Journal of the Research Institute of Industrial Technology. Vol. 21, 2004

Automatic Corpus Construction for Linear Alignment

Jae-Hoon Kim
Department of Computer Engineering
Korea Maritime University
#1 Dongsam-Dong, Yeongdo-Gu
Busan 606-791, Korea
jhoon@mail.hhu.ac.kr

Kong Joo Lee
Department of Computer Science & Engineering
Ewha Womans University
11-1 Daehyun-Dong, Seodaemoon-Ku,
Seoul 120-750, Korea
kjlee007@ewha.ac.kr

Abstract

This paper presents a method for automatically building aligned corpora for transliteration, grapheme-phoneme conversion, morphological analysis, and so on. To achieve this goal, we use an unsupervised linear alignment algorithm, which inserts null characters into both of two aligned strings in case that two input strings are different from each other in length. This may cause some problem like search space explosion. To alleviate the problem, we modify the algorithm not to contain the null characters in the aligned source strings. Using the modified algorithm, aligned corpora are automatically constructed. We have shown the usability of our approach by applying the constructed aligned corpora to several areas such as Korean-to-English back-transliteration, English grapheme-to-phoneme conversion, and construction of a finite state transducer for Korean morphological analysis. Consequently, we have observed that the aligned corpora are very useful although they are automatically constructed.

1 Introduction

Recently, a range of annotated corpora like part-of-speech tagged corpora (Marcus et al, 1994; Kim and Kim, 1995) and parallel corpora (Hansards, 2001; Shin, 1996) are widely used in natural lan-

consuming works, but also presupposes accurate skills of individuals with good linguistic knowledge that are to be annotated. The time and cost are still magnificent even though semi-automatic annotation systems are used.

To lighten the cost, we present a method for automatically constructing aligned corpora for transliteration, grapheme-phoneme conversion, morphological analysis, and so on. To do this, we propose an unsupervised linear alignment algorithm. A linear alignment refers to the procedure of aligning two different strings (source and target strings)[1] by looking for pairs of characters (or substrings) that are in the same order in the strings. Table 1[2] shows the linear alignment of an English word 'corpus' and its pronunciation /K/AO/R/P/AH/S/[3] as an example. As you can see in Table 1, each character (or grapheme) in the word has been aligned with each phonetic symbol (or phoneme) in its pronunciation in order one by one.

word	c	o	r	p	u	s
pronunciation	K	AO	R	P	AH	S

Table 1: The linear alignment of an English word 'corpus' and its pronunciation /K/AO/R/P/AH/S/

However, the lengths of two input strings are not always same as shown in Table1. In order to align two different strings in length, null characters are required (Yvon, 1994). As an example, see Table 2[4], which shows the linear alignment of an

[1] Hereafter a terminology 'string' can be intermixed with 'sequence'.

[2] There are generally two systems for representing phones: International Phonetic Alphabet (IPA) and ARPAbet (Jurafsky and Martin, 2000). In this paper, we use the latter AR-

English word 'exercise' and its pronunciation /EH/K/S/ER/S/AY/Z/.

word	e	x	ϕ	e	r	c	i	s	e
pronunciation	EH	K	S	ER	ϕ	S	AY	Z	ϕ

Table 2: The linear alignment of an English word 'exercise' and its pronunciation "/EH/K/S/ER/S/AY/Z/".

The null characters may cause some problem when we conduct some applications using an aligned corpus including the null characters. For example, suppose that we adopt a classifier based on machine learning techniques (Mitchell, 1997) like decision tree, instance-based algorithms, and so on. In general, the classifier produces a class (eg. a phoneme /AY/) for a given feature vector (eg. a trigram ['c', 'i', 's'] centered at the character 'i' in 'exercise'). Consider the null character in the first row of Table 2. A phoneme /S/ corresponding to the null character cannot be generated through the classifier because, in the source sequence, the position of the null character to be aligned cannot be predicted in advance. As a result, the problem can make system complexity increase considerably as well as can cause search space explosion for finding classes.

