

Resolving Sense Ambiguity of Korean Nouns Based on Concept Co-occurrence Information

You-Jin Chung

Jong-Hyeok Lee

Div. of Electrical and Computer Engineering

Pohang University of Science and Technology (POSTECH) and Advanced Information
Technology Research Center(AITrc)

San 31 Hyoja-dong, Nam-gu, Pohang 790-784, R. of Korea

prizer@postech.ac.kr

jhlee@postech.ac.kr

fax: +82-54-279-5699

fax: +82-54-279-5699

Abstract

From the view point of the linguistic typology, Korean and Japanese have many grammatical similarities which enable it to easily construct a sense-tagged Korean corpus through an existing high-quality Japanese-to-Korean machine translation system. The sense-tagged corpus may serve as a knowledge source to extract useful clues for word sense disambiguation (WSD). This paper addresses a disambiguation model for Korean nouns, whose execution is based on the concept codes extracted from the sense-tagged corpus and the semantic similarity values over a thesaurus hierarchy. By the help of the automatically constructed sense-tagged corpus, we overcome the knowledge acquisition bottleneck. Also, we show that the performance of word sense disambiguation can be improved by combining several base classifiers. In an experimental evaluation, the proposed model using a majority voting achieved an average precision of 77.75% with an improvement over the baseline by 15.00%, which is very promising for real world MT systems.

1 Introduction

Generally, a Korean homograph may be translated into a different Japanese equivalent depending on which sense is used in a given context. Thus, noun sense disambiguation is essential to the selection of an appropriate Japanese target word in Korean-to-Japanese translation.

Much research on word sense disambiguation has revealed that several different types of information can contribute to the resolution of lexical ambiguity. These include surrounding words (an unordered set of words surrounding a target word), local collocations (a short sequence of words near a target word, taking word order into account), syntactic relations (selectional restrictions), parts of speech, morphological forms, semantic context, etc (McRoy, 1992, Yarowsky, 1992, Ng and Zelle, 1997).

To extract such information, various types of knowledge sources have been utilized such as machine-readable dictionaries (MRD), thesauri, and computational lexicons. Since most MRDs and thesauri were created for human use and display inconsistencies, these resources have clear limitations. Sense-tagged corpora have been used as the most useful knowledge source for WSD. However, despite the value of sense-tagged corpora, two major obstacles impede the acquisition of lexical knowledge from corpora: the difficulties of manually sense-tagging a training corpus, and data sparseness (Ide and Veronis, 1998). Manual sense-tagging of a corpus is extremely costly, and at present, very few sense-tagged corpora are available.

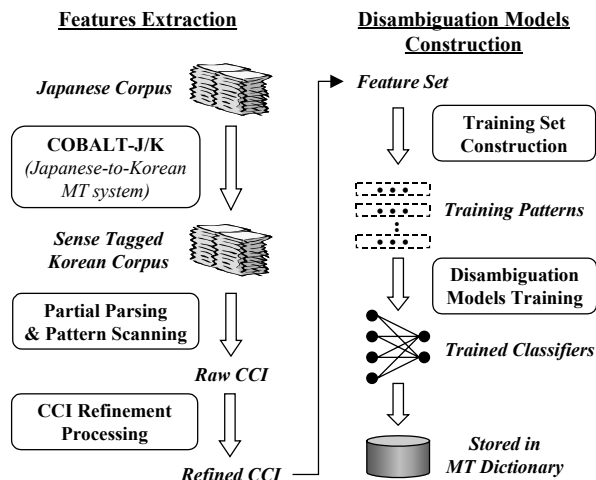


Figure 1. System Architecture

In our WSD approach, we construct a sense-tagged corpus automatically by using a method based on similarities between Korean and Japanese. Our disambiguation model is based on the work of Li *et al* (2000), especially focusing on the practicality of the method for application to real world MT systems. We alleviate the data sparseness problem by adopting a concept-based processing and reduce the number of features to a practical size by refinement processing.

This paper is organized as follows. Section 2 presents the overall system architecture. Section 3 explains the automatic construction of a sense-tagged Korean corpus and the extraction of refined features for word sense disambiguation. Section 4 describes the construction of feature set and the learning of disambiguation models. In Section 5, the experimental results are given, showing that the proposed method may be useful for WSD in a real text. In this paper, Yale Romanization is used to represent Korean expressions.

