하이브리드 방법을 이용한 개선된 문장경계인식

이충희○ 장명길 서영훈**
한국전자통신연구원 지식마이닝연구팀
**충북대학교 전자정보대학 컴퓨터공학과
{forever, mgjang}@etri.re.kr, **yhseo@chungbuk.ac.kr

Advanced detection of sentence boundaries based on hybrid method

Chung-Hee Lee○ Myung-Gil Jang Young-Hoon Seo**
Electronics and Telecommunications Research Institute (ETRI)

요 약

본 논문은 다양한 형태의 웹 문서에 적용하기 위해서, 언어의 통계정보 및 후처리 규칙에 기반 하여 개선된 문장경계 인식 기술을 제안한다. 제안한 방법은 구두점 생략 및 띄어쓰기 오류가 빈번한 웹 문서에 적용하기 위해서 문장경계로 사용될 수 있는 모든 음절을 대상으로 학습하여 문장경계 인식을 수행하였고, 문장경계인식 성능을 최대화 하기 위해서 다양한 실험을 통해 최적의 자질 및 학습데이터를 선정하였고, 다양한 기계학습 기반 분류 모델을 비교하여 최적의 분류모델을 선택하였으며, 학습데이터에 의존적인 통계모델의 오류를 규칙에 기반 해서 보정하였다.
성능 실험은 다양한 형태의 문서별 성능 측정을 위해서 문어체와 구어체가 복합적으로 사용된 신문기사와 블로그 문서(평가셋1), 문어체 위주로 구성된 세종말뭉치와 백과사전 본문(평가셋2), 구두점 생략 및 띄어쓰기 오류가 빈번한 웹 사이트의 게시판 글(평가셋3)을 대상으로 성능 측정을 하였다. 성능척도로는 F-measure를 사용하였으며, 구두점만을 대상으로 문장경계 인식 성능을 평가한 결과, 평가셋1에서는 96.5%, 평가셋2에서는 99.4%를 보였는데, 구어체의 문장경계인식이 더 어려움을 알 수 있었다. 평가셋1의 경우에도 규칙으로 후처리한 경우 정확률이 92.1%에서 99.4%로 올라갔으며, 이를 통해 후처리 규칙의 필요성을 알 수 있었다. 최종 성능평가로는 구두점만을 대상으로 학습된 기본 엔진과 모든 문장경계후보를 인식하도록 개선된 엔진을 평가셋3을 사용하여 비교 평가하였고, 기본 엔진(61.1%)에 비해서 개선된 엔진이 32.0% 성능 향상이 있음을 확인함으로써 제안한 방법이 웹 문서에 효과적임을 입증하였다.

주제어: 문장경계인식, 기계학습

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 구문분석기나 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 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 생략 하는 등의 다양한 형태로 작성된 것들이 많다.

따라서 이런 비전문가가 작성한 웹 문서를 대상으로 문서검색, 질의응답, 정보 추출 등을 하기 위해서는 웹 문서의 다양성을 커버할 수 있는 문장경계인식 기술이 필요하다. 본 논문은 문어체 뿐 아니라 다양한 구어체를 처리할 수 있고, 특별히 인터넷에 있는 웹 문서에서 자주 발생하는 구두점 생략이나 띄어쓰기 오류도 커버할 수 있도록 문장경계 대상이 될 수 있는 모든 음절을 대상으로 문장경계를 인식하는 기술을 제안하다.

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 빨랐다. 기계학습 기반 분류모델은 학습데이터에 의존적이므로 학습데이터에 없거나 잘못된 정보에 의해서 오류를 발생시킬 수 있다. 본 논문은 그런 분류모델에 의한 오류를 후처리 규칙에 의해서 보정하였고 정확도가 향상되는 것을 알 수 있었다.

2. 관련연구

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 결합시켜서18개 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. 임희석, 한군희[12]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. Liu, Stolcke, Shriberg, and Harper[13]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[14]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. 박수혁, 임해창[15]은 구두점만을 문장경계 인식대상으로 고려하고, 12개의 자질을 사용해서 다양한 기계학습 분류모델을 문장경계인식기에 적용하였고, 최고 성능을 보이는 Decision Tree와 Random Forest 알고리즘을 사용해서 세종코퍼스를 대상으로 실험 하여 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, 대상 문서를 신문기사와 같이 구두점이 명확한 의미로 사용되는 것들을 대상으로 하였으므로 99% 이상의 성능을 보인다.

