웹 문서를 위한 개선된 문장경계인식 방법
(Improved Sentence Boundary Detection Method for Web Documents)

이 충 희 † 장 명 길 †† 서 영 훈 †††
(Chung-Hee Lee) (Myung-Gil Jang) (Young-Hoon Seo)

요 약 본 논문은 다양한 형태의 웹 문서에 적용하기 위해서, 언어의 통계정보 및 후처리 규칙에 기반하여 개선한 문장경계 인식 기술을 제안한다. 제안한 방법은 구두점 생략 및 띄어쓰기 오류가 빈번한 웹 문서에 적용하기 위해서 문장경계로 사용될 수 있는 모든 종결어미를 대상으로 학습하여 문장경계 인식을 수행하였다. 또한 문장경계인식 성능을 최대화하기 위해서 다양한 실험을 통해 최적의 자질 및 학습데이터를 선정하였고, 학습데이터에 의존적인 통계모델의 오류를 규칙에 기반 해서 보정하였다.

성능 실험은 다양한 문서별 성능 측정을 위해서 구두점이 주로 문장경계로 사용된 문어체 위주의 평가셋1(신문기사와 블로그 문서)과 구두점 생략 및 띄어쓰기 오류가 빈번한 웹 문서 위주의 평가셋2(웹 사이트의 게시판 글)를 대상으로 성능을 측정하였다. 평가 척도로는 F-measure를 사용하였으며, 기존 연구와 동일하게 구두점만을 문장경계 대상으로 학습한 기본 모델을 만들어서 실험한 결과, 평가셋1에 대해서 96.5%의 성능을 보였지만, 평가셋2에 대해서는 56.7%로 매우 저조한 성능을 보였다. 제안하는 개선 방법은 기본 모델을 웹 문서의 특징을 반영시키도록 자질 및 엔진을 개선시켰고, 최종 모델을 평가셋2로 평가한 결과, 96.3%의 성능을 보여서 39.6%의 성능 향상이 있음을 확인하였다.

키워드 : 문장경계인식, 기계학습, Support Vector Machine

Abstract In this paper, we present an approach to sentence boundary detection for web documents that builds on statistical-based methods and uses rule-based correction. The proposed system uses the classification model learned offline using a training set of human-labeled web documents. The web documents have many word-spacing errors and frequently no punctuation mark that indicates the end of sentence boundary. As sentence boundary candidates, the proposed method considers every Ending Eomis as well as punctuation marks. We optimize engine performance by selecting the best feature, the best training data, and the best classification algorithm. For evaluation, we made two test sets; Set1 consisting of articles and blog documents and Set2 of web community documents. We use F-measure to compare results on a large variety of tasks, Detecting only periods as sentence boundary, our basis engine showed 96.5% in Set1 and 56.7% in Set2. We improved our basis engine by adapting features and the boundary search algorithm. For the final evaluation, we compared our adaptation engine with our basis engine in Set2. As a result, the adaptation engine obtained improvements over the basis engine by 39.6%. We proved the effectiveness of the proposed method in sentence boundary detection.

Key words : Sentence boundary detection, Machine Learning, Support Vector Machine

· 이 논문은 2009 한글 및 한국어 정보처리 학술대회에서 '하이브리드 방법을 이용한 개선된 문장경계인식'의 제목으로 발표된 논문을 확장한 것임

† 정 회 원 : 한국전자통신연구원 지식마이닝팀 선임연구원
forever@etri.re.kr
†† 비 회 원 : 한국전자통신연구원 지식마이닝팀 책임연구원
mgjang@etri.re.kr
††† 종신회원 : 충북대학교 컴퓨터공학과 교수
yhseo@chungbuk.ac.kr

논문접수 : 2009년 11월 12일
심사완료 : 2010년 3월 19일

정보과학회논문지 : 소프트웨어 및 응용 제37권 제6호(2010.6)

1. 서 론

'문장'의 사전적인 의미는 '의사를 전달하는 최소의 단위'로 정의되어 있으며, 전통 문법에서는 '비교적 완전하고 독립된 의사전달 단위다'라고 정의하고 있다[1]. 문장은 구문분석이나 의미분석 등의 언어학적 분석 작업에서는 가장 기본이 되는 단위이며, 문장경계 인식 성능이 언어학적 분석 작업에 미치는 영향력은 매우 크다.

문어체로 되어 있는 문서의 경우에 문장경계는 대부분 마침표, 느낌표, 물음표 등의 문장 기호들로 구분이 되지만, 문장 내부에서 다른 의미로도 사용되므로 수작업이나 규칙 또는 통계적인 방법에 의한 자동화 방법에 의해 문장의 끝 여부를 결정해 줘야 한다. 구어체의 경우에는 문장 기호들이 문장경계 이외의 목적으로 더욱 다양하게 사용되므로 문어체보다 문장경계인식 작업이 더욱 어렵다. 최근에는 인터넷을 통해 일반인들이 작성한 문서들이 매우 많으며, 이런 웹 문서들은 문서 검색이나 정보 추출 시에 유용한 자료로 사용될 수 있다. 하지만 이런 웹 문서들은 사용자의 문법오류에 의한 띄어쓰기 오류나 오타 등이 많고, 사회적으로 유행하는 글 작성 행태에 맞춰서 구두점을 전혀 다른 의미로 사용하거나, 구두점을 생략하는 등의 다양한 형태로 작성된 것들이 많다.

따라서 이런 비전문가가 작성한 웹 문서를 대상으로 문서검색, 질의응답, 정보추출 등을 하기 위해서는 웹 문서의 다양성을 커버할 수 있는 문장경계인식 기술이 필요하다. 하지만 기존에 발표된 연구들은 문장경계를 인식하기 위해서 일반 문서만을 대상으로 문장경계인식기를 학습하였고 문장경계 후보도 구두점만을 대상으로 하였다. 그러므로 일반 문서를 대상으로 평가하였을 때 99% 이상의 성능을 보였어도 웹 문서에서도 동일한 성능을 보인다는 보장이 없고 관련해서 보고된 연구도 없다. 그에 따라 본 논문은 문어체 뿐 아니라 다양한 구어체를 처리할 수 있고, 특별히 인터넷 웹 문서에서 자주 발생하는 구두점 생략이나 띄어쓰기 오류도 커버할 수 있도록 웹 문서의 특징을 반영하고, 문장경계 대상이 될 수 있는 모든 종결어미를 대상으로 문장경계를 인식하는 기술을 제안한다.

