음절 N-Gram과 어절 통계 정보를 이용한 한국어 띄어쓰기 시스템*

부산대학교 컴퓨터공학과
최성자 · 강미영 · 허희근 · 권혁철

Korean Word Spacing System Using Syllable N-Gram and Word Statistic Information

Sung-Ja Choi, Mi-Young Kang, Hee-Keun Heo, Hyuk-Chul Kwon
Korean Language Processing Lab, School of Electrical & Computer Engineering, Pusan National University, Busan, Korea

요 약

본 논문은 정제된 대용량 말뭉치로부터 얻은 음절 n-gram과 어절 통계를 이용한 한국어 자동 띄어쓰기 시스템을 제안한다. 한 문장 내에서 최적의 띄어쓰기 위치는 Viterbi 알고리즘에 의해 결정된다. 통계 기반 연구에 고유한 문제인 데이터 부족 문제, 학습 말뭉치 의존 문제를 개선하기 위하여 말뭉치를 확장하고 실험을 통해 얻은 매개변수를 사용하고 최장 일치 Viable Prefix를 찾아 어절 목록에 추가한다. 본 연구에 사용된 학습 말뭉치는 33,641,511어절로 구성되어 있으며 구어와 문어를 두루 포함한다.

서 론(1장)

한국어 문서 자동 처리에서 띄어쓰기는 중요한 변수로 작용한다. 예를 들어 밖에 "내 아들은 집(밖에)없다"라는 문장이 입력되었을 때 '밖에'라는 형태가 앞서는 어절에 붙어있느냐 띄어 씌었느냐에 따라 그것이 속한 문장의 뜻이 변한다. '밖에'가 앞서는 어절에 붙어있으면 '오직'이라는 의미를 나타내는 조사로 분석이 되고, 띄어쓰기 됐으면 장소를 나타내는 보어로도 해석된다.

자동 띄어쓰기는 잘못 띄어쓴 어절을 올바르게 띄어쓰는 작업으로 맞춤법 검사, 정보 검색, 문자인식 후 줄 경계를 복원, 문자-음성 변환 시에 올바른 띄어쓰기로 복원해 주는 데 활용된다.

본 논문에서는 한국어의 통계 정보를 이용하여 자동으로 띄어쓰는 방법을 제안한다. 띄어쓸 확률은 정제된 대량의 말뭉치에서 추출한 어절의 빈도와 음절 n-gram의 빈도로부터 최우추정(Maximum likelihood estimation)에 의해 계산한다.

통계 정보를 이용한 기법으로 접근함으로써 나타나는 자료 부족 문제(Data sparseness)를 해결하기 위해 viable-prefix에 기반한 '최장일치법(Longest match strategy)'을 이용하여 후보 어절을 추가하였다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 한국어 자동 띄어쓰기와 관련된 이전의 연구들을 간략하게 소개하고, 3장에서는 자동 띄어쓰기에 사용될 어절의 통계 정보와 음절 n-gram의 통계 정보에 관한 설명을 하도록 한다. 4장에서는 어절과 음절 n-gram의 빈도를 이용한 자동 띄어쓰기 알고리즘을 소개한다. 5장에서는 성능 실험 결과를 보이고, 6장에서 결론을 맺는다.

관련 연구(2장)

한국어 문서의 띄어쓰기와 관련한 연구는 세 부류로 구분할 수 있다. 통계 기반 방법과 규칙 기반 방법, 그리고 두 기법을 통합한 방법에 관한 연구가 있다. 통계 기반 방법은 띄어쓰기 시스템을 단순하게 구축할 수 있고, 계산적으로도 부담이 적다. 그러나 통계 정보를 추출한 말뭉치의 성격에

*본 연구는 부산대학교 연구(보조)비(4년과제)에 의한 연구임.
†E-mail : heya5@pusan.ac.kr
E-mail : kmyoung@pusan.ac.kr
E-mail : riniya@pusan.ac.kr
E-mail : hckwon@pusan.ac.kr

따라 유사 분야의 문장들에 대해서는 우수한 성능을 보이지만 다른 분야에 대해서는 정확도가 떨어질 수 있다. 규칙 기반 방법은 중의성을 해결하는데 성능이 우수하지만 구축하고 유지 관리하는데 부담이 크다.

