Augmentation of Modality Translation Rules in Korean-to-English Machine Translation by Rule Learning

Seong-Bae Park **Jeong-Woo Son** **Yoon-Shik Tae**
Department of Computer Engineering
Kyungpook National University
Daegu 702-701, Korea
{sbpark,jwson,shtae}@sejong.knu.ac.kr

Abstract

The modality is usually expressed by compound verbs in Korean, while it is done by auxiliary verbs in English. Thus, it is important to distinguish Korean compound verbs with modality from others for developing a practical Korean-English machine translation system. In this paper, we describe a new method for processing Korean compound verbs. The precise rules are first generated by human experts, and then they are augmented by a machine learning method. The weakness of human-generated rules, low recall, is overcome by the machine-generated rules. The cross validation results show that the presented method achieves 85.78 % of F-score on the average, which is 43.63% improvement over the human-generated rules only.

1 Introduction

Modality is generally defined to be a grammatical category of a speaker's *subjective* degree of commitment to the expressed proposition. Thus, it is one of important and critical tasks for implementing practical machine translation systems. Without the correct translation of modality, the correct meaning of a sentence can not be delivered in machine translation.

The modality is also important in that it shows the peculiarity of different languages. For instance, the modality is usually expressed by auxiliary verbs in English, while it is done by auxiliary declinable words and some incomplete nouns in Korean. Especially the study on Korean modality processing has been performed under the name 'compound verb processing' since the modality is expressed by a compound verb in most cases. Thus, for correct Korean-English machine translation, it is important to capture the boundary of compound verbs and their sense.

In spite of its importance, up to now, the study on the translation of modality has not been a main focus in Korean-English machine translation, whereas there have been some studies in generating modality of Korean verbs in English-Korean machine translation (An et al., 1995). This is because it is believed that modality in Korean can be translated by a few simple rules. However, this is not true actually. It is not an easy task to identify the boundary of compound verbs for Korean, since an identical verb can be used as an independent verb and as an auxiliary verb. A verb should be determined by its context whether it is independent or auxiliary.

In order to identify the compound verbs, there have been a few studies on Korean compound verb processing. Among the limited number of pieces of previous work on compound verbs, the rule-based approach was the main stream. (Kim et al., 1995) proposed a rule-based system for processing Korean compound verbs. They presented 43 rules for detecting compound verbs from Korean sentences. In similar way, (Lee, 1989) derived 36 rules for extracting the aspect, mood, and modality in Korean. The main drawback of the rule-based systems is that the recall could be extremely low even with relatively high precision.

In addition, the rules demand too much human efforts since they are constructed by human experts who have profound knowledge of the target task. This is because the performance of rule-based methods depends on the quality of the rules. Thus, in machine learning community, a number of methods have been proposed that automatically learn the rules from data. (Clark and Niblett, 1989) proposed the CN2 program that uses the general-to-specific beam search (Mitchell, 1997), and (Fürnkfranz and Widmar, 1994) proposed the IREP algorithm. (Cohen, 1995) improved the IREP to produce the RIPPER algorithm, while (Cohen and Singer, 1999) presented the SLIPPER algorithm which adopted a boosting technique into rule learning.

