Lightweight Named Entity Extraction for Korean Short Message Service Text

Choong-Nyoung Seon[1], JinHwan Yoo[1], Harksoo Kim[2], Ji-Hwan Kim[1] and Jungyun Seo[3]
[1] Department of Computer Science and Engineering, Sogang University
Shinsu-dong, Mapo-gu, Seoul 121-742 - Korea
[e-mail: wilowisp@sogang.ac.kr, yoomsg@gmail.com, kimjihwan@sogang.ac.kr]
[2] Department of Computer and Communications Engineering, Kangwon National University
192-1, Hyoja 2-dong, Chuncheon-si, Gangwon-do 200-701 - Korea
[e-mail: nlpdrkim@kangwon.ac.kr]
[3] Department of Computer Science and Interdisciplinary Program of Integrated Biotechnology, Sogang University
Shinsu-dong, Mapo-gu, Seoul 121-742 - Korea
[e-mail: seojy@sogang.ac.kr]
*Corresponding author: Ji-Hwan Kim

Received September 13, 2010; revised December 9, 2010; revised January 21, 2011; accepted February 8, 2011; published March 31, 2011

Abstract

In this paper, we propose a hybrid method of Machine Learning (ML) algorithm and a rule-based algorithm to implement a lightweight Named Entity (NE) extraction system for Korean SMS text. NE extraction from Korean SMS text is a challenging theme due to the resource limitation on a mobile phone, corruptions in input text, need for extension to include personal information stored in a mobile phone, and sparsity of training data. The proposed hybrid method retaining the advantages of statistical ML and rule-based algorithms provides fully-automated procedures for the combination of ML approaches and their correction rules using a threshold-based soft decision function. The proposed method is applied to Korean SMS texts to extract person's names as well as location names which are key information in personal appointment management system. Our proposed system achieved 80.53% in F-measure in this domain, superior to those of the conventional ML approaches.

Keywords: Named entity (NE) extraction, machine learning (ML), rule-based, lightweight

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MEST)(2010-0027-797).

DOI: 10.3837/tiis.2011.03.006

1. Introduction

Not long ago, a mobile phone was just a communication device to make calls on the move. However, this small electronic device started to evolve to a context-aware one in order to alleviate memory problems. The users can effectively manage appointments without taking pains of remembering them using mobile phone's function such as PIMS (Personal Information Management System). The design goal to make a mobile phone a personal memory aid is to minimize the user's effort to memorize daily experiences.

The research in this paper aims to find efficient way to extract person's names and location names from Korean SMS texts, which are key information for personal appointment management system. The extraction of these entities from SMS texts presents significant technical challenges because of the following four characteristics of which implementation has: 1) it should be lightweight in terms of computation and memory requirements, because it runs on a mobile phone with limited resources. Recent NE systems running on servers or PCs demand excessive number of context features. 2) It requires robustness to corruption in input text. There is clear distinction between formal writing and SMS text, such as lack of punctuation and deliberate use of misspelling, shortcuts and emoticons. Especially, Korean SMS text lacking in spacing and without uppercase letter in NEs causes more difficulties than European languages. 3) It requires easy extension to include personal information. The most-frequently used names of a user are normally stored in the phonebook. The location names of user's interest can be easily obtained from location-based services. This personalized information would greatly enhance the performance of NE extraction from the user's point of view. 4) It estimates suitable probabilities for unseen sequences and copes well with this data sparsity problem. As SMS involves a lot of personal messages, it costs a great deal to gather a sufficient corpus. Thus, it is very difficult to collect such amount of corpus as it is required to measure the exact frequency of a word or syllable sequences.

In the last decades, substantial efforts have been made and impressive achievements have been obtained in the area of NE extraction. The best source of information relating to NE extraction system descriptions is the proceedings of the MUC (Message Understanding Conference) [1], the 1999 DARPA broadcast news workshop[2], and the 2002 and 2003 CoNLL (Conferences on Natural Language Learning) [3][4]. These proceedings contain the results of performance evaluations as well as system descriptions for each participating system in the evaluation. In the DARPA broadcast news workshop, a HMM (Hidden Markov Model) based model showed the best result [2]. At CoNLL-2003 which devoted to NE extraction for text input, a variety of machine learning techniques including ME (Maximum Entropy), HMM, CRFs (Conditional Random Fields), and SVM (Support Vector Machine) was presented [4].

As shown in the results, the SVM method seems to outperform other techniques. Extension of the binary classification using SVM is normally applied to n-ary classification. SVM requires relatively excessive memory capacity and computation power in this domain. In consideration of these requirements, CRFs [5] is chosen as baseline statistical classifier rather than SVM in this research. HMM is not considered either, because CRFs shows better performance than HMM in many fields, which results from the relaxation of the independence assumptions required by HMM. And ME [6] is also chosen as a statistical classifier. ME-based techniques showed the best performance at CoNLL-2003 and ME classifier is suitable in a limited memory environment because it requires lesser memory than CRFs. As a result,

depending on the size of available memory and minimum performance requirement, the type of a ML classifier is determined in this research.

In a stochastic method, linguistic information is obtained indirectly through large tables of statistics. Under a mobile phone environment, the size of context features should be restricted. In addition, a stochastic system is apt to encounter the difficulties in estimating probabilities from sparse training data in many circumstances [7]. In contrast to the stochastic methods, the rule-based method encodes linguistic information directly in a set of simple rules. The advantages of the rule-based method over the stochastic method include its smaller storage requirements, absence of need for less-descriptive models as in back-off, and its easy extension using expert linguistic knowledge and personal information due to its conceptually reasonable rules. However, one of the disadvantages of previous rule-based systems is to require too much time on the run to process the trained rules since they are applied to the input text sequentially one-by-one.