본 논문의 처음 접근법, 통계 기반 방법으로는 강승식(2001)의 임의의 두 음절 χ_i와 χ_{i+1}사이에 공백이 삽입될 확률을 이용해 자동 띄어쓰기를 하는 방법이 있다. 공백이 삽입될 확률 계산은 다음과 같다.

$$P(x_i, x_{i+1}) = 0.25\times P_R(x_{i-1}, x_i) + 0.5\times P_M(x_i, x_{i+1}) + 0.25\times P_L(x_{i+1}, x_{i+2})$$

이 값이 임계치(threshold) 0.375를 넘으면 두 음절 사이에 공백을 삽입한다. 위 식에서 P_R은 음절 쌍의 오른쪽에 공백이 위치할 확률이고, P_M은 음절 쌍 사이에 공백이 위치할 확률이다. P_L은 음절 쌍의 왼쪽에 공백이 위치할 확률이다. 이 방법의 공백 재현율은 97.7%이다.

이도길(2003)의 자동 띄어쓰기 방법은 기존의 통계 기반 자동 띄어쓰기 방법에서 이전의 띄어쓰기 상태를 고려하지 않기 때문에 발생하는 문제점을 극복하는 방안으로 자동 띄어쓰기를 품사 부착과 같은 분류 문제로 간주하고 은닉 마르코프 모델을 확장한 띄어쓰기 모델을 제안한다. 이 모델에서 띄어쓰기 태그열 T와 음절열 S의 결합 확률 $P(T, S)$를 최대로 하는 띄어쓰기 태그열 T를 찾는다. 태그열 T를 찾는 두 방법 태그 우선 모델과 음절 우선 모델 중 음절 우선 모델이 더 좋은 결과를 나타낸다(K=1, J=2, L=2, I=2일 때 가장 성능이 좋다.).

$$\begin{aligned} P(T,S) &= P(s_1)P(t_1|s_1)\prod_{i=2}^{n} P(s_i|s_{1,i-1}, t_{1,i-1}) \\ &\quad P(t_i|s_{1,i}, t_{1,i-1}) \\ &\approx \prod_{i=1}^{n} P(s_i|s_{i-K,i-1}, t_{i-J,i-1}) \\ &\quad P(t_i|s_{i-L,i}, t_{i-I,i-1}) \end{aligned}$$

이 방법의 음절 단위 정확도는 98.33%이고 어절 단위 정확도는 93.06%이다.

본 논문은 비교적 간단하게 구할 수 있는 어절 빈도와 음절간 띄어쓸 확률을 주변수로 사용하는 모델에 기반한 효율적인 시스템을 구현한다.

어절과 음절 N-Gram 통계(3장)

음절 n-gram과 어절의 빈도 추출을 위해 사용한 말뭉치는 총 33,641,511어절로 이루어져 있다. 이 말뭉치는 A신문 2000년, 2001년 기사와 B신문 2002년 기사, 총 3년분의 신문 기사와 TV 뉴스 방송 원고 3년분을 정제한 것이다. 문어체, 구어체 등 다양한 표현방식의 한글 어휘를 포함한 말뭉치를 이용함으로써 통계 기반 방식이 학습 말뭉치의 영향을 크게 받아 생기는 문제점을 줄이도록 하였다(Table 1).

자동 띄어쓰기에 이용하기 위한 정보로 우선 어절 단위 데이터를 추출하였다. 추출된 어절들의 총 개수는 33,641,511개이며, 어절들의 총 가짓수는 1,949,866개이다. 총 추출된 어절에는 아라비안 숫자, 로마자, 다양한 특수문자 등이 포함되어 있다. 숫자는 소수점을 포함한 숫자, 한 자리(0~9), 두 자리(10~99), 세 자리(100~999), 네 자리(1000~9999), 그 이상의 숫자 단위로 그룹화하여 빈도를 조사하였다.

조사한 빈도를 이용하여 본 논문에서 자동 띄어쓰기에 사용되는 임의의 음절 연속이 독립된 어절로 존재할 확률 $P(\omega_i)$는 전체 어절 수에 대한 특정 어절 ω_i의 빈도로 간단하게 구한다($freq(\omega)$는 전체 말뭉치 어절 수이고, $freq(\omega_i)$는 특정 어절 ω_i의 빈도이다.).

$$P(w_i) = \frac{freq(w_i)}{freq(w)} \qquad (1)$$

다음으로, 말뭉치에서 자동 띄어쓰기를 위해 음절 n-gram의 통계 데이터를 추출하였다. 추출한 통계 데이터는 한 음절(unigram)의 좌우에 공백이 나타나는 빈도와 두 음절(bigram) 사이 공백이 나타나는 빈도이다.

Table 2는 한 음절의 오른쪽 혹은 왼쪽에 무조건 공백이 나타나는 음절과 무조건 음절이 나타나는 음절의 개수를 보여준다(χ는 말뭉치에 나타난 각각의 음절이다.).

