자율 학습을 이용한 선형 정렬 말뭉치 구축

이 공 주† · 김 재 훈††

요 약

본 논문에서는 자율 선형 정렬 알고리즘을 이용하여 선형 정렬 말뭉치를 구축하는 방법을 제안한다. 기존의 자율 선형 정렬 알고리즘을 이용하여 선형 정렬 말뭉치를 구축할 경우, 두 문자열의 길이가 서로 다르면 정렬된 두 문자열(입력열과 출력열)에 모두 공백문자가 나타난다. 이 방법을 그대로 사용하면 정렬 말뭉치의 구축은 용이하나 정렬된 말뭉치를 이용하는 응용 시스템에서는 탐색 공간이 기하급수적으로 늘어날 뿐 아니라 구축된 정렬 말뭉치는 다양한 기계학습 방법에 두루 사용될 수 없다는 문제가 있다. 본 논문에서는 이들 문제를 최소화하기 위해서 입력열에는 공백문자가 나타나지 않도록 기존의 자율 선형 정렬 알고리즘을 수정하였다. 이 알고리즘을 이용해서 한영 음차 표기 및 복원, 영어 단어의 발음 생성, 영어 발음의 단어 생성, 한국어 형태소 분리 및 복원을 위한 정렬 말뭉치를 구축하였으며, 간단한 실험을 통해, 그들의 실용성을 입증해 보였다.

Construction of Linearly Aliened Corpus Using Unsupervised Learning

Kong Joo Lee† · Jae-Hoon Kim††

ABSTRACT

In this paper, we propose a modified unsupervised linear alignment algorithm for building an aligned corpus. The original algorithm inserts null characters into both of two aligned strings (source string and target string), because the two strings are different from each other in length. This can cause some difficulties like the search space explosion for applications using the aligned corpus with null characters and no possibility of applying to several machine learning algorithms. To alleviate these difficulties, we modify the algorithm not to contain null characters in the aligned source strings. We have shown the usability of our approach by applying it to different areas such as Korean-English back-transliteration, English grapheme-phoneme conversion, and Korean morphological analysis.

키워드 : 자율 학습(Unsupervised Learning), 편집거리(Edit Distance), 선형 정렬(Linear Alignment), 한영/영한 음차복원(Korean-English (back) Transliteration), 음소-자소 변환(Grapheme-Phoneme Conversion), 한국어 형태소 분리(Korean Morphological Segmentation)

1. 서 론

품사 부착 말뭉치와 병렬 말뭉치와 같은 언어정보 부착 말뭉치[1]는 자연언어처리 분야에서 매우 널리 사용되고 있다[2]. 이러한 말뭉치의 대부분은 수동이나 반자동으로 구축되고 있으며, 이러한 말뭉치를 구축하는 데는 많은 시간과 노력이 필요하다[3]. 뿐만 아니라 말뭉치 구축은 고도의 숙련된 전문가에 의해서 구축되지 않으면 많은 오류를 범하게 되어 자칫하면 어렵게 구축된 말뭉치가 쓸모없게 될 수도 있다. 본 논문은 이런 문제를 다소 완화시키기 위해서 제한된 분야이기는 하지만 사람의 개입을 최소화하는 방법으로 말뭉치를 구축하고자 한다. 즉, 음차(音借) 변환이나 자소/음소 변환과 같은 영역에서 자율 선형 정렬 알고리즘을 이용해서 사람이 개입하지 않고 정렬된 말뭉치를 구축하는 방법을 제안한다. 또 구축된 정렬 말뭉치를 다양한 분야에 적용해 봄으로써 그 유용성을 보이고자 한다.

선형 정렬은 두 문자열(입력열과 출력열)에 대해서 문자들 사이의 대응 관계가 서로 교차되지 않는 정렬 방법이며, <표 1>[1]에서 영어 단어 "corpus"와 그 발음 "/K/AO/R/P/AH/S/"[2]의 선형 정렬을 보이고 있다. <표 1>에서 영어 단어의 각 문자와 발음의 각 음소가 차례로 하나씩 나란히 정렬되고 있다. 이와 같이 길이가 같은 두 문자열을 정렬할 경우, 대부분은 문자와 음소가 정확히 하나씩 정렬될 수 있

† 정 회 원 : 경인여자대학 컴퓨터정보기술학부 교수
†† 정 회 원 : 한국해양대학교 컴퓨터공학과 교수
논문접수 : 2003년 8월 1일, 심사완료 : 2004년 5월 13일

1) 발음 표기 방법에는 크게 IPA(International Phonetic Alphabet) 방법과 ARPAbet 방법이 있다[4]. 본 논문에서 사용되는 표기법은 ARPAbet 방법을 사용한다.

2) 본 논문에서 발음을 표기할 경우 음소들 사이를 구별하기 위해 기호 '/'를 사용한다.

으나, 길이가 서로 다른 두 문자열을 정렬할 경우에는 상황이 조금 다르다.

〈표 1〉 영어 단어 "corpus"와 그 발음 "/K/AO/R/P/AH/S/"의 선형 정렬

단 어	c	o	r	p	u	s
발 음	K	AO	R	P	AH	S

<표 2>를 살펴보자. <표 2>에서 정렬된 문자열에는 원래의 문자열과는 달리 공백문자(empty character) 'Ø'를 사용하고 있다. <표 2>의 단어와 발음의 길이 차이는 1이지만, 정렬된 결과에는 3개의 공백문자가 사용되었다. 이 정렬을 자세히 관찰해보면, 일반적으로 'x'는 "/K/S/"로 발음되며, 또한 "er"은 "/ER/"로 발음되며 단어 끝에 나오는 "se"는 "/Z/"로 발음되어야 하기 때문에 3개의 공백문자를 사용해서 이들이 정확하게 정렬될 수 있도록 하였다.