It shows clearly that there is no universal classifier that may be suited to all of the above four requirements. This fact also leads to the consideration of a hybrid of classifiers. A rule-based post-processing method [8] was presented in Bioinformatics. In this work, global patterns are generated using the Smith-Waterman local alignment algorithm. The presented rule-based method modifies the results of CRF-based NE extraction. For generating the global patterns, a vast corpus is needed. Owing to the difficulties in collecting SMS text, this method is not appropriate for application of NE extraction in SMS text. As a knowledge-based method, a method using SCFG (Stochastic Context Free Grammar) was presented [9]. However this method is difficult to be applied into SMS text because the SMS texts often have noise data and these noise data disturb the analysis of knowledge.

This paper proposes an efficient hybrid method combining the advantages of statistical ML systems and a rule-based system. In this method, a rule-based classifier is used as a post processor to modify incorrect results from a statistical ML classifier using correction rules. However, one of the disadvantages of a rule-based system is that it is based on hard decision rules. This introduces the necessity of soft decision which considers the confidence level of the results produced by ML classifiers. The novelty of this paper is that a threshold-based soft decision function automatically determines the result between the results of a statistical ML and the modified results by the correction rules. The selection of the results depends on the confidence level of the statistical ML classifier.

The rest of this paper is organized as follows. In Section 2, our hybrid method using statistical learning algorithms and correction rules is presented together with threshold-based decision function. Then, in Section 3, experiments and their results are described. Finally, conclusion is discussed in Section 4.

2. Hybrid NE Extraction Method

This section presents our hybrid NE extraction approach which combines a statistical classifier with correction rules generated by TBL (Transformation-Based Learning). In Section 2.1, the learning phase of the proposed method is described. Then, the execution phase of the method is explained in Section 2.2.

2.1 Learning Phase

Fig. 1 illustrates the procedures in the learning phase of the proposed methods. Statistical ML model is learned using training corpus. Then the method generates a set of correction rules which modify the errors produced by a statistical classifier. These correction rules are

generated using TBL. Finally the threshold value for the decision function is calculated at the same time in order to select the results.

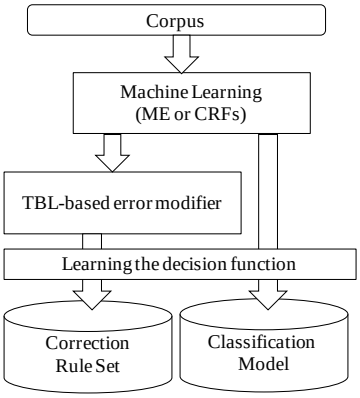


Fig. 1. Procedures in the learning phase of the proposed NE extraction

2.1.1 Syllable-level NE Extraction Using Machine Learning Technique

The aim of the system is to generate exact types of person's names and location names, as well as their exact boundaries. Due to the lack of spacing in Korean SMS text, we determine syllable as the basic recognition unit of the system. The training corpora are generated in accordance with B(egin)/I(ntermediate)/E(nd)/O(ther) syllable-level tags. Extraction results are generated using the same tags. **Table 1** describes the tags used in the system.

Table 1. Syllable-level tag set

Tag	Descriptions
O	Syllable not used in a person's name and a location name
LB	First syllable of a location name
LI	Intermediate syllable of a location name
LE	Last syllable of a location name
PB	First syllable of a person's name
PI	Intermediate syllable of a person's name
PE	Last syllable of a person's name

Two different statistical classifiers are constructed using ME and CRFs. These classifiers use syllable-level unigrams and bigrams. The selection of a classifier between ME and CRFs depends on the minimum requirement for performance and the size of available memory space. According to the experimental results shown in Section 3.2, CRFs shows better performance than ME. However, ME requires only 1/4 of memory space for the same amount of training data. Experimental details will be presented in Section 3.

2.1.2 Error-correction Rule Learning

After the initial syllable-level tags are annotated by statistical classifier, correction rules are generated using TBL algorithm. Syllable-level tags are compared to the true values of tags, and errors are counted. For all syllables whose tags are incorrect, the correction rules to recognize these tags correctly are generated and stored. These stored rules are then applied,

and the resulting number of improvements on the whole training data is calculated. The correction rules are generated according to their appropriate rule templates.

Table 2 shows the 65 rule templates used in this system. The major advantage of a rule-based method is relatively-small storage requirement due to simple structure of its rules. Therefore, it is possible for a rule-based method to incorporate longer distance lexical information. Rule templates consist of pairs of characters and a subscript. The italicized letters *s* and *t* represents that templates are related to syllables and syllable-level tags respectively. Subscripts show the relative distance from the current syllable, where 0 means the current word, -1 means the previous syllable, 1 means the next syllable and [1,3] means the presence of the particular syllable on positions 1, 2 or 3. Rule templates have one more slot which indicates the syllable-level tag of the change after the rule is activated.