Hence, we are eager to construct aligned corpora without the null characters. This is our goal. We, however, desire that the null characters be only removed from aligned source sequences. Namely, there may be the null characters in aligned target sequences. This should not get into any trouble for achieving our goal.

Now we return to discuss a method for the linear alignment. There is an unsupervised linear alignment algorithm, which is the integration of an expectation maximization (EM) method (Ristad and Yianilos, 1998) and a linear alignment algorithm (Allison et al, 1999). The former is used to estimate stochastic edit distances, which are the probabilities to perform edit operations (insertion, deletion or substitution) to align a source sequence with a target sequence. The latter is employed to align two strings using the estimated stochastic edit distances and a dynamic programming technique. The linear alignment algorithm generates aligned in order that null characters disappear in aligned source sequences. See more details in Section 3.

Anyway, an aligned corpus constructed using the modified algorithm is uni-directional, but not bi-directional. In other words, two aligned corpora (eg. grapheme-to-phoneme and phoneme-to-grapheme corpora) are different from each other. Unlike in methods for corpus construction, this is not a big problem in this paper because they can be constructed automatically.

The remainder of this paper is organized as follows. In Section 2, we begin with an overview of processing steps in this work. In Section 3, we describe modification of an unsupervised linear alignment algorithm and in Section 4, we discuss construction of aligned corpora in several fields. In Section 5, we mention applications of the aligned corpora and in Section 6, we finally draw conclusions and further studies.

2 Overview

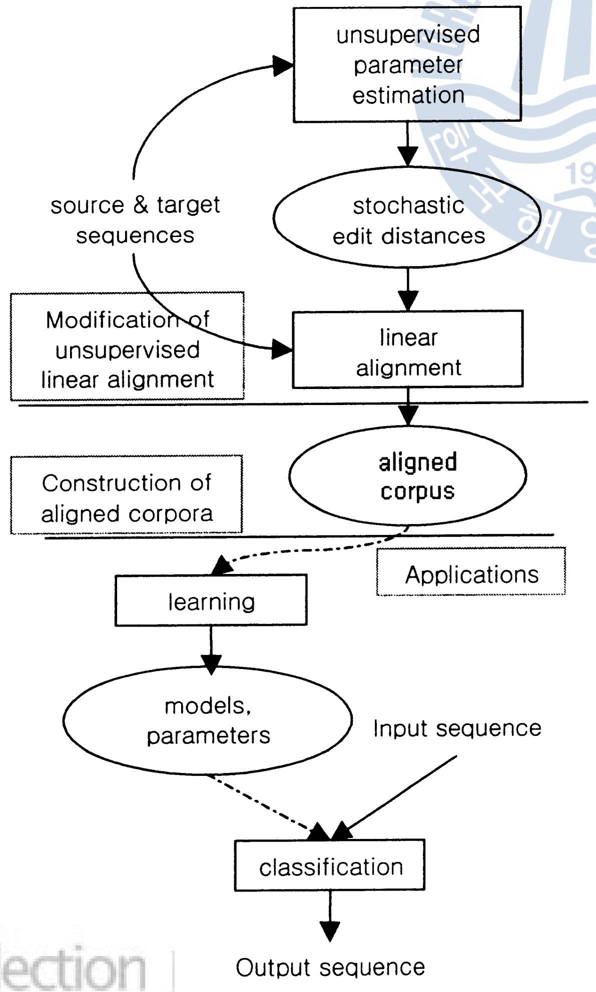
In this section, we describe an overview of processing steps in this work, which is shown in Figure 1. The processing consists of three steps: 1) modification of an unsupervised linear alignment algorithm, 2) construction of aligned corpora, and 3) applications of the aligned corpora to real systems. Each step is described in three subsequent sections in detail.

The first step is the modification of an unsupervised linear alignment algorithm (Ristad and Yianilos, 1998; Allison et al, 1999), which is composed of two sub-algorithms: one is an unsupervised learning algorithm known as the EM algorithm for estimating stochastic edit distances and the other is a linear alignment algorithm based on a dynamic programming technique for constructing aligned corpora. The modification is conducted on both of them.