〈표 2〉 길이가 다를 경우 선형 정렬의 예 : 영어 단어 "exercise"와 그 발음 "/EH/K/S/ER/S/AY/Z/"

단 어	e	x	Ø	e	r	c	i	s	e
발 음	EH	K	S	ER	Ø	S	AY	Z	Ø

일반적으로 선형 정렬을 위해 동적 프로그래밍 정렬(dynamic programming(DP) alignment) 방법[4, 5]이 주로 사용된다. 이 알고리즘은 문자 간의 편집거리(edit distance)를 이용하는데, 여기서 말하는 편집거리는 대응되는 두 문자가 편집(혹은 정렬)에 필요한 행위 (즉, 삽입, 삭제, 대체)의 최소 횟수를 말한다. **선형 정렬 말뭉치 구축에서 DP 정렬 알고리즘의 문제점은 정렬된 문자열에 공백문자가 사용되어 이를 이용하는 응용 시스템에서 결과를 찾기 위한 탐색 공간이 기하급수적으로 늘어난다는 것이다.**

DP 정렬에서 공백문자의 사용은 큰 문제가 되지 않는다. 그러나 정렬된 말뭉치를 이용하여 응용 시스템을 구축할 경우에는 응용 시스템의 분류 알고리즘(classification algorithm)에 따라서 큰 문제가 될 수 있다. 분류 알고리즘으로 C4.5[6]를 사용한다고 가정하자[3]. <표 2>의 정렬 말뭉치를 C4.5로 학습하기 위해서는 <표 3>과 같은 학습 데이터로 변환해야 한다. <표 3>에서 c_i와 p_i는 각각 현재 문자와 그 문자에 대응되는 발음기호이고, c_{i-k}와 c_{i+k}은 각각 c_i의 k번째 이전 문자와 이후 문자를 의미하고 p_{i-k}와 p_{i+k}는 각각 c_{i-k}와 c_{i+k}에 대응하는 발음기호이다. 기호 'Δ'는 단어의 시작과 끝을 의미한다. <표 3>에서 보듯이 입력 단어 "exercise"에 존재하지 않은 문자 'Ø'이 c_i에 추가되어 있다. 응용 시스템에서 이를 그대로 이용할 경우, 언제 어디에서 문자 'Ø'를 추가해야 하는지 알 수 없으며 또한 몇 개를 추가해야 하는지도 알 수 없다. 때문에 일반적인 DP 정렬 알고리즘을 이용하여 <표 2>와 같이 구축된 정렬 말뭉치는 다양한 기계학습 방법[7]에 이용될 수 없다. 만약 그대로 사용하기 위해서는 기계학습 방법에 적용하기 전에 어떤 형태로든 가공이 되어야 한다.

〈표 3〉 〈표 2〉의 정렬 말뭉치를 이용한 C4.5의 학습 데이터

번 호	입 력 자 질						정 답
	c_{i-2}	p_{i-2}	c_{i-1}	p_{i-1}	c_i	c_{i+1}	p_i
1	Δ	Δ	Δ	Δ	e	x	EH
2	Δ	Δ	e	EH	x	Ø	K
3	e	EH	x	K	Ø	e	S
4	x	K	Ø	S	e	r	ER
5	Ø	S	e	ER	r	c	Ø
6	e	ER	r	Ø	c	i	S
7	r	Ø	c	S	i	s	AY
8	c	S	i	AY	s	e	Z
9	i	AY	s	Z	e	Δ	Ø

본 논문에서는 이런 문제를 최소화하기 위해서 **정렬된 입력 문자열에 공백문자가 나타나지 않도록 정렬 말뭉치를 구축한다.** 이렇게 정렬된 말뭉치는 <표 2>에서 가지고 있던 양방향성(bi-directionality)이 사라지게 된다. 즉 입력열이 영어 단어 혹은 발음일 경우에 따라 서로 다른 정렬 말뭉치를 구축해야 한다. 그러나 본 논문에서는 자율 학습 알고리즘[5]을 사용하기 때문에 이것은 큰 문제가 되지 않는다. 기존의 DP 정렬 알고리즘을 그대로 사용할 수 없기 때문에 본 논문에서 [5]의 DP 정렬 알고리즘을 수정하여 정렬 말뭉치를 구축하고, 구축된 말뭉치를 음차 복원 및 표기 등과 같은 몇몇 응용 분야에 적용해서 그 유용성을 보이고자 한다.

본 논문의 구성은 다음과 같다. 2장에서는 선형 정렬 말뭉치의 구축 및 응용 시스템에 대해서 개략적으로 기술하고, 3장에서는 기존의 자율 선형 정렬 알고리즘의 수정에 대해서 기술한다. 4장에서는 수정된 자율 선형 정렬 알고리즘을 이용한 정렬 말뭉치 구축 방법에 대해서 설명하고, 5장에서는 구축된 정렬 말뭉치를 이용한 응용 시스템의 성능에 대해서 기술한다. 끝으로 6장에서 결론을 맺고 앞으로의 연구 방향을 제시한다.

