Developing Methodology for Korean Particle Error Detection

Markus Dickinson
Indiana University
md7@indiana.edu

Ross Israel
Indiana University
raisrael@indiana.edu

Sun-Hee Lee
Wellesley College
slee6@wellesley.edu

Abstract

We further work on detecting errors in post-positional particle usage by learners of Korean by improving the training data and developing a complete pipeline of particle selection. We improve the data by filtering non-Korean data and sampling instances to better match the particle distribution. Our evaluation shows that, while the data selection is effective, there is much work to be done with preprocessing and system optimization.

1 Introduction

A growing area of research in analyzing learner language is to detect errors in function words, namely categories such as prepositions and articles (see Leacock et al., 2010, and references therein). This work has mostly been for English, and there are issues, such as greater morphological complexity, in moving to other languages (see, e.g., de Ilarraza et al., 2008; Dickinson et al., 2010). Our goal is to build a machine learning system for detecting errors in post-positional particles in Korean, a significant source of learner errors (Ko et al., 2004; Lee et al., 2009b).

Korean postpositional particles are morphemes that attach to a preceding nominal to indicate a range of linguistic functions, including grammatical functions, e.g., subject and object; semantic roles; and discourse functions. In (1), for instance, *ka* marks the subject (function) and agent (semantic role).[1] Similar to English prepositions, particles can also have modifier functions, adding meanings of time, location, instrument, possession, and so forth.

(1)	Sumi-**ka**	John-**uy**	cip-**eyse**	ku-**lul**	twu	sikan-**ul**	kitaly-ess-ta.
	Sumi-SBJ	John-GEN	house-LOC	he-OBJ	two	hours-OBJ	wait-PAST-END

'Sumi waited for John for (the whole) two hours in his house.'

We treat the task of particle error detection as one of particle selection, and we use machine learning because it has proven effective in similar tasks for other languages (e.g., Chodorow et al., 2007; Oyama, 2010). Training on a corpus of well-formed Korean, we predict which particle should appear after a given nominal; if this is different from the learner's, we have detected an error. Using a machine learner has the advantage of being able to perform well without a researcher having to specify rules, especially with the complex set of linguistic relationships motivating particle selection.[2]

We build from Dickinson et al. (2010) in two main ways: first, we implement a presence-selection pipeline that has proven effective for English preposition error detection (cf. Gamon et al., 2008). As the task is understudied, the work is preliminary, but it nonetheless is able to highlight the primary areas of focus for future work. Secondly, we improve upon the training data, in particular doing a better job of selecting relevant instances for the machine learner. Obtaining better-quality training data is a major issue for machine learning applied to learner language, as the domain of writing is different from news-heavy training domains (Gamon, 2010).

[1] We use the Yale Romanization scheme for writing Korean.

[2] See Dickinson and Lee (2009); de Ilarraza et al. (2008); Oyama (2010) for related work in other languages.

2 Particle error detection

2.1 Pre-processing

Korean is an agglutinative language: Korean words (referred to as *ecels*) are usually composed of a root with a number of functional affixes. We thus first segment and POS tag the text, for both training and testing, using a hybrid (trigram + rule-based) morphological tagger for Korean (Han and Palmer, 2004). The tagger is designed for native language and is not optimized to make guesses for ill-formed input. While the POS tags assigned to the learner corpus are thus often incorrect (see Lee et al., 2009a), there is the more primary problem of segmentation, as discussed in more detail in section 4.

2.2 Machine learning

We use the Maximum Entropy Toolkit (Le, 2004) for machine learning. Training on a corpus of well-formed Korean, we predict which particle should appear after a given nominal; if this is different from what the learner used, we have detected an error. It is important that the data represent the relationships between specific lexical items: in the comparable English case, for example, *interest* is usually found with *in*: ***interest** in/*with learning*.