2 System Architecture

Our disambiguation method consists of two phases. The first phase is the extraction of features for WSD and the second phase is the construction of disambiguation models. (see Figure 1.)

For practical reasons, a reasonably small number of features is essential to the design of disambiguation models. To construct a feature set of a reasonable size, we adopt Li's method (2000), based on concept co-occurrence information (CCI). CCI are concept codes of words which co-occur

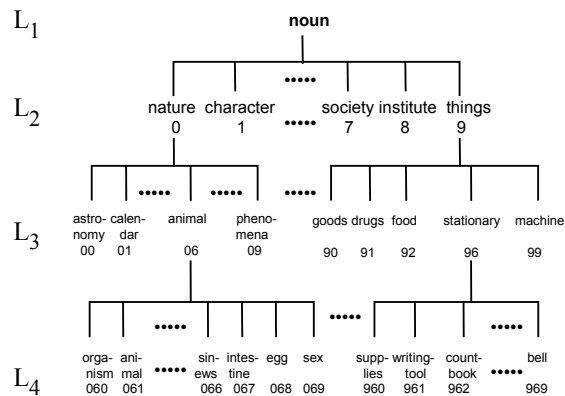


Figure 2. Concept hierarchy of the Kadokawa thesaurus

with the target word for a specific syntactic relation.

In accordance with Li's method, we automatically extract CCI from a corpus by constructing a sense-tagged Korean corpus. To accomplish this, we apply a Japanese-to-Korean MT system. Next, we extract CCI from the constructed corpus through partial parsing and scanning. To eliminate noise and to reduce the number of CCI, refinement processing is applied to the extracted raw CCI set. After completing refinement processing, we use the remaining CCI as features for disambiguation. The obtained feature set and the trained disambiguation models are stored in a dictionary for MT system.

3 Extraction of Features for WSD

3.1 Automatic Construction of Sense-tagged Corpus

Japanese and Korean are very similar in word order and lexical properties. Also, they have many nouns in common derived from Chinese characters. Because almost all Japanese common nouns represented by Chinese characters are monosemous, little transfer ambiguity is exhibited in Japanese-to-Korean translation of nouns, and we can obtain a sense-tagged Korean corpus of a good quality by using those linguistic similarities between Korean and Japanese.

For automatic construction of the sense-tagged corpus, we used a Japanese-to-Korean MT system

Table 1. Structure of CCI Patterns

CCI type	Structure of pattern
type ₀	unordered co-occurrence words
type ₁	noun + noun or noun + noun
type ₂	noun + <i>uy</i> + noun
type ₃	noun + other particles + noun
type ₄	noun + <i>lo/ulo</i> + verb
type ₅	noun + <i>ey</i> + verb
type ₆	noun + <i>eygey</i> + verb
type ₇	noun + <i>eyse</i> + verb
type ₈	noun + <i>ul/lul</i> + verb
type ₉	noun + <i>i/ka</i> + verb
type ₁₀	verb + relativizer + noun

called COBALT-J/K¹. In the transfer dictionary of COBALT-J/K, nominal and verbal words are annotated with concept codes of the Kadokawa thesaurus (Ohno and Hamanishi, 1981), which has a 4-level hierarchy of about 1,100 semantic classes, as shown in Figure 2. Concept nodes in level L_1 , L_2 and L_3 are further divided into 10 subclasses.

We made a slight modification of COBALT-J/K to enable it to produce Korean translations from a Japanese text, with all content words tagged with specific concept codes at level L_4 of the Kadokawa thesaurus. As a result, a sense-tagged Korean corpus of 1,060,000 sentences can be obtained from the Japanese corpus (*Asahi Shinbun*, Japanese Newspaper of Economics, etc.).

The quality of the constructed sense-tagged corpus is a critical issue. To evaluate the quality, we collected 1,658 sample sentences (29,420 eojeols²) from the corpus and checked their precision. The total number of errors was 789, and included such errors as morphological analysis, sense ambiguity resolution and unknown words. It corresponds to the accuracy of 97.3% (28,631 / 29,420 eojeols). The number of ambiguity resolution errors was 202 and it took only 0.69% of the overall corpus (202 / 29,420 eojeols). Considering the fact that the overall accuracy of the constructed corpus exceeds 97% and only a few sense ambiguity resolution errors were found in the Japanese-to-Korean

¹ COBALT-J/K (Collocation-Based Language Translator from Japanese to Korean) is a high-quality practical MT system developed by POSTECH.