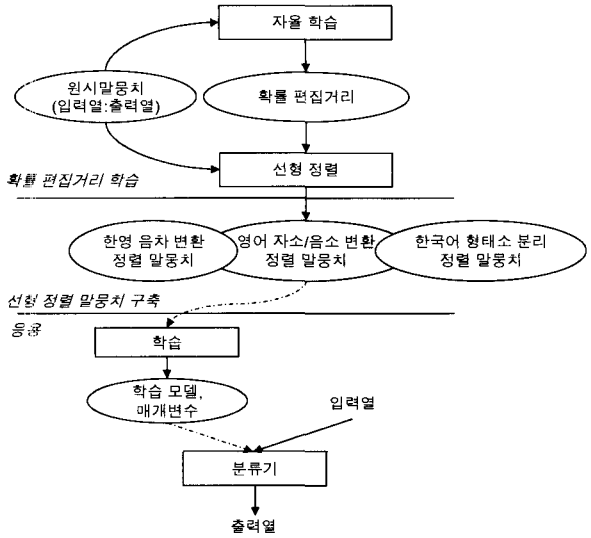
2. 선형 정렬 말뭉치의 구축 및 응용 시스템

(그림 1)은 본 논문에서 개발된 선형 정렬 말뭉치 구축 시스템과 그 응용 시스템의 관계를 개략적으로 보이고 있다.

본 논문에서 개발된 시스템은 크게 세 단계로 이루어졌다. 첫 번째 단계는 확률 편집거리(stochastic edit distance)의 학습 단계이고, 두 번째 단계는 선형 정렬 말뭉치의 구

3) 공개된 대부분의 기계학습 알고리즘에서는 <표 3>과 같은 학습자료 형식을 사용한다.

축 단계이고, 마지막 단계는 구축된 선형 정렬 말뭉치의 응용 단계이다. 이하에서는 이들 각 단계에 대해서 좀더 구체적으로 기술하고자 한다.



(그림 1) 선형 정렬 말뭉치 구축 및 응용 시스템의 개관

확률 편집거리의 학습 단계는 DP 정렬 알고리즘의 매개변수인 편집거리를 학습하는 단계이다. 앞에서 언급했듯이 본 논문에서는 기존의 DP 정렬 알고리즘[5]을 수정해서 사용하는데, 수정에 대한 구체적인 내용은 3장에서 기술할 것이다. 일반적으로 DP 정렬 알고리즘의 매개변수로 편집거리를 사용하는데, 본 논문에서는 확률 편집거리를 사용한다. 이 편집거리의 추정 방법(estimation method)으로 EM 알고리즘을 사용하며, EM 알고리즘은 자율학습(unsupervised learning)으로 미리 정렬된 말뭉치를 사용하지 않는다[4, 5]. 학습 방법은 원시 말뭉치인 입력열과 출력열 쌍을 EM 알고리즘에 입력하여 각 입력 문자에 대한 편집 연산(edit operation : 삭제, 대체, 삽입)의 가능 정도를 편집거리로서 추정한다. 추정된 편집거리를 이용해서 다음 단계에서 설명할 선형 정렬 알고리즘에 적용하여, 주어진 입력열에 대해 정확한 출력열을 찾는지를 검사한다. 이 검사를 통해서 오류가 충분히 작을 때까지 이와 같은 과정을 계속 반복한다.

선형 정렬 말뭉치의 구축 단계는 학습 단계에서 충분히 학습된 확률 편집거리를 모델로 수정된 DP 정렬 알고리즘을 수행함으로써 선형 정렬 말뭉치를 구축한다. 수정된 DP 정렬 알고리즘의 입력은 원시 말뭉치의 입력역과 출력열 쌍이며, 출력은 <표 1>과 같이 입력열과 출력열에 포함된 문자들의 정렬 결과이다. 본 논문의 **정렬 결과는 일반적인 정렬 결과와는 다르게 정렬된 입력열에는 공백문자 'Ø'가 포함되지 않는다**. 이런 정렬 결과는 1장에서 기술한 기존의 DP 정렬 알고리즘에 의해서 정렬된 말뭉치의 문제점을 해소할 수 있다. 본 논문에서는 한영 음차 표기 및 복원, 영어 단어의 발음 생성과 영어 발음의 단어 생성, 한국어 형태소 분리 및 복원을 위한 정렬 말뭉치를 구축하였으며 각 응용 분야의 원시 말뭉치의 구축 방법과 규모에 대해서는 4장에서 자세히 기술할 것이다.

선형 정렬 말뭉치의 응용 단계는 구축된 선형 정렬 말뭉치의 유용성을 검증하기 위해서 다양한 응용 분야에 적용하는 단계이다. 본 논문은 영어 단어를 소리나는 대로 한글로 표기하는 한영 음차 표기와 영어 단어를 발음 기호로 변환하는 영어 단어의 발음 생성 그리고 한국어 어절을 분리하여 형태소의 사전표현과 표층표현으로 정렬하는 한국어 형태소 분리 및 복원 분야에 적용하였으며, 좀더 구체적인 내용은 5장에서 다룰 것이다.

