한국어 품사 모호성 해소를 위한 통계적 모델

이상호[O*], 김재훈[*], 조정미[*], 서정연[*]
한국과학기술원 전산학과[*]

A Stochastic Model for Lexical Disambiguation in Korean

Sangho Lee[O*], Jae-Hoon Kim[*], Cho, Jeong Mi[*], Jungyun Seo[*]
Department of Computer Science, KAIST[*]

요 약

종래의 자연언어 처리 시스템에서는 품사 모호성이 그대로 구문 분석기의 입력으로 사용되었으나, 최근에 와서 품사 모호성 해소에 관한 연구가 활발히 진행되고 있다. 본 논문에서는 품사 모호성 해소를 위한 두개의 통계적 모델인 경로 기반 태깅(path-based tagging) 모델과 상태 기반 태깅(state-based tagging) 모델을 설명한다. 그리고, 하나의 최적 품사열만을 구할 경우 단어당 94 %내외의 정확률을 가지므로 N개의 최적 품사열을 구하는 다중 출력 태거(multiple tagger)에 대해 각각 설명한다. 끝으로 한국어에 이러한 통계적 모델들을 적용한 결과와 발생되는 문제점들을 논한다[1].

1. 서론

본 논문에서는 한국어 품사 모호성 해소를 위한 통계적 모델을 설명한다[2]. 종래의 품사 모호성에 관한 연구는 가장 적당한 품사열을 찾는 문제로 인식되어 하나의 품사열만을 구하는 것이 초점이었으나, 대부분의 품사 태거는 단어당 94%내외의 정확률을 가지고 있다. 그렇다면, 만약 20개의 단어로 이루어진 문장에 대해 그 문장의 품사가 모두 옳을 확률은 $0.94^{20} = 0.29$가 된다. 이것은 구문 분석기에서 볼 때, 품사 태깅된 문장에 대한 신뢰도는 0.29가 되고 품사 태깅의 오류로 말미암아 구문 분석이 실패하는 경우가 발생된다[Macklovitch 1992]. 이러한 문제에 대한 해결책으로 하나의 최적 품사열 뿐만 아니라, 여러개의 적절한 품사열을 출력하는 다중 출력 품사 태거 모델이 개발되었다[Foster 1991, Weischedel 1993][3]. 이러한 다중 출력 태거의 특징은 PCFG(Probabilistic Context Free Grammar)를 이용하는 파서의 전단부로 이용되어 파스 트리(parse tree)의 단말 노드 (terminal node)의 확률로 다중 출력 태거가 내주는 신뢰도를 사용할 수 있는 장점을 갖고 있다.

한편, 태깅 방법에는 전체 단어열에 대한 품사열(sentence level)을 최적화시키는 경로 기반 태깅(path-based tagging)과 각 단어의 품사(word level)를 최적화시키는 상태 기반 태깅(state based tagging)으로 크게 나뉠 수 있다. 본 논문에서는 이 두 모델들을 한국어에 적용한 결과와 각각의 문제점들을 논한다[4].

본 논문의 구성은 2장에서 통계적 품사 모호성 해소 모델을 소개하고, 3장에서 각 모델을 한국어에 적용한 결과에 대해서 논하고, 4장에서 결론을 맺고자 한다.

2. 통계적 품사 모호성 해소 모델

품사 모호성 해소란 주어진 단어가 품사 모호성을 갖을 때, 하나의 최적 품사를 구하는 것을 말하고 크게 규칙에 기반한 모델[Brill 1992], 통계에 기반한 모델[Church 1988, Merialdo 1994]로 나뉜다. 본 논문은 통계에 기반한 모델을 논하고자 한다. 통계적 품사 태깅 모델은 주로 은닉 마르코프 모델(Hidden Markov Model)을 기반으로 한다. 품사 태깅을 위한 은닉 마르코프 모델에서 어떤 방법에 의해서 가장 적절한 품사열을 찾을 것인가에 따라서 경로 기반 태깅과 상태 기반 태깅으로 나뉜다.