² An Eojeol is a Korean syntactic unit consisting of a content word and one or more function words.

Table 2. Concept codes and frequencies in CFP ($\{\langle C_i, f_i \rangle\}$, type₂, *nwun*(eye))

Code	Freq.	Code	Freq.	Code	Freq.	Code	Freq.
028	19	107	8	121	7	126	4
143	8	160	5	179	7	277	4
320	8	331	6	416	7	429	22
433	4	501	13	503	10	504	11
505	6	507	12	508	27	513	5
530	6	538	11	552	4	557	7
573	5	709	5	718	5	719	4
733	5	819	4	834	4	966	4
987	9	other*	210				

* ‘other’ in the table means the set of concept codes with the frequencies less than 4.

translation of nouns, we regard the generated sense-tagged corpus as highly reliable.

3.2 Extraction of Raw CCI

Unlike English, Korean has almost no syntactic constraints on word order as long as the verb appears in the final position. The variable word order often results in discontinuous constituents. Instead of using local collocations by word order, Li *et al.* (2000) defined 13 patterns of CCI for homographs using syntactically related words in a sentence. Because we are concerned only with noun homographs, we adopt 11 patterns from them excluding verb patterns, as shown in Table 1. The words in bold indicate the target homograph and the words in italic indicate Korean particles.

For a homograph W , concept frequency patterns (CFPs), i.e., $(\{\langle C_1, f_1 \rangle, \langle C_2, f_2 \rangle, \dots, \langle C_k, f_k \rangle\}, type_i, W(S_i))$, are extracted from the sense-tagged training corpus for each CCI type i by partial parsing and pattern scanning, where k is the number of concept codes in $type_i$, f_i is the frequency of concept code C_i appearing in the corpus, $type_i$ is an CCI type i , and $W(S_i)$ is a homograph W with a sense S_i . All concepts in CFPs are three-digit concept codes at level L_4 in the Kadokawa thesaurus. Table 2 demonstrates an example of CFP that can co-occur with the homograph ‘*nwun*(eye)’ in the form of the CCI type₂ and their frequencies.

3.3 CCI Refinement Processing

The extracted CCI set is too numerous and too noisy to be used in a practical system, and must to be further selected. To eliminate noise and to reduce the number of CCI to a practical size, we ap-

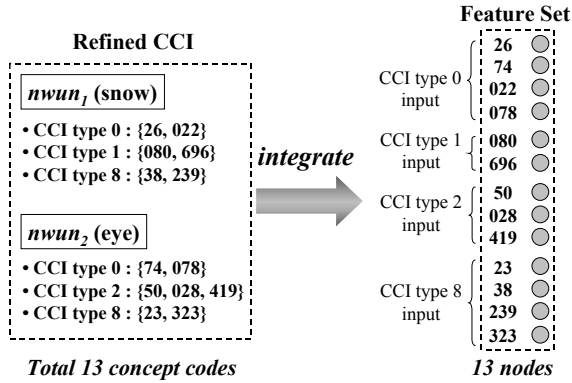


Figure 3. Construction of Feature Set for ‘nwun’

ply the refinement processing to the extracted CCI set. CCI refinement processing is composed of 2 processes: concept code discrimination and concept code generalization.

3.3.1 Concept Code Discrimination

In the extracted CCI set, the same concept code may appear for determining the different meanings of a homograph. To select the most probable concept codes, which frequently co-occur with the target sense of a homograph, Li defined the discrimination value of a concept code using Shannon’s entropy. A concept code with low entropy has a large discrimination value. If the discrimination value of the concept code is larger than a threshold, the concept code is selected as useful information for deciding the word sense. Otherwise, the concept code is discarded.

3.3.2 Concept Code Generalization

After concept discrimination, co-occurring concept codes in each CCI type must be further selected and the code generalized. To perform code generalization, Li adopted Smadja’s work (Smadja, 1993) and defined the code strength using a code frequency and a standard deviation in each level of the concept hierarchy. The generalization filter selects the concept codes with a strength larger than a threshold. We perform this generalization processing on the Kadokawa thesaurus level L_4 and L_3 .