For example, let's consider a generated rule 'if t_0=*O* and s_1= *(ap)* then change t_0 to *LE*.' This rule is generated using a rule template '$s_0\ t_1$.' This rule means that if the current syllable tag is '*O*' and the next syllable is ' *(ap)*' then change the current syllable tag into '*LE*.' The improvement for each possible correction rule is updated each time when a correction rule is generated. From all possible correction rules, the rule which causes the greatest improvement is applied to the current training data and the training data file is updated. If there is any change in the tags which affect any of the other rules, then the improvements from those other rules are also updated.

Table 2. 65 rule templates used in training of TBL-based error modifier.

Syllable	Tag	Rule templates		
s_{-2} s_{-1} s_0 s_1 s_2	t_0 $t_{-1}\ t_0$ $t_0\ t_1$ $t_{-2}\ t_{-1}$ $t_1\ t_2$	$s_{-2}\ t_0$ $s_{-2}\ t_{-1}\ t_0$ $s_{-2}\ t_0\ t_1$ $s_{-2}\ t_{-2}\ t_{-1}$ $s_{-2}\ t_1\ t_2$ $s_{-1}\ t_0$ $s_{-1}\ t_{-1}\ t_0$ $s_{-1}\ t_0\ t_1$ $s_{-1}\ t_{-2}\ t_{-1}$	$s_{-1}\ t_1\ t_2$ $s_0\ t_0$ $s_0\ t_{-1}\ t_0$ $s_0\ t_0\ t_1$ $s_0\ t_{-2}\ t_{-1}$ $s_0\ t_1\ t_2$ $s_1\ t_0$ $s_1\ t_{-1}\ t_0$	$s_1\ t_0\ t_1$ $s_1\ t_{-2}\ t_{-1}$ $s_1\ t_1\ t_2$ $s_2\ t_0$ $s_2\ t_{-1}\ t_0$ $s_2\ t_0\ t_1$ $s_2\ t_{-2}\ t_{-1}$ $s_2\ t_1\ t_2$
$s_{-1}\ s_0$ $s_0\ s_1$ $s_{-1}\ s_1$ $s_{-2}\ s_{-1}$ $s_1\ s_2$	t_0 $t_{-1}\ t_0$ $t_0\ t_1$ $t_{-2}\ t_{-1}$ $t_1\ t_2$	$s_{-1}\ s_0\ t_0$ $s_{-1}\ s_0\ t_{-1}\ t_0$ $s_{-1}\ s_0\ t_0\ t_1$ $s_{-1}\ s_0\ t_{-2}\ t_{-1}$ $s_{-1}\ s_0\ t_1\ t_2$ $s_0\ s_1\ t_0$ $s_0\ s_1\ t_{-1}\ t_0$ $s_0\ s_1\ t_0\ t_1$ $s_0\ s_1\ t_{-2}\ t_{-1}$	$s_0\ s_1\ t_1\ t_2$ $s_{-1}\ s_1\ t_0$ $s_{-1}\ s_1\ t_{-1}\ t_0$ $s_{-1}\ s_1\ t_0\ t_1$ $s_{-1}\ s_1\ t_{-2}\ t_{-1}$ $s_{-1}\ s_1\ t_1\ t_2$ $s_{-2}\ s_{-1}\ t_0$ $s_{-2}\ s_{-1}\ t_{-1}\ t_0$	$s_{-2}\ s_{-1}\ t_0\ t_1$ $s_{-2}\ s_{-1}\ t_{-2}\ t_{-1}$ $s_{-2}\ s_{-1}\ t_1\ t_2$ $s_1\ s_2\ t_0$ $s_1\ s_2\ t_{-1}\ t_0$ $s_1\ s_2\ t_0\ t_1$ $s_1\ s_2\ t_{-2}\ t_{-1}$ $s_1\ s_2\ t_1\ t_2$
$s_{[-3,-1]}$ $s_{[1,3]}$	t_0 $t_{-1}\ t_0$ $t_0\ t_1$ $t_{-2}\ t_{-1}$ $t_1\ t_2$	$s_{[-3,-1]}\ t_0$ $s_{[-3,-1]}\ t_{-1}\ t_0$ $s_{[-3,-1]}\ t_0\ t_1$ $s_{[-3,-1]}\ t_{-2}\ t_{-1}$ $s_{[-3,-1]}\ t_1\ t_2$	$s_{[1,3]}\ t_0$ $s_{[1,3]}\ t_{-1}\ t_0$ $s_{[1,3]}\ t_0\ t_1$ $s_{[1,3]}\ t_{-2}\ t_{-1}$ $s_{[1,3]}\ t_1\ t_2$	$s_{[-3,-1]}$ $s_{[1,3]}$
Ø	$t_{-1}\ t_0$ $t_0\ t_1$ $t_{-2}\ t_{-1}\ t_0$ $t_{-1}\ t_0\ t_1$ $t_0\ t_1\ t_2$		$t_{-1}\ t_0$ $t_0\ t_1$ $t_{-2}\ t_{-1}\ t_0$ $t_{-1}\ t_0\ t_1$ $t_0\ t_1\ t_2$	

The reason why we use TBL as a postprocessor is that we can easily extend the correction rules by using personalized data. The more the user uses, the more rules are accumulated and the better the NE extraction system works. For example, *'성수(Seongsu)'* may be used as a location name for a village in Seoul or as a person's name. At the release time of a mobile phone, the classification result of *'성수(Seongsu)'* depends on its frequencies used as location names and person's names in the training corpus. The more the user uses, the more rules are accumulated, the better the NE extraction system. After analyzing the personal information of a user, such as GPS logs and names listed in his/her phonebook, the user frequently visits *'성수(Seongsu)'* area then he/she normally uses *'성수(Seongsu)'* as a location name. but if he/she frequently make calls to *'성수(Seongsu)'*, then *'성수(Seongsu)'* is more likely used as a person's name. As a result, the performance of the system is incrementally improvement using personal information.