태깅 모델을 설명하기 전에 먼저, 품사 집합과 단어 집합을 다음과 같이 정의한다.

$$T = \{t^1, t^2, \ldots, t^r\}$$
$$W = \{w^1, w^2, \ldots, w^w\}$$

[1] 본 연구는 한국 통신의 정기기초 과제 "차등 통역 전화 개발을 위한 대화체 기계번역에 관한 연구"의 부분 지원을 받은 것입니다.

[2] 품사 모호성 해소 모델을 품사 태거(간략히 태거)라고 하자.

[3] 하나의 최적 품사열을 내주는 태거를 유일 출력 태거(Single Tagger)라 하고, 하나 이상의 품사열과 신뢰도를 내주는 태거를 다중 출력 태거(Multiple Tagger)라 하자.

[4] 상태 기반 태깅(state-based tagging)과 경로 기반 태깅(path-based tagging)을 [Merialdo 1994]에서는 각각 Viterbi Tagging , ML(Maximum Likelihood) Tagging이라 한다.

$$\begin{array}{cccc} w_1 & w_2 & \cdots & w_n \\ (w_1,t_1) & (w_2,t_1) & \cdots & (w_n,t_1) \\ (w_1,t_2) & (w_2,t_2) & \cdots & (w_n,t_2) \\ \vdots & \vdots & \ddots & \vdots \\ (w_1,t_\omega) & (w_2,t_\omega) & \cdots & (w_n,t_\omega) \end{array}$$

그림 1: (단어,품사)를 노드로 표현한 격자(trellis)

여기서 τ와 ω는 각각 품사 집합의 크기와 단어 집합의 크기를 의미한다.

위와 같이 정의된 품사 집합과 단어 집합에 대해 품사 모호성이란 임의의 단어 w가 여러개의 품사 ($T' \subset T$)를 가진 경우를 의미한다. 결국 품사 모호성 해소란 주어진 단어열에 대해 각 단어마다 T'안에 있는 특정 품사를 선택하는 것을 의미한다. 이 때, 어떤 기준으로 품사를 선택하느냐에 따라 품사 모호성 해소 모델은 달라지게 된다.

2.1 경로 기반 태깅 모델

경로 기반 태깅 모델을 좀 더 수식적으로 설명하면 [수식 1]과 같다.

$$\phi(w_{1..n}) = \mathrm{argmax}_{t_{1..n}} P(t_{1..n}|w_{1..n}) \quad (1)$$

[수식 1]은 주어진 단어열 $w_{1..n}$을 가장 잘 설명하는(어울리는) 품사열 $t_{1..n}$을 선택하는 것이다. 즉, [수식 1]은 주어진 단어열 $w_{1..n}$에 대해 (w_i,t_k)를 [그림 1]에서 보듯이 격자의 노드로 표현하고, 모든 가능한 경로 중 최적 경로를 찾는 문제가 된다[5].

한편, [수식 1]을 다음과 같이 전개할 수 있다.

$$\begin{aligned} \phi(w_{1..n}) &= \mathrm{argmax}_{t_{1..n}} P(t_{1..n}|w_{1..n}) && (2) \\ &= \mathrm{argmax}_{t_{1..n}} \frac{P(w_{1..n}|t_{1..n})P(t_{1..n})}{P(w_{1..n})} && (3) \\ &= \mathrm{argmax}_{t_{1..n}} P(w_{1..n}|t_{1..n})P(t_{1..n}) && (4) \end{aligned}$$

[수식 3]에서 $P(w_{1..n})$은 상수가 되므로 [수식 4]를 만족하는 $t_{1..n}$을 구하면 된다. 한편, [수식 4]에서 사용되는 확률들을 직접 구하는 것은 자료 부족(sparse data)문제를 갖고 있기 때문에, 마르코프 가정(markov assumption)을 사용해서 $P(w_{1..n}|t_{1..n})$와 $P(t_{1..n})$을 [수식 5, 6]과 같이 각각 정의한다.

