheme boundaries, hence presupposing a segmentation of a word into its component morphemes, our spelling recovery templates are conditioned on the POS tag assigned to the entire word.

[5] Srinivas's 1997 SuperTagger is publicly available from the XTAG Project webpage: `http://www.cis.upenn.edu/~xtag`.

[6] While the Korean Treebank is smaller than the usual data set used in English POS tagging (the 1m-word Penn English Treebank (Marcus et al., 1993)) the size of the Treebank is big enough to provide statistically significant results. Hence, we did not find the need to conduct *n*-fold cross validation experiments.

[7] The tag set used in our morphological tagger is a subset of the Korean Treebank tag set. In the tag set for the tagger, EAU and ECS are collapsed to ECS, and affixal tags such as XPF, XSF, XSV and XSJ are not included. In particular, in the training data, the POS tag for words with XSV tag were converted to VV, and XSJ tag were converted to VJ.

[8] The longest match requirement employed in lemma/inflection identification is similar to the one employed in a finite-state transducer with Directed Replacement, as described in Karttunen (1996).

[9] A reviewer asks if we assume that all morphological processes are via overt suffixes. Although there may be empirical and theoretical motivations for positing null morphemes in Korean, we did not model this in our system. The main reason for not doing so is that the Treebank we used did not have any annotations for null morphemes.

References

Chan, Jeongwon, Geunbae Lee, and Jong-Hyeok Lee: 1998, ‘Generalized Unknown Morpheme Guessing for Hybrid POS Tagging of Korean’, in *Proceedings of the Sixth Workshop on Very Large Corpora*, Montreal, Canada, pp. 85–93.

Church, Kenneth: 1988, ‘A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text’, *Computer Speech and Language* **5**, 19–54.

Ezeiza, N., I. Alegria, J. M. Arriola, R. Urizar, and I. Aduriz: 1998, ‘Combining Stochastic and Rule-based Methods for Disambiguation in Agglutinative Languages’, in *COLING-ACL ’98: 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, Montreal, Canada, pp. 379–384.

Good, I. J.: 1953, ‘The Population Frequencies of Species and the Estimation of Population Parameters’, *Biometrika* **40**, 237–264.

Hajič, Jan and Barbora Hladká: 1998, ‘Tagging Inflective Languages: Prediction of Morphological Categories for a Rich, Structured Tagset’, in *COLING-ACL ’98: 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, Montreal, Canada, pp. 483–490.

Hajič, Jan, Pavel Krbec, Pavel Květoň, Karel Oliva, and Vladimír Petkevič: 2001, ‘Serial Combination of Rules and Statistics: A Case Study in Czech Tagging’, in *Association for Computational Linguistics 39th Annual Meeting and 10th Conference of the European Chapter*, Toulouse, France, pp. 260–267.

Hakkani-Tür, Dilek Z., Kemal Oflazer, and Gökhan Tür: 2002, ‘Statistical Morphological Disambiguation for Agglutinative Languages’, *Computers and the Humanities* **36**, 381–410.

Han, Chung-hye and Na-Rae Han: 2001, ‘Part of Speech Tagging Guidelines for Penn Korean Treebank’, IRCS Report 01-09, IRCS, University of Pennsylvania.

Han, Chung-hye, Na-Rare [sic] Han, Eon-Suk Ko, and Martha Palmer: 2002, ‘Development and Evaluation of a Korean Treebank and its Application to NLP’, in *LREC 2002: Third International Conference on Language Resources and Evaluation*, Las Palmas de Gran Canaria, Spain, pp. 1635–1642.

Hong, Y., M. W. Koo, and G. Yang: 1996, ‘A Korean Morphological Analyzer for Speech Translation System’, in *ICSLP 96: The Fourth International Conference on Spoken Language Processing*, Philadelphia, PA, pp. 676–679.

Karttunen, Lauri: 1996, ‘Directed Replacement’, in *34th Annual Meeting of the Association for Computational Linguistics*, Santa Cruz, California, pp. 108–115.

Katz, Slava M.: 1987, ‘Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer’, *IEEE Transaction on Acoustics, Speech and Signal Processing* **35**, 400–401.

Lee, Sang-Zoo, Jun-ichi Tsujii, and Hae-Chang Rim: 2000, ‘Lexicalized Hidden Markov Models for Part-of-speech Tagging’, in *Proceedings of the 18th International Conference on Computational Linguistics, COLING 2000 in Europe*, Saarbrücken, Germany, pp. 481–487.

Lim, Hewui Seok, Jin-Dong Kim, and Hae-Chang Rim: 1997, ‘A Korean Part-of-speech Tagger using Transformation-based Error-driven Learning’, in *Proceedings of the 1997 International Conference on Computer Processing of Oriental Languages*, Hong Kong, pp. 456–459.

Lim, Heui-Suk, Sang-Zoo Lee, and Hae-Chang Rim: 1995, ‘An Efficient Korean Morphological Analyzer Using Exclusive Information’, in *International Conference on Computer Processing of Oriental Languages, ICCPOL ’95*, Honolulu, HI.

Marcus, Mitch, Beatrice Santorini, and M. Marcinkiewicz: 1993, ‘Building a Large Annotated Corpus of English’, *Computational Linguistics* **19**, 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.

Ratnaparkhi, Adwait: 1996, ‘A Maximum Entropy Model for Part-of-speech Tagging’, in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Philadelphia, PA, pp. 133–142.

Sarkar, Anoop and Chung hye Han: 2002, ‘Statistical Morphological Tagging and Parsing of Korean with an LTAG Grammar’, in *Proceedings of the 6th International Workshop on Tree Adjoining Grammars and Related Formalisms TAG+6*, Venice, Italy, pp. 48–56.

Srinivas, B.: 1997, ‘Complexity of Lexical Descriptions and its Relevance to Partial Parsing’. Ph.D. thesis, Department of Computer and Information Sciences, University of Pennsylvania.

Tufiş, Dan, Péter Dienes, Csaba Oravecz, and Tamás Váradi: 2000, ‘Principled Hidden Tagset Design for Iterated Tagging of Hungarian’, in *LREC 2000: 2nd International Conference on Language Resources and Evaluation*, Athens, Greece, pp. 1421–1426.

Weischedel, Ralph, Richard Schwartz, Jeff Palmucci, Marie Meteer, and Lance Ramshaw: 1993, ‘Coping with Ambiguity and Unknown Words through Probabilistic Models’, *Computational Linguistics* **19**, 359–382.

Yoon, Juntae, C. Lee, S. Kim, and M. Song (윤준태, 이충희, 김선호, 송만석): 1999, ‘연세대 형태소 분석기 Morany: 말뭉치로부터 추출한 대량의 어휘 데이타베이스에 기반한 형태소 분석’ [Morphological Analyzer of Yonsei University Morany: Morphological Analysis based on Large Lexical Database Extracted from Corpus], in *Proceedings of the 11th Conference on Hangul and Korean Language Information Processing* (제11회 한글 및 국어정보처리 학술대회 논문집), pp. 92–98.