한국어의 경우에도 기존 연구들은 문장경계 후보로 구두점만을 대상으로 고려하고 있으며, 구두점이 모호하게 사용되는 경우가 많은 한국어의 특징 때문에 규칙보다는 기계학습 방법에 의해서 문장경계 모호성을 해소하였다

하지만, 일반인이 작성하는 웹 문서의 다양한 문장들의 경우에는 구두점이 생략되거나 띄어쓰기 오류가 빈번히 발생하므로 기존 연구를 웹 문서에 적용하기에는 무리가 있다. 이에 본 논문은 구두점 외의 문장경계에 사용되는 모든 음절을 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 최적화 시키는 것으로, 최적의 자질과 학습데이터를 선택하고, 최종적으로 최적의 분류 모델을 선택하는 일이다.

3.1 통계적 자질

3.1.1 자질 집합

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

1) 문장 경계 후보
   문장경계 모호성을 해소할 대상 후보로는 모호성 해소의 어려움에 따라 3개의 레벨로 구분하였다.
   - 1레벨: 3개의 구두점(마침표, 물음표, 느낌표)
   - 2레벨: 문장종결에 사용될 수 있는 15개 어미(다,네,오,어,지,나,군,라,니,가,까,게,자,세,요)
   - 3레벨: 문장종결에 사용된 모든 음절(문장경계 태깅 말뭉치로부터 435개 추출)
2) 문장경계 다음 음절의 공백 여부
   문장경계 모호성 해소에 문장경계 다음 음절의 공백 여부가 중요한 역할을 하기 때문에 자질로 고려하였다.

3,4) 문장경계 앞/뒤 1번째 음절
   문장경계 바로 이전과 이후에 나타난 음절 정보

5,6) 문장경계 앞/뒤 2번째 음절

문장경계 2번째 이전과 이후에 나타난 음절 정보

7,8) 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.

9,10) 문장경계 앞/뒤 1번째 토큰의 길이

문장경계 바로 이전과 이후에 나타난 토큰의 길이

3.1.2 자질 선택

가. 실험 환경

- 문장경계 후보: 1레벨의 구두점만을 대상
- 학습데이터: 신문 기사와 블로그 문서로 구성된 32,469문장
- 평가셋: 신문기사, 블로그, 에세이 등의 구어체와 문어체가 섞인 3,455문장 (Set1)
- 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 성능을 측정 하였고, 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)

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

2) 자질별 추가에 따른 성능 변화

모든 구두점을 문장경계로 인식했을 때의 기준선 성능에 비해서, 자질을 1개씩 누적해서 추가함에 따른 성능 변화를 실험하였다. 실험 결과, 자질2를 제외하고는 자질을 추가할 때마다 성능 향상이 있었고, 자질3이 가장 큰 성능 향상을 가져왔다. 자질2는 처음으로 추가된 자질로 추가 시 기준선에 비해 1.5% 성능저하가 있었지만, 기준선에 비해 Precision이 대폭 개선되었고, 추가 실험을 통해 다른 자질과 함께 사용되는 경우는 성능향상에 도움이 된다는 것을 알 수 있었다.

추가자질	Precision	Recall	F	Impr.
기준선	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%

3) 자질별 제외에 따른 성능 변화

모든 자질을 고려한 경우와 비교해서, 특정 1개의 자질을 빼는 경우에 따른 성능 변화를 측정하였다. 실험 결과, 어떤 자질을 빼도 성능 저하가 발생하므로 모든 자질이 유용함을 알 수 있었다. 특히, 문장경계 후보 바로 앞 음절에 대한 자질3의 경우에 가장 큰 성능 저하가 발생하여, 앞의 1,2,3번 실험 모두에서 중요한 자질임을 알 수 있었다.

제거자질	Precision	Recall	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. 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.

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 된다는 것을 알 수 있었고, 특히 자질3의 중요성이 확인되었다. 최종 자질로 10개 모두를 선택하였다.

3.2 최적의 분류모델 선택

다양한 기계학습 기반 분류모델을 선정해서 동일한 자질과 학습데이터를 사용해서 학습하고, 동일한 평가셋에 대해서 비교 평가를 수행하였다. 실험 환경은 3.1.2와 동일하다. 평가에 사용된 분류모델은 다음과 같다.

- ME: Maximum Entropy 모델
- SVM_light: 기본적인 SVM 모델
- SVM_p: 최적화 알고리즘으로 L-BFGS를 사용한 SVM 모델
- SVM_SGD: 최적화 알고리즘으로 Stochastic Gradient Descent(SGD)를 사용한 SVM 모델
- SVM_fsmo: Fixed-threshold SMO를 사용한 structural SVM 모델
- CRF: Conditional Random Fields 모델

실험한 결과는 아래와 같다. 알고리즘에 따른 성능 변화는 최대 1.5% 정도의 미미한 차이가 있었지만, 최고 성능을 낸 SVM_fsmo를 최종 분류 모델로 선정하였다.