The second step is the construction of three aligned corpora for Korean-English transliteration, English grapheme-phoneme conversion, and Korean morphological segmentation and analysis. Each aligned corpus consists of two sub-corpora, due to its uni-directionality. For example, Korean-to-English back-transliteration and English-to-

to-English back-transliteration and English-to-Korean transliteration should be generated in case of Korean-English transliteration.

In the third step, we apply the aligned corpora to three applications, which are Korean-to-English back-transliteration, English grapheme-to-phoneme conversion, and construction of finite-state transducers (FST) for Korean morphological analysis. Each application system have two sub-systems: one is a learning system for estimating parameters depending on a machine learning algorithm and the other is a classification system for determining classes as output sequences based on a classification model and its parameters for given input sequences.


unsupervised parameter estimation
source & target sequences
stochastic edit distances
Modification of unsupervised linear alignment
linear alignment
aligned corpus
Construction of aligned corpora
Applications
learning
models, parameters
Input sequence
classification
Output sequence

3 Modification of Unsupervised Linear Alignment Algorithm

In this section, we describe modification of an existing unsupervised linear alignment algorithm (Ristad and Yianilos, 1998; Allison et al, 1999). As mentioned before, null characters must be used for aligning two input strings of different lengths. To remove this problem, we modify the algorithm in order that null characters disappear in aligned source strings, but may appear in aligned target strings.

One character in each source string is inevitably aligned with zero or more characters in the corresponding target string. Consequently, insertion operations for the linear alignment should be modified not to allow any null characters in the aligned source strings.

First, we define three kinds of edit operations for linear alignment: insertion, deletion, and substitution as follows:

1. insertion: $\phi \rightarrow$ b
   "Inserts a 'b' into target without effecting the source";
2. deletion: a $\rightarrow \phi$
   "Deletes an 'a' from source without effecting the target";
3. substitution: a $\rightarrow$ b
   "Substitutes a 'b' into target for an 'a' in source;

To achieve our goal, the insertion operation is modified as:

1'. insertion: a $\rightarrow$ c:b
   "Substitutes a 'b' into target for an 'a' in source and inserts a 'c' into the front position of 'b' in the target"

According to the modification of the insertion operation, the unsupervised learning algorithm described by Ristad and Yianilos (1998) should be modified, but the modification is not presented here in detail because except something related to insertion operations, most part of the modified algorithm is same as that of Ristad and Yianilos (1998) and the space of this paper is not enough.

Figure 2 shows the modified linear alignment algorithm from Allison et al (1999). In Figure 2, a where, $\Pr(x)$ is a probability of x, and $\Pr(deletion)$, $\Pr(insertion)$, and $\Pr(substitution)$ are probabilities that edit operations of deletion, insertion, and substitution are performed to align two strings, respectively. |y| represents the length of y.

Table 3 shows the linear alignment generated by the modified algorithm. The character 'x' in word 'exercise' is aligned with a subsequence /K:S/ in the phoneme sequence /EH/K/S/ER/S/AY/Z. This result is very different from that in Table 2. As a result, the null characters in the aligned source sequence of Table 2 disappear in Table 3.

word	e	x	e	r	c	i	s	e
pronunciation	EH	K:S	ER	ϕ	S	AY	Z	ϕ

Table 3: The linear alignment generated by the modified linear alignment algorithm for the example in Table 2.

4 Construction of Aligned Corpora Using Linear Alignment Algorithm

In this section, we describe construction of three aligned corpora: 1) Korean-English transliteration, 2) English grapheme-phoneme conversion, and 3) construction of a finite-state transducer for Korean morphological analysis. Every aligned corpus except the last one has two sub-corpora.