2.2 Execution Phase

Fig. 2 illustrates the procedures in the execution phase of the proposed method. The system using the proposed method is mainly divided into three parts; statistical ML classifier, rule-based error modifier, and decision function.

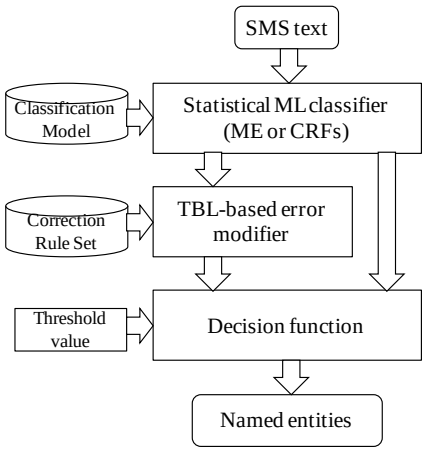


Fig. 2. Procedures in the execution phase of the proposed hybrid NE extraction method

In the execution phase, the proposed method uses the model of the statistical classifier and the correction rules that are learned in the learning phase. When an SMS text is provided as an input, the proposed method annotates the syllable-level classes to the SMS text using the statistical classifier to extract named entities. The annotated syllable-level tags and SMS text are used for the input of TBL-based error modifier. The correction rules of the TBL-based error modifier are applied to the input text one-by-one according to a given order. If the conditions for a correction rule are met, the correction rule is triggered. As previously stated, one of the disadvantages of a rule-based system is that it is based on the hard decision rules. In following section, a soft decision method is introduced. This method is based on threshold which considers the confidence level of the results produced by ML classifiers.

2.2.1 Decision Function

If there is a difference between the result of the statistical classifier and that of the TBL-based error modifier, the difference value d is calculated using p_m, the output probability of the statistical classifier as follows:

$$d\left(s_i, m_i, r_i\right) = p_m\left(m_i \middle| s_i\right) - p_m\left(r_i \middle| s_i\right) \tag{1}$$

where s_i is i-th syllable; m_i is the result tag of a statistical ML classifier at s_i; r_i is the result tag of a TBL-based modifier at s_i; p_m is the output probability of a statistical ML classifier; $p_m(m_i|s_i)$ is the output probability of m_i at s_i; $p_m(r_i|s_i)$ is the output probability of r_i at s_i.

If d is small, the result tag of statistical classifier would seem to be ambiguous because other tags have similar probability. Using the difference value d, we define the decision function D, which selects the appropriate syllable tag between m_i and r_i at the syllable s_i, as follows:

$$D(s_i) = \begin{cases} m_i & \text{if } d(s_i, m_i, r_i) \geq \alpha \\ r_i & \text{otherwise} \end{cases} \tag{2}$$

where α is the minimum value of d can trust the statistical ML classifier.

According to (2), $D(s_i)$ becomes m_i when p_m of the tag estimated by a ML classifier given s_i ($p_m(m_i|s_i)$) is sufficiently greater than p_m of the tag generated by a rule-based modifier given s_i ($p_m(r_i|s_i)$). In other words, $D(s_i)$ becomes m_i when the result produced by ML classifiers is confident enough. As a consequence, the result produced by a rule-based modifier is accepted exclusively if the difference between $p_m(m_i|s_i)$ and $p_m(r_i|s_i)$ is less than α.

The value α is automatically calculated in the learning phase through the following process: Denote c_i as the reference tag for s_i which is the syllable at the position i. For all syllables in validation data, a set, V, is defined as collection of tuples as in (3):

$$V = \left\{ \left(s_i, m_i, r_i, c_i\right) \mid m_i \in T, r_i \in T, c_i \in T \text{ and } s_i \in S \right\} \tag{3}$$

where c_i is the reference tag of s_i; T is the set of syllable-tags; S is the set of syllables.

Let R be a set of tuples where the incorrect results annotated by a ML classifier are substituted correctly by the rule-based modifier. In contrast to R, define M be a set of tuples where the correct results generated by a ML classifier are incorrectly revised by the rule-based classifier. R and M, which are the subsets of V, are represented as follows:

$$\begin{aligned} M &= \left\{ \left(s_i, m_i, r_i, c_i\right) \mid m_i = c_i \ \& \ r_i \neq c_i \right\},\ M \subset V \\ R &= \left\{ \left(s_i, m_i, r_i, c_i\right) \mid m_i \neq c_i \ \& \ r_i = c_i \right\},\ R \subset V \end{aligned} \tag{4}$$

The average value d is calculated for each set R and M as follows:

$$\overline{d_M} = \frac{1}{|M|}\sum_{i=1}^{|M|} d\left(s_i, m_i, r_i\right), \quad \left(s_i, m_i, r_i, c_i\right) \in M$$
$$\overline{d_R} = \frac{1}{|R|}\sum_{j=1}^{|R|} d\left(s_j, m_j, r_j\right), \quad \left(s_j, m_j, r_j, c_j\right) \in R \tag{5}$$

Finally, the threshold value α is determined as the mean value of $\overline{d_M}$ and $\overline{d_R}$.

$$\alpha = \frac{1}{2}\left(\overline{d_M} + \overline{d_R}\right) \tag{6}$$

In general, if both training data and held-out data are sufficient, an optimum threshold value in held-out data is also considered the optimum threshold value in test data. However if not, the threshold value does not completely guarantee the performance in test data. Because obtaining a sufficient SMS corpus is costly, it is very difficult to collect sufficient amount of the training data and held-out data. Therefore the optimum threshold value in held-out data is not appropriate in SMS domain.