제안한 문장경계 인식기는 기계학습 기반 분류모델에 의해서 학습되고 수행되며, 분류모델에 사용되는 자질들은 언어에 독립적인 자질들 위주로 사용되었다. 분류모델은 다양한 모델을 비교한 결과, FSMO를 사용한 structured SVM[2]이 가장 좋은 성능을 보였고 학습속도도 느리지 않았다. 기계학습 기반 분류모델은 학습데이터에 의존적이므로 학습데이터에 없거나 잘못된 정보에 의해서 오류를 발생시킬 수 있다. 본 논문은 그런 분류모델에 의한 오류를 후처리 규칙에 의해서 보정하였고, 정확도 향상을 확인하였다.

2. 관련 연구

문장경계인식을 위해 지금까지 사용된 방법에는 규칙에 기반한 방법과 기계학습 방법에 의한 것이 있다. 초기에는 대부분 규칙에 기반해서 인식하였고, 최근의 연구는 주로 기계학습 방법을 이용하고 있다.

규칙 기반 연구에는 Grefenstette and Tapanainen[3]가 구두점의 문장경계 여부를 판단하기 위해서 정규 표현식을 이용하여 애매성을 해소하였고, Brown 말뭉치를 대상으로 실험한 결과, 숫자표현에 대해서 93.64%, 축약어에 대해서는 99.07%의 정확률을 보였다. O'Neil [4]은 3개의 간단한 규칙으로 영어문장에 대해서 95%의 정확률을 보인다고 설명하였다. Stamatatos, Fakotakis, and Kokkinakis[5]는 Transformation based learning (TBL)을 이용해서 태깅 말뭉치로부터 문장경계인식 규칙을 자동으로 추출하는 방법을 제안했고, 7,274개의 문장으로부터 자동 추출된 312개의 규칙을 이용해서 8,736개의 문장을 평가해서 99.4%의 정확률을 보였다.

기계학습 기반 연구에는 규칙기반 연구보다 훨씬 다양한 연구들이 진행되었다. Riley and Michael[6]은 구두점 주변 단어의 출현확률 및 구두점이 발견된 어절의 클래스 등의 자질을 추출하였고, AP News 2,500만 단어를 학습해서 만들어진 Decision Tree(C4.5)를 이용해서 Brown 말뭉치를 평가하여 99.8%의 정확률을 보였다. Palmer and Hearst[7,8]는 구두점 주변 단어에 대한 품사의 확률정보 및 20개의 토큰 정보를 이용하여 Feed-forward Neural Network를 학습한 결과 98.5%의 정확률을 보였고, 추가 연구로 결정 트리와 신경망, 품사 정보를 포함하는 사전을 이용하여 Wall Street Journal (WSJ) 말뭉치에서 98.5%의 정확도를 보였다. Reynar and Ratnaparkhi[9]는 구두점 후보가 발생한 앞/뒤 토큰의 확률정보를 이용하였으며, Maximum Entropy (ME) 기법을 이용하여 WSJ 및 Brown 말뭉치에서 각각 98.0%, 97.5%의 정확률을 보였다. Mikheev[10]는 문장경계인식기를 품사 태깅 기술과 결합시켜서 18개 품사에 대한 태깅 결과를 이용해서 문장경계인식을 수행하였고, Brown 말뭉치에 대해서 99.8%, WSJ 말뭉치에 대해서 99.61%의 정확률을 보였다. Wang and Huang[11]은 문장경계인식을 위해서 8개의 자질을 추출하였고 3개의 알고리즘(규칙기반, HMM, ME)을 이용해서 문장경계인식 성능을 비교하였다. WSJ 말뭉치를 대상으로 평가해서, 규칙기반은 76.95%, HMM은 94.46%, ME는 97.62%의 결과를 얻었다. Liu, Stolcke, Shriberg, and Harper[12]는 음석인식 결과에 대한 문장경계인식

기술에 대한 것으로 HMM, ME, Critical Random Fields (CRF) 3개의 알고리즘을 비교하였다. 자질은 음성 자질, n-gram 단어, 품사 태깅 결과, 청킹 결과, 그리고 단어 클래스를 사용하였고, Broadcast News를 대상으로 실험해서 HMM은 96.47%, ME는 96.48%, 그리고 CRF는 96.53%의 정확률을 보였다. Pan and Shaw [13]는 서수, 이니셜, 축약어 정보에 기반 해서 주변 토큰의 확률정보와 규칙을 자동으로 확장함으로써 문장경계를 인식하는 기술을 제안하였고, 언어독립적인 자질을 사용해서 영어권의 10개 언어에 대해서 실험한 결과, 신문기사를 대상으로 평균 98.74%의 정확률을 보였다.

한국어에 대한 것도 최근에 연구되고 있는데, 임희석, 한군희[14]는 후보 구두점 자체의 확률, 앞/뒤 발생하는 음절 그리고 인용부호의 개수를 자질로 이용하였으며, kNN 알고리즘으로 ETRI, KAIST 코퍼스에서 각각 96.73%, 98.64%의 정확률을 보였다. 또한 두 코퍼스를 모두 학습한 경우에는 98.82%의 정확률을 보였다. 박수혁, 임해창[1]은 구두점만을 문장경계 인식대상으로 고려하고, 12개의 자질을 사용해서 다양한 기계학습 분류모델을 문장경계인식기에 적용하였고, 최고 성능을 보이는 Decision Tree와 Random Forest 알고리즘을 사용해서 세종코퍼스를 대상으로 실험하여 Decision Tree는 98.4%, Random Forest는 99.1%의 성능을 얻을 수 있었다.