4.1 Korean-English transliteration

More recently, we often use many transliterated words in Korean documents. For example, here are Korean words transliterated from English words: "파일 (file)", "뉴욕 (New York)", "메모리 (memory)", and "가톨릭 (catholic)". In machine translation and multilingual information retrieval, it is very important to restore the transliterated words to their original words in order to look for synonyms or translation equivalents.

From Great Korean Dictionary (NAKL, 2000), we collect 23,576 pairs of transliterated words in Korean and their origin words in Roman alphabet. Most transliterated words come from English, but, some of them come from Japanese, French, Latin, Russian, German, and so on. Among 23,576 pairs, we collect 14,547 English-Korean transliterated

alignment algorithm described in Section 3, we have automatically aligned the selected pairs to

```
Modified linear alignment algorithm
Input: two strings,  S[1..n] and T[1..m]
Output:  pairs P[0..n]
Method:
  // M[0..n, 0..m] :
  //   M[i,j] = Prob.  of sequence  from <0,0> to <i,j>
  //   for forward processing
  // B[0..n, 0..m] :
  //   B[i,j] = Best predecessor at <i, j>
  //   for backward processing
  // P[0..n] :
  //   P[i] = indexes of the i-th aligned pairs

  M[0,0] = 1;  B[0,0] =<-1, -1>
  for i ← 1 to n do
    M[i, 0] = M[i-1, 0] * c(S[i], "ϕ");
    B[i, 0] = <i-1, 0>;
  enddo
  for j ← 1 to m do
    M[0, j] = M[0, j-1] * c(S[0], T[j-1:j]);
    B[0, j] = <0, j-1>;
  enddo
  for i ← 1 to n do
    for j ← 1 to m do
      M[i,j] = max(
        M[i-1, j-1] * c(S[i], T[j]), // substitution
        M[i-1, j] * c(S[i], "ϕ"),    // deletion
        M[i, j-1] * c(S[i], T[j-1:j]) // insertion
      );
      B[i, j] = argmax (
        M[i-1, j-1] * c(S[i], T[j]), // substitution
        M[i-1, j] * c(S[i], "ϕ"),    // deletion
        M[i, j-1] * c(S[i], T[j-1:j]) // insertion
      );
    enddo
  enddo
  P[n] = B[n, m];
  for i ← n-1 to 0
    P[i] = P[i+1];
  return P;
```

Figure 2: The modified linear alignment algorithm.

construct aligned corpora for Korean-to-English back-transliteration (Table 4) and English-to-Korean transliteration (Table 5).

Table 4 and 5 illustrate the part of the two aligned corpora. We decompose Korean transliterated words into a sequence of Hangeul Jamo[5] as aligning the pairs. All consonants in Jamo can be used as both of leading and trailing consonants.

[5] Hangeul is the Korean Alphabet and basically consists of 14

ated words into a sequence of Hangeul Jamo[5] as aligning the pairs. All consonants in Jamo can be used as both of leading and trailing consonants. For example, A letter ‘ㄹ’ is one of consonant Jamo and a word “가톨릭 (catholic)” in Table 4 has two ‘ㄹ’s: One is used as the trailing consonant (eg. ‘ㄹ’ in ‘톨’) and the other is the leading consonant (eg. ‘ㄹ’ in ‘릭’). In order to distinguish these from each other in a decomposed sequence, we add a special mark ‘_’ to the end of trailing Jamo. As shown in Table 4 and 5, the alignments are very understandable and realistic, but not reversible.

We have observed several phonetically-interesting alignments in Table 4 and 5. First, the leading consonant ‘ㅇ’ in the first example ‘가운’ in Table 4 has no sound value, so it is aligned with the null character ‘ϕ’. Actually most ‘ㅇ’ characters are aligned with null characters in the aligned corpus for Korean-to-English back-transliteration since every leading consonant ‘ㅇ’ has no sound value in Korean. Second, a sub-string ‘th’ in English words can be transliterated into ‘▪’, ‘▪’, or ‘▪’ in Korean usually. Such the alignment can be found in the third example ‘▪ ▪ ▪’ of Table 4. Next, the silent letters like ‘p’ in ‘pseudo’ are coupled with the null characters as shown in Table 5.