Classifier	Prec.	Recall	F-measure
ME	0.969	0.947	0.958
SVM_light	0.976	0.953	0.965
SVM_p	0.974	0.955	0.965
SVM_SGD	0.934	0.967	0.950
SVM_fsmo	**0.976**	**0.954**	**0.965**
CRF	0.968	0.946	0.957

3.3 학습말뭉치에 따른 성능 비교

3.2까지의 실험을 통해 최적의 자질과 분류모델을 선정하였다. 이렇게 선정된 기본 문장경계 인식기를 사용해서 이번에는 학습말뭉치에 따른 성능 변화를 실험하였다.

가. 실험 환경

- 문장경계 후보: 1레벨의 구두점만을 대상
- 분류 모델: SVM_fsmo
- 평가셋: 신문기사, 블로그, 에세이 등의 구어체와 문어체가 섞인 3,455문장 (Set1)

나. 학습데이터별 성능 평가

학습데이터는 아래와 같이 5가지를 대상으로 실험하였다.

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

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

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 평가셋에 따른 성능 비교

3.3까지의 실험을 통해 최적의 자질과 학습데이터, 그리고 분류모델을 선정하였다. 이렇게 선정된 기본 문장경계 인식기를 사용해서 이번에는 평가셋의 차이에 따른 성능을 비교하였다. 비교를 위해 사용된 평가셋은 아래와 같다.

- 평가셋1: 신문기사, 블로그, 에세이 등의 구어체와 문어체가 섞인 3,455문장 (Set1)
- 평가셋2: 주로 문어체로 구성된 백과사전 본문과 세종말뭉치의 1,775문장 (Set2)

각각의 평가셋 자체의 난이도를 보기 위해서 모든 구두점을 문장경계로 인식하는 기준선 성능도 측정하였고, 평가셋별 결과는 아래와 같다.

가) 평가셋1 (Set1)

	Prec.	Recall	F
기준선	0.775	1.0	0.873
SVM_fsmo	0.976	0.954	0.965

나) 평가셋2 (Set2)

	Prec.	Recall	F
기준선	0.910	1.0	0.953
SVM_fsmo	0.994	0.995	0.995

실험결과, 대부분 문어체로 구성된 평가셋2가 평가셋1보다 성능이 좋았다. 기준선 성능 평가를 통해 평가셋1이 문장경계가 모호하게 사용되는 경우가 많다는 것을 알 수 있었다. 기존 연구의 경우에 대부분 98% 이상의 성능을 보이는 것도, 문어체 문장 위주로 학습하고 평가하였기 때문이며, 4장의 개선된 문장경계 인식기에 대한 실험을 통해 기존 연구가 웹 문서와 같이 구어체가 훨씬 많은 문서에서는 성능이 떨어짐을 알 수 있다.

3.5 규칙 후처리에 의한 성능 개선

마지막 실험으로, 이번에는 기계학습 기반 분류모델의 자주 발생하는 오류에 대해서 규칙에 의해 후처리하는 기능을 추가한 후 성능을 평가하였다. 사용된 규칙은 아래와 같다.

- 규칙1: 괄호 사이에서 문장경계인식 금지
- 규칙2: 따옴표 사이에서 내부 문장경계인식 금지

	Prec.	Recall	F
SVM_fsmo	0.921	0.983	0.951
+규칙	**0.994**	**0.983**	**0.989**

실험 결과, 분류모델의 오류로 과생성 되던 문장들이 규칙 후처리를 통해 없어지면서 precision이 대폭 향상되었고, 그에 따라 전체 성능도 3.8% 향상되었다.

4. 개선된 문장 경계 인식

웹 문서에 대한 문장경계인식 성능 측정을 위해서 웹 사이트 게시판 등의 웹 문서로부터 수집된 295개의 문장들로 구성된 3차 평가셋을 만들었다. 주로 구어체로 구성된 3차 평가셋의 구두점 생략 정도를 확인하기 위해서 모든 구두점을 문장경계로 인식한 기준선 성능을 측정하였고, 결과는 아래와 같다.

	Prec.	Recall	F
기준선	0.848	0.615	0.713

문장경계로 구두점이 대부분 사용되는 평가셋1과 평가셋2에 비해서 생략된 구두점이 많기 때문에 Recall이 상당히 떨어졌다.