For personalization, the correction rules will be extended to personalized data such as phone book, GPS data, and SMS text. According to accumulating personalized data, the distribution of the accumulated data may have difference the distribution estimated from the held-out data. Since the calculation method is required to easily update the threshold value, we determine threshold α as the arithmetic mean of two mean difference values rather than a specific value.

3. Experiment

Baseline methods and our proposed scheme are applied to Korean SMS text. Each performance is evaluated to extract person's names and location names. The setup procedure for each experiment is described in Section 3.1, followed by Section 3.2 where evaluation results are analyzed.

3.1 Experimental Setup

The corpus was collected from 80 undergraduates using their mobile phones. They were requested to write SMS texts that contain appointments. The corpus consists of 6,170 Korean SMS text messages. Baseline methods and the proposed method are applied to these text messages to extract person's names as well as location names which are key information for personal appointment management. Two experts in the natural language processing annotated B/I/E/O syllable-level tags according to the tag descriptions in **Table 1**. The corpus consists of 1,075 person's names and 5,514 location names, respectively. In order to analyze the number of misspellings and missing spaces, space errors and misspellings were counted from the first 1,000 text messages in the corpus. One or more misspellings were observed in 20.6 percent of the text and 91.6 percent of the text showed space-related errors. 34.2 percent of the text did not contain any space across the whole text.

These misspellings and incorrect spacing errors will cause negative effect on the detection of POS (Part of speech) boundaries and as a result, on the extraction of person's names and location names. In order to measure this negative effect on the POS boundary detection, we used the POS tagger released from Sejong project [10] which is known as one of the best

Korean POS taggers. This tagger assumes exact space between the words. **Table 3** shows the results of this POS boundary detection. Only 30.07% of the NE boundaries were detected successfully. This result represents that the traditional extraction methods for named entity using a word as base unit are not suitable for extraction from the SMS text. Because the proposed methods use a syllable as a base unit, they are less affected by the error propagation of the base-unit analyzer compared with the traditional methods.

Table 3. Detection rate of POS boundary of NEs in Korean SMS texts (Source of POS tagger: Sejong project)

	Both boundaries are correctly detected	**Only one of the boundaries is correctly detected**
Detection rate	30.07%	33.51%

In order to implement baseline statistical ML classifiers and rule-based error modifier, we use Maxent toolkit [11] for ME, CRF++ toolkit [12] for CRFs, and fnTBL toolkit [13] for TBL. Five-fold cross validation is applied and held-out data are treated as validation data for the estimation of α in the decision function. In order to learn TBL rules and calculate α value, we divide the corpus into training data, held-out data, and test data (the ratio is 3:1:1).

3.2 Experimental Results

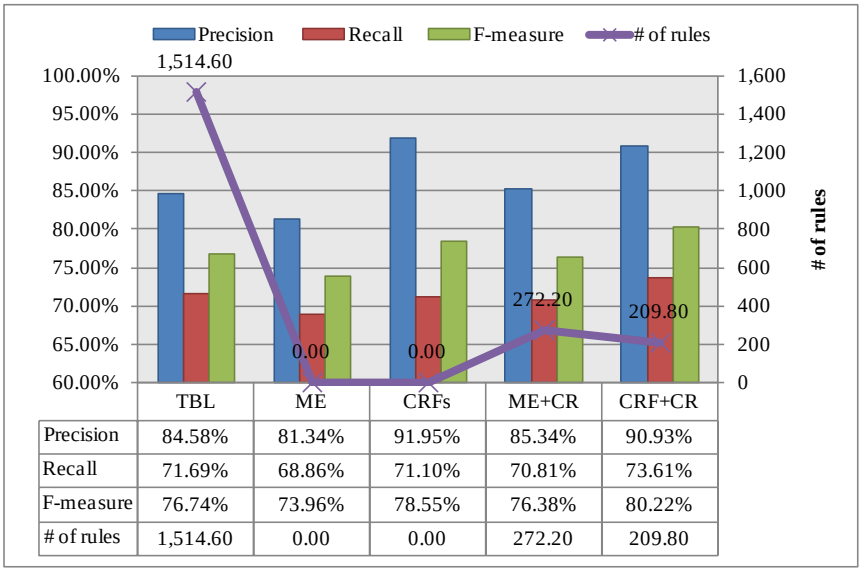
In order to evaluate the proposed methods, we performed three types of experiment. First of all, we compare the overall performance between the traditional methods and the proposed ones. Secondly, we show the variations of performance and memory requirement by varying the size of training corpus. Thirdly, we analyze the effect of the decision function.

3.2.1 Performance Comparison

We implemented five different systems. The first three are baseline systems which use only a single ML classifier such as TBL, ME, and CRFs. The other two systems, ME+CR and CRF+CR, are the proposed methods which combine an ML classifier and the rule-based error modifier. ME+CR and CRF+CR use ME and CRFs as ML classifiers, respectively. **Fig. 3** shows the results of five systems for the syllable-level extraction of person's names and location names. The performance is measured in terms of precision, recall, and F-measure. **Fig. 3** also shows the number of rules, if a system uses rule-based system as an ML classifier or an error modifier.