Treating the ends of nominal elements as possible particle slots, we break classification into two steps: 1) Is there a particle? (Yes/No); and 2) What is the exact particle? Using two steps eases the task of actual particle prediction: with a successful classification of negative and positive instances, there is no need to handle nominals that have no particle in step 2. To evaluate our parameters for obtaining the most relevant instances, we keep the task simple and perform only step 1, as this step provides information about the usability of the training data. For actual system performance, we evaluate both steps.

In selecting features for Korean, we have to account for relatively free word order (Chung et al., 2010). We follow our previous work (Dickinson et al., 2010) in our feature choices, using a five-word window that includes the target stem and two words on either side for context (see also Tetreault and Chodorow, 2008). Each word is broken down into: stem, affixes, stem_POS, and affixes_POS. We also have features for the preceding and following noun and verb, thereby approximating relevant selectional properties. Although these are relatively shallow features, they provide enough lexical and grammatical context to help select better or worse training data (section 3) and to provide a basis for a preliminary system (section 4).

3 Obtaining the most relevant instances

We need well-formed Korean data in order to train a machine learner. To acquire this, we use web-based corpora, as this allows us to find data similar to learner language, and using web as corpus (WaC) tools allows us to adjust parameters for new data (Dickinson et al., 2010). However, the methodology outlined in Dickinson et al. (2010) can be improved in at least three ways, outlined next.

3.1 Using sub-corpora

Web corpora can be built by searching for a set of seed terms, extracting documents with those terms (Baroni and Bernardini, 2004). One way to improve such corpora is to use better seeds, namely, those which are: 1) domain-appropriate (e.g., about traveling), and 2) of an appropriate level. In Dickinson et al. (2010), we show that basic terms result in poor quality Korean, but slightly more advanced terms on the same topics result in better-formed data.

Rather than use all of the seed terms to create a single corpus, we divide the seed terms into 13 separate sets, based on the individual topics from our learner corpus. The sub-corpora are then combined to create a cohesive corpus covering all the topics. For example, we use 10 *Travel* words to build a subcorpus, 10 *Learning Korean* words for a different subcorpus, and so forth. This means that terms appropriate for one topic are not mixed with terms for a different topic, ensuring more coherent web documents. Otherwise, we might obtain a *Health Management* word, such as *pyengwen* ('hospital'), mixed with a *Generation Gap* word, such as *kaltung* ('conflict')—in this case, leading to webpages on war, a topic not represented in our learner corpus.

3.2 Filtering

One difficulty with our web corpora is that some of them have large amounts of other languages along with Korean. The keywords are in the corpora, but there is additional text, often in Chinese, English, or Japanese. These types of pages are unreliable for our purposes, as they may not exhibit natural Korean. By using a simple filter, we check whether a majority of the characters in a webpage are indeed from the Korean writing system, and remove pages beneath a certain threshold.

3.3 Instance sampling

Particles are often dropped in colloquial and even written Korean, whereas learners are more often required to use them. It is not always the case that the web pages contain the same ratio of particles as learners are expected to use. To alleviate this over-weighting of having no particle attached to a noun, we propose to downsample our corpora for the machine learning experiments, by removing a randomly-selected proportion of (negative) instances. Instance sampling has been effective for other NLP tasks, e.g., anaphora resolution (Wunsch et al., 2009), when the number of negative instances is much greater than the positive ones. In our web corpora, nouns have a greater than 50% chance of having no particle; in section 3.4, we thus downsample to varying amounts of negative instances from about 45% to as little as 10% of the total corpus.

3.4 Training data selection

In Dickinson et al. (2010), we used a Korean learner data set from Lee et al. (2009b) for development. It contains 3198 ecels, 1842 of which are nominals, and 1271 (≈70%) of those have particles. We use this same corpus for development, to evaluate filtering and down-sampling. Evaluating on (yes/no) particle presence, in tables 1 and 2, recall is the percentage of positive instances we correctly find and precision is the percentage of instances that we classify as positive that actually are. A baseline of always guessing a particle gives 100% recall, 69% precision, and 81.7% F-score.