한 음절의 좌우에 공백이 오는 빈도를 이용해 자동 띄어쓰기에 사용되는 중요한 값은 음절 χ의 왼쪽에 공백이 올

Table 1. 말뭉치의 어절 통계

	총 어절의 가짓수	총 어절의 개수
(A) A신문의 2년분 신문기사	1,563,864	18,959,461
(B) B신문의 1년분 신문기사	650,555	9,432,167
(C) TV 뉴스 방송 원고	409,933	5,249,883
총 합	1,949,866	33,641,511

Table 2. 공백 정보와 관련한 한 음절(unigram) 통계

	χ의 왼쪽의 공백 정보		χ의 오른쪽의 공백 정보	
	왼쪽에 항상 공백이 오는 경우	왼쪽에 항상 음절이 오는 경우	오른쪽에 항상 공백이 오는 경우	오른쪽에 항상 음절이 오는 경우
χ의 가짓수	551	1,396	539	1,751
χ의 총 개수	29,217	633,640	51,037	201,556

Table 3. 공백 정보와 관련한 음절 쌍(bigram) 통계

		음절 쌍 사이에 항상 공백이 나타나는 경우 : $P_{innerS}(\chi y)=1$	음절 쌍 사이에 항상 공백이 나타나지 않는 경우 : $P_{innerS}(\chi y)=0$	음절 쌍의 왼쪽에 항상 공백이 나타나는 경우 : $P_{leftS}(\chi y)=1$
한글로만 된 음절 쌍	가짓수	211,839	56,342	19,392
	총 개수	5,921,230	8,002,297	586,821
아라비안 숫자, 로마자, 알파벳, 특수문자 등이 포함된 음절 쌍	가짓수	124,437	59,391	28,160
	총 개수	847,340	424,791	167,881

확률 $P_{LS}(\chi)$이다('_'는 공백을 뜻하고, $freq(\chi)$는 음절 χ가 말뭉치에서 나타난 총 횟수이고, $freq(_\chi)$는 음절 χ의 왼쪽에 공백이 나타난 총 횟수이다.).

$$P_{LS}(x)=\frac{freq(\_x)}{freq(x)} \qquad (2)$$

만약 $P_{LS}(\chi)$가 1이면 음절 χ는 항상 왼쪽에 공백이 오는 것이다.

본 논문에서는 오른쪽 공백 정보는 사용하지 않는다. 교착어인 한국어는 오른쪽으로 계속 기능어들을 붙여 확장할 수 있는데 오른쪽 공백 정보를 이용하면 이러한 어미 확장에 대비할 수 없게 된다.

다음으로, 말뭉치에서 음절 쌍 사이에 공백이 오는 경우의 빈도와 그렇지 않은 경우의 빈도를 추출하고, 이 정보를 이용하여 음절 쌍 사이에 공백이 올 확률 $P_{innerS}(\chi y)$과 음절 쌍의 왼쪽에 공백이 올 확률 $P_{leftS}(\chi y)$을 다음과 같이 구하였다("_"는 공백을 뜻하고, χy는 음절 χ와 음절 y로 이루어진 음절 쌍을 나타낸다. $freq(\chi y)$는 χ와 y가 이웃하여 나타난 모든 경우의 빈도를 뜻하고, $freq(\chi y)$는 χ와 y사이에 공백이 나타난 빈도를 뜻한다. $freq(_\chi y)$는 χ와 y가 이웃하여 나올 때 χ의 왼쪽에 공백이 나타난 빈도를 나타낸다.).

$$P_{innerS}(xy)=\frac{freq(x\_y)}{freq(xy)} \qquad (3)$$

$$P_{leftS}(xy)=\frac{freq(\_xy)}{freq(xy)} \qquad (4)$$

이렇게 구한 $P_{innerS}(\chi y)$의 값이 1일 경우 χ와 y사이에는 항상 공백이 있는 것이다. 또한 $P_{leftS}(\chi y)$가 1일 경우 χ와 y가 이웃하였을 때 의 왼쪽에서는 항상 공백이 있는 것이다. 이러한 음절 쌍의 개수를 Table 3, 4에서 보인다.

앞서 말한 한 음절의 오른쪽에 공백이 오는 정보를 사용하는 것이 위험함을 설명했던 바와 같이 음절 쌍의 오른쪽 공백 정보를 자동 띄어쓰기에 이용하는 것도 위험하므로 Table 3에서 왼쪽에 공백이 오는 경우에 관한 음절 쌍 개수만 보였다.