기존 연구와 같이 영어권의 경우, 모든 연구가 구두점만을 대상으로 문장경계 모호성을 해소하는 방법에 대한 것이며, 주로 축약어나 서수의 모호성만을 해소하면 되는 간단한 문제이므로 규칙기반 방법이 초기에 많이 사용되었고 평균 95% 이상의 성능을 보였다. 최근에는 기계학습 방법이 주로 사용되는데, 학습 및 평가 문서를 신문기사와 같이 구두점이 명확한 의미로 사용되는 것들을 대상으로 하였으므로 99% 이상의 성능을 보인다. 한국어의 경우에도 기존 연구들은 주로 문어체의 일반 문서를 대상으로 학습하고 평가하였고, 문장경계 후보도 구두점만을 대상으로 고려하고 있으며, 구두점이 모호하게 사용되는 경우가 많은 한국어의 특징 때문에 규칙보다는 기계학습 방법에 의해서 문장경계 모호성을 해소하였다.

하지만, 일반인이 작성하는 웹 문서의 경우에는 구두점이 생략되거나 띄어쓰기 오류가 빈번히 발생하므로 기존 연구를 웹 문서에 적용하기에는 무리가 있다. 이에 본 논문은 구두점 외의 문장경계에 사용되는 모든 종결어미를 대상으로 문장경계 모호성을 해소할 수 있는 방법을 제안한다.

3. 기본 문장 경계 인식

본 논문에서 제안하는 개선된 문장경계 인식 기술을 적용하기 전에 선행되어야 할 일은, 기계학습 기반 분류모델을 최적화하기 위해서 최적의 자질과 학습데이터를 선택하고, 최종적으로 최적의 분류 모델을 선택하는 일이다. 3장에서는 기존 연구와 동일한 기술로 구현된 문장 인식기를 기본 문장 경계 인식기라고 명명하고, 기본 문장 경계 인식기를 최적화하는 과정을 자세히 다룬다.

3.1 통계적 자질

3.1.1 자질 집합

문장경계 모호성 해소를 위한 기계학습 기반 분류모델에 사용된 자질은 아래와 같이 10개 자질을 사용하였다.

자질1) 문장 경계 후보

문장경계 모호성을 해소할 대상 후보로는 모호성 해소의 어려움에 따라 3개의 단계로 구분하였다.

- 단계1: 3개의 구두점(마침표, 물음표, 느낌표)
- 단계2: 문장종결에 사용될 수 있는 15개 어미 (다, 네,오,어,지,나,군,라,니,가,까,게,자,세,요)
- 단계3: 문장종결에 사용된 모든 음절(문장경계 태깅 말뭉치로부터 435개 추출) (본 논문에서는 단계3은 사용하지 않음)

자질2) 문장경계 다음 음절의 공백 여부

문장경계 모호성 해소에 문장경계 다음 음절의 공백 여부가 중요한 역할을 하기 때문에 자질로 고려하였다.

자질3,4) 문장경계 앞/뒤 1번째 음절

문장경계 바로 이전과 이후에 나타난 음절 정보

자질5,6) 문장경계 앞/뒤 2번째 음절

문장경계 2번째 이전과 이후에 나타난 음절 정보

자질7,8) 문장경계 앞/뒤 1번째 토큰

문장경계 바로 이전과 이후에 나타난 토큰 정보로, 추출되는 토큰의 형태는 같은 종류의 2byte 글자(한글, 일본어, 한자, 기타)와 1byte 글자(숫자, 영어, 기타)를 1개의 토큰으로 추출한다.

자질9,10) 문장경계 앞/뒤 1번째 토큰의 길이

문장경계 바로 이전과 이후에 나타난 토큰의 길이

3.1.2 자질 선택

가. 실험 환경

- 문장경계 후보: 단계1의 구두점만을 대상
- 학습데이터: 신문 기사와 블로그 문서로 구성된 32,469문장
- 평가셋: 신문기사, 블로그, 에세이 등의 구어체와 문어체가 섞인 3,455문장 (평가셋1)
- 분류 모델: SVM_light

나. 자질별 기여도

기계학습 방법에서는 어떤 자질을 사용하느냐에 따라 성능에 영향을 받으므로 자질별 기여도를 다양한 방법에 의해 비교 실험하였다.

1) 자질별 단독 사용

각 자질을 단독으로 사용했을 때의 문장경계 인식 성

표 1 자질별 단독사용 실험결과

자질종류	Precision	Recall	F-measure
자질2	**0.951**	0.784	0.859
자질3	0.949	0.920	**0.934**
자질4	0.892	0.975	0.932
자질5	0.869	0.957	0.911
자질6	0.849	0.977	0.909
자질7	0.851	0.963	0.903
자질8	0.846	**0.985**	0.910
자질9	0.842	0.962	0.898
자질10	0.862	0.926	0.893

능을 측정하였고(표 1), 실험결과 자질3이 가장 성능이 좋았고 F-measure로 93.4%의 성능을 보였다. 자질별로 precision이나 recall에 각각 더 좋은 자질이 있음을 알 수 있다. (1번 자질은 필수 자질로 평가에서 제외하고 이후 실험에서도 제외하였음)

2) 자질별 추가에 따른 성능 변화

모든 구두점을 무조건 문장경계로 인식했을 때의 Baseline 성능에 비해서, 자질을 1개씩 누적해서 추가함에 따른 성능 변화를 실험하였다(표 2).