3. 자율 학습 및 선형 정렬 알고리즘의 수정

선형 정렬 방법은 은닉 마르코프 모델(hidden Markov model, HMM)[9], 정보이론(information-theoretic)[10], 동적 프로그래밍 방법(Dynamic programming)[11] 등 매우 다양한 방법으로 다루어지고 있다. 본 논문에서는 확률 이론에 기반을 둔 동적 프로그래밍 방법을 이용한다. 본 절에서는 Ristad와 Yilanilos에 의해서 제안된 기존의 자율 학습(unsupervised learning) 및 선형 정렬 알고리즘[5, 10]의 수정에 대해서 기술한다. 1장에서 이미 언급했듯이 입력열과 출력열의 길이가 다를 경우, 정렬된 문자열에 공백문자가 사용되어 구축된 정렬 말뭉치는 결정트리[6]와 SVM[8]과 같은 기존의 기계학습 알고리즘에 그대로 적용할 수 없다.

본 논문에서는 다른 여러 기계학습 방법에서 자유롭게 사용할 수 있는 정렬 말뭉치를 구축하는데 목적이 있다. 이를 달성하기 위해서 본 논문에서는 정렬된 입력열에 공백문자가 들어가지 않도록 기존의 알고리즘을 수정한다. 이렇게 수정된 알고리즘에서 입력열에 속한 각 문자에 대응하는 출력열의 문자의 길이는 0개 이상이 된다. 먼저 일반적인 선형 정렬 알고리즘의 편집 연산(삽입, 삭제, 대체)을 정의하면 아래와 같다.

① 삽입(Ø → b) : 입력열의 변화 없이 출력열에 문자 'b'를 삽입한다.
② 삭제(a → Ø) : 출력열의 변화 없이 입력열로부터 문자 'a'를 삭제한다.
③ 대체(a → b) : 입력열의 문자 'a'를 출력열에 포함된 문자 'b'로 대체한다.

위에서 정의된 편집 연산을 보아 알 수 있듯이 정렬된 입력열에 공백문자가 포함되지 않기 위해서 삽입연산을 수정해야 한다. 수정된 결과는 아래와 같다.

①' 삽입(a → c : b) : 입력열의 문자 'a'를 출력열에 포함될 문자 'b'로 대체하고, 출력열의 문자 'b' 바로 전에 있는 문자 'c'를 출력열에 삽입한다.

삽입 연산의 수정에 따라서 기존의 자율 학습 알고리즘[5]과 선형 정렬 알고리즘[2, 10]이 수정되어야 한다. 자율 학습 알고리즘의 경우는 삽입 연산 이외에 대부분은 그대로 사용하기 때문에 지면 관계상 구체적인 수정 내용은 본 논문에서 다루지 않을 것이다. (그림 2)는 수정된 선형 정렬 알고리즘을 보이고 있다. (그림 2)에서 표기법 <i, j>는 행렬에서 i번째 행과 j번째 열의 위치를 표현하고, 삽입 연산에서 T[i : j](i < j)는 문자열 "T[i] T[i+1] … T[j]"를 의미한다. 행렬 M[0..n, 0..m]은 <0, 0>에서 <i, j>까지의 정렬 확률로서 Pr(S[0 : i], T[0 : j])를 의미하며, B[0..n, 0..m]은 <i, j>에서 어떤 연산을 수행하는지를 저장하는데 이 정보를 이용해서 정렬된 각 문자의 쌍을 쉽게 찾을 수 있다. 그리고 함수 c(x, y)는 확률 편집거리로서 식 (1)과 같이 정의된다.

$$c(x, y) = \begin{cases} \Pr(delete) \times \Pr(x) & \text{if } y = \varnothing \\ \Pr(insert) \times \Pr(y) & \text{if } |y| > 1 \\ \Pr(match) \times \Pr(x, y) & otherwise \end{cases} \quad (1)$$

식 (1)에서 $P(delete)$는 삭제 확률이고, $P(insert)$는 삽입 확률이며, $P(match)$는 대체 확률이고, |y|는 문자열 y의 길이를 의미한다.

수정된 선형 정렬 알고리즘

```
입력 : 두 문자열 S[1..n]과 T[1..m]
출력 : 두 문자열의 정렬 결과를 표현하는 행렬 P[1..n]
방법 :
  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], "∅")
    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]), // 대체연산
        M[i-1, j] × c(S[i], '∅'), // 삭제연산
        M[i, j-1] × c(S[i], T[j-1 : j]) // 삽입연산
      ) ;
      B[i, j] = argmax (
        M[i-1, j-1] × c(S[i], T[j]), // 대체연산
        M[i-1, j] × c(S[i], '∅'), // 삭제연산
        M[i, j-1] × c(S[i], T[j-1 : j]) // 삽입연산
      ) ;
    enddo
  enddo
  P[n] = B[n, m] ;
  for i = n-1 to 0
    P[i] = P[i+1] ;
  return P ;
```

(그림 2) 두 문자열을 정렬하기 위한 동적 프로그래밍 알고리즘

수정된 선형 정렬 알고리즘에 의해서 단어 "exercise"와 발음 "/EH/K/S/ER/S/AY/Z/"를 선형적으로 정렬하면 <표 4>와 같다. <표 4>(ㄱ)은 입력열이 단어이고 출력열이 발음일 경우이며, <표 4>(ㄴ)은 입력열이 발음이고 출력열이 단어이다. 이처럼 수정된 선형 정렬 알고리즘을 사용할 경우에는 양방향성을 가지지 못한다는 단점을 가지고 있으나, 본 논문은 자율 학습 알고리즘을 사용하기 때문에 큰 문제가 없다. <표 4>(ㄱ)에서 단어 "exercise"의 'x'는 "/K:S/"와 정렬됨을 볼 수 있으며 's'는 "/Z/"와 정렬되고 마지막 'e'는 공백문자와 정렬됨을 볼 수 있다. <표 4>(ㄱ)은 <표 2>보다 훨씬 더 직관적인 정렬이라고 할 수 있다. <표 4>(ㄴ)에서 "/ER/"은 "e:r"로 정렬되었고, "/Z/"는 "s:e"로 정렬되었다. 이것도 또한 <표 2>의 정렬보다 더 직관적인 정렬일 것이다.