Table 4. 총 음절 쌍(bigram) 통계

	한글로만 된 음절 쌍	아라비안 숫자, 로마자, 알파벳, 특수문자 등이 포함된 음절 쌍
가짓수	391,732	198,512
총 개수	90,235,529	7,032,467

통계적 자동 띄어쓰기(4장)

다음과 같은 임의의 문장 W가 있다고 할 때, 최우추정에 의해 최대 값을 가지는 W를 찾는 것이 최적의 띄어쓰기를 하는 방법이다.

$$W=\{\, \omega(i_0, i_1),\ \omega(i_1, i_2),\ \cdots,\ \omega(i_{n-1}, i_n) \,\} \qquad (5)$$

$\omega(i_{n-1}, i_n)$는 띄어쓰는 단위인 어절로 공백 위치 i_{n-1}과 i_n사이에 위치한 음절들의 연속을 뜻한다. i_n은 n번째 띄어쓸 위치를 나타내고, i_0는 문장 W의 초기 위치를 나타낸다.

본 논문에서 제안하는 자동 띄어쓰기 방법은 세 단계로 이루어진다. 첫 번째 단계에서 음절 n-gram 통계를 이용하여 자동 띄어쓰기 대상인 문장에서 항상 공백이 위치할 곳을 찾아 띄어쓰도록 한다. 항상 공백이 위치하는 곳은 한 음절의 왼쪽에 공백이 위치할 확률 $P_{LS}(\chi)$과 음절 쌍의 왼쪽에 공백이 위치할 확률 $P_{leftS}(\chi y)$, 그리고 음절 쌍 사이에 공백이 위치할 확률 $P_{innerS}(\chi y)$을 통해 찾아낸다. 이렇게 항상 띄어쓸 위치를 미리 찾아 띄어씀으로써 분석해야할 음절의 연속을 가능한 짧게 만들어줄 수 있다. 이는 다음 단계에서 어절 빈도 통계를 이용하여 처리해야할 때 시스템의 처리 속도를 향상시킨다.

두 번째 단계에서는 어절 확률과 음절 bigram을 사용하여 적절한 띄어쓰기 위치를 찾아낸다. i_{k-1}지점과 i_k지점 사이에 오는 어절의 확률을 $P(\omega(i_{k-1}, i_k))$로 나타낸다. 음절 쌍 사이에 공백이 위치할 확률은 $P_{innerS}(\sigma(i_{k-1})\ \sigma(i_{k+1}))$로 $\sigma(i_{k-1})\ \sigma(i_{k+1})$는 어절 $\omega(i_{k-1}, i_k)$와 $\omega(i_k, i_{k+1})$사이에 있는 공백 좌우에 위치한 음절 쌍을 뜻한다. 어절 확률과 음절 쌍

사이에 공백이 위치할 확률을 이용하여 Viterbi algorithm으로 가장 적절한 띄어쓰기 위치를 찾아낸다. Viterbi algorithm으로 가능한 모든 문장의 확률 값 W중 가장 큰 값을 가지는 경우를 채택하도록 한다. W의 값이 같은 경우에는 무작위로 한 문장을 채택한다. 어절 확률과 음절 쌍 사이에 공백이 위치할 확률을 이용하여 띄어쓰기 위치를 정하는 알고리즘은 다음과 같다.

$$\underset{W}{argmax}\left[\prod_{k=1}^{n} P(w_{i_{k-1},i_k})\frac{P_{innerS}(\sigma_{(i_k-1)}\sigma_{(i_k+1)})}{1-P_{innerS}(\sigma_{(i_k-1)}\sigma_{(i_k+1)})}\right] \quad (9)$$

$$C=\frac{P_{innerS}(\sigma_{(i_k-1)}\sigma_{(i_k+1)})}{1-P_{innerS}(\sigma_{(i_k-1)}\sigma_{(i_k+1)})} \quad (10)$$

C는 공백 좌우의 음절 쌍 $\sigma(i_{k-1})\ \sigma(i_{k+1})$를 붙여쓸 확률에 대한 띄어쓸 확률이다. 어절의 곱을 통한 문장의 확률 계산은 입력 문장을 많이 띄어쓰도록 할수록 작은 값을 가지게 한다. 이에, 띄어쓴 부분이 띄어쓸 확률이 큰 위치일 경우 보상 값을 가지도록 한다. C를 이용해 어절의 확률에 기반하여 구한 문장의 확률에 대해 띄어쓸 확률이 큰 위치에서 띄어쓰는 문장은 부가 값을 더 가지도록 값을 조정하게 된다.