가운	ㄱ	ㅏ	ㅇ	ㅜ	ㄴ_				
gown	g	o	ϕ	w	n				
메저	ㅁ	ㅔ	ㅈ	ㅓ					
measure	m	e:a	s	u:r:e					
가톨릭	ㄱ	ㅏ	ㅌ	ㅗ	ㄹ_	ㄹ	ㅣ	ㄱ_	
catholic	c	a	t:h	o	l	ϕ	i	c	
메커니즘	ㅁ	ㅔ	ㅋ	ㅓ	ㄴ	ㅣ	ㅈ	ㅡ	ㅁ_
mechanism	m	e	c:h	a	n	i	s	ϕ	m
슈도코드	ㅅ	ㅠ	ㄷ	ㅗ	ㅋ	ㅗ	ㄷ	ㅡ	
pseudocode	p:s:e	u	d	o	c	o	d	e	

Table 4: The part of the aligned corpus for Korean-to-English back-transliteration.

gown	g	o	w	n						
가운	ㄱ	ㅏ	ㅇ:ㅜ	ㄴ_						
measure	m	e	a	s	u	r	e			
메저	ㅁ	ㅔ	ϕ	ㅈ	ㅓ	ϕ	ϕ			

catholic	c	a	t	h	o	l	i	c		
가톨릭	ㄱ	ㅏ	ㅌ	ϕ	ㅗ	ㄹ:ㄹ_	ㅣ	ㄱ		
mechanism	m	e	c	h	a	n	i	s	m	
메커니즘	ㅁ	ㅔ	ㅋ	ϕ	ㅓ	ㄴ	ㅣ	ㅈ	ㅡ:ㅁ_	
pseudocode	p	s	e	u	d	o	c	o	d	e
슈도코드	ϕ	ㅅ	ϕ	ㅠ	ㄷ	ㅗ	ㅋ	ㅗ	ㄷ	ㅡ

Table 5: The part of the aligned corpus for English-to-Korean transliteration.

4.2 English grapheme-phoneme conversion

The grapheme-phoneme conversion is one of real problems, to which a linear alignment approach is applicable very well. Some systems for grapheme-phoneme conversion are bi-directional (Knight and Graehl, 1998). In other words, they can convert a word as a sequence of characters into its pronunciation as a sequence of phonemes, and vice verse. To solve this problem is very trivial, only if there is a finite set of words and its complete pronouncing dictionary. The number of words, however, continues to grow steadily since newly coined words appear frequently in documents of various domains and on the Internet. Because we do not have any pronunciation information in the dictionary for these new words, an automatic grapheme-phoneme conversion system can be very useful in speech synthesis and speech recognition.

From CMU pronouncing dictionary (CMU, 1997), we obtain 127,069 pairs of English words ant their pronunciations. After we remove pairs with special symbols like ‘#’, ‘%’, and so on, and normalize accent representations by eliminating digits in phonetic symbols (eg. /AH0/ → /AH/). Among the 127,069 pairs, we choose 45,000 pairs for this application. In the same way as described in Section 4.1, we can construct the aligned corpora for English grapheme-to-phoneme (Table 6) and English phoneme-to-grapheme (Table 7).

exotic	e	x	o	t	i	c
IH G Z AA T IH K	IH	G:Z	AA	T	IH	K
usage	u	s	a	g	e	
Y UW S AH JH	Y:UW	S	AH	JH	ϕ	
island	i	s	l	a	n	d
AY L AH N D	AY	ϕ	L	AH	N	D
taxi	t	a	x	i		
T AE K S IY	T	AE	K:S	IY		

REYSIHZAHM	R	EY	S	IH	Z	AH	M
racism	r	a	c	i	s	ϕ	m

Table 7: The part of the aligned corpus for English phoneme-to-grapheme conversion