$$\begin{aligned} P(w_{1..n}|t_{1..n}) &= \Pi_{i=1}^{n} P(w_i|t_i) && (5) \\ P(t_{1..n}) &= \Pi_{i=1}^{n} P(t_i|t_{i-1}) && (6) \end{aligned}$$

[5] 경로 기반 태깅 모델(path-based tagging model)이란 이름은 여기서 나온 것이다.

step 1 Initialization for all t_1^k
$1 \le k \le$ number of w_1's ambiguities
$\delta_1(k) = P(t_1^k|INI)P(w_1|t_1^k)$
$\Psi_1(k) = 1$
step 2 Recursion
for all w_i ($2 \le i \le$ number of words)
for all t_i^k ($1 \le k \le$ number of w_i's ambiguities)
$\delta_i(k) = \max_m[\delta_{i-1}(m)P(t_i^k|t_{i-1}^m)]P(w_i|t_i^k)$
$\Psi_i(k) = \mathrm{argmax}_m[\delta_{i-1}(m)P(t_i^k|t_{i-1}^m)]$
step 3 Path Backtracking
$S_n = \mathrm{argmax}_k[\delta_n(k)]$
loop i = n - 1 to 1
$S_i = \Psi_{i+1}(S_{i+1})$

그림 2: Viterbi 알고리즘

[수식 5, 6]을 [수식 4]에 적용하면 결국 $\phi(w_{1..n})$은 [수식 7]과 같이 된다.

$$\phi(w_{1..n}) = \mathrm{argmax}_{t_{1..n}} \Pi_{i=1}^{n} P(w_i|t_i)P(t_i|t_{i-1}) \quad (7)$$

[수식 7]이 의미하는 것은, 최적 품사열이란 연속된 품사들의 품사 전이 확률과 특정 품사에서 특정 단어가 나타날 어휘 확률을 계속 곱한 값 중 가장 높은 값을 내는 품사열을 말하는 것이다. [수식 7]에서 계산되는 값 중 우리는 최적 품사열만을, 혹은 N개의 최적 품사열들을 원할 수도 있다. 일반적으로 전자의 태거를 유일 출력 태거(single tagger), 후자의 태거를 다중 출력 태거(multiple tagger)라고 한다. 특히 후자의 경우는 전체 가능한 품사열의 수가 M개라고 할 때 $N(N \ll M)$개의 최적 품사열을 파서에게 줄 수 있는 장점이 있다.

한편, [수식 7]에서 최적 품사열의 값과 그 값의 상태열을 구하는 방법으로 [그림 2]에 기술되어 있는 Viterbi 알고리즘을 이용할 수 있다. [그림 2]에서 t_i^k는 w_i의 가능한 품사들 중 k번째 품사를 의미하고 'INI'은 문장 시작 기호이다.

일반적으로 [그림 2]에 기술된 알고리즘을 이용하여 유일 출력 태거를 구현할 때, 대부분의 경우 적중률이 94%내외이다. 이것은 영어 뿐만 아니라 한국어도 마찬가지여서, 나머지 6%를 해결하기 위해서는 상위 단계의 지식을 이용해야 한다. 이렇게 지식을 계층적으로 나누고, 그것을 이용할 경우, 낮은 단계의 지식을 이용하는 모듈은 N개의 후보들을 신뢰도와 함께 상위 단계의 모듈로 넘겨주게 되며, 마찬가지로 품사 모호성 해소 모듈도 N개의 최적 품사열을 파서에게 줄 수 있다. 그러면, 경로 기반 태깅 방법을 이용하여 N개의 최적 품사열을 구해보자.

일반적으로, 격자(trellis)에서 N개의 최적 경로를 찾는 알고리즘은 음성 인식 분야에서 많이 이용되었다. 본 논문에서는 그 중 tree-trellis 알고리즘을 설명한다[Soong 1991, Schwartz 1991].