이후, 통계 정보를 이용한 기법으로 접근함으로써 나타나는 자료 부족 문제를 해결하기 위해 viable-prefix에 기반한 '최장일치법(Longest match strategy)'을 이용하여 후보 어절을 추가하였다. 말뭉치에서 추출한 어절과 함께 최장일치법으로 찾은 어절을 후보로 두고 띄어쓸 위치를 찾도록 하였다. 이때, 말뭉치에서 추출한 어절은 각각의 $P(\omega(i_{k-1},i_k))$확률 값을 가지고, 최장일치법으로 찾은 어절에 대해서는 학습을 통하여 얻어진 특정 확률 값을 부여하였다.

식(9)로 문장의 확률을 계산하면 문장이 길어지면 길어질수록 0에서 1사이의 값을 가지는 어절의 확률이 계속해서 곱해지게 되므로, 문장이 길면 문장의 확률은 아주 작아지게 되기도 한다. 이는 underflow 문제를 야기한다. 이 문제를 해결하기 위해 문장의 확률 계산을 log 연산으로 변환하여 한다. 또, 보상 값 C의 영향력을 더 크게 하기 위해 log를 취한 C에 특정 값 m을 곱한다. 최종적으로 어절 빈도와 음절 bigram을 이용하여 자동 띄어쓰기를 하는 알고리즘은 다음과 같다.

$$\underset{W}{argmax}\left[\sum_{k=1}^{n}\left\{\log P(w_{i_{k-1},i_k}) + m\log\frac{P_{innerS}(\sigma_{(i_k-1)}\sigma_{(i_k+1)})}{1-P_{innerS}(\sigma_{(i_k-1)}\sigma_{(i_k+1)})}\right\}\right] \quad (11)$$

$$m=1.162$$

$P_{innerS}(\chi y)$가 1이면 C는 무한대 값을 가진다. 그러나 이 경우에 자동 띄어쓰기의 첫 단계에서 항상 띄어쓰는 위치에서 미리 문장을 잘라 두고 이 단계를 통과하게 되므로 문장의 확률 계산시에는 문제가 되지 않는다. $P_{innerS}(\chi y)$가 0일 때는 C값이 0이 되고, 이때는 식(11)상으로는 보상 값을 부여하지 않게 된다. 그러나 이 경우는 절대 띄어쓰지 않는 곳에 띄어쓴 경우이므로 보상 값을 부여하지 않는 것에 그치지 않고 C값이 0으로 나오게 띄어쓰는 경우가 있는 어절의 연속 W를 결과 값에서 제외하도록 처리한다.

말뭉치에서 추출한 어절의 통계와 음절 bigram 통계를 이용한 자동 띄어쓰기는 자료 부족 문제로 인해 추출한 어절에서 찾을 수 없는 문자열은 확률 값이 주어지지 않아 W값을 계산하지 못하고 최고 값을 가지는 띄어쓰기 위치도 구해내지 못하고 연산을 종료하게 되는 경우가 있다. 이렇듯 결과를 내지 못하는 경우에는 음절 bigram 통계만을 이용해 다음과 같은 확률이 임계치(threshold) 기준으로 띄어쓰도록 한다.

띄어쓰는 임계치 : $P_{innerS}(\chi y)\geq 0.835$ (12)

붙여쓰는 임계치 : $P_{innerS}(\chi y)\leq 0.003$

실험 및 평가(5장)

본 논문에서 제안한 자동 띄어쓰기 시스템을 구현하여 세 가지 유형의 실험 데이터에 대해 실험하였다.

실험 데이터는 학습한 말뭉치에서 추출한 내부 데이터가 있다. 또, 다양한 어휘를 균형적으로 포함한 21세기 세종계획의 말뭉치 자료에서 다양한 분야의 데이터를 추출하였다. 다양한 어휘에 대한 성능 실험을 위해 인터넷상에 오른 신문기사에 대해 독자가 남긴 100자 평을 실험 데이터로 하였다.

실험은 학습 말뭉치를 단계별로 추가하면서 통계 정보를 추출한 말뭉치의 양에 따른 자동 띄어쓰기 성능을 비교하였다. 이 실험에서 단계별로 추가하는 말뭉치 크기의 비율에 맞추어 각 말뭉치에서 내부 데이터를 추출하였다. 또, 말뭉치 관련 실험으로 정제되지 않은 말뭉치에서 어절과

Table 5. 실험 데이터들의 양

실험 데이터		데이터의 양		
		문장 수	어절 수	음절 수
100자 평에서 추출한 데이터		500	3,126	9,992
내부 데이터	A(100%)	500	6,225	19,776
	A(67%)+B(33%)	500	6,061	18,952
	A(56%)+B(28%)+C(16%)	500	5,932	18,521
세종 말뭉치에서 추출한 데이터		2,000	13,877	40,320

음절 bigram 통계 정보를 추출한 경우와 정제된 말뭉치에서 어절과 음절 bigram 통계 정보를 추출한 경우를 비교하였다(Table 5).