Antecedent of Rules	English Attribute	Antecedent of Rules	English Attribute
(1) $w_{i-1} \in \{ah, uh\}$ & $w_i \in \{it, duh\}$	*Complete*	(24) $w_{i-1} = da$ & $w_i = chi$	*Suppose*
(2) $w_{i-1} \in \{ah, uh\}$ & $w_i = gyesi$	*Continue*	(25) $w_{i-1} = nun$ & $w_i = semchi$	
(3) $w_{i-1} = go$ & $w_i \in \{it, gyesi\}$	*Progressive*	(26) $w_{i-1} = ji$ & $w_i = mal$	*Prohibition*
(4) $w_{i-1} \in \{ah, uh\}$ & $w_i \in \{ga, oh\}$		(27) $w_{i-1} \in \{go, da\}$ & $w_i = mal$	*Emphasis*
(5) $w_{i-1} = ge$ & $w_i = gul$		(28) $w_{i-1} \in \{ah, uh\}$ &	
(6)$w_{i-1} \in \{gon, gonun\}$ & $w_i = ha$	*Used to*	$w_i \in \{dae, paji, sat, tuji\}$	
(7) $w_{i-1} = gi$ & $w_i = sijakha$	*Begin to*	(29) $w_i = nun$ &	*Pretend to*
(8) $w_{i-1} = uh$ & $w_i = dul$		$w_i \in \{yangha, chukha, cheha\}$	
(9) $w_{i-1} \in \{ah, uh\}$ &	*Possesive*	(30) $w_{i-1} \in \{ah, uh\}$ &	*Passive*
$w_i \in \{not, gaji, du\}$		$w_i = ji$	
(10) $w_{i-1} \in \{ryu, ryugo\}$ &	*Intend to*	(31) $w_{i-1} = ji$ &	*Negative*
$w_i \in \{dul, ha\}$		$w_i \in \{an, aniha, motha\}$	
(11) $w_{i-1} = goja$ & $w_i = ha$		(32) $w_{i-1} \in \{ah, uh\}$ & $w_i \in \{ju, dri\}$	*Support to*
(12) $w_{i-1} = myun$ & $w_i = ha$	*Wish to*	(33) $w_{i-1} \in \{ge, dorok\}$ &	*Causative*
(13) $w_i = go$ & $w_i = sip$		$w_i \in \{ha, mandul\}$	
(14) $w_{i-1} \in \{nunga, na\}$ & $w_i = bo$	*Guess*	(34) $w_{i-1} = go$ & $w_i \in \{na, mal\}$	*Finish*
(15) $w_{i-2} = ul$ & $w_{i-1} = gut$		(35) $w_{i-1} \in \{ah, uh\}$ &	
& $w_i = gat$		$w_i \in \{nae, buri, chiwoo\}$	
(16) $w_{i-1} \in \{ul, un\}$ &		(36) $w_{i-1} \in \{ah, uh\}$ & $w_i = boi$	*Guess*
$w_i \in \{dutha, dutsip, bupha, sungsip\}$		(37) $w_{i-1} = ge$ & $w_i = saengi$	
(17) $w_{i-1} \in \{nunga, ulka\}$ & $w_i = sip$		(38) $w_{i-1} \in \{ul, un\}$ & $w_i = bo$	*Try to*
(18) $w_{i-1} = um$ & $w_i = jikha$		(39) $w_{i-1} = ge$ & $w_i = dwe$	*Get to*
(19) $w_{i-2} = ul$ & $w_{i-1} = soo$ & $w_i = it$	*Can*	(40) $w_{i-1} \in \{ginun, gido\}$ & $w_i = ha$	*Admit*
(20) $w_{i-1} = do$ & $w_i = dwe$	*May*	(41) $w_{i-1} \in \{ah, uh\}$ &	
(21) $w_{i-1} = ul$ & $w_i = manha$	*Worthy to*	$w_i = maji - an$	
(22) $w_{i-2} = ul$ & $w_{i-1} = soo$ & $w_i = up$	*Cannot*	(42) $w_{i-4} = ul$ & $w_{i-3} = soo$	*Inevitable*
(23) $w_{i-1} \in \{ahya, uhya\}$ &	*Must*	$w_{i-2} = bak$ &$w_{i-1} = eh$ & $w_i = up$	
$w_i \in \{ha, dwe\}$		(43) $w_{i-1} = ul$ & $w_i = punha$	*Barely*

Table 1: The hand-crafted rules for modality translation. These rules are proposed in (Kim et al., 1995) and followed in the experiments below.

(Park and Zhang, 2003) proposed a hybrid method of hand-crafted rules and memory-based learning for text chunking. To overcome the limit of the hand-crafted rules they adopted memory-based learning, where the memory-based learner is trained with only the exceptions of the rules. Thus, the memory-based learner acted as if it were an exception handler of the rules. However, this method can be applied to the cases only when the hand-craft rules are strong enough for the target task. If the rules do not show high performance by themselves, then the overall performance would be not satisfactory.

In this paper, we propose a new approach to modality translation in Korean-English machine translation. The modality translation is based on compound verb recognition, and the translation is performed by the if-then rules. A few precise rules for detecting compound verbs in a sentence are first prepared by human experts, and then they are augmented by a machine learning approach. Therefore, they can be compensated for their low recall by the additional rules automatically generated from a corpus. For consistency with the previous hand-crafted rules, a rule-learning method is adopted in this paper. Thus, what remains finally is the increased number of if-then rules.