Tree-trellis 알고리즘은 Viterbi 알고리즘에서 $\delta_i(k)$를 모두 구

$$
\begin{aligned}
h(i,k) &= \delta_i(k) \\
g(i,k) &= P(w_n|t_n^x)P(t_n^y|t_{n-1}^y) \\
&\quad \cdots P(t_{i+1}^v|t_i^u) \\
f(i,k) &= g(i,k)\cdot h(i,k)
\end{aligned}
$$

표 1: tree-trellis 알고리즘에서 $f(i,k), g(i,k), h(i,k)$

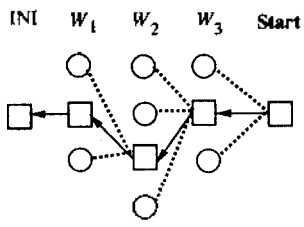

표 2: 최적 품사열을 찾은 상태

한 후, 격자의 오른쪽에서 왼쪽으로 A^* 알고리즘을 이용하여 경로를 찾는 알고리즘이다. A^* 알고리즘은 일반적으로 시작 노드로부터 목적 노드까지 가는 최적의 경로를 찾을 때 사용하는 알고리즘으로 현재까지 오는데 걸린 경로 값을 $g(i)$로, 목적 노드까지 가는데 걸리는 경로 값을 $h(i)$로 표현하여 전체 경로 값을 $f(i) = g(i) + h(i)$로 표현한다. A^* 알고리즘은 대부분의 경우 $h(i)$가 아닌 휴리스틱 $h^*(i)$를 사용하지만 Viterbi 알고리즘의 $\delta_i(k)$를 구한 상태는 그 값이 곧 $h(i)$가 된다.

결국, tree-trellis 알고리즘은 $h^*(i) = h(i)$이 되는 특징으로 경로의 정확한 값을 구할 수 있다. 여기서는 Viterbi 알고리즘의 표현법을 이용하여 $h(i,k) = \delta_i(k)$로 표현한다. $g(i,k)$는 문장 마지막에서 현재 단어의 t_i^k까지의 경로 값이고, $h(i,k)$은 t_i^k부터 t_1^*까지의 경로 값으로 전체 경로 값 $f(i,k)$은 $g(i,k)\cdot h(i,k)$가 된다. [표 1]에 $f(i,k), g(i,k), h(i,k)$가 각각 정의되어 있다. N개의 최적 품사열은 처음으로 구한 최적 품사열의 나머지 트리를 이용하여 최적 품사열을 구하는 방법으로 N개를 구한다. 예를 들면, 세개의 단어로 된 문장의 첫번째 최적 품사열을 찾은 후, 트리 상태가 [그림 2]와 같을 때 두번째 최적 품사열을 찾기 위해서는 원으로 표기된 7개의 $f(i,k)$값 중 가장 큰 노드부터 경로를 찾기 시작한다.

2.2 상태 기반 태깅 모델

상태 기반 태깅 모델은 경로 기반 태깅 모델과 달리 주어진 문장 내에서 임의의 단어에 대해 가장 적당한 품사를 찾는 것이다. 이것은 [수식 8]로 정의된다.

$$
\begin{aligned}
\phi(w_{1..n})_i &= \mathrm{argmax}_{t_i^k} P(t_i = t_i^k|w_{1..n}) && (8) \\
&= \mathrm{argmax}_{t_i^k} \sum_{t_{1..n}:t_i=t_i^k} P(w_{1..n}, t_{1..n}) && (9)
\end{aligned}
$$

step 1 Initialization for all t_1^k
$1 \le k \le$ number of w_1's ambiguities
$\alpha_1(k) = P(t_1^k|INI)P(w_1|t_1^k)$
step 2 for all w_i ($2 \le i \le$ number of words)
for all t_i^k ($1 \le k \le$ number of w_i's ambiguities)
$\alpha_i(k) = [\sum_m \alpha_{i-1}(m)P(t_i^k|t_{i-1}^m)]P(w_i|t_i^k)$