표 2 자질별 추가에 따른 성능 평가

추가자질	Prec.	Rec.	F	Impr.
Baseline	0.776	1.0	0.874	0.0%
+자질2	0.951	0.784	0.859	-1.5%
+자질3	**0.940**	**0.954**	**0.947**	**8.7%**
+자질4	**0.968**	**0.948**	**0.957**	**1.1%**
+자질5	**0.972**	**0.950**	**0.961**	**0.3%**
+자질6	0.971	0.951	0.961	0.0%
+자질7	0.971	0.951	0.961	0.0%
+자질8	0.973	0.949	0.961	0.0%
+자질9	0.977	0.947	0.962	0.1%
+자질10	0.976	0.951	0.964	0.2%

실험 결과, 자질2를 제외하고는 자질을 추가할 때마다 성능 향상이 있었고, 자질3이 가장 큰 성능 향상을 가져왔다. 자질2는 처음으로 추가된 자질로 추가 시 Baseline에 비해 1.5% 성능저하가 있었지만, Baseline에 비해 Precision이 대폭 개선되었고, 추가 실험을 통해 다른 자질과 함께 사용되는 경우는 성능향상에 도움이 된다는 것을 확인하였다.

3) 자질별 제외에 따른 성능 변화

모든 자질을 고려한 경우와 비교해서, 특정 1개의 자질만을 빼는 경우에 따른 성능 변화를 측정하였다(표 3).

실험 결과, 어떤 자질을 빼도 성능 저하가 발생하므로 모든 자질이 유용함을 알 수 있고, 특히 문장경계후보 바로 앞 음절에 대한 자질3의 경우에 가장 큰 성능 저하가 발생하여, 1,2,3번 실험 모두에서 중요한 자질임이 확인되었다.

표 3 자질별 제외에 따른 성능 평가

제거자질	Prec.	Rec.	F	Impr.
ALL	0.976	0.951	0.964	0.0%
-자질2	0.976	0.949	0.963	-0.1%
-자질3	**0.937**	**0.953**	**0.945**	**-1.9%**
-자질4	**0.966**	**0.955**	**0.960**	**-0.3%**
-자질5	0.976	0.948	0.962	-0.2%
-자질6	0.976	0.949	0.963	-0.1%
-자질7	0.977	0.948	0.962	-0.1%
-자질8	0.973	0.953	0.963	-0.1%
-자질9	0.973	0.952	0.962	-0.1%
-자질10	0.977	0.947	0.962	-0.2%

4) 자질별 조합에 따른 성능 변화

9개의 자질을 1개부터 9개까지 모든 경우를 고려해서 조합해서 성능을 측정하였다(표 4). 실험결과 F-measure는 모든 자질을 사용하였을 때 가장 성능이 좋았고, Precision은 자질10을 제외한 나머지를 사용했을 때 가장 높고, Recall은 자질8과 자질10 만을 사용했을 때 가장 높았다.

표 4 자질별 조합에 따른 성능 평가

개수	Precision (조합)	Recall (조합)	F (조합)
1	0.951 (2)	0.985 (8)	0.934 (3)
2	0.968 (3+4)	**0.994** (8+10)	0.958 (3+4)
3	0.976 (3+4+9)	0.987 (2+6+10)	0.961 (3+4+7)
4	0.976 (2+3+4+9)	0.962 (2+5+6+8)	0.961 (2+3+4+9)
5	0.976 (2+3+4+5+9)	0.960 (2+5+6+7+8)	0.962 (2+3+4+5+9)
6	0.975 (2+3+4+5+6+9)	0.958 (2+5+6+7+8+9)	0.962 (2+3+4+5+6+9)
7	0.976 (2+3+4+5+6+7+9)	0.956 (2+4+5+6+7+8+10)	0.962 (2+3+4+5+6+7+9)
8	**0.977** (2+3+4+5+6+7+8+9)	0.955 (2+3+4+5+6+8+9+10)	0.963 (3+4+5+6+7+8+9+10)
9	0.976 (ALL)	0.951 (ALL)	**0.964** (ALL)

5) 최종 자질 선택

앞에서의 4가지 실험을 통해서 10개의 자질은 모두 미미하더라도 성능 향상에 도움이 되었고, 특히 자질3의 중요성이 확인되었다. 그러므로 최종 자질로 10개 모두를 선택하였다.

3.3 학습말뭉치에 따른 성능 비교

3.2절까지의 실험을 통해 최적의 자질을 선정하였다. 이렇게 선정된 값들로 문장경계 인식기를 만들어 이번에는 학습말뭉치에 따른 성능 변화를 실험하였다.

가. 실험 환경

- 문장경계 후보: 단계1의 구두점만을 대상
- 평가셋: 신문기사, 블로그, 에세이 등의 구어체와 문어체가 섞인 3,455문장(평가셋1)

나. 학습데이터별 성능 평가

학습데이터는 아래와 같이 5가지를 대상으로 실험하였다.

- Article 1(A1): 2007년 4,5월 신문기사
- Blog 1(B1): Allblog 사이트로부터 추출된 블로그
- Sejong 1(S1): 세종말뭉치로부터 추출된 다양한 장르의 문서
- Article 2(A2): 2007년 6,7월 신문기사
- Blog 2(B2): 이글루스 사이트로부터 추출된 블로그

학습데이터는 크게 3종류로 신문기사, 블로그, 다양한 장르의 세종말뭉치로 구분되며, 신문기사가 A1, A2로 구분된 이유는 신문기사는 날짜별 이슈에 따라서 내용이 많이 차이 나므로 월별로 구분을 해서 2개를 만들었고, 블로그도 주요 사이트 2개를 각각 구분해서 내용에 따라 구분하였다.

성능평가는 5가지의 말뭉치를 1개부터 5개까지 모든 조합을 사용해서 학습하였고, 평가 척도별 최고 성능의 조합 결과는 아래와 같다(표 5).

표 5 학습말뭉치별 성능 평가

조합	Prec.	Recall	F
A1+A2	**0.990**	0.823	0.898
B2	0.910	**0.969**	0.939
A2+B2	0.976	0.954	**0.965**
ALL	0.985	0.919	0.951

실험결과, 모든 말뭉치를 사용하는 것보다 A2와 B2만을 사용하는 것이 F-measure가 최고 성능을 보였고, precision과 recall의 최고성능을 보이는 말뭉치도 모두 달랐다. 실험을 통해서, 기계학습 분류모델의 학습데이터 의존도를 알 수 있었고, 적용하는 대상에 따라서는 학습데이터도 선별적으로 사용해야 한다는 것을 확인하였다.