Table 1 shows the results of the MaxEnt system for step 1, using training data built for the topics in the data with filter thresholds of 50%, 70%, 90%, and 100%—i.e., requiring that percentage of Korean characters—as well as the unfiltered corpus. The best F-score is with the filter set at 90%, despite the size of the filtered corpus being smaller than the full corpus. Accordingly, we use the 90% filter on our training corpus for the experiments described below.

Threshold	100%	90%	70%	50%	Full
Ecel	67k	9.6m	10.3m	11.1m	12.7m
Instances	37k	5.8m	6.3m	7.1m	8.4m
Accuracy	74.75	81.11	74.64	80.29	80.46
Precision	80.03	86.14	79.65	85.41	85.56
Recall	84.50	86.55	84.97	86.15	86.23
F-score	82.20	**86.34**	82.22	85.78	85.89

Table 1: Step 1 (particle presence) results with filters

The results for instance sampling are given in table 2. We experiment with positive to negative sampling ratios of 1.3/1 (≈43% negative instances), 2/1 (≈33%), 4/1 (≈20%), and 10/1 (≈10%). We select the 90% filter, 1.3/1 downsampling settings and apply them to the training corpus (section 3.1) for all experiments below.

P/N ratio	10/1	4/1	2/1	1.3/1	1/1.05
Instances	3.1m	3.5m	4.3m	5m	5.8m
Accuracy	74.75	77.85	80.23	81.59	81.11
Precision	73.38	76.72	80.75	84.26	86.14
Recall	99.53	97.48	93.71	90.17	86.55
F-score	84.47	85.86	86.74	**87.12**	86.34

Table 2: Step 1 (presence) results with instance sampling

One goal has been to improve the web as corpus corpus methodology for training a machine learning system. The results in tables 1 and 2 reinforce our earlier finding that size is not necessarily the most important variable in determining the usefulness or overall quality of data collected from the web for NLP tasks (Dickinson et al., 2010). Indeed, the corpus producing best results (90% filter, 1.3:1 downsampling) is more than 3 million instances smaller than the unfiltered, unsampled corpus.

4 Initial system evaluation

We have obtained an annotated corpus of 25 essays from heritage intermediate learners,[3] with 299 sentences and 2515 ecels (2676 ecels after correcting spacing errors). There are 1138 nominals, with 93 particle errors (5 added particles, 35 omissions, 53 substitutions)—in other words, less than 10% of particles are errors. There are 979 particles after correction. We focus on 38 particles that intermediate

[3]Heritage learners have had exposure to Korean at a young age, such as growing up with Korean spoken at home.

students can be reasonably expected to use. A particle is one of three types (cf. Nam and Ko, 2005): 1) case markers, 2) adverbials (cf. prepositions), and 3) auxiliary particles.[4]

Table 3 gives the results for the entire system on the test corpus, with separate results for each category of particle, (**Case**, **Adv.**, and **Aux.**) as well as the concatenation of the three (**All**). The accuracy presented here is in terms of only the particle in question, as opposed to the full form of root+particle(s). Step 2 is presented in 2 ways: **Classified**, meaning that all of the instances classified as needing a particle by step 1 are processed, or **Gold**, in which we rely on the annotation to determine particle presence. It is not surprising, then, that **Gold** experiments are more accurate than **Classified** experiments, due to step 1 errors and also preprocessing issues, discussed next.

		Step 1	Step 2	
Data	#		Classified	Gold
Case	504	95.83%	71.23%	72.22%
Adv.	205	82.43%	30.24%	32.68%
Aux.	207	89.37%	31.41%	35.74%
All	916	91.37%	53.05%	55.13%

Table 3: Accuracy for step 1 (particle presence) & step 2 (particle selection), with number (#) of instances

Preprocessing For the particles we examine, there are 135 mis-segmented nominals. The problem is more conspicuous if we look at the entire corpus: the tagger identifies 1547 nominal roots, but there are only 1138. Some are errors in segmentation, i.e., mis-identifying the proper root of the ecel, and some are problems with tagging the root, e.g., a nominal mistagged as a verb. Table 4 provides results divided by cases with only correctly pre-processed ecels and where the target ecel has been mis-handled by the tagger. This checks whether the system particle is correct, ignoring whether the whole form is correct; if full-form accuracy is considered, we have no way to get the 135 inaccurate cases correct.