After processing, the system stores the refined conceptual patterns ($\{C_1, C_2, C_3, \dots\}, type_i, W(S_i)$) as a knowledge source for WSD of real texts. These refined CCI are used as features for disambiguation models.

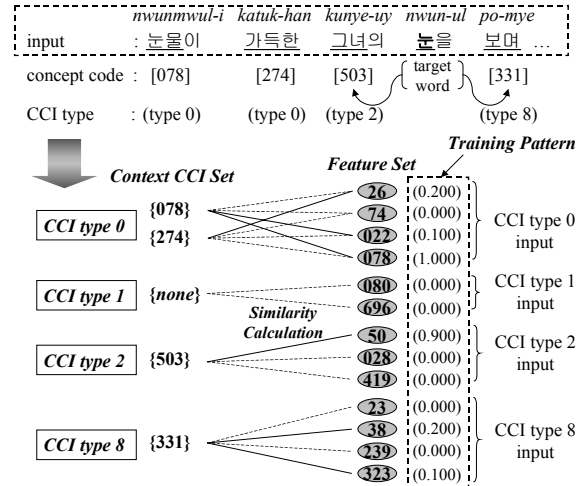


Figure 4. Construction of Training Pattern by Using Concept Similarity Calculation

The more specific description of the CCI extraction is explained in Li (2000).

4 Construction of Disambiguation Models

4.1 Feature Set Construction

The feature set is constructed by integrating the extracted CCI into a single vector. Figure 3³ demonstrates a construction example of the feature set for the homograph ‘nwun’ with the sense ‘snow’ and ‘eye’. The left side is the extracted CCI for each sense after refinement processing. We construct the feature set for ‘nwun’ by merely integrating the concept codes in CCI set of both senses. The resulting feature set is partitioned into several subgroups depending on their CCI types, i.e., type 0, type 1, type 2 and type 8. Since the extracted CCI set are different according to each word, each homograph has a feature set of its own.

4.2 Extraction of Training Patterns

After constructing the feature set for WSD, we extract training patterns for each homograph from the previously constructed sense-tagged corpus. The construction of training pattern is performed in the following 2 steps.

Step 1. Extract CCI from the context of the target homograph. The window size of the context is

³ The concept codes in Figure 3 are simplified ones for the ease of illustration. In reality there are 87 concept codes for ‘nwun’.

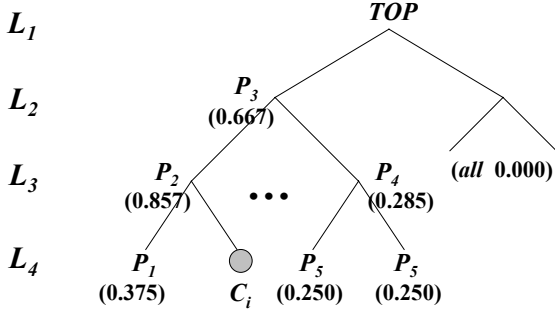


Figure 5. Concept Similarity on the Kadokawa Thesaurus Hierarchy

a single sentence. Consider, for example, the sentence in Figure 4 which has the meaning of “Seeing her eyes filled with tears, ...”. The target homograph is the word ‘*nwun*’. We extract its CCI from the sentence by partial parsing and pattern scanning. In Figure 4, the words ‘*nwun*’ and ‘*kunye*(her)’ with the concept code 503 have the relation of <noun + *uy* + noun>, which corresponds to ‘*CCI type 2*’ in Table 1. There is no syntactic relation between the words ‘*nwun*’ and ‘*nwunmul*(tears)’ with the concept code 078, so we assign ‘*CCI type 0*’ to the concept code 078.

Similarly, we can obtain all pairs of CCI types and their concept codes appearing in the context. All the extracted <*CCI-type: concept codes*> pairs are as follows: {<*type 0: 078,274*>, <*type 2: 503*>, <*type 8: 331*>}.

Step 2. Obtain the training pattern by calculating concept similarities between concept codes in the context CCI set and the feature set. Concept similarity calculation is performed only between the concept codes with the same CCI-type. This score represents that how much each node of the network relates to clues appearing in the target context. The calculated concept similarity score is assigned to each feature node as the activation strength for it.