3.4 기본 문장경계인식기 최적화 결과

3장에서는 3.3절까지의 실험을 통해서 일반 문서에 대해서 학습되고 구두점만을 대상으로 문장경계를 인식하는 기본 문장경계인식기를 최적화하는 방법에 대해서 실험하였고, 최종적으로 아래와 같은 결과를 얻었다.

- 문장경계후보: 단계1
- 통계자질: 10가지
- 학습데이타: A2+B2

분류모델 최적화 실험은 CRF, ME, SVM 모델을 비교하였고, 모델 간에는 1% 내외의 차이만을 보였지만 SVM이 가장 좋은 성능을 보였으므로 4장부터의 실험에서는 분류 모델로 기본적으로 SVM을 사용하였다.

4. 개선된 문장 경계 인식

4.1 웹 문서의 특징

웹 문서는 일반인들이 작성하는 글이 많고 연령대도 다양하므로, 주로 전문가가 작성하는 일반 문서와 구별되는 특징이 있고, 주요 특징과 예제 문장은 다음과 같다.

- 구두점 생략이 빈번

예제문장: 샤오랑은 오른쪽 눈, 다른아이는 왼쪽눈에 안대를 하고 있는 걸로 봐선 샤오랑의 **다른모습인건가** 샤오랑이 선이라면,. 그는 악., 뭐 그런? 농락당한 **기분이었다**-_ㅠ 사쿠라의 힘으로 그는 다시 **잠들었다**-_-;;

- 띄어쓰기 오류 및 오타 많음

예제문장: 아;;입장권있으면 풀 **무료이용맞구요** 빌리지라는게 케리비안베이에서 당일치기로 **노는게아니라** 1박2일 2박3일 **이런식으로놀예** 한마디로 숙소입니다

- 구두점을 다양한 용도로 사용

예제문장: 약간 패닉상태에 빠져버리는 일도(..) 있고.. **T.T… 하.지.만.** 나는 안 그랬엉

4장에서는 이런 웹 문서에 대해서 기존의 문장경계인식 기술을 그대로 적용하는 경우의 성능을 평가하고, 웹 문서를 위해서 문장경계인식기를 개선시키는 방법에 대해서 설명한다.

4.2 웹 문서 성능평가를 위한 평가셋

웹 문서에 대한 문장경계인식 성능 측정을 위해서 웹 사이트 게시판 등의 웹 문서로부터 수집된 3,104개의 문장들로 구성된 2차 평가셋을 만들었고, 웹 문서의 특징을 알아보기 위해서 2차 평가셋에 대한 2가지 Baseline 평가를 하였다.

표 6 평가셋2의 Baseline 평가 결과

	Prec.	Recall	F
Baseline	0.327	0.424	0.369

첫 번째 실험은, 주로 구어체로 구성된 2차 평가셋의 구두점 생략 정도를 확인하기 위해서 모든 구두점만을 문장경계로 인식한 Baseline 성능을 측정하였고, 결과는 표 6과 같다.

문장경계로 구두점만이 사용되는 평가셋1(recall 1.0)에 비해서 평가셋2의 Baseline 성능은 recall이 0.424이므로 웹 문서는 57.6% 정도는 문장경계에서 구두점이 생략된다는 것을 확인하였고, precision이 32.7%이므로 구두점 자체도 문장종결 이외의 기능으로 훨씬 다양하게 사용됨을 확인하였다.

두 번째 실험은, 구두점 이외의 모든 종결어미도 무조건 문장경계로 고려하는 Baseline2 성능을 측정하였다(표 7).

표 7 평가셋2의 Baseline2 평가 결과

	Prec.	Recall	F
Baseline2	0.213	1.0	0.351

이번실험은 문장경계로 사용될 수 있는 모든 종결어미 후보까지 고려할 경우의 모호성을 측정하기 위한 것이며 실험결과 21.3%의 정확도를 보였다. 즉, 78.7%의 모호성이 존재하여 22.5%(평가셋1:0.775)의 모호성이 존재하는 일반 문서에 비해서 훨씬 문장경계인식이 어려움을 알 수 있다.

4.3 웹 문서에 대한 기본 모델 실험

이번 실험에서는 3장까지의 실험을 통해 최종적으로 얻어진 기본 문장경계인식기를 웹 문서에 그대로 적용해 봤다.

가. 실험 환경

- 문장경계 후보: 단계2의 문장종결어미
- 학습데이터: A2+B2
- 평가셋: 평가셋2

나. 실험 결과

이번 실험을 위해서 문장종결어미에 대해서도 문장경계를 인식하도록 기본 모델을 수정해서 실험하였고 결과는 아래와 같다(표 8).

표 8 평가셋2에 대한 기본 모델 성능 평가

	Prec.	Recall	F
기본모델	0.622	0.520	0.567

기본 모델은 구두점만을 대상으로 분류모델이 학습되었기 때문에, Baseline2보다는 성능이 좋지만 일반문서에서의 성능(평가셋1:0.965)에 비해서는 많이 떨어지는 결과를 얻었다. 즉, 기존에 연구되어진 문장경계인식 기술을 웹 문서에 그대로 적용하기에는 문제가 있음을 알 수 있다.

4.4 웹 문서를 위한 개선된 문장경계 인식

4.4.1 개선 작업

웹 문서에 대한 문장경계인식 성능을 개선하기 위해서 작업한 내용은 다음과 같다.

- 개선1: 학습 대상 문장경계후보 확장

기본 모델의 경우에는 문장경계후보로 3개의 구두점에 대해서만 주변 문맥 정보를 학습하였지만, 웹 문서의 경우에는 구두점이 생략되는 경우가 많고 이런 경우에는 주변 문맥도 다른 경우가 빈번히 발생한다. 그래서 개선1에서는 문장경계후보로 15개의 종결어미를 추가로 고려해서 학습하였다. 또한 4.2절의 Baseline2 실험에서 확인되듯이 확장된 문장경계후보들은 구두점에 비해서 문장종결 이외의 기능으로 사용되는 경우가 훨씬 많으므로 문장경계 후보의 부정자질도 추가 학습하였다.