Error detection While our goal now is to establish a starting point, the ultimate, on-going goal of this work is to develop a robust system for automatically detecting errors in learner data. Thus, it is necessary to measure our performance at actually finding the erroneous instances extracted from our test corpus. Table 5 provides results for step 2 in terms of our ability to detect erroneous instances. We report precision and recall, calculated as in figure 1.

[4]Full corpus details will be made available at: `http://cl.indiana.edu/~particles/`.

		Step 1	Step 2	
Data	#		Classified	Gold
Accurate	781	94.24%	55.95%	58.13%
Inaccurate	135	74.81%	36.29%	38.51%

Table 4: Overall accuracy divided by accurate and inaccurate preprocessing

	Case	Adv.	Aux.	All
Precision	28.82%	7.69%	5.51%	15.45%
Recall	87.50%	100%	77.78%	88.00%

Table 5: Error detection (using Gold step 1)

From the set of *erroneous* instances:	
True Positive (TP)	ML class $\neq$ student class
False Negative (FN)	ML class = student class
From the set of *correct* instances:	
False Positive (FP)	ML class $\neq$ student class
True Negative (TN)	ML class = student class
Precision (P)	$\frac{TP}{TP+FP}$
Recall (R)	$\frac{TP}{TP+FN}$

Figure 1: Precision and recall for error detection

4.1 Discussion and Outlook

One striking aspect about the results in table 3 is the gap in accuracy between case particles and the other two categories, particularly in step 2. This points at a need to develop independent systems for each type of particle, each relying on different types of linguistic information. Auxiliary particles, for example, include topic particles which—similar to English articles (Han et al., 2006)—require discourse information to get correct. Still, as case particles comprise more than half of all particles in our corpus, the system is already potentially useful to learners.

Comparing the rows in table 4, the dramatic drop in accuracy when moving to inaccurately-processed cases shows a clear need for preprocessing adapted to learner data. While it is disconcerting that nearly 15% (135/916) of the cases have no chance of resulting in a correct full form, the results indicate that we can obtain reliable accuracy (cf. 94.24%) for predicting particle presence across all types of particles, assuming good morphological tagging.

From table 5, it is apparent that we are overguessing errors; recall that only 10% of particles are erroneous, whereas we more often guess a different particle. While this tendency results in high recall, a tool for learners should have higher precision, so that correct usage is not flagged. However, this is a first attempt at error detection, and simply knowing that precision is low means we can take steps to solve this deficiency. Our training data may have too many possible classes in it, and we have not yet accounted for phonological alternations; e.g. if the system guesses *ul* when *lul* is correct, we count a miss, even though they are different realizations of the same morpheme.

To try and alleviate the over-prediction of errors, we have begun to explore implementing a confidence filter. As a first pass, we use a simple filter that compares the probability of the best particle to the probability of the particle the learner provided; the absolute difference in probabilities must be above a certain threshold. Table 6 provides the error detection results for each type of particle, incorporating confidence filters of 10%, 20%, 30%, 40%, 50%, and 60%. The results show that increasing the threshold at which we accept the classifier's answer can significantly increase precision, at the cost of recall. As noted above, higher precision is desirable, so we plan on further developing this confidence filter. We may also include heuristic-based filters, such as the ones implemented in *Criterion* (see Leacock et al., 2010), as well as a language model approach (Gamon et al., 2008).

Finally, we are currently working on improving the POS tagger, testing other taggers in the process, and developing optimal feature sets for different kinds of particles.

Acknowledgments

We would like to thank the IU CL discussion group and Joel Tetreault for feedback at various points.