In general, an English character 'x' is mapped into /G/Z/ or /K/S/ as phoneme sequences. The first and the fourth row in Table 6 show these mappings, respectively. In Table 7, their reverse alignments are found and phonemes /Z/ and /S/ are aligned with the null character 'ϕ' as a silent sound under the context following the mapping (/K/ → 'x'). This alignment may cause some errors uncommonly. An English letter 'u' often pronounces as a phoneme sequence /Y/UW/. Here, a phoneme /Y/ represents a semivowel and the alignment can be seen in the second row of Table 6. Its reverse alignment is to align the semivowel with the null character, and is found in Table 7. A silent letter 's' in 'island' is also paired with null character, and the alignment is found in Table 6. Most aligned pairs in Table 6 and 7 seem to be very understandable.

4.3 Construction FST for Korean Morphological Analysis

Finite-state transducers (FST) have been widely used for morphological analysis (Antworth, 1990; Lee, 1992). In general, this is known as two-level model, which is bi-directional. This model represents the correspondences between lexical forms (eg. "try+s") and their surface forms (eg. 'tries'). If we view this model in aligning context, the correspondences are alignments between lexical forms and surface forms for morphological analysis. From Korean POS tagged corpus (Kim and Kim, 1995), we get 172,703 pairs of Korean eojeols[6] and their morphological structures. Using the unsupervised linear alignment algorithm described in Section 3, we have automatically aligned the pairs to construct the aligned corpus for FST for Korean morphological analysis (Table 8).

Table 8 shows the part of the alignments between lexical forms and surface forms for Korean morphological analysis. As you can see in Table 8, both of lexical and surface forms are represented as decomposed Jamo sequences, and a leading consonant 'ㅇ' is omitted in the sequences (Lee, 1992) because it does not have sound value in Korean, as mentioned above. A symbol '+' indicates a separator between morphemes. A lexical form reveals how to segment a sequence of morphemes and what is the root form for a surface. For example, the character 'ㅜ' in the surface form '아름다운' of the first pair in Table 8 is restored to the root form 'ㅂ' in the lexical form. This restoration is one of notorious phonological variations in Korean. The pairs in the second row in Table 8 demonstrates another phonological variation called contraction (eg. "ㄱ ㅕ ㅆ_" → "ㄱ:ㅣ +:ㅓ ㅆ_:+").

아름다운	ㅏ	ㄹ	ㅡ	ㅁ_	ㄷ	ㅏ	ㅜ	ㄴ	
아름답+ㄴ	ㅏ	ㄹ	ㅡ	ㅁ_	ㄷ	ㅏ	ㅂ_:+	ㄴ	
남겼다	ㄴ	ㅏ	ㅁ_	ㄱ	ㅕ	ㅆ_	ㄷ	ㅏ	
남기+었+다	ㄴ	ㅏ	ㅁ_	ㄱ:ㅣ	+:ㅓ	ㅆ_:+	ㄷ	ㅏ	
결정적	ㄱ	ㅕ	ㄹ_	ㅈ	ㅓ	ㅇ_	ㅈ	ㅓ	ㄱ_
결정+적	ㄱ	ㅕ	ㄹ_	ㅈ	ㅓ	ㅇ_:+	ㅈ	ㅓ	ㄱ_
우려되고	ㅜ	ㄹ	ㅕ	ㄷ	ㅚ	ㄱ	ㅗ		
우려+되+고	ㅜ	ㄹ	ㅕ:+	ㄷ	ㅚ:+	ㄱ	ㅗ		
증가는	ㅈ	ㅡ	ㅇ_	ㄱ	ㅏ	ㄴ	ㅡ	ㄴ_	
증가+는	ㅈ	ㅡ	ㅇ_	ㄱ	ㅏ:+	ㄴ	ㅡ	ㄴ_	

Table 8: The part of the aligned corpus as the correspondences between lexical and surface forms.

5 Three Applications

In this section, we describe the application of the aligned corpora constructed in Section 4 into three domains, which are Korean-to-English back-transliteration, English grapheme-to-phoneme, and Korean morphological analysis. Our goal is only to show the usability of the aligned corpora, but not to improve the performance of models nor to find the best model for each application. Therefore, we only adopt a basic and simple model such as an HMM, and do not enrich the model with any other information like dictionaries and heuristics.