$Csim(C_i, P_j)$ in Equation 1 is used to calculate the concept similarity between C_i and P_j , where $MSCA(C_i, P_j)$ is the most specific common ancestor of concept codes C_i and P_j , and *weight* is a weighting factor reflecting that C_i as a descendant of P_j is preferable to other cases. That is, if C_i is a descendant of P_j , we set *weight* to 1. Otherwise, we set *weight* to 0.5.

$$Csim(C_i, P_j) = \frac{2 \times level(MSCA(C_i, P_j))}{level(C_i) + level(P_j)} \times weight \quad (1)$$

The similarity values between the target concept C_i and each P_j on the Kadokawa thesaurus hierarchy are shown in Figure 5. These similarity values are computed using Equation 1. For example, in ‘*CCI-type 0*’ part calculation, the relation between the concept codes 274 and 26 corresponds to the relation between C_i and P_4 in Figure 5. So we assign the similarity 0.285 to the feature node labeled by 26. As another example, the concept codes 503 and 50 have a relation between C_i and P_2 and we obtain the similarity 0.857. If more than two concept codes exist in one CCI-type, such as <*type 0: 078, 274*>, the maximum similarity value among them is assigned to the input node, as in Equation 2.

In Equation 2, C_i is the concept code of the feature set, and P_j is the concept codes in the <*CCI-type: concept codes*> pair which has the same CCI-type as C_i .

$$InputVal(C_i) = \max_{P_i} (Csim(C_i, P_j)) \quad (2)$$

The use of concept similarity scheme gives another advantage. By adopting this concept similarity calculation, we can achieve a broad coverage of the method. If we use the exact matching scheme instead of concept similarity, we may obtain only a few concept codes matched with the features. Consequently, sense disambiguation would fail because of the absence of clues.

4.3 Learning of Disambiguation Models

Using the obtained feature set and training patterns, we learned 4 types of disambiguation models, such as neural network, decision tree, support vector machine and majority voting system. Neural network and decision tree have been used in a lot of pattern recognition problems because of their strong capability in classification. And recently, support vector machine have generated a great interest in the community of machine learning due to its excellent generalization performance in a wide variety of learning problems.

From a statistical point of view, if the size of sample is small, generating different classifiers about the sample and combining them may result in more accurate prediction of new patterns. On the other hand, based on a computational view, if the sample is large enough, the nature of learning algorithm could lead to getting stuck in local optima. Therefore, a classifier combination is a way to expand the hypothesis space to represent the true

Table 3. Evaluation Results for Decision Tree with Different Pruning Levels

Pruning Confidence Level	Precision (correct / applied)
Level = 10%	76.26% (546/716)
Level = 15%	77.38% (561/725)
Level = 25%	77.30% (555/718)
Level = 40%	76.72% (547/713)

(※ number of test samples : 942)

Table 4. Evaluation Results for Support Vector Machine with Different Kernel Functions

Kernel Function	Precision (correct / applied)
Linear	79.89% (556/696)
Polynomial (degree=2)	80.60% (565/701)
Polynomial (degree=3)	79.92% (569/712)
RBF (width=0.5)	80.80% (568/703)
RBF (width=1.0)	80.66% (563/698)
RBF (width=2.0)	79.71% (554/695)

(※ number of test samples : 942)

function (Ardeshir, 2002). In our experiment, we adopted a majority voting system for combining base classifiers. A majority voting selects the sense of the test pattern based on receiving more than half votes of base classifiers.

To find the best parameters for decision tree and support vector machine, we evaluated performance of each classifier on various parameters. For this evaluation, we used 942 test samples extracted from KIBS (Korean Information Base System) corpus. Table 3 and 4 are the evaluation results for decision tree and support vector machine respectively and we selected parameters which showed the best performance. The parameter settings for our system are listed below.

[Decision Tree (DT)]

- C4.5 Decision Tree Generator
- Pruning confidence level : 15%

[Neural Network (NN)]

- 2-layer network

[Support Vector Machine (SVM)]

- SVM light
- Kernel : RBF (width = 0.5)

[Majority Voting (MV)]

- Base classifiers : DT, NN, SVM

5 Experimental Evaluation

Our WSD approach is a hybrid method, which combines the advantage of knowledge-based and corpus-based methods. Figure 6 shows our overall WSD algorithm. For a given homograph, sense disambiguation is performed as follows. First, we search a collocation dictionary. The Korean-to-Japanese translation system COBALT-K/J has an MWTU (Multi-Word Translation Units) dictionary, which contains idioms, compound words, collocations, etc. If a collocation of the target word exists in the MWTU dictionary, we simply determine the sense of the target word to the sense found in the dictionary. This method is based on the idea of ‘one sense per collocation’. Next, we verify the selectional restrictions of verbs described in the dictionary. If we cannot find any matched patterns for selectional restrictions, we apply the machine learning classifiers. If we fail in all the previous stages, we assign the most frequently appearing sense in the training corpus to the target word.