		Adv	Aux	Case	All
10%	P	10.0%	6.3%	29.9%	16.3%
	R	100%	77.8%	67.8%	73.3%
20%	P	13.5%	7.8%	32.6%	18.0%
	R	100%	77.8%	50.0%	60.0%
30%	P	20.0%	8.3%	36.1%	20.8%
	R	100%	66.7%	39.3%	50.7%
40%	P	19.4%	14.3%	48.6%	26.9%
	R	60.0%	66.7%	30.4%	38.7%
50%	P	23.1%	16.7%	57.9%	32.1%
	R	30.0%	44.4%	19.6%	24.0%
60%	P	40.0%	26.7%	72.3%	45.2%
	R	20.0%	44.4%	14.3%	18.7%

Table 6: Error detection with confidence filters

References

Marco Baroni and Silvia Bernardini. 2004. Bootcat: Bootstrapping corpora and terms from the web. In *Proceedings of LREC 2004*, pages 1313–1316.

Martin Chodorow, Joel Tetreault, and Na-Rae Han. 2007. Detection of grammatical errors involving prepositions. In *Proceedings of the 4th ACL-SIGSEM Workshop on Prepositions*, pages 25–30. Prague.

Tagyoung Chung, Matt Post, and Daniel Gildea. 2010. Factors affecting the accuracy of korean parsing. In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 49–57. Los Angeles, CA, USA. URL `http://www.aclweb.org/anthology/W10-1406`.

Arantza Díaz de Ilarraza, Koldo Gojenola, and Maite Oronoz. 2008. Detecting erroneous uses of complex postpositions in an agglutinative language. In *Proceedings of COLING-08*. Manchester.

Markus Dickinson, Ross Israel, and Sun-Hee Lee. 2010. Building a korean web corpus for analyzing learner language. In *Proceedings of the 6th Workshop on the Web as Corpus (WAC-6)*. Los Angeles. URL `http://jones.ling.indiana.edu/~mdickinson/papers/dickinson-israel-lee10.html`.

Markus Dickinson and Chong Min Lee. 2009. Modifying corpus annotation to support the analysis of learner language. *CALICO Journal*, 26(3).

Michael Gamon. 2010. Using mostly native data to correct errors in learners' writing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 163–171. Los Angeles, California. URL `http://www.aclweb.org/anthology/N10-1019`.

Michael Gamon, Jianfeng Gao, Chris Brockett, Alexander Klementiev, William Dolan, Dmitriy Belenko, and Lucy Vanderwende. 2008. Using contextual speller techniques and language modeling for esl error correction. In *Proceedings of IJCNLP*. Hyderabad, India.

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, Martin Chodorow, and Claudia Leacock. 2006. Detecting errors in english article usage by non-native speakers. *Natural Language Engineering*, 12(2).

S. Ko, M. Kim, J. Kim, S. Seo, H. Chung, and S. Han. 2004. *An analysis of Korean learner corpora and errors*. Hanguk Publishing Co.

Zhang Le. 2004. *Maximum Entropy Modeling Toolkit for Python and C++*. URL `http://homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html`.

Claudia Leacock, Martin Chodorow, Michael Gamon, and Joel Tetreault. 2010. *Automated Grammatical Error Detection for Language Learners*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool.

Chong Min Lee, Soojeong Eom, and Markus Dickinson. 2009a. Towards analyzing korean learner particles. Talk given at CALICO '09 Pre-Conference Workshop on Automatic Analysis of Learner Language. Tempe, AZ.

Sun-Hee Lee, Seok Bae Jang, and Sang kyu Seo. 2009b. Annotation of korean learner corpora for particle error detection. *CALICO Journal*, 26(3).

Ki-shim Nam and Yong-kun Ko. 2005. *Korean Grammar (phyocwun kwuke mwunpeplon)*. Top Publisher, Seoul.

Hiromi Oyama. 2010. Automatic error detection method for japanese particles. *Polyglossia*, 18.

Joel Tetreault and Martin Chodorow. 2008. The ups and downs of preposition error detection in esl writing. In *Proceedings of COLING-08*. Manchester.

Holger Wunsch, Sandra Kübler, and Rachael Cantrell. 2009. Instance sampling methods for pronoun resolution. In *Proceedings of RANLP 2009*. Borovets, Bulgaria.