실험 데이터로 사용하기 위해 추출한 데이터는 문장별로 구분한 후 공백을 모두 없애고 자동 띄어쓰기 시스템으로 띄어쓰기를 하였다. 성능을 계산을 위해 공백이 삽입될 수 있는 모든 음절 위치에 대하여 띄어써야 할 위치에 띄어쓴 경우와 붙여써야 할 위치에 붙여쓴 경우의 비율(공백 재현율, P_{space}), 시스템이 반환한 전체 어절 수 중에 시스템이 제대로 반환한 어절 수의 비율(Word Precision, P_{word}), 실험 데이터의 전체 정답 어절 수에 대한 시스템이 제대로 반환한 어절 수의 비율(Word Recall, P_{word}), 문장 단위 정확도(P_W)를 계산하였다(복합 명사 고려하여 띄어써도 되고 붙여써도 되는 위치에서는 둘 다 옳은 것으로 간주하였다.).

$$P_{space} = \frac{\text{올바르게 띄어쓴 위치 수}}{\text{공백이 삽입될 수 있는 모든 음절 위치 수}} \times 100(\%)$$

$$P_{word} = \frac{\text{시스템이 올바르게 반환한 어절 수}}{\text{시스템이 반환한 전체 어절 수}} \times 100(\%)$$

$$R_{word} = \frac{\text{시스템이 올바르게 반환한 어절 수}}{\text{실험 데이터의 전체 정답 어절 수}} \times 100(\%) \quad (13)$$

$$P_W = \frac{\text{올바르게 띄어쓴 문장 수}}{\text{전체 문장 수}} \times 100(\%)$$

학습 말뭉치 크기와 정제 정도의 차이에 따라서 성능 실험 결과는 다음과 같은 차이를 보여주었다.

Table 6을 보면 학습 말뭉치의 크기가 커질수록 자동 띄어쓰기 시스템의 정확도가 높아지는 것을 볼 수 있다. 말뭉치(A)에서만 어절과 음절 통계를 추출하여 띄어쓰기를 하였을 때보다 말뭉치(A)와 (B)에서 얻어낸 통계 값을 이용하여 띄어쓰기를 하였을 때 각 실험 데이터들의 어절 단위 정확도 P_{word}의 평균값이 1.02% 더 높게 나타난다. 말뭉치 (A), (B), (C)에서 추출한 통계 값을 이용하여 띄어쓰기를 할 경우 말뭉치(A)에서만 추출했을 때보다 평균적으로 1.49% 더 좋은 결과를 내는 것을 확인하였다. 아래 Fig. 1은 말뭉치 크기별 성능 변화를 보여준다.

정제된 말뭉치에서 어절과 음절 통계를 추출하였을 때와 정제되지 않은 원시 말뭉치에서 어절과 음절의 통계를 추출하였을 때의 차이는 Table 6에서 첫 번째 테스트 말뭉치 (A)에서만 통계 정보를 추출하였을 때와 표의 가장 아래에 있는 말뭉치 (A)를 정제하지 않은 상태에서 통계 정보를 추출하였을 때를 비교하면 알 수 있다. 정제된 말뭉치에서 통계 정보를 추출한 것이 각 실험 데이터의 정확도

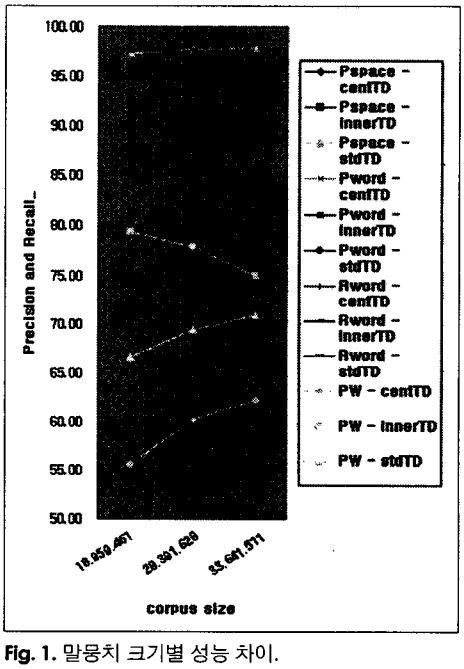

Fig. 1. 말뭉치 크기별 성능 차이.