For an experimental evaluation, 15 Korean noun homographs were selected, along with a total of 1,200 test sentences in which one homograph appears (2 senses : 12 words, 3 senses : 2 words, 4 senses : 1 word). The test sentences were randomly selected from the KIBS corpus.

The baseline results are shown in Table 5, where the result A is the case when the most frequent sense was taken as the answer and the result B is the case when COL and SR stages were applied previously. Symbols COL, SR, ML and MFS in Table 5 and 6 indicate 4 stages of our method in

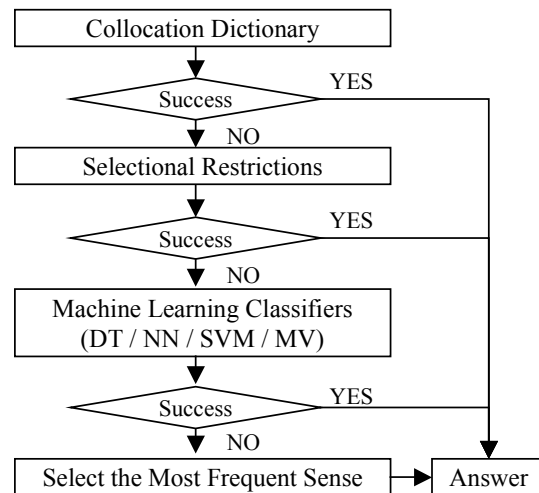


Figure 6. The Proposed WSD Algorithm

Table 5. Baseline Performance

* Precision (correct # / applied #)

Stage	Baseline A	Baseline B
COL	N/A (0/0)	100% (21/21)
SR	N/A (0/0)	91.14% (216/237)
MFS	62.75% (753/1200)	64.23% (605/942)
Total	62.75% (753/1200)	70.17% (842/1200)

Table 6. Comparison Results of Classifiers

* Precision (correct # / applied #)

Stage	Model 1 [DT]	Model 2 [NN]	Model 3 [SVM]	Model 4 [MV]
COL	100% (21/21)			
SR	91.14% (216/237)			
ML	77.38% (561/725)	78.93% (558/707)	80.80% (568/703)	81.84% (568/694)
MFS	53.00% (115/217)	57.87% (136/235)	51.46% (123/239)	51.61% (128/248)
Total	76.08% (913/1200)	77.58% (931/1200)	77.33% (928/1200)	77.75% (933/1200)

Figure 6, respectively. Table 6 is the comparison results of 4 machine learning classifiers. To compare models with the same condition, we controlled the number of test samples which each model is applied to about 700.

As shown in the table, the majority voting system showed the best performance above all other single classifiers and exceeded the baseline A by 15.00%. Even if we exclude the help of the collocation information and selectional restrictions described in the dictionary, we achieved the improvement of 7.58% over the baseline B. This result is very promising for real world MT systems and indicates that word sense disambiguation can be improved by classifier combination. Among the single classifiers, SVM was better than DT and NN (see ML stage in Table 6). Interestingly, however, when followed by MFS stage, NN overtook the performance of SVM.

The results of classifiers for each word are shown in Table 7. A shadowed cell indicates the best classifier on the word. Although the majority voting recorded the best results on only 2 words, it showed good results on other words steadily. We can recognize that the best classifier is different for each word. Some words have a decision tree as the best classifier and some have a neural network. From this observation, we guess that each word

Table 7. Comparison Results of Classifiers for Each

Word * Precision (correct # / applied #)