The TBL-based error modifier requires only about 18 percent (for ME+CR) and about 14 percent (for CRF+CR) of rules compared to the number of rules in which only TBL is used. Based on this comparison result, reducing the number of rules used in our proposed method not only decreases memory requirement but also enhances runtime speed greatly.

The proposed error modifier successfully improves performance by 2.42 percent for ME and by 1.67 percent for CRFs in terms of F-measure, respectively. In order to confirm whether these performance improvements have statistical significance, we performed paired Student's *t-test*. **Table 4** shows the *p* values of Student's *t-test* between the base classifiers and the proposed methods. The *p* values are evaluated as 0.001639 and 0.000913 in ME+CR and CRF+CR, respectively. As these *p* values are smaller than 0.01, we can confirm that the performance resulted in the application of our proposed methods is statistically significant under 0.01 significance level.



	TBL	ME	CRFs	ME+CR	CRF+CR
Precision	84.58%	81.34%	91.95%	85.34%	90.93%
Recall	71.69%	68.86%	71.10%	70.81%	73.61%
F-measure	76.74%	73.96%	78.55%	76.38%	80.22%
# of rules	1,514.60	0.00	0.00	272.20	209.80

Fig. 3. Performance comparison

Table 4. *P* values evaluated by paired Student's *t-test* between base classifiers and proposed methods (F-measure is used as *t-test* criterion)

Fold	ME	ME+CR	CRFs	CRF+CR
1	74.26%	77.52%	80.69%	82.34%
2	73.91%	76.63%	76.89%	79.10%
3	71.88%	73.74%	77.96%	79.43%
4	74.71%	75.93%	79.43%	81.52%
5	75.02%	78.09%	77.77%	78.70%
p value		0.001639		0.000913

3.2.2 Performance Analysis by Varying the Size of Training Corpus

The performance improvement using TBL-based error modifier is further analyzed by varying the size of training corpus. The size of training data is changed from one to three in the five-fold cross validation. N-fold training in **Table 5** and **Fig. 4** means that the proposed methods are learned with *n* folds of five-fold cross validation. Held-out data and testing data are fixed with one-fold data, respectively.

Table 5 shows the comparison of the memory requirement between the base classifiers and the proposed methods. The memory requirements in **Table 5** show the memory footprints of the trained models.

Table 5. Memory requirement according to size of training data

Method	One-fold training (Kbytes)	Two-fold training (Kbytes)	Three-fold training (Kbytes)
ME	307.73	397.02	473.53
ME+CR	332.95	419.46	494.84
CRFs	1441.55	1797.46	2093.78
CRFs+CR	1464.06	1816.34	2110.20

When ME+CR is compared with ME, the sizes of additional memory space required by TBL-based error modifier are only 25.22 KB, 22.44 KB and 21.31 KB for one-fold, two-fold and three-fold training data, respectively. Similarly, when CRFs+CR is compared with CRFs, the sizes of additional memory space required by TBL-based error modifier are only 22.51 KB, 18.88 KB and 16.42 KB, for training data of one-fold, two-fold and three-fold training data, respectively. These results show that the proposed methods require less than 25KB of additional memory regardless of the size of training data. Thus these methods can be used effectively under the condition of limited resource.

Fig. 4 shows the comparison of F-measure between the base classifiers and the proposed methods.

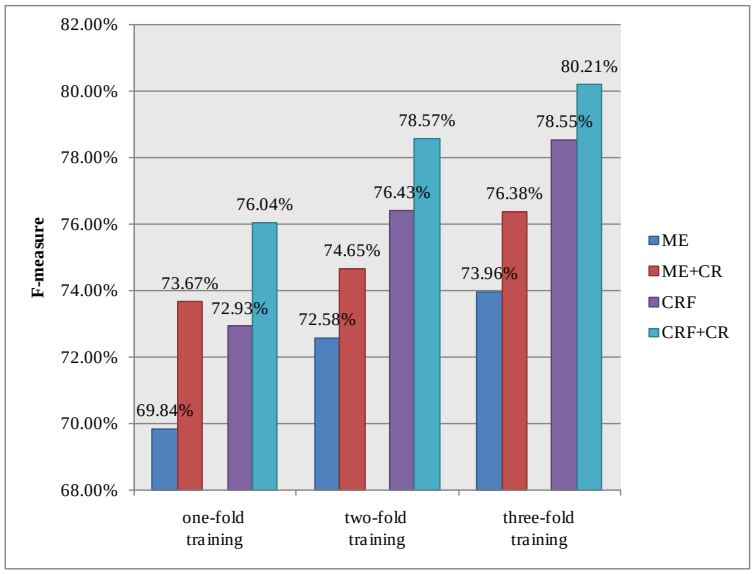


Fig. 4. F-measure according to the size of training data

ME+CR shows better results over all test conditions compared with ME. The amount of each improvement by applying TBL-based error modifier is measured as 3.83 %, 2.07 % and 2.42 % in terms of F-measure for training data of one-fold, two-fold and three-fold training data, respectively. The F-measure of ME reaches to the F-measure of ME+CR for one-fold, when ME uses up to three-fold training data. The similar results are observed for CRFs and CRF+CR. CRF+CR also shows better results over all test conditions compared with CRFs. The amount of each improvement by applying TBL-based error modifier is measured as 3.11 %, 2.14 % and 1.66 % in terms of F-measure for training data of one-fold, two-fold and three-fold training data, respectively. In order for the F-measure of CRFs to reach to the level of the F-measure of CRF+CR for two-fold, three-fold training data is used, which is 1.5 times greater than the training data of CRF+CR.