〈표 4〉 수정된 선형 정렬 알고리즘에 의한 정렬의 예

(ㄱ) 입력열이 단어이고 출력열이 발음일 경우

단어(입력열)	e	x	e	r	c	i	s	e
발음(출력열)	EH	K:S	ER	∅	S	AY	Z	∅

(ㄴ) 입력열이 발음이고 출력열이 단어일 경우

발음(입력열)	EH	K	S	ER	S	AY	Z
단어(출력열)	e	x	∅	e:r	c	i	s:e

4. 선형 정렬 말뭉치의 구축

본 장에서는 3장에서 기술한 수정된 선형 정렬 알고리즘을 이용해서 자연언어처리에서 필요로 하는 선형 정렬 말뭉치의 구축 사례를 소개하고자 한다.

4.1 한영 음차 표기 및 복원을 위한 정렬 말뭉치

최근 한국어 문서에는 음차 표기된 많은 외래어가 사용된다. 영어로부터 음차 표기된 단어 "파일 (file)", "뉴욕(New York)", "메모리(memory)", "가톨릭(catholic)" 등이 그 예이다. 기계번역이나 정보검색과 같은 분야에서는 대역어나 동의어를 찾기 위해 음차 표기된 외래어를 원래의 단어로 복원해야 한다[12]. 이와는 반대로, 번역하기 곤란한 신조어의 경우 외국어를 그대로 표기할 수도 있으나, 경우에 따라서는 외국어를 한국어로 음차 표기한다. 이와 같이 외국어를 음차 표기하거나 음차된 외래어를 복원하기 위해서 외국어 단어와 음차 표기된 외래어 단어의 정렬이 필요하다.

본 논문에서는 영어와 한국어 사이의 음차 표기 및 복원을 위한 정렬 말뭉치를 구축하고자 한다. 자율 선형 정렬을

위한 학습 데이터는 단어 단위로 음차 표기된 한국어와 영어 단어 쌍이다. 본 논문에서는 표준국어대사전[13]의 "외래어 표기 용례"로부터 23,576개의 단어 쌍을 수집하였으나 이들 중에서 영어 외에 일어, 불어, 노어 등에서 나온 단어를 제외하고 14,590개의 단어 쌍을 학습을 위해서 사용하였다. 3장에서 기술한 자율 학습 및 선형 정렬 알고리즘을 이용하여 자동적으로 구축하였으며, 그 결과의 일부를 <표 5>에 실었다. <표 5>(ㄱ)에서 입력열은 음차 표기된 한국어 단어의 자소열이고 출력열은 영어 단어의 자소열이며, <표 5>(ㄴ)에서 입력열은 영어 단어의 자소열이고 출력열은 음차 표기된 한국어 단어의 자소열이다. <표 5>에서 꼬리표 '_'를 달고 있는 한글 자소는 받침을 의미한다.

〈표 5〉 한영 음차 표기 및 복원을 위한 선형 정렬 말뭉치

(ㄱ) 한영 음차 복원을 위한 선형 정렬

가 운	ㄱ	ㅏ	ㅇ	ㅜ	ㄴ_				
gown	g	o	Ø	w	n				
메 저	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	Ø	i	c	
메커니즘	ㅁ	ㅔ	ㅋ	ㅓ	ㄴ	ㅣ	ㅈ	--	ㅁ_
mechanism	m	e	c:h	a	n	i	s	Ø	m
슈도코드	ㅅ	ㅠ	ㄷ	ㅗ	ㅋ	ㅗ	ㄷ	ㅡ	
pseudocode	p:s:e	u	d	o	c	o	d	e	

(ㄴ) 영한 음차 표기를 위한 선형 정렬

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

본 논문에서는 정렬된 선형 말뭉치로부터 음운적으로 몇 가지 사실을 발견할 수 있었다. 첫째로 한국어 초성의 'ㅇ'은 음가가 없기 때문에 대체로 'Ø'로 정렬된다. 그 현상을 <표 5>(ㄱ)의 "가운"에서 찾아볼 수 있다. "메저"에서 "ㅓ"에 "u:r:e"가 정렬되고, "가톨릭"에서 "ㅌ"가 "t:h"로 정렬되고, "메카니즘"에서 "ㅋ"가 "c:h"로 정렬됨을 볼 수 있다. 'ㅅ'은 "p:s:e"로 정렬되는데 이는 "pseudo-"로 시작하는 단어에서 'p'는 발음되지 않는 현상을 반영하고 있다. <표 5>(ㄴ)에서는 영어 단어 "catholic"의 'l'이 "ㄹ_:ㄹ"로 정렬되었다. 이는 일반적으로 'l'을 받침의 'ㄹ'과 초성의 'ㄹ'로 중복해서 표기하는 현상을 반영해 주고 있다. <표 5>에서 보는 바와 같이, 대체적으로 발음과 문자들의 대응이 잘 이루어졌으나, 학습말뭉치 내에 일부의 약어 등이 포함되어 약간의 문제를 일으키는 경우도 있었다.