Table 6. 학습 말뭉치의 크기와 학습 말뭉치의 정제 정도에 따른 자동 띄어쓰기 정확도

말뭉치 크기	실험 데이터	P_{space}	P_{word}	R_{word}	P_W
정제된 말뭉치(A) : 18,959,461 어절	100자 평에서 추출한 데이터	96.19	83.43	88.93	55.60
	내부 데이터	99.32	97.67	96.53	79.40
	세종 말뭉치에서 추출한 데이터	97.29	88.99	92.60	66.60
정제된 말뭉치(A)+정제된 말뭉치(B) : 28,391,628 어절	100자 평에서 추출한 데이터	96.73	85.50	90.44	60.20
	내부 데이터	99.20	97.40	96.01	77.80
	세종 말뭉치에서 추출한 데이터	97.66	90.25	93.59	69.35
정제된 말뭉치(A)+정제된 말뭉치(B)+정제된 말뭉치(C) : 33,641,511 어절	100자 평에서 추출한 데이터	97.03	86.68	91.20	62.20
	내부 데이터	99.08	97.07	95.38	75.00
	세종 말뭉치에서 추출한 데이터	97.82	90.82	94.03	70.85
정제되지 않은 말뭉치(A) : 18,996,289 어절	100자 평에서 추출한 데이터	95.45	80.14	85.51	50.00
	내부 데이터	97.66	92.61	86.12	35.00
	세종 말뭉치에서 추출한 데이터	96.70	86.51	90.08	59.65

가 평균적으로 3.61% 더 높게 나타난다. Fig. 2는 정제된 말뭉치에서 통계 정보를 추출하는 경우와 정제되지 않은 말뭉치에서 통계 정보를 추출하는 경우의 성능 차이를 보여준다.

앞서 설명하였듯이 자료 부족 문제로 인해 추출한 어절에서 찾을 수 없는 문자열의 발생으로 띄어쓰기 위치를 구해내지 못하고 연산을 종료하게 된다. 이와 같이 결과를 내지 못하는 경우는 시스템에 **완화(smoothing)기법**을 적용함으로써 보완할 수 있다. 즉, 음절 bigram 통계만을 이용해 얻은 임계치(threshold) 이상이면 띄어쓰도록 하고 결과로 반환하게 하는 것이다(12 참고).

Table 7은 완화(smoothing)기법을 적용하기 전과 후의 성능 차이를 보여준다(학습 말뭉치는 말뭉치 (A), (B), (C)를 모두 합한 33,641,511 어절로 된 말뭉치를 사용하였다.).

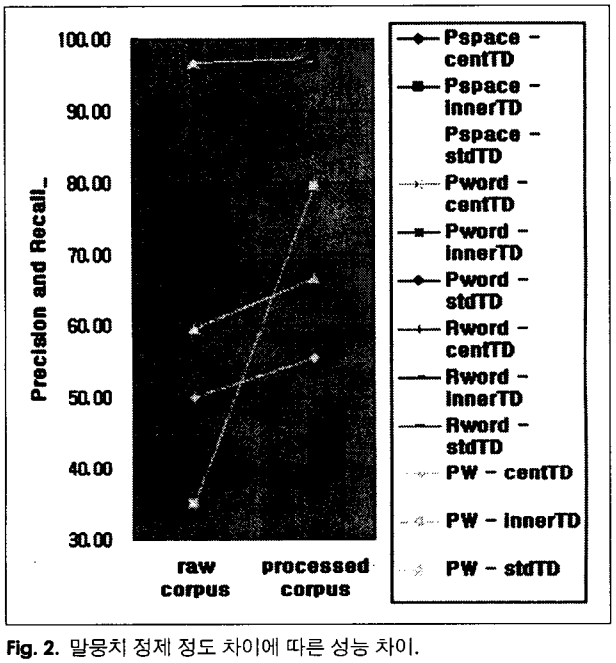

Fig. 2. 말뭉치 정제 정도 차이에 따른 성능 차이.

완화(smoothing)기법을 적용한 후 100자 평에서 추출한 데이터, 내부 데이터, 세종 말뭉치에서 추출한 데이터의 정확도가 평균적으로 0.6% 향상되었다.

다음으로, **viable-prefix에 기반한 '최장일치법(Longest match strategy)'을 이용**하여 후보 어절을 추가하였다. 말뭉치에서 추출한 어절과 함께 최장일치법으로 찾은 어절을 후보로 두고 띄어쓸 위치를 찾도록 하였다. 아래 Table 8은 viable-prefix에 기반한 '최장일치법'을 이용하여 후보 어절을 추가한 경우에 시스템의 성능을 나타내고 있다.