The CRFs classifier and the ME classifier used in the proposed method show different characteristics each other. When the same size of training data is used, the performance of CRF+CR is better than that of ME+CR. The F-measure of ME+CR reaches to that of CRF+CR for one-fold training data, when ME+CR uses up to three-fold training data. However, when it comes to the memory requirement, ME+CR is better than CRF+CR. When the same size of training data is used, ME+CR requires about 22 percent of memory compared to CRF+CR. Even when ME+CR uses more corpus (A case of ME+CR with 60-training data and CRF+CR

with 40-training data), ME+CR requires only 33.8 percent compared to CRF+CR. In other words, CRF+CR is an efficient method when the training data is small or high performance is required. In contrast to CRF+CR, ME+CR is an efficient method when the memory is limited.

3.2.3 The Effect of the Decision Function

In this section, the effectiveness of the proposed decision function is validated in details. For all tuples in set *R* and set *M*, their corresponding difference values *d* are measured by using CRFs as the baseline statistical classifier. The number of tuples is counted according to the ranges of their difference values. The ranges move with the increment of 0.1 from 0.0 to 1.0. As defined in Section 2.2, *R* is the set of tuples where the incorrect results annotated by a ML classifier are substituted correctly by the rule-based modifier, and *M* is the set of tuples where the correct results generated by a ML classifier are incorrectly revised by the rule-based classifier. The aim of the decision function is to maintain the number of the tuples in *R*, to which the rule-based modifier is applied, as many as we can while reducing the number of the tuples in *M*, to which the modifier is applied.

Fig. 5 shows the number of tuples in *R* and *M* according to ranges where their difference values are located. The dark line with rhombus shaped points expresses the number of tuples in *R* and the other light line with square shaped points demonstrates those in *M*.

According to the definition of *R* and *M*, the difference from the accumulated number of tuples for in the dark line to that in the light line means the amount of improvement when the rule-based error modifier is applied. For the left area of the intersection point (about d=0.54) in **Fig. 5**, the application of the rule-based error modifier causes positive effects, because the dark line is on the above of the light line. However, for the right area of the intersection point, the application of the rule-based error modifier deteriorates the performance, since the light line is above the dark line.

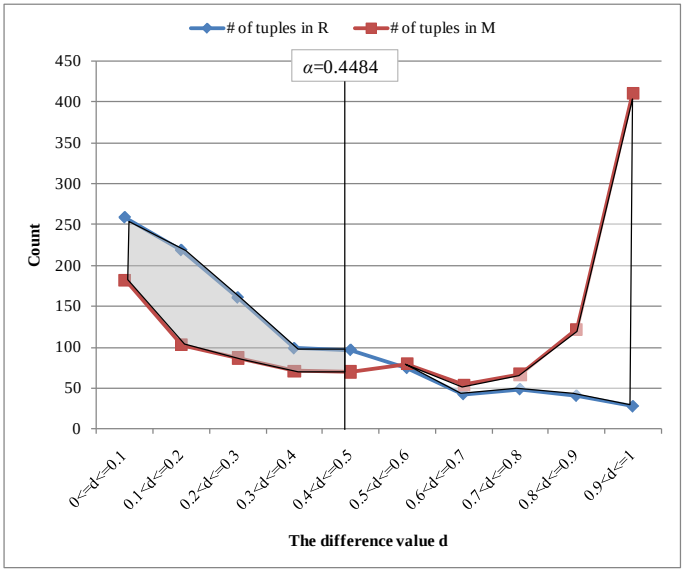


Fig. 5. The number of tuples in *R* and *M* according to ranges where their difference values *d* are located. The shade area illustrates the increased number of correctly annotated tuples when the decision function is applied. (Blue line with rhombus points: no. of tuples in *R*. Red line with square points: no. of tuples in *M*. 0.484 is estimated threshold value based on the method described in Section 2.2)

The proposed decision function determines the threshold value and prevents the application of the rule-based error modifier from the tuples whose difference value is greater than the threshold value. As a result, the amount of performance improvement using the application of the rule-based error modifier is maximized. According to the characteristics of the distribution for the number of tupes in *R* and *M* depicted in **Fig. 5**, the optimal value for the threshold is the value on *d* axis of the intersection point. As the threshold value estimated by the proposed method (0.4484) is quite close to this optimal value, and the area between the two lines from the estimated threshold value and the optimal value is quite small, it can be drawn that our proposed decision function operates very effectively. As a result, the number of correctly annotated tuples is increased as the size of the shaded area, after applying the proposed decision function.