Word	DT	NN	SVM	MV
<i>kasa</i>	93.24% (69/74)	77.46% (55/71)	77.94% (53/68)	89.39% (59/66)
<i>kancang</i>	88.93% (47/56)	85.25% (52/61)	88.89% (48/54)	87.72% (50/57)
<i>keli</i>	79.17% (19/24)	57.69% (15/26)	76.92% (30/39)	83.33% (20/24)
<i>kyengki</i>	69.39% (34/49)	77.27% (34/44)	70.21% (33/47)	70.83% (34/48)
<i>kyengpi</i>	84.21% (32/38)	75.56% (34/45)	76.19% (32/42)	77.27% (34/44)
<i>kwutu</i>	86.44% (51/59)	90.57% (48/53)	87.27% (48/55)	87.72% (50/57)
<i>nwun</i>	91.84% (45/49)	93.48% (43/46)	93.62% (44/47)	91.67% (44/48)
<i>tali</i>	52.94% (27/51)	52.38% (22/42)	54.29% (19/35)	52.63% (20/38)
<i>pwuca</i>	82.61% (57/69)	86.67% (39/45)	87.10% (54/62)	85.94% (55/64)
<i>swumyen</i>	66.67% (22/33)	65.38% (34/52)	83.33% (30/36)	80.56% (29/36)
<i>yongki</i>	62.07% (36/58)	83.33% (35/42)	73.33% (33/45)	75.56% (34/45)
<i>uysa</i>	81.82% (9/11)	78.00% (39/50)	78.00% (39/50)	83.33% (35/42)
<i>yenki</i> (3 senses)	52.08% (25/48)	68.75% (22/32)	66.67% (20/30)	65.52% (19/29)
<i>censin</i> (3 senses)	93.55% (58/62)	93.22% (55/59)	98.08% (51/52)	96.49% (55/57)
<i>cenlyek</i> (4 senses)	68.18% (30/44)	79.49% (31/39)	79.49% (31/39)	76.92% (30/39)

may have disambiguation property of its own and require a different machine learning method according to its property. So if we can identify the disambiguation characteristics of words, we will be able to improve the system performance by applying a different classifier for each word.

6 Conclusion

To resolve sense ambiguities in Korean-to-Japanese MT, this paper has proposed a practical word sense disambiguation method using concept co-occurrence information. We showed that sense-tagged Korean corpus can be generated easily by using Japanese corpus and a machine translation system. In an experimental evaluation, the pro-

posed WSD model using a majority voting achieved an average precision of 77.75% with an improvement over the baseline by 15.00%. This result indicates that word sense disambiguation can be improved by combining base classifiers and the concept co-occurrence information-based approach is very promising for real world MT systems.

We plan further research on feature selection. Compared with the surface form information of lexical words, the concept codes are somewhat diluted information as clues for WSD. Thus we will be able to improve the performance of system if we add other features to our disambiguation model, such as lexical words and part of speech of surrounding words. Also, we have a plan to develop a new similarity measure to find the more feasible similarity values for our system.

Acknowledgements

This work was supported by the Korea Science and Engineering Foundation (KOSEF) through the Advanced Information Technology Research Center(AITrc).

References

- Ardeshir, G. (2002) Decision Tree Simplification for Classifier Ensembles. PHD Thesis, University of Surrey, U.K.
- Ide, N. and Veronis, J. (1998) *Word Sense Disambiguation: The State of the Art*. Computational Linguistics, Vol.24, No.1, pp.1-40
- Li, H. F., Heo, N. W., Moon, K. H., Lee, J. H. and Lee, G. B. (2000) *Lexical Transfer Ambiguity Resolution Using Automatically-Extracted Concept Co-occurrence Information*. International Journal of Computer Processing of Oriental Languages, Vol.13, No.1, pp.53-68
- McRoy, S. (1992) *Using Multiple Knowledge Sources for Word Sense Discrimination*. Computational Linguistics, Vol.18, No.1, pp.1-30
- Ng, H. T. and Zelle, J. (1997) *Corpus-Based Approaches to Semantic Interpretation in Natural Language Processing*. AI Magazine, Vol.18, No.4, pp.45-64
- Ohno, S. and Hamanishi, M. (1981) *New Synonym Dictionary*. Kadokawa Shoten, Tokyo
- Smadja, F. (1993) *Retrieving Collocations from Text: Xtract*. Computational Linguistics, Vol.19, No.1, pp.143-177
- Yarowsky, D. (1992) *Word-Sense Disambiguation Using Statistical Models of Roget's Categories Trained*

on Large Corpora. In *Proceedings, COLING-92*. Nantes, pp. 454-460