- 개선2: 웹 문서를 위한 자질 확장

웹 문서의 경우, 특히 게시판의 경우에는 한 라인에 한 문장을 쓰는 경우가 많다. 그래서 일반 문서와는 다르게 뉴라인캐릭터가 음절이나 토큰 자질로 고려될 수가 있다. 그래서 음절 자질 중 후보 다음 음절을 고려하는 자질4와 자질6, 그리고 후보 다음 토큰을 고려하는 자질8의 경우에는 뉴라인캐릭터도 고려하도록 엔진을 수정하였다.

```
7월 25일 부터 울릉도 여행하려고 합니다{S1}
우선 배편은 다행이고 예약이 되었는데...{S2}
숙박시설을 알아보던중 결정할수 없어 경험하신 분들의 추천을 받을려고 합니다{S3}
```

게시판의 특징을 보여주는 문장은 다음과 같다.

- 1번 문장: 7월 25일 부터 울릉도 여행하려고 합니다
- 문장경계후보: {S1}
- 자질4
  - √ 기본모델: 우
  - √ 개선모델: \n (뉴라인캐릭터)

위 게시판 문장 중 1개 문장에 대한 자질 추출의 예는 다음과 같다.

- 개선3: 학습데이타 최적화 및 확장

3.3절에서 실험한 기본 모델에 대한 학습데이타 최적화 실험에서는 모든 학습데이타를 사용하는 것보다 A2와 B2 데이터만을 사용하는 것이 가장 성능이 좋았다. 하지만 평가셋2는 일반 문서가 아닌 웹 문서 위주로 구성되어 있으므로 주변 문맥 정보가 다르므로 새로운 최적화된 학습데이타를 분석할 필요가 있다. 실험 결과, 블로그 문서로 구성된 B1과 B2를 사용한 경우에 A2와 B2를 사용한 것보다 1.2% 정도의 성능 향상이 있었으므로 이후 실험에서는 B1과 B2로 학습된 모델을 사용하였다. 또한 웹 문서의 경우에는 다양한 사용자들의 글이 많으므로 개인에 따라 다양한 형태로 문장을 구성하여 문장종결후보의 앞, 뒤 문맥 정보도 일반 문서에 안 나타나는 것들이 많이 나타난다. 그래서 개선3에서는 웹 문서로 구성된 추가 학습데이타로 8,046문장을 구축하여 분류 모델을 추가로 학습하였다.

- 개선4: 근접 후보 제거

단계2의 문장경계후보의 경우에는 문장종결이 아닌 경우에도 많이 사용되므로 후보들이 근접해서 발생하는 경우가 많다. 예를 들면 문장 "물론 밖에 다 나와있는

곰돌이네도 있었습니다{S} 자동차군보다 귀여웠어요{S} 어쨌든 나는 요기가 더 좋다니까요{S}"의 경우에는 실제 문장경계는 3개지만 후보는 17개나 발생한다. 이런 경우에는 근접한 후보 중에 실제 후보는 1개만 대상인 경우가 많으므로, 후보의 우선순위에 따라 1개만 후보로 남기고 나머지는 후보로 고려하지 않았다. 우선순위는 3.1.1절의 자질1에 나온 단계1>단계2 순으로 정하였다. 즉, 단계1이 단계2보다는 더 확실한 후보라는 가정 하에 정하였다.

• 개선5: 문장경계후보의 생성 방법 수정

기본 모델에서는 대상 문서의 문장경계후보를 한꺼번에 생성한 후, 후보의 앞/뒤 문장을 이용해서 대상 후보의 모호성을 해소 하였다. 하지만 웹 문서의 경우에는 후보의 개수 및 모호성이 많아지면서 개선4에서 언급한 문제에 의해 후보의 앞/뒤의 후보 문장의 길이가 짧아지면서 추출되는 자질에 오류가 발생하였다. 그래서 개선5에서는 앞에서부터 후보를 1개씩 순차적으로 생성하면서 모호성을 해소하도록 수정하였다.

위의 5가지 개선 작업이 실제로 문장경계 인식기의 성능에 어떤 영향을 주는지 평가셋2를 대상으로 실험하였다. 실험은 각 개선작업을 1개씩 추가하면서 각각의 성능을 측정하여 개별적인 성능 향상 여부를 분석하였다.

가. 실험 환경

• 문장경계 후보: 단계2의 문장종결어미

• 평가셋: 평가셋2

나. 실험 결과

비교 대상으로는 단계2의 후보를 무조건 문장종결대상으로 고려한 Baseline2를 평가 대상으로 고려하였고, 추가 비교작업으로, 개선 모델은 기본 모델의 문제점에 대해서 개선하는 작업을 하였으므로 기본 모델도 비교 대상으로 고려하였다(표 9).

표 9 기본모델 튜닝결과(개선별 성능 평가)

	Prec.	Recall	F	Impr.
Baseline2	0.213	1.0	0.351	-
기본모델	0.622	0.520	0.567	21.6%
+개선1	**0.935**	**0.504**	**0.655**	**8.8%**
+개선2	0.934	0.505	0.656	0.1%
+개선3	0.984	0.550	0.706	5.0%
+개선4	**0.990**	**0.925**	**0.956**	**25.0%**
+개선5	0.990	0.938	0.963	0.7%

위 실험 결과에 나와 있듯이 성능 향상이 미미한 것도 있지만 모든 개선 작업이 성능 향상에 도움을 주는 것을 확인하였다. 모든 개선 작업이 반영된 최종 개선모델의 경우에 Baseline2에 비해서는 61.2%, 기본모델에 비해서는 39.6%가 향상되었다.