4.2 영어 자소/음소 정렬 코퍼스 구축

자소/음소 선형 정렬 말뭉치를 이용하면 쉽게 자소와 음소 사이의 변환 시스템을 구축할 수 있다. 자소를 음소로 변환하는 것은 주어진 단어에 대한 발음기호를 찾는 것이고 반대로 음소를 자소로 변환하는 것은 주어진 발음기호에 대한 단어를 찾는 것이다. 전자의 경우는 음성합성(speech synthesis)에 응용되고, 후자는 음성인식(speech recognition)에 응용된다. 이들 문제는 단어가 유한하다고 가정하면 아주 쉽게 해결될 수 있다. 즉, 주어진 단어나 발음기호를 이용해서 데이터베이스를 검색함으로써 음성합성과 음성인식을 수행할 수 있다. 그러나 시대의 변화에 따라 신조어가 계속 생겨나고, 이러한 새로운 단어에 대해서는 사전이나 데이터베이스 검색 없이 발음기호로 변환할 수 있어야 하며, 동시에 임의로 생성된 발음에 대해서 가장 적절한 단어를 찾을 수 있어야 한다. 이렇게 하기 위해서는 입력 단어에 대해 자소 단위를 음소로 변환하고 이를 결합하여 주어진 단어에 대한 발음기호를 생성한다. 이 경우 하나의 자소에 대해서 대응되는 음소가 하나 이상인 경우가 빈번하게 발생된다. 경우에 따라서는 두 자소가 하나의 음소를 생성하기도 하고, 하나의 자소가 두 개의 음소로 대응되기도 한다. 음소를 자소로 변환할 경우도 마찬가지이다. 이와 같은 문제를 해결하기 위해서는 자소-음소 간 정렬된 코퍼스를 이용하여 두 입력열 간의 최소 편집거리를 학습하고 이를 이용하여 자소열을 음소열로, 또는 음소열을 자소열로 쉽게 바꿀 수 있다.

본 논문에서는 영어 단어/발음 정렬을 위해 CMU 발음 사전[14]으로부터 127,069개의 단어/발음 쌍을 추출하였다. 이들 중에서 '%'와 '#' 등이 포함된 단어는 제거하였고, 발음기호에서 악센트를 표기하기 위한 숫자를 제거하여 45,000개의 쌍을 학습 데이터로 선정하였다. 이 학습 데이터와 3장에서 기술한 자율 선형 정렬 알고리즘을 이용해서 자동적으로 영어 단어/발음 정렬 말뭉치를 구축하였으며 그 결과의 일부를 <표 6>에 실었다. <표 6>(ㄱ)에서 입력열은 단어이고 출력열은 발음이며, 이와는 반대로 <표 6>(ㄴ)의 입력열은 발음이고 출력열은 단어이다.

4.1절에서와 마찬가지로 <표 6>에서도 몇 가지 음운 현상을 찾을 수 있다. <표 6>(ㄱ)에서 입력열 "exotic"와 "taxi"의 'x'는 각각 "/G/Z/"와 "/K/S/"로 정렬되는데 비록 발음이 다르지만 항상 일정한 유형으로 잘 정렬됨을 알 수 있

었다. 이와 같은 현상이 발음에서 단어로 정렬될 경우에도 "/Z/"나 "/S/"가 공백문자 'Ø'로 일관성 있게 정렬됨을 관찰할 수 있었다. 또 영어 문자 'u'는 일반적인 "/UW/"로 정렬되는데 단어의 첫머리에서는 반모음 "/Y/"가 첨가되어 "/Y/UW/"로 정렬됨이 자주 관찰되었다. 이 현상은 <표 6>(ㄱ)의 단어 "usage"에서 찾아볼 수 있으며, 그 반대의 정렬은 <표 6>(ㄴ)에서 찾아볼 수 있다. <표 6>(ㄱ)에서 영어 단어 "island"에서 's'는 묵음으로 공백문자 'Ø'로 정렬됨을 관찰할 수 있었다. 그 역정렬은 <표 6>(ㄴ)에서 "/AY/"가 "i:s"로 정렬됨을 볼 수 있다. 이와 같은 정렬은 일반적으로 "/AY/"에 대응하는 정렬은 아니다. 그러나 "island"와 같은 특수한 문맥에서는 발생될 수 있다.

〈표 6〉 자소열과 음소열 간의 선형 정렬 말뭉치

(ㄱ) 영어 단어의 발음 생성을 위한 선형 정렬(음성합성)

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	Ø	
island	i	s	l	a	n	d
AY L AH N D	AY	Ø	L	AH	N	D
taxi	t	a	x	i		
T AE K S IY	T	AE	K:S	IY		
racism	r	a	c	i	s	m
R EY S IH Z AH M	R	EY	S	IH	Z:AH	M

(ㄴ) 영어 발음의 단어 생성을 위한 선형 정렬(음성인식)

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

4.3 형태복원을 위한 오토마타 생성

형태소 분석을 위한 유한 상태 변환기(finite-state transducer)가 널리 사용되고 있으며[15, 16], 이는 사전표현(예 : "try+s")과 표층표현(예 : "tries")에 대한 단어열의 대응관계를 유한 상태 변환기로 표현한 것이다. 이 대응 관계가 바로 사전표현 단어열과 표층표현 단어열에 대한 정렬이며, 이 정렬은 형태소 분석은 물론 형태소 분리 및 복원을 위해서 사용될 수 있다. 형태소 분석의 경우는 정렬 말뭉치 이외에 각 단어의 품사정보가 저장되어 있는 형태소 분석 사전이 필요하다.