The rest of this paper is organized as follows. Section 2 explains the hand-crafted rules for translating Korean compound verbs. Section 3 describes the learning model of the rules and their application, and Section 4 presents the experimental results. Finally, section 5 draws conclusions.

2 Hand-Crafted Rules for Modality Translation

A compound verb in Korean usually consists of two or more verbs and sometimes it is composed of verbs and an incomplete noun. In addition, they appear in a sentence successively. Thus, basically morphological analysis will do for identifying a compound verb. Table 1 lists 43 rules for compound verb translation rules for Korean-English machine translation. This table is proposed by (Kim et al., 1995), and the rules in this

table are designed by a human expert through the thorough investigation of about 14,000 sentences.

All the rules use only morphological information. Thus, the compound verbs can be analyzed only with a morphological analyzer. Since there are a number of high-performance morphological analyzers that are publicly available, they can be processed without much effort. In this paper, HAM (Hangul Analysis Module)[1] proposed by (Kang and Kim, 1994) is used for morphological analysis.

Most of the rules use only a previous word as a context information. Out of 43 rules, only three rules require two precedent words, and just one rule needs four precedent words as a context. Therefore, the compound verbs can be processed by a *bigram* model, and the 4 remaining rules are applied to an input sentence independently from the other 39 rules.

Each rule in the table is interpreted as an *if-then* rule. For instance, a rule given by

$$w_{i-2} = ul \textbf{ and } w_{i-1} = soo \textbf{ and } w_i = it$$

with its English attribute '*Can*' is translated into

$$\textbf{If}(w_{i-2} = ul \textbf{ and } w_{i-1} = soo \textbf{ and } w_i = it) \textbf{ Then } English\ Target = Can.$$

That is, when w_{i-3} is a verb, '*muk* (eat)', a word sequence "*muk ul soo it*" is translated into "can eat". All the rules are not translated into an auxiliary verb. Some of them are translated into a TO infinitive following a normal verb, and some are into a THAT clause following a verb. This is because an auxiliary verb in Korean is mapped to various parts-of-speech in English.

Even though these rules are carefully designed, they have a significant flaw. Since they are carefully crafted, they generally show high accuracy. However, they are generated considering narrow context due to the limitation of the human cognition ability. As a result, the recall could be extremely low. This can be verified empirically in Section 4.2.

[1]This morphological analyzer can be downloaded from http://nlp.kookmin.ac.kr/~sskang/index.html.

```
function RuleLearning(data)
begin
 RuleSet := φ
 while ∃ positive examples ∈ data do
   rule := GrowRule(data)
   Add rule to RuleSet.
   Remove examples covered by rule
     from data.
   if Accuracy(rule) ≤ P/(P+N) then
     return RuleSet
   endif
 endwhile
return RuleSet
end
```

Figure 1: The rule-learning algorithm.

3 Augmented Rules by Machine Learning

3.1 Rule Learning for Additional Modality Translation Rules

Since the hand-crafted rules are not good enough to handle all cases of compound verbs, they should be supplemented by some other methods. In this paper, they are augmented by adding the additional rules that are automatically generated from a corpus.

In oder to generate new rules automatically, a machine-learning approach is used that learns *if-then* style rules. The compound verb processing can be considered as a classification task in the viewpoint of machine learning. Suppose that data $S =< (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N) >$ are given where $\mathbf{x}_i \in \mathbb{R}^n$ and $y_i \in \{true, false\}$. $\mathbf{x}_i$ is a contextual representation of a verb w_i. If w_i is within a compound verb, the class of $\mathbf{x}$ is *true*. It is $false$ otherwise. Then, the compound verb processing is to estimate a function $f : \mathbb{R}^n \rightarrow \{true, false\}$. That is, our task is to determine whether a verb w_i expressed as $\mathbf{x}$ with its context is located within a compound verb.

To estimate f, a rule learning algorithm is used that is based on the IREP (Incrementally Reduced Error Pruning) algorithm proposed by (Fürnkfranz and Widmar, 1994). Figure 1 shows the rule learning algorithm used in this paper. This algorithm is equivalent to the IREP except that it does not have a rule-pruning step.