가장 크게 성능을 향상시킨 개선4의 경우에, 구두점에 비해서 근접해서 많이 발생하는 단계2의 문장경계 후보들의 특징을 고려해서 문장경계인식 알고리즘을 수정한 것이 크게 도움이 되었고, 두 번째로 도움이 된 개선1의 경우에는 단계2의 문장경계 후보는 구두점에 비해서 다른 의미로 사용되는 경우가 훨씬 많기 때문에 각 후보의 부정 자질을 추가로 학습한 것이 도움이 되었다. 하지만 생각보다는 학습데이타 추가에 의한 개선1의 성능 향상이 아주 크지는 않았는데 그 원인은 기존 학습데이타에도 블로그 문서 등의 웹 문서가 포함되어 있어서 추가 학습데이타의 영향력이 적었던 것으로 분석된다.

지금까지의 실험으로 웹 문서의 경우에는 기존 문장경계인식 기술을 그대로 사용하면 성능이 크게 떨어짐을 확인하였고, 본 논문에서 제안한 개선 방법이 성능 향상에 효과적임이 입증되었다.

4.4.2 추가 실험

이번 실험에서는 4.4.1절에서 개선시킨 문장경계인식기가 3장에서 만들어진 기본문장경계인식 기능에 어떤 영향을 주는 지를 확인하려고 한다. 즉, 웹 문서만을 위해서 개선시킨 작업이 일반 문서의 경우에는 부작용이 크다면 개선 모델의 활용성에 큰 제약이 있을 것이므로 그에 대한 검증을 하려고 한다.

평가셋은 구두점만이 문장경계로 사용되고 일반문서로 구성된 평가셋1을 사용하였고, 실험은 2가지에 대해서 진행되었다.

첫 번째 실험은 문장경계후보로 구두점만을 대상으로 인식하였고, 개선모델은 2개의 모델을 만들어 기본모델과 비교하였다(표 10). 개선모델1과 2는 기본모델과 동일하게 문장경계 후보로 단계1만을 인식하도록 다시 학습시켰으며, 개선모델 1과 2의 차이는 학습데이타를 다르게 사용한 것으로 개선모델1은 웹 문서인 블로그1(B1), 블로그2(B2), 추가 학습데이타로 학습된 모델이고, 개선모델2는 일반문서인 신문기사2(A2)와 블로그2(B2)로 학습된 모델이다.

표 10 단계1 평가: 기본모델 vs. 개선모델

Lv1	Prec.	Recall	F	Impr.
기본모델	0.976	0.954	0.965	-
개선모델1	0.961	0.948	0.955	-1.0%
개선모델2	**0.958**	**0.982**	**0.970**	**+0.5%**

실험결과를 분석해보면, 개선모델1이 기본모델에 비해서 1% 성능저하가 있는 것은 웹 문서로 구성된 학습데이터가 일반 문서의 특징을 덜 반영해서 그렇다. 하지만 웹 문서에도 일반 문서의 특성을 많이 포함하므로 성능 저하가 크지는 않았다. 개선모델2는 기본모델과 동일한

학습데이터로 학습된 것으로, 개선모델에서 새로 추가된 자질 및 엔진 개선작업만이 결과에 영향을 주었고, 실험결과, 추가된 자질 및 엔진 개선작업이 0.5% 정도 성능을 향상시키는 것이 확인되었다. 즉, 개선 작업이 웹 문서 뿐 아니라 일반문서에 대해서도 필요한 작업임이 확인되었다.

두 번째 실험은 문장경계후보로 문장종결어미도 대상으로 인식하였고 결과는 다음과 같다(표 11).

표 11 단계2 평가: 기본모델 vs. 개선모델

Lv2	Prec.	Recall	F	Impr.
기본모델	0.976	0.954	0.965	-
개선모델1	0.935	0.948	0.941	-2.3%
개선모델2	0.927	0.979	0.952	-1.3%

이번 실험은 구두점만이 문장경계로 사용되는 문서를 대상으로 구두점 외의 종결어미까지 문장경계후보로 고려하는 경우에 어떤 영향을 주는가를 보기 위한 것으로, 문장종결에 구두점만이 사용되었는지를 문서마다 미리 알 수 없는 경우에 문장종결어미까지 후보로 무조건 고려할 때의 영향력을 확인하기 위한 실험이다.

예상대로 구두점만을 보는 것보다 많게는 2.3% 정도의 성능 저하를 초래했다. 즉, 실제 문장경계가 아닌 후보들까지 고려하므로 문장경계를 잘못 인식하는 경우들이 발생해서 성능저하가 있었다. 하지만 엔진이 그런 오류들에 대한 오류율이 낮아서 성능 저하가 크지는 않았다. 이 실험을 통해서 엔진의 robustness까지 확인하였다.

4.4.2절의 실험 결과를 정리하면 다음과 같다.

• 첫 번째 실험: 기본모델 vs. 개선모델1

학습데이터 문제: 웹 문서를 위해서 구성된 학습데이터는 일반 문서에 대한 문장경계인식에서는 1% 정도의 성능 저하를 초래했지만, 웹 문서도 일반 문서의 특징을 가지고 있어서 성능 저하가 크지는 않다.

• 첫 번째 실험: 기본모델 vs. 개선모델2

개선모델의 추가 자질 및 개선작업 기여도: 웹 문서를 위해 추가된 자질 및 개선작업은 일반 문서에서도 긍정적인 영향을 준다. 또한 추가된 작업에 의해서 고려하는 경우가 다양해지면서 정확도는 떨어졌지만 재현율이 많이 올라가서 전체적인 성능은 향상되었다.

• 두 번째 실험

문장경계후보: 웹 문서를 위해서 정의된 19개의 문장경계후보를 구두점만 문장경계로 사용된 문서에 적용한 경우 2.3%의 성능 저하를 초래하지만, 성능 저하가 크지는 않으므로 구두점만 사용되는 문서를 포함해서 모든 문서에 대해서 개선 모델을 적용하기에 무리가 없음이 확인되었다.