본 논문에서는 사전을 이용하지 않기 때문에 형태소 분리 및 복원을 위한 정렬 말뭉치를 구축하고자 한다. 선형 정렬 알고리즘을 학습하기 위해 KAIST 품사 부착 말뭉치[17]를 사용하였다. 이 말뭉치는 한국어 어절과 그 어절에 형태소 분석 결과가 부착되어 있으며, 전체 어절 수는 172,703개이다. 학습은 자소 단위로 이루어지므로 어절과 형태소는 모두 자소 단위로 풀어쓰기 되었다. 이 과정에서 정렬의 편의를 위해서 음가가 없는 초성 'ㅇ'은 제거하였고[15], 초성과 종성을 서로 구별하기 위해 종성에 특수문자 '_'를 붙였으며 형태소 간의 분리자로서 특수문자 '+'를 사용하였다. 이와 같은 과정을 통해서 구축된 학습 데이터를 자율 선형 정렬 알고리즘으로 학습하였으며, <표 7>에 그 결과의 일부를 보이고 있다.

〈표 7〉 형태 복원을 위한 정렬 결과

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	ㅆ_:+	ㄷ	ㅏ	
결정적	ㄱ	ㅕ	ㄹ_	ㅈ	ㅣ	ㅇ_	ㅈ	ㅓ	ㄱ_
결정+적	ㄱ	ㅕ	ㄹ_	ㅈ	ㅣ	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_	

<표 7>의 입력열은 한국어 어절이고 출력열은 어절의 형태소 분리 및 복원 결과이다. <표 7>에서 어절 "아름다운"에서 'ㅜ'는 "ㅂ_:+"로 정렬되었다. 이는 'ㅂ'불규칙 현상을 보이고 있는데, 즉 두 형태소가 결합하는 위치에서 받침의 'ㅂ'은 "아/어"나 'ㄴ'으로 시작하는 형태소를 만나면 'ㅜ'로 변하는 현상이 정렬에서도 그대로 보이고 있다. 또한 어절 "남겼다"는 "이+어"가 "겨"로 축약됨을 보이고 있다. 이 정렬 결과로부터 형태소 분리 및 복원에 대한 유한 상태 변환기의 정보를 추출할 수 있으며, 이것은 정보검색의 색인기를 구축하는데 매우 중요한 자료가 될 수 있다.

5. 구축된 정렬 말뭉치의 적용

본 절에서는 4장에서 구축된 세 종류의 말뭉치를 이용해서 한영 음차 복원, 영어 발음 생성, 한국어 형태소 분리에 적용하여 그 유용성을 살펴보고자 한다. 본 논문은 시스템이나 모델의 정확성을 개선하고자 하는 목적이 아니기 때문에, HMM[18]을 사용해서 각 응용 시스템을 구축하였으며, 사전이나 여러 가지 경험규칙을 사용해서 성능을 개선

하기 위한 노력을 전혀 하지 않았다.

한영 음차 복원 시스템의 입력은 음차 표기된 한국어 단어이고 출력은 이에 대응하는 영어 단어이다. 즉, 영어 단어에 대한 정보가 전혀 주어지지 않는 상황에서 입력된 한국어 단어로부터 영어 단어를 찾아내는 시스템이다. 영어 발음 생성 시스템은 영어 단어가 입력으로 주어질 때, 그 단어의 발음을 생성해 주는 시스템을 말한다. 또한 한국어 형태소 분리 및 복원 시스템은 한국어 어절을 입력으로 주었을 때, 주어진 어절을 형태소 단위로 분리하고 필요할 시에는 형태소를 복원하는 시스템을 말한다. 이들 세 응용 시스템 모두 어떠한 사전 정보를 전혀 사용하지 않았다.

5.1 실험 환경

모든 응용 시스템에서 기본 분류기는 N개의 결과를 출력할 수 있는 HMM 분류기[19]를 사용한다. <표 8>은 실험에 사용될 말뭉치의 규모이다. 이 말뭉치들을 이용해서 각 응용 시스템을 학습시키고 그 성능을 평가해 보고자 한다.

〈표 8〉 학습 및 시험 말뭉치의 규모

응 용 분 야	말뭉치 규모		
	전 체	학 습	시 험
① 한영 음차 복원	14,590	13,720	870
② 영어 발음 생성	45,000	44,000	1,000
③ 한국어 형태소 분리 및 복원	172,703	131,581	41,122

5.2 성능 평가

성능 측도는 정확률을 사용하며, 정확률은 주어진 입력에 대해서 가능한 10개의 후보를 생성하고 그 중에 정확한 출력이 포함되어 있을 비율을 의미한다. <표 9>는 각 응용 시스템의 성능을 보이고 있다.