The IREP is originally composed of two greedy algorithms. The first greedy algorithm, `GrowRule` in Figure 1, constructs a rule at a time, and then the second greedy algorithm

simplifies the learned rule. After that, all examples covered by the new rule are removed from the training set (`data`). The principle used to construct a rule in `GrowRule` is that more positive examples and less negative examples should be covered by the rule. It repeatedly adds conditions to rule r_0 with an empty antecedent. In each i-th stage, a more specialized rule r_{i+1} is made by adding single condition to r_i. The added condition in constructing r_{i+1} is the one with the largest *information gain* (Quinlan, 1993) relative to r_i, where the information gain is defined as

$$Gain\,(r_{i+1}, r_i) = \\ T^+_{i+1} \cdot \left(-\log \frac{T^+_i}{T^+_i + T^-_i} + \log \frac{T^+_{i+1}}{T^+_{i+1} + T^-_{i+1}} \right).$$

Here, T^+_i and T^-_i are the number of positive and negative examples covered by r_i accordingly. The conditions are added until the information gain becomes 0.

The information gain used is larger than or equal to zero. Thus, all the generated rules always cover some positive examples in `data` and it is guaranteed that the algorithm will eventually terminate. However, it is possible that there could be a number of rules that cover only a few positive examples, which causes too much computation for noisy data. To keep these rules from being added to `RuleSet`, the learning process stops if the accuracy of the generated rule is less than $P/(P+N)$, where P is the number of positive examples in `data`, and N is that of negative examples.

In the IREP, the second greedy algorithm that prunes a newly-learned rule performs the simplification of the rules in order to avoid the overfitting of `GrowRule` function. This is because `GrowRule` adds condition until the information gain gets 0. Fürnkfranz and Widmar thought that too specific rules may be generated through `GrowRule`. Thus, they suggested the `PruneRule` function that prunes the learned rule by dropping the conditions one by one. However, this step does harm at least in our task of the Korean compound verb. Since a number of rules will be generated by the proposed method, the high precision of a single rule is far more important than its general performance considering both precision and recall. That is, each rule in `RuleSet` should be specific with high precision. Therefore, `PruneRule` is not adopted in the proposed method in Figure 1. The RIPPER algorithm proposed by (Cohen, 1995) is

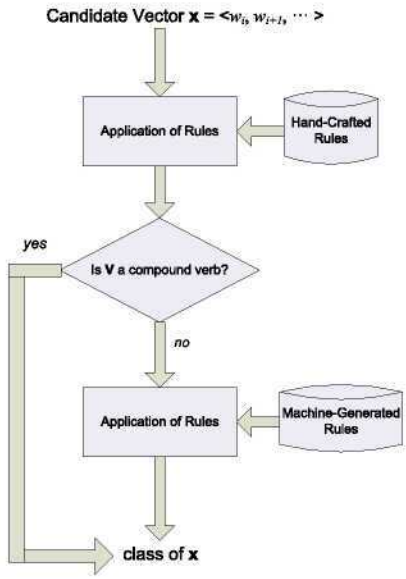

Figure 2: The process that determines whether a vector **v** is a compound verb or not using both hand-crafted rules and machine-generated rules.

different from the proposed method in that it tries to change the conditions of each rule produced by the IREP, not to change the number of the conditions.

3.2 Combination of Hand-Crafted Rules and Machine-Generated Rules

Since we have two sets of rules (hand-crafted rules and machine-generated rules), it is important to determine how to harmonize them. In this paper, the role which is played by machine-generated rules is to support the hand-crafted rules by handling the cases ignored by them.

Since the hand-crafted rules need not learning, the learning process described in Section 3.1 is applied only to the machine-generated rules. Their training set `data` is composed of the instances that are not processed by the hand-crafted rules. In training, therefore, we first apply the hand-crafted rules to the whole data. The instances that are not solved by them are gathered into a repository `data`. With `data`, the machine-generated rules are constructed through `RuleLearning`.

The precision and recall level of the hand-crafted rules is quite different from those of the machine-generated rules. Thus, they have to be treated independently. That is, both