5. 결 론

본 논문에서는 기존에 연구되어진 문장경계인식기가 웹 문서에서도 좋은 성능을 보이는지 확인하기 위해서 먼저 기존 연구와 동일한 기능을 수행하는 기본 문장경계인식기로서, 일반 문서를 대상으로 학습하고 구두점만을 대상으로 문장경계를 인식하는 기본 모델을 만들었고 다양한 실험을 통해 최적화시켰다.

이렇게 만들어진 기본 문장경계인식기를 그대로 웹 문서에 사용한 결과, 56.7%의 매우 낮은 성능을 보임을 확인하였다. 이에 따라 기본 모델의 성능 개선을 위해서 웹 문서의 특징을 반영하기 위한 5가지 개선 작업을 하였고, 최종적으로 만들어진 개선 모델은 기본 모델에 비해서 39.6%의 성능 개선을 보여서, 본 논문에서 제안한 방법이 효과적임을 확인하였다.

마지막으로 추가 실험을 통해, 웹 문서의 특징을 반영하여 개선시킨 문장경계인식기가 일반 문서에 어떻게 영향을 주는지 확인하였다. 그 결과, 개선된 모델에 추가된 자질 및 엔진 개선 작업은 일반 문서에 대해서도 성능 향상에 도움이 됨을 알 수 있었고, 다양한 문장경계 후보를 보는 것이 구두점만이 문장경계로 사용되는 문서에 대해서는 성능을 감소시키지만 성능 저하가 크지 않으므로 개선된 모델을 일반 문서를 포함한 대부분의 문서에 그대로 적용해도 문제가 없음을 확인하였다.

향후 연구로는 현재의 자질 외에 더욱 효과적인 자질을 찾아보고, 더욱 다양한 분류모델을 문장경계 인식에 적용해서 인식성능 및 속도에 더욱 최적화된 모델을 알아보려고 한다. 또한 한국어가 아닌 다른 언어에 대해서도 실험을 해서 자질의 언어 독립적 특징을 확인하려고 한다.

참 고 문 헌

[1] S. H. Park, H. C. Rim, "Sentence Boundary Detection Using Machine Learning Techniques," *Proc. of the 35th KIISE Springl Conference*, vol.15, no.1, pp.122-124, 2008. (in Korean)

[2] C. Lee, M. G. Jang, "Fast Training of Structured SVM Using Fixed-Threshold Sequential Minimal Optimization," *Journal of ETRI Journal*, vol.31, no.2, Apr. pp.121-128, 2009.

[3] G. Grefenstette, P. Tapanainen, "What is a word, what is a sentence? problems of tokenization," *Proc. of the 3rd International Conference on Computational Lexicography*, pp.79-87, 1994.

[4] J. O'Neil, "Doing Things with Words, Part Two: Sentence Boundary Detection," URL: http://www.attivio.com/blog/57-unified-information-access/263-doing-things-with-words-part-two-sentence-boundary-detection.html#ixzz0QOiR1kVm, 2008.

[5] S. Fakotakis, Kokkinakis, "Automatic extraction of rules for sentence boundary disambiguation," *Proc. of ACAI'99*, 1999.
[6] Riley, Michael, "Some Applications of Tree-based Modeling to Speech and Language Indexing," *Proc. of the DARPA speech and natural language workshop*, pp.339-352, 1989.
[7] D. D. Palmer, M. A. Hearst, "Adaptive sentence boundary disambiguation," *Proce. of the fourth conference on Applied natural language processing*, 1994.
[8] D. D. Palmer, M.A.Hearst, "Adaptive Multilingual Sentence Boundary Disambiguation," *Journal of Computational Linguistics*, vol.23, no.2, pp.241-267, 1997.
[9] J. C. Reynar, A. Ratnaparkhi, "A Maximum Entropy Approach to Identifying Sentence Boundaries," *Proc. of the Fifth Conference on Applied Natural Language Processing*, pp.16-19, 1997.
[10] A. Mikheev, "Tagging Sentence Boundaries," *Proc. of NACL'2000*, pp.264-271, 2000.
[11] H. Wang, Y. Huang, "Bondec: A Sentence Boundary Detector," URL: http://nlp.stan-ford.edu/courses/cs224n/2003/fp/huangy/final_project.doc, 2003.
[12] Y. Liu, A. Stolcke, E. Shriberg, M. Harper, "Using conditional random fields for sentence boundary detection in speech," *Proc. of ACL'05*, 2005.
[13] T. Kiss, J. Strunk, "Unsupervised Multilingual Sentence Boundary Detection," *Journal of Computational Linguistics*, vol.32 Issue 4, 2006.
[14] H. S. Lim, K. H. Han, "Korean Sentence Boundary Detection Using Memory-based Machine Learning," *Journal of The Korea Contents Society*, vol.4 no.4, pp.133-139, 2004. (In Korean)

이 충 희

1996년 한양대학교 전자계산학과(학사) 2001년 연세대학교 컴퓨터과학과(석사) 2001년~현재 한국전자통신연구원 선임연구원. 관심분야는 자연어처리, 정보검색, 질의응답

장 명 길

1988년 부산대학교 계산통계학과(이학사). 1990년 부산대학교 계산통계학과(이학석사). 2002년 충남대학교 컴퓨터과학과(이학박사). 1990년~1998년 5월 시스템공학연구소 선임연구원. 1998년 6월~현재 한국전자통신연구원 지식마이닝연구팀 팀장. 관심분야는 정보검색, 질의응답, 자연어처리

서 영 훈

1983년 서울대학교 컴퓨터공학과(학사) 1985년 서울대학교 컴퓨터공학과(석사) 1991년 서울대학교 컴퓨터공학과(박사) 1994년~1995년 미국 Carnegie Mellon 대학 기계번역센터 객원교수. 1988년~현재 충북대학교 전자정보대학 컴퓨터공학과 교수. 관심분야는 자연언어처리, 한국어 구문분석, 한영 기계번역, 정보검색, 질의응답시스템