Table 7, 8을 보면 viable-prefix에 기반한 '최장일치법'을 이용하여 후보 어절을 추가한 후 자동 띄어쓰기 시스템의 정확도가 smoothing 단계까지 만으로 띄어쓰기를 했을 때보다 4.15% 높아졌음을 알 수 있다.

결 론(6장)

본 논문은 음절 n-gram과 어절 통계를 이용하여 효과적인 자동 띄어쓰기 시스템을 구현하였다. 말뭉치에서 추출한 통계를 이용한 자동 띄어쓰기 처리 시 발생하는 자료 부족 문제와 노이즈 문제를 극복하기 위해 어절이 다양하고, 큰 말뭉치에서 어절 빈도와 음절 n-gram 통계 정보를 추출하였다. 대량의 말뭉치에서 추출한 어절들로도 해결하지 못하는 자료 부족 문제를 해결하기 위해 '최장일치법'을 사용하였다. 또, 어절의 양을 늘려주는 접근과 다르게 임계치(threshold)를 정하여 추출한 어절 통계 정보를 기반으로 완화(smoothing)기법을 사용함으로써 띄어쓰기

Table 7. 자동 띄어쓰기 시스템의 smoothing 전, 후 정확도

	실험 데이터	P_{space}	P_{word}	R_{word}	P_W
Smoothing 전	100자 평에서 추출한 데이터	96.19	86.14	88.07	58.20
	내부 데이터	99.08	96.99	95.31	74.20
	세종 말뭉치에서 추출한 데이터	97.01	89.63	91.50	66.80
Smoothing 후	100자 평에서 추출한 데이터	97.03	86.68	91.20	62.20
	내부 데이터	99.08	97.07	95.38	75.00
	세종 말뭉치에서 추출한 데이터	97.82	90.82	94.03	70.85

Table 8. viable-prefix에 기반한 '최장일치법'을 이용하여 후보 어절을 추가하였을 때 시스템의 정확도

실험 데이터	viable-prefix에 기반한 '최장일치법'을 이용하여 후보 어절을 추가하였을 때			
	P_{space}	P_{word}	R_{word}	P_W
100자 평에서 추출한 데이터	98.29	92.97	94.18	77.20
내부 데이터	98.93	96.77	94.44	71.60
세종 말뭉치에서 추출한 데이터	99.26	97.27	97.51	89.35

문제를 효과적으로 해결할 수 있었다.

그러나 여러 실험 결과들을 통해서 통계 기반 띄어쓰기 시스템이 노이즈 문제와 자료 부족 문제를 근본적으로 해결하지 못한다는 것을 관찰할 수 있었다.

따라서, 향후 과제로 규칙 기반 시스템과 통계 기반 시스템을 적절히 합성한 효율적인 시스템 개발이 필요하다 하겠다.

REFERENCES

Manning CD, Schutze H(2001) : *Foundations of Statistical Natural Language Processing, The MIT Press, Cambridge, London*

심철민, 권혁철(1996) : "연어 정보에 기반한 한국어 철자 검사와 교정기의 구현", 정보과학회 논문지, *23*권 *8*호, 한국정보과학회, *pp776-785*

강미영, 권혁철(2003) : "효율적인 문서처리를 위한 띄어쓰기 교정 기법 개선", *2003* 봄 학술발표논문집(*B*), 한국정보과학회, *pp486-488*

강승식(2001) : "음절 *bigram*를 이용한 띄어쓰기 오류의 자동 교정", 음성과학회 논문지, 제*8*권 제*2*호, *pp83-90*

이도길, 이상주, 임희석, 임해창(2003) : "한글 문장의 자동 띄어쓰기를 위한 두 가지 통계적 모델", 정보과학회 논문지 : 소프트웨어 및 응용 제*30*권 제*4*호, pp*358-370*

Kim SN, Nam HS, H.CH. Kwon(1999) : *"Correction Methods of Spacing Words for Improving the Korean Spelling and Grammar Checkers" Proceedings of 5th Natural Language Processing Pacific Rim Symposium, pp415-419*

Kang SS, Woo CW(2001) : *"Automatic segmentation of words using syllable bigram statistics" Proceedings of the 6th Natural Language Processing of the 6th Natural Language Processing Pacific Rim Symposium, pp.729-732*

강승식(2002) : 한국어 형태소 분석과 정보 검색, 홍릉과학출판사, 서울