〈표 9〉 각 응용 시스템의 정확률

응 용 분 야	정 확 률(%)
① 한영 음차 복원	51
② 영어 발음 생성	61
③ 한국어 형태소 분리 및 복원	99

한국어 형태소 분리 및 복원 시스템의 정확률을 제외하고는 정확률이 그다지 높지 않다. 앞에서도 언급했듯이 정확률을 높이는 연구를 앞으로 좀더 수행할 계획이며, 사전이나 각 응용 분야에 적합한 경험규칙 혹은 다양한 문맥을 사용할 수 있는 분류기를 사용함으로써 이들 성능은 충분히 개선될 수 있다고 확신한다.

6. 결 론

본 논문에서는 자율 선형 정렬 알고리즘을 이용하여 자동으로 정렬 말뭉치를 구축하는 방법을 기술하였다. 본 논문에서는 기존의 자율 선형 정렬 알고리즘을 수정하여 입력열에 나타날 수 있는 공백 문자를 제거하였으며, 이 알고리즘을 이용하여 정렬 말뭉치를 구축할 경우, 입력 문자열에 따라 다른 말뭉치가 구축되어야 한다. 즉 구축된 말뭉치는 양방향성을 잃게 된다. 한영 음차변환의 예를 들면, 음차 표기된 한국어를 영어 단어로 변환할 경우와 영어 단어를 음차 표기된 한국어로 변환할 경우의 말뭉치가 다르게 된다. 본 논문에서는 제안된 알고리즘을 실제 응용 분야(한영 음차 표기 및 복원, 영어 자소/음소 변환, 한국어 형태 분리 및 복원)에 적용해 보았다. 그 결과 대부분의 선형 정렬이 요구되는 응용 분야에 잘 적용됨을 알 수 있었다. 앞으로의 연구는 선형 정렬된 코퍼스를 이용하여 입력열을 출력열로 변환하는 변환 시스템의 성능 향상에 집중해야 할 것이다.

참 고 문 헌

[1] 국립국어연구원, 21세기 세종계획 성과발표 및 토론회 자료집, 2004.

[2] Manning, C. D. and Schutze, H. *Foundations of Statistical Natural Language Processing*, The MIT Press, 1999.

[3] Marcus, M. P., Santorini, B. and Marcinkiewicz, M. A. "Building a large annotated corpus of English : The Penn Treebank," *Computational Linguistics*, 19(2), pp.313-330, 1993.

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

[5] Ristad, E., Yianilos, P., "Learning String Edit Distance," *IEEE Tr. on Pattern Analysis and Machine Intelligence*, 20(2), pp.522-532, 1998.

[6] Qualian J. R., *C4.5 : Programs for Machine Learning*, San Mateo, CA : Morgan Kaufmann Publishers, 1993.

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

[8] Burges, C. J. C., "A tutorial on support vector machines for pattern recognition," *Knowledge Discovery and Data Mining*, 2(2), 1998.

[9] Krogh, A., Brown, M., Mian, I. S., Sjolander, K. and Haussler, D. "Hidden Markov models in computational biology : Applications to protein modeling," *Journal of Molecular Biology*, 235, pp.1501-1531, 1994.

[10] Allison, L., Powell, D. and Dix, T. I. "Comptession and Approximate Matching," *The Computer Journal*, 42(1), pp. 1-10, 1999.

[11] Breimer, E. A. A Learning Approach for Designing Dy-

namic Programming Algorithms, http://www.cs.rpi.edu/~breime/slide/, 2000.

[12] 이재성, 다국어 정보검색을 위한 영-한 음차 표기 및 복원 모델, 한국과학기술원 박사학위논문, 1999.

[13] 국립국어연구원, 표준대국어사전, (주)두산동아, 2000.

[14] CMU, CMU Pronouncing Dictionary, http://www.speech.cs.cmu.edu/speech/.

[15] 이성진, Two-Level 한국어 형태소 해석, 한국과학기술원, 전산학과, 석사학위 논문, 1992.

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

[17] 김재훈, 김길창, 한국어에서의 품사 부착 말뭉치의 작성 요령 : KAIST 말뭉치, 한국과학기술원, 전산학과, CS-TR-95-99, 1995.

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

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

이 공 주

e-mail : kjoolee@kic.ac.kr

1992년 서강대학교 전자계산학과(학사)

1994년 한국과학기술원 전산학과(공학석사)

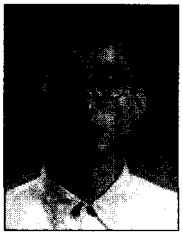
1998년 한국과학기술원 전산학과(공학박사)

1998년~2003년 (주) 한국마이크로소프트 연구원

2003년 이화여자대학교 컴퓨터학과 전임 강사

2004년~현재 경인여자대학 컴퓨터정보기술학부 전임 강사

관심분야 : 자연언어처리, 자연어인터페이스, 기계번역, 정보검색



김 재 훈

e-mail : jhoon@mail.hhu.ac.kr

1986년 계명대학교 전자계산학과(학사)

1988년 한국과학기술원 전산학과(공학석사)

1996년 한국과학기술원 전산학과(공학박사)

1988년~1997년 한국전자통신연구원, 선임 연구원

1997년~1999년 한국해양대학교, 컴퓨터공학과, 전임강사

2000년~2002년 한국과학기술원 첨단정보기술연구센터, 연구원

2001년~2002년 USC, Information Sciences Institute, 방문연구원

1999년~현재 한국해양대학교 컴퓨터공학과 부교수

관심분야 : 자연언어처리, 한국어 정보처리, 정보검색, 정보추출