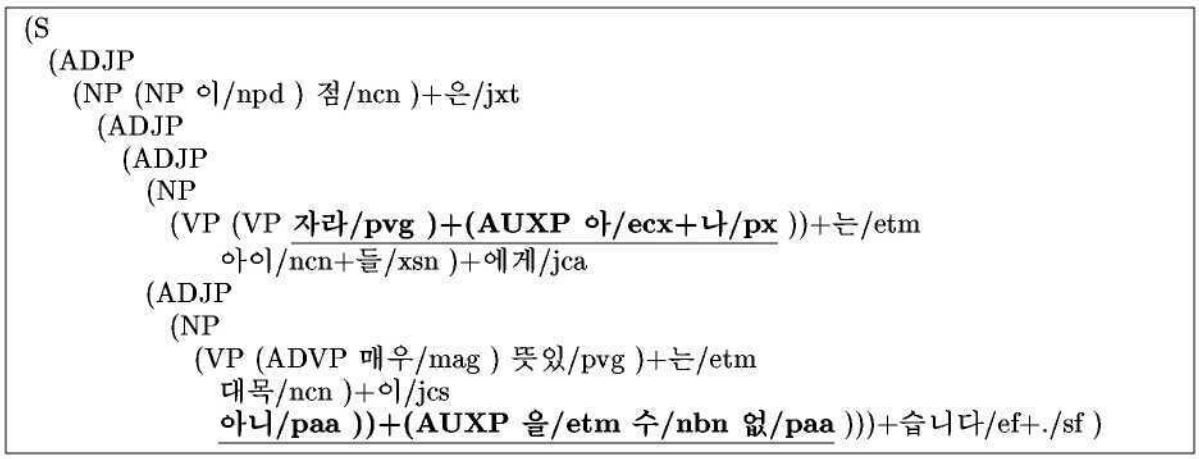

Figure 3: An example sentence of the STEP 2000 parsed corpus.

rules should behave as an independent classifier. When rules and a machine learning method are combined, it is important to decide their application order (Golding and Rosenbloom, 1996). They proved empirically that it is reasonable to apply the rules first when they are efficient and accurate enough to begin with. Since the hand-crafted rules in this paper have high precision and low recall and the machine-generated rules cover most exceptions of them, they must be applied ahead of machine-generated rules.

Figure 2 depicts how to determine the compound verb. When each case is represented as a vector $\mathbf{x} \in \mathbb{R}^n$, it is first given to the hand-crafted rules. If the hand-crafted rules can determine that $\mathbf{x}$ is a compound verb, the classification stops in this moment. Otherwise, the machine-generated rules classify the class of $\mathbf{x}$. Since machine-generated rules are constructed using the cases that the hand-crafted rules can not determine their class, they can determine the class of $\mathbf{x}$ regardless that this output is correct or not.

4 Experiments

4.1 Dataset

Since there is no standard data for English-Korean machine translation, we prepare a dataset for compound verb processing using a freely available parsed corpus. The dataset is derived from the product of STEP 2000 project supported by Korean government. The corpus consists of 12,092 sentences with 111,658 phrases and 321,328 words.

Figure 3 shows an example sentence in the STEP 2000 parsed corpus. The underlined phrase presents two instances of a compound verb. The first one is "자라(jara)/pvg+아(ah)/ecx 나(na)/px" and the second is "아니(ani)/paa+을(ul)/etm 수(soo)/nbn 없(up)/paa"[2]. However, only one of them can be processed by the hand-crafted rules. The second is determined by the rule numbered (22) in Table 1, but the first remains with its class undetermined.

Each instance on which the hand-crafted rules do not make a decision is transformed into a vector through n-gram model. That is, in order to determine w_i, the words $w_{i+1}, \ldots, w_{i+(n-1)}$ are also considered. Since every Korean word consists of two components (a stem and an ending), n words make up a vector with $2n$ dimensions. In addition, a vector of w_i is not considered as a training instance unless w_i is a verb, since the target task is a compound verb processing.

The total number of instances used to train `RuleLearning` is 15,827. Among them 5,795 instances (36.62%) are positive, and 10,032 (63.38%) are negative. Thus, the baseline accuracy is 63.38%. The whole dataset is divided into two parts: training set (90%) and test set (10%), and in all the experiments below 10-fold cross validation is performed.

4.2 Experimental Results

To evaluate all the methods, we use the contingency table method which is widely used in information retrieval and psychology. In this method, accuracy, recall and precision are de-

[2] pvg, paa, px, ecx, etm, and nbn are POS marks.

	Answer should be *true*	Answer should be *false*
The model says *true*	a	b
The model says *false*	c	d

Table 2: The contingency table for evaluation.

	Precision (%)	Recall (%)
1	93.79	27.40
2	93.69	27.19
3	93.62	27.18
4	93.85	27.04
5	93.65	27.12
6	93.81	27.09
7	93.55	27.36
8	93.75	27.24
9	93.73	27.07
10	93.56	27.22
Average	93.70 ± 0.10	27.19 ± 0.12

Table 3: The 10-fold cross validation results of hand-crafted rules only.