5.1 Experimental Environments

A classifier adopted in this work calculates *N*-best sequences matching a given input sequence most properly. To do this, we employ a second ordered only adopt a basic and simple model such as an HMM, and do not enrich the model with any other information like dictionaries and heuristics.

5.1 Experimental Environments

A classifier adopted in this work calculates *N*-best sequences matching a given input sequence most properly. To do this, we employ a second ordered hidden Markov model (Rainber, 1989) and an *N*-best path algorithm (Huang et al, 1994). Table 9 shows the size of corpora for the preliminary experiments. All the corpora are constructed in Section 4. We divide them into training and test corpora according to circumstances of each aligned corpora in arbitrary way.

Applications	training	test
1) K-to-E back-transliteration	13,720	870
2) English grapheme-to-phoneme	44,000	1,000
3) Korean morphological analysis	131,581	41,122

Table 9: The size of corpora used in preliminary experiments.

5.2 Performance

Performance for each application is shown in Table 10. In this work, the performance is measured in terms of accuracy, which is the ratio of the occurrences with the correct answer among 10-best candidate sequences generated by the classifier.

Applications	Accuracy (%)
1) K-to-E back-transliteration	51
2) English grapheme-to-phoneme	61
3) Korean morphological analysis	99

Table 10: The accuracy performance in preliminary experiments.

The accuracies except the third result are lower than that we expect. We will leave room for the improvement for future studies, and certainly believe that the improvement should be outstanding if additional information such as dictionary or/and heuristics could be used.

models (Lee, 1999; Jung et al, 2000), a weighted finite-state transducer (WFST) (Knight and Graehl, 1998), and decision trees (Bahl et al, 1989). Their performances are very different from each other depending on models and experimental environments. Hence, we cannot compare ours with others objectively. The accuracy of the first application in Table 10 is very close to that of a direct transliteration method proposed by Lee (1999), which is conceptually similar to that proposed by this paper. Our approach for back-transliteration should be improved by using additional information such as a higher order model (Jung et al, 2000), a word model (Knight and Graehl, 1998), a larger amount of training sets extracted from real text, and so on.

Many researchers conduct grapheme-to-phoneme conversion using several methods such as rule induction (van Coile, 1991), instance-based learning (Daelemans and van den Bosch, 2001), neural networks (Sejnowsk and Rosenberg, 1987), hidden Markov model (Rainber, 1989), and analogy-based model (Yvon, 1994). The performance of these models also vary from each other depending on their models and experimental environments. Like the first application, its performance should be easily improved by using syntactic information like part-of-speech, some heuristics, words model, and so on.

A study on automatic construction of FST for Korean morphological analysis has hardly performed, and so our approach is very creative. The accuracy of third application is very higher than that of others as you can see in Table 10. Moreover, the accuracy reaches 98% under even 3-best candidate sequences. Note that the average number of morphological structures generated by Korean morphological analyzers (Kim, 1998) is approximately nine. It is very useful for information retrieval to look for index terms from running text by concatenating characters in the output sequences. It can also be expanded into POS tagging easily.

6 Conclusions and Further Study

In this paper, we propose a new method for constructing aligned corpora for transliteration, graphalthough they are automatically constructed and have vastly saved time and cost to build them.

The performance of the applications is not our main concern in this work. Hence, a further direction of this study will be to develop practical systems with acceptable accuracies for three applications based on the aligned corpora.

References

Allison, L., Powell, D. and Dix, T. I. 1999. Compression and approximate matching. *The Computer Journal*, 42(1):1-10.

Antworth, E. L. 1990. *PC-KIMMO: A Two-level Processor for Morphological Analysis*. Summer Institute of Linguistics.