그림 3: 전방 계산(forward computation)

step 1 Initialization for all t_n^k
$1 \le k \le$ number of w_n's ambiguities
$\beta_n(k) = 1$
step 2 loop i = n - 1 to 1
$\beta_i(k) = [\sum_m P(t_{i+1}^m|t_i^k)P(w_{i+1}|t_{i+1}^m)\beta_{i+1}(m)]$

그림 4: 후방 계산(backward computation)

[수식 8]은 결국 [그림 1]에서 모든 가능한 경로에 대해 특정 노드를 통과할 확률을 의미한다. 이것은 은닉 마르코프 모델에서 모델 파라미터를 재 계산할 때 쓰이는 식과 동일하다. [그림 3]에 임의의 단어 w_i까지 모든 가능한 경로의 확률값을 더하는 전방 계산(forward computation)과 [그림 4]에 마지막 단어로부터 임의의 단어 w_i까지 모든 가능한 경로의 확률값을 더하는 후방 계산(backward computation)이 각각 보여진다.

한편, 주어진 단어열이 발생할 확률은 [그림 3]에서 $\alpha_n(k)$를 모두 더한 값이다[수식 10].

$$
P(w_{1..n}) = \sum_{t_{1..n}} P(t_{1..n}, w_{1..n}) = \sum_k \alpha_n(k) \qquad (10)
$$

그러므로, 모든 가능한 경로에 대해 특정 노드 t_i^k를 통과할 확률은 [수식 11]과 같다.

$$
P(t_i = t_i^k|w_{1..n}) = \frac{\alpha_i(k)\beta_i(k)}{P(w_{1..n})} \qquad (11)
$$

결국, 상태 기반 태깅 방법은 임의의 단어 w_i에 대해 t_i^k를 [수식 11]을 이용하여 모두 구한 후, 그 값들 중 가장 높은 값을 갖는 품사가 각 상태에서의 최적의 품사가 되고, 각 상태에서의 N개의 최적 품사를 구하기 위해서는 가장 높은 값부터 차례대로 N개를 선택하면 된다.

3. 실험

본 논문의 목적은 2장에서 설명한 경로 기반 태깅 방법과 상태 기반 태깅 방법을 구현하고 이를 한국어에 적용하는 것이다.

한편, 한국어로 적용할 때 문제점은 w_i를 한국어의 어절로 본다면 [수식 1]에서 t_i 자체가 품사열이 될 수 있다는 것이다. 즉, "나는"와 형태소 분석 결과 중 "나/동사 + 는/어미"를 선택할 때 t_i는 <동사,어미>가 된다. 만약 이러한 현상을 인정하고 [수식 7]에

필요한 $P(w_i|t_i), P(t_i|t_{i-1})$를 구하고자 한다면, 예를 들어 품사열 <동사,어미>가 주어졌을 때 '나는'이 발생할 확률을 구하고자 하는 것과 같다. 이 경우 자료 부족(sparse data)문제를 포함하게 된다. 결론적으로 한국어에 적용을 하기 위해서는 각각의 확률들을 자료 부족(sparse data)문제가 발생하지 않게 다시 나누는 과정이 필요하게 된다.

이러한 문제점 때문에 본 논문에서는 w_i를 하나의 형태소로 보았다. 하지만 기본 단위를 형태소로 할 때 역시 한국어 때문에 생기는 또다른 문제가 발생한다. 주어진 어절에 대해 한개 이상의 형태소 분석 결과가 있고, 각 형태소 분석 결과가 서로 다른 갯수의 형태소를 갖을 때, [수식 7]에서 확률을 곱하는 횟수가 다를 수 있다. 이 때 형태소열이 긴 분석 결과는 값이 작아지기 때문에 이것이 최적 품사열로 선택될 여지가 별로 없다. 그렇기 때문에 이러한 문제를 보상할 수 있는 가중치가 필요하게 되고 이것을 소위 정규화(path normalization)문제라고 한다 [이 윤재 1993].