4. Conclusion

We propose a hybrid method of machine learning algorithms and a rule-based algorithm to implement a lightweight NE extraction system for Korean SMS text. In this proposed method, a rule-based classifier is used as a post processor to modify incorrect results from a statistical ML classifier using correction rules. We propose a threshold-based decision function which automatically determines the result between the results of a statistical ML and the modified results by the correction rules. The proposed method improves the results of NE extraction: in terms of F-measure, by 2.42 percent for ME and by 1.67 percent for CRFs, while it only requires 18 percent (for ME+CR) and 14 percent (for CRF+CR) of rules respectively, compared to the number of rules in TBL only. Because of the amount of performance improvement and the small number of additional rules, the proposed method provides a very good solution for the four criteria of NE extraction from Korean SMS text. The selection of a statistical ML algorithm between CRF and ME depends on system resource requirements as well as performance requirements. When the memory requirement is a critical condition, ME+CR requires fewer than 500K memory.

References

[1] R. Grishman and B. Sundheim, "Message understanding conference – 6: A brief history," in *Proc. of the 16th Conference on Computational Linguistics*, pp.466-471, 1996. Article (CrossRef Link)

[2] M. Przybocki, J. Fiscus, J. Garofolo and D. Pallet, "HUB-4 information extraction evaluation," in *Proc. of the DARPA Broadcast News Workshop*, pp.13-18, 1999. Article (CrossRef Link)

[3] T. Sang and F. Erik, "Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition," in *Proc. of the 6th Conference on Natural Language Learning*, pp.1-4, 2002. Article (CrossRef Link)

[4] T. Sang, F. Erik and F. De Meulder, "Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition," in *Proc. of Conference on Natural Language Learning*, pp.142-147, 2003. Article (CrossRef Link)

[5] J. Lafferty, A. McCallum and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. of 18th International Conference on Machine Learning*, pp.282-289, 2001. Article (CrossRef Link)

[6] A. Berger, S. Pietra and V. Pietra, "A maximum entropy approach to natural language processing," *Computational Linguistics*, vo.22, no.1, pp.39-71, 1996. Article (CrossRef Link)

[7] E. Brill, "Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging," *Computational Linguistics*, vo.21, no.4, pp.543-565, 1995. Article (CrossRef Link)

[8] R. Tsai, C. Sung, H. Dai, H. Hung, T. Sung and W. Hsu, "NERBio: using selected word conjunctions, term normalization, and global patterns to improve biomedical named entity recognition," *BMC Bioinformatics 7* (Suppl. 5):S11, 2006. Article (CrossRef Link)
[9] R. Feldman, B. Rosenfeld and M. Fresko, "TEG—a hybrid approach to information extraction," *Knowledge and Information Systems*, vo.9, no.1, pp. 1-18, 2006. Article (CrossRef Link)
[10] The National Institute of the Korean Language, "Final Report on Achievements of 21st Sejong Project: Electronic Dictionary," *the National Institute of the Korean Language*, 2007.
[11] Z. Le, "Maximum Entropy Modeling Toolkit for Python and C++," Available from: http://homepages.inf.ed.ac.uk/lzhang 10/maxent_toolkit.html.
[12] T. Kudo, "CRF++: Yet Another CRF toolkit," Available at http://chasen.org/~taku/software/CRF++/.
[13] G. Ngai and R. Florian, "Transformation-based learning in the fast lane," in *Proc. of North American Chapter of the Association for Computational Linguistics on Language technologies*, pp.40-47, 2001. Article (CrossRef Link)

Choong-Nyoung Seon is a Computer Science PhD candidate at Sogang University, Seoul. He received his BS and MS in Computer Science from Sogang University in 1999 and 2001, respectively. He has worked as a senior researcher for four years at Diquest Inc., a major information retrieval company in Korea. His research interests include natural language understanding, information extraction, and natural language generation.

Jin-hwan Yoo is a research engineer at Center for Intelligent Robotics in Korea Institute of Science and Technology. He received his BS in Computer Science from Suwon University in 2006. He received his MS in Computer Science from Sogang University, Seoul. His research interests include natural language understanding, information extraction, and named entity recognition.

Harksoo Kim is an assistant professor of computer and communications engineering at Kangwon National University. He received his BA in Computer Science from Konkuk University in 1996. He received his MS and PhD in Computer Science from Sogang University in 1998 and 2003, respectively. He visited the CIIR at the University of Massachusetts in Amherst as a research fellow in 2004. In 2005, he began working for the Electronics and Telecommunications Research Institute (ETRI) as a senior researcher. His research interests include natural language processing, data mining, information retrieval, and question-answering.

Ji-Hwan Kim received the B.E. and M.E. degrees in Computer Science from KAIST (Korea Advanced Institute of Science and Technology) in 1996 and 1998 respectively and Ph.D. degree in Engineering from the University of Cambridge in 2001. From 2001 to 2007, he was a chief research engineer and a senior research engineer in LG Electronics Institute of Technology, where he was engaged in development of speech recognizers for mobile devices. In 2005, he was a visiting scientist in MIT Media Lab. Since 2007, he has been an assistant professor in the Department of Computer Science and Engineering, Sogang University. His research interests include spoken multimedia content search, speech recognition for embedded systems and dialogue understanding.

Jungyun Seo is a professor of computer science at Sogang University. He obtained a BS in Mathematics from Sogang University in 1981. He continued his studies in the Department of Computer Science at the University of Texas, Austin, receiving his MS and PhD in Computer Science in 1985 and 1990, respectively. He returned to Korea in 1991 to join the faculty of the Korea Advanced Institute of Science and Technology (KAIST) in Taejon, where he led the Natural Language Processing Laboratory in the Computer Science Department. In 1995, he moved to Sogang University in Seoul and became a full professor in March 2001. His research interests include multi-modal dialogues, statistical methods for NLP, machine translation, and information retrieval.