Bahl, L. R., de Souza, P. V., Gopalakrishnan, P. S., Nahamoo, D. and Picheny, M. H. 1989. Decision tree for phonological rules in continuous speech. *Proceedings of Proceedings of the International Conference on Acoustics, Speech and Signal*, pp.185-188

CMU. 1997. *CMU Pronouncing Dictionary*. http://www.speech.cs.cmu.edu/speech/

Daelemans, W. and van den Bosch, A. 2001. TreeTalk: Memory-based word pronemisation. (ed.) R. I. Damper, *Data-driven techniques in Speech Synthesis*, Kluwer, pp. 149-172.

Hansards. 2001. *Aligned Hansards of the 36th Parliament of Canada Release 2001-1a*, the Natural Language Group of the USC Information Sciences Institute – Ulrich Germann, Editor.

Huang, E.-F., Soong, F. K., and Wang, H.-C. 1994. The use of tree-trellis search for large-vocabulary mandarin polysyllabic word speech recognition. *Computer Speech and Language*, 8: 39-50.

Jung, S. Y., Hong, S. L., and Paek, E . (2000). An English to Korean Transliteration Model of Extended Markov Window. *Proceedings of the 18th International Conference on Computational Linguistics*. pp. 383-389.

Jurafsky, A. and Martin, J. H. 2000. *An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice-Hall.

Kim, J.-H. 1998. Reduction of morphological ambiguity in Korean using morphological sub-

Kim, J.-H. and Kim, G. C. 1995. *Guidelines on Building a Korean Part-of-Speech Tagged Corpus : KAIST Corpus*, Technical report CS-TR-95-99, Department of Computer Science, Korea Advanced Institute for Science and Technology.

Knight, K. and Graehl, J. 1998. Machine Transliteration. *Computational Linguistics* 24(4):599-612.

Lee, J.-S. 1999. *An English-Korean Transliteration and Retransliteration Model for Cross-lingual Information Retrieval*, Ph.D. Dissertation, Department of Computer Science, Korea Advanced Institute for Science and Technology.

Lee, S.-J. 1992. *A Two-Level Morphological Analysis in Korean*, Master Thesis, Department of Computer Science, Korea Advanced Institute for Science and Technology.

Marcus, M. P. and Santorini, B. and Marcinkiewicz, M. A. 1994. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, 19(4): 313-330.

Mitchell, T. M. 1997. *Machine Learning*, McGraw-Hill.

NAKL, 2000. *Pyo-Jun Great Korean Dictionary*. © Doosan Dong-a and The National Academy of the Korean Language.

Rainber, L. R. 1989. A tutorial on hidden Markov models and selected application in speech recognition. *Proceedings of the IEEE*, 77(2): 257-286.

Ristad, E. and Yianilos, P. 1998. Learning string edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(2): 522-532.

Sejnowski T. J. and Rosenberg C. R. 1987. Parallel network that learn to pronounce English text. *Complex Systems*, 1:145-168.

Shin, J.-J. 1996. *Aligning a parallel Korean-English corpus at word and phrase level*, Master Thesis, Department of Computer Science, Korea Advanced Institute for Science and Technology.

van Coile, B. 1991. Inductive learning of pronunciation rules with the DEPES system. *Proceedings of the International Conference on Acoustics, Speech and Signal*, pp. 745-748.

Yvon, F. 1994. Self-learning techniques for grapheme-to-phoneme conversion. *Proceedings of the*

Sejnowski T. J. and Rosenberg C. R. 1987. Parallel network that learn to pronounce English text. *Complex Systems*, 1:145-168.

Shin, J.-J. 1996. *Aligning a parallel Korean-English corpus at word and phrase level*, Master Thesis, Department of Computer Science, Korea Advanced Institute for Science and Technology.

van Coile, B. 1991. Inductive learning of pronunciation rules with the DEPES system. *Proceedings of the International Conference on Acoustics, Speech and Signal*, pp. 745-748.

Yvon, F. 1994. Self-learning techniques for grapheme-to-phoneme conversion. *Proceedings of the 2nd Onomastica Research Colloquium*. London.