본 논문에서 실험은 학습 말뭉치(training corpus)로 약 35,000어절의 태깅된 말뭉치에서 확률값을 추출하고, 약 20,000어절의 실험 말뭉치(test corpus)에 적용하였다. 태깅 방법에 대한 평가는 [표 3]과 같이 형태소 단위, 어절 단위, 문장 단위로 적중율을 조사하였다.

	문장 단위	어절 단위	형태소 단위
상태 기반	57 %	92.26 %	94.06 %
경로 기반	57 %	92.17 %	94.03 %

표 3: 태깅 방법의 평가

[표 3]은 두 가지 태깅 방법의 최적 품사열의 비교인데 거의 차이가 없음을 알 수 있다. 이것은 영어에 대해 실험한 [Merialdo 1994]에서 얻은 결론과 같아, 한국어도 두 가지 태깅 방법에 대해 큰 영향을 받지 않는 것을 알 수 있다.

4. 결론

본 논문에서는 경로 기반 태깅 방법과 상태 기반 태깅 방법의 유일 출력 태거와 다중 출력 태거에 대해 각각 설명하였다. 한국어로 적용하기 위해서 형태소 단위의 태깅을 하였고, 두 가지 방법의 최적 품사열을 비교한 결과 거의 차이가 없음을 발견하였다. 한편, 두 가지 태깅 방법의 다중 출력 태거를 혼합적으로 이용할 경우, PCFG(Probabilistic Context Free Grammar)를 이용하는 파서에게 적당한 품사 태거가 될 수 있다고 생각한다. 본 실험에서는 정규화(path normalization)를 하지 않았으며 만약 어절 단위의 태깅을 위해서는 자료 부족(sparse data)문제를 해결해야 한다.

참고 문헌

[Macklovitch 1992] E. Macklovitch, "Where the Tagger Falters,", *Proc. of the 4th Int'l Conf. Theoretical and Methodological Issues in Machine Translation(TMI-92)*, pp.113 - 126, 1992.

[Foster 1991] G. F. Foster, "Statistical Lexical Disambiguation," Master's Thesis, McGill Univ. School of Computer Science, Montreal, Canada, 1991.

[Weischedel 1993] R. Weischedel, R.Scewartz, J.Ralmucci, M.Meteer and L.Rawshaw, "Coping with Ambiguity and Unknown Words through Probabilistic Models," *Computational Linguistics*, vol.19, no.2, pp.359 - 382, 1993.

[Merialdo 1994] B. Merialdo, "Tagging English Text with a Probabilistic Model," *Computational Linguistics*, vol.20, no.2, pp.156 - 171, 1994.

[Church 1988] K. W. Church, "A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text," *Proceedings of Applied Natural Language Processing*, Austin, Texas, 1988.

[Brill 1992] E. Brill, "A Simple Rule-Based Part of Speech Tagger," *Proc. of the 3rd Conf. on Applied Natural Language Processing*, Trento, Italy, pp.153 - 155, April, 1992.

[Soong 1991] Frank K. Soong and Eng-Fong Huang, "A Tree-Trellis Based Fast Search for Finding the N Best Sentence Hypotheses in Continuous Speech Recognition," *IEEE International Conference on Acoustic Speech and Signal Processing*, pp.546 - 549, 1991.

[Schwartz 1991] Richard Schwartz and Steve Austin, "A Comparison of Several Approximate Algorithms For Finding Multiple (N-BEST) Sentence Hypotheses," *IEEE International Conference on Acoustic Speech and Signal Processing*, pp.701 - 704, 1991.

[이 윤재 1993] 이 윤재, 한국어 문서 태깅 시스템의 설계 및 구현, 한국과학기술원 석사학위 논문, 1993.

[임 철수 1994] 임 철수, HMM을 이용한 한국어 품사 태깅 시스템 구현, 한국과학기술원 석사학위 논문, 1994.

[김 재훈 1994] 김 재훈, 서경연, 품사 태깅: 검토 및 다루기 어려운 문제들, 한국과학기술원, 인공지능연구센타, CAIR-TR-94-53, 1994.