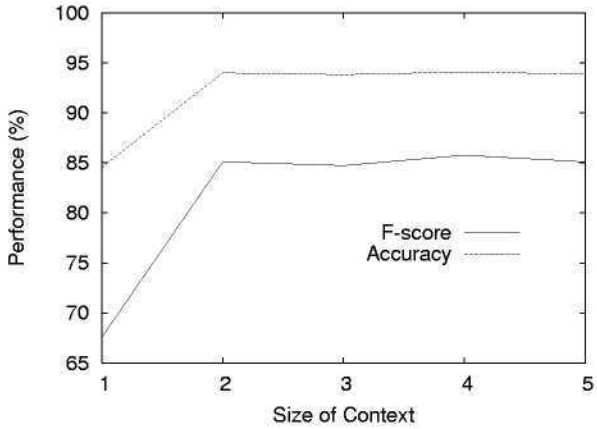

Figure 4: The performance as a function of context size.

fined as follows:

$$\begin{aligned} accuracy &= \frac{a+d}{a+b+c+d} \cdot 100\% \\ recall &= \frac{a}{a+c} \cdot 100\% \\ precision &= \frac{a}{a+b} \cdot 100\%, \end{aligned}$$

where a, b, c and d are defined in Table 2. The F_β-*score* which combines precision and recall is defined as

$$F_\beta = \frac{(\beta^2+1) \cdot recall \cdot precision}{\beta^2 \cdot recall + precision},$$

where β is the weight of recall relative to precision. We use $\beta = 1.0$, which corresponds to equal weighting of the two measures.

Table 3 shows the performance of the hand-crafted rules. Their average precision and recall are 93.70% and 27.19% respectively. As expected, while their precision is very high, only about one fourth of compound verbs are found by the hand-crafted rules. The small standard deviation indicates that the hand-crafted rules are stable.

Figure 4 illustrates what size of context is suitable for compound verb processing. Both F-score and accuracy is low when the context size is 1, but gets stable with high value when it is larger than 1. The hand-crafted rules in Table 1 use the context size of one, since one context size is composed of two words as explained in Section 4.1. The fact that the performance gets high from context size 2 implies that their context size is too narrow. The best performance is obtained when the context size is 4. This coincides with the fact that a compound verb composed of more than four words can not be found in the corpus. Figure 5 shows the number of rules generated by the proposed rule learning algorithm. For the context size from 2, the number of rules is around 40. Especially, it is 45 when the context size is 4. It means that we have at most 88 rules in total since there are 43 hand-crafted rules.

In Section 3.1, we insisted that the `PruneRule` function in the original IREP will do harm at least for our target task. Table 4 improves it empirically. The performance is measured with context size 4. since the best performance is obtained when the context size is 4. When we apply `PruneRule` to our task, `RuleLearning` yields 61.21 of F-score and 87.33% of accuracy. However, when `PruneRule` is removed, the performance goes up to 85.78 of F-score and 94.11% of accuracy. Thus, we can conclude that the rule-pruning step in the IREP is harmful at least for our task.

Finally, Table 5 shows the overall performance of the proposed method. The proposed

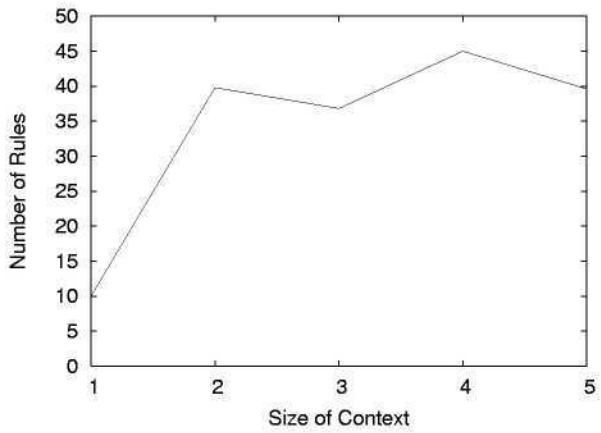

Figure 5: The number of rules generated by `RuleLearning` for various context size.

	F-score	Accuracy
With Pruning	61.21	87.33%
Without Pruning	85.78	94.11%

Table 4: The effect of pruning a rule in `RuleLearning`.

method yield 85.78 of F-score and 94.11% of accuracy. The improvements achieved over the hand-crafted rules are 43.63 in F-score and 0.41% in accuracy. As seen above, the precision of the hand-crafted rules is very high, but their F-score is low due to their low recall. The fact that the combination of both hand-crafted and machine-generated rules gives high F-score implies that the machine-generated rules are helpful in increasing the recall of the hand-crafted rules. Therefore, it can be reported that the proposed method is far more efficient than the hand-crafted rules alone. In addition, the accuracy is also 30.73% higher than that of the baseline model.

5 Conclusions

In this paper we have proposed a new method for compound verb processing in Korean-English machine translation by combining the hand-crafted rules and machine-generated rules. Our method is basically based on the hand-crafted rules, and the cases that are not processed by them are handled by a set of machine-generated rules. In order to generate rules automatically from a corpus, a modified version of the IREP algorithm is used. Since these rules are trained using the instances that are not processed by the hand-crafted rules, they play a role of supporting the hand-crafted. Therefore, in classification of unknown instances, the hand-crafted rules are first used. Only when the hand-crafted rules can not decide whether an instance is a compound verb or not, the machine-generated rules are then applied.

Method	F-score	Accuracy
Baseline	N/A	63.38%
Hand-Crafted Rules	42.15	93.70%
Proposed Method	**85.78**	**94.11%**

Table 5: The final experimental result of the proposed method.

The experiments was performed using a dataset derived from STEP 2000 corpus, and showed that the proposed method gives an F-score of 85.78 and an accuracy of 94.11%. This is an improvement of 30.73 % in accuracy over the baseline model which gives always negative answers. In addition, the proposed method outperforms the hand-crafted rules with the improvement of 43.63 in F-score and 0.41% in accuracy. Therefore, we can conclude that the proposed method is more efficient than the hand-crafted rules alone.

The hand-crafted rules are sufficient as translation rules for Korean-English machine translation. However, the machine-generated rules are insufficient for translation, since they can decide only the compound verb boundaries. Thus, our future work will be to make the output of the machine-generated rules be not a boolean value but the target English expression.

Acknowledgements

This work was supported by the Korea Research Foundation Grant funded by the Korean Government (KRF-2005-202-D00465).

References

An, D.-U., Kim, G.-C., and Kim, J.-H. 1995. Corpus-based modality generation for Korean predicates. *Literary and Linguistic Computing*, 10(1):1–10.

Clark, P. and Niblett, T. 1989. The CN2 induction algorithm. *Machine Learning*, 3(1):261–284.

Cohen, W. and Singer, Y. 1999. A simple, fast, and effective rule learner. In *Proceedings of the 16th National Conference on Artificial Intelligence*, pages 335–342, Orlando, Florida.

Cohen, W. 1995. Fast effective rule induction. In *Proceedings of the 12th International Conference on Mahcine Learning*, pages 115–123, Lake Tahoe, California.

Fürnkfranz, J. and Widmar, G. 1994. Incremental reduced error pruning. In *Proceedings of the 11th International Conference on Machine Learning*, pages 70–77, San Francisco, California.

Golding, A. and Rosenbloom, P. 1996. Improving accuracy by combining rule-based and case-based reasoning. *Artificial Intelligence*, 87:215–254.

Kang, S.-S. and Kim, Y.-T. 1994. Syllable-based model for the Korean morphology. In *Proceedings of the 15th International Conference on Computational Linguistics*, pages 221–226, Kyoto, Japan.

Kim, K.-C., Lee, K.-O., and Lee, Y.-S. 1995. Korean compound verbals processing driven by morphological analysis. *Journal of KISS*, 22(9):1384–1393.

Lee, S.-H. 1989. A study on extraction of aspect and modality information in Korean. *Korean Journal of Cognitive Science*, 1(2):255–257.

Mitchell, T. 1997. *Machine Learning.* The McGaw-Hill Companies.

Park, S.-B. and Zhang, B.-T. 2003. Text chunking by combining hand-crafed rules and memory-based learning. In *Proceedings ofthe 41st Annual Meeting of the Association for Computational Linguistics*, pages 497–504, Sapporo, Japan.

Quinlan, R. 1993. *C4.5: Programs for Machine Learning.* Morgan Kaufmann Publisher.