4.1 웹 문서에 대한 기본 모델 실험

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

가. 실험 환경
- 문장경계 후보: 3레벨의 모든 음절
- 분류 모델: SVM_fsmo
- 학습데이터: A2+B2
- 평가셋: 평가셋3

나. 실험 결과

이번 실험을 위해서 구두점만을 문장경계후보로 고려하지 않고, 학습데이터에서 문장경계에 사용됐었던 모든 음절을 대상으로 문장경계를 인식하도록 기본 모델을 수정해서 실험하였고 결과는 아래와 같다.

	Prec.	Recall	F
기본모델	0.563	0.667	0.611

기본 모델은 구두점만을 대상으로 분류하도록 학습이 되었기 때문에, 기준선보다도 성능이 더 낮았다. 이번 실험을 통해, 본 연구의 기본 모델을 포함해서 기존의 연구결과를 웹 문서에 그대로 적용하기에는 무리가 있음을 알 수 있었다.

4.2 웹 문서를 위한 개선된 문장경계 인식

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

- 튜닝1: 자질1의 문장경계후보에 종결어미 및 문장경계로 사용된 모든 음절을 추가하여 추가 학습. 문장경계의 부정자질로 각 후보의 문장경계로 사용되지 않은 경우에 대해서도 추가 학습
- 튜닝2: 웹 문서의 경우에 한 라인에 한 문장을 적는 경우가 많으므로, 뉴라인캐릭터를 음절 및 토큰 자질에 추가
- 튜닝3: 3.3절의 실험을 통해 일반 문서의 경우에 A2와 B2의 학습데이터를 사용하는 것이 가장 성능이 좋았지만, 웹 문서에 대한 성능평가에서는 B1과 B2를 사용하는 것이 가장 성능이 좋았으므로 학습데이터를 B1,B2로 교체함
- 튜닝4: 문장경계후보의 생성 규칙 수정 기본 모델에서는 대상 문서의 문장경계후보를 한꺼번에 생성한 후, 앞/뒤 문장을 이용해서 대상 후보의 모호성을 해소 하였지만, 웹 문서의 경우에는 레벨2,3의 후보를 모두 대상으로 하기 때문에 생성되는 후보가 근접해서 많이 발생하므로, 앞에서부터 1개씩 후보를 생성하면서 모호성을 해소하도록 수정함
- 튜닝5: 근접해서 발생하는 문장경계 후보들의 경우, 우선순위에 따라 1개를 삭제함 (우선순위: lv1>lv2>lv3)

위와 같은 5가지 작업에 따른 성능 비교를 다음과 같이 실험하였다.

가. 실험 환경 (최종 모델)
- 문장경계 후보: 3레벨의 모든 음절
- 분류 모델: SVM_fsmo
- 학습데이터: B1+B2
- 평가셋: 평가셋3

나. 실험 결과

	Prec.	Recall	F	Impr.
기준선	0.848	0.615	0.713	0.0%
기본모델	0.563	0.667	0.611	-10.2%
튜닝1	**0.959**	**0.727**	**0.827**	**21.6%**
튜닝2	0.967	0.729	0.831	0.4%
튜닝3	0.967	0.744	0.841	1.0%
튜닝4	**0.957**	**0.888**	**0.921**	**8.0%**
튜닝5	0.968	0.896	0.931	1.0%

최종 실험 결과, 개선된 모델에 사용된 모든 튜닝작업이 성능 향상에 기여 하였고, 개선모델이 기준선에 비해서는 22%, 기본모델에 비해서는 32%가 향상되었다. 가장 크게 성능을 향상시킨 튜닝1의 경우에, 레벨2,3의 문장경계 후보의 경우에 구두점에 비해서 다른 의미로 사용되는 경우가 훨씬 많기 때문에 각 후보의 부정 자질을 추가로 학습한 것이 크게 도움이 된 것으로 분석되었다. 두 번째로 성능 향상에 도움이 된 튜닝4의 경우에는 구두점에 비해서 근접해서 많이 발생하는 레벨2,3의 문장경계 후보를 고려해서 문장경계인식 알고리즘을 수정한 것이 성능 향상에 도움이 되었다.
결론적으로 4장의 최종 실험으로부터 웹 문서의 경우에는 기존 문장경계인식 기술로는 성능이 크게 떨어짐을 확인하였고, 본 논문에서 제안한 방법이 성능 향상에 효과적임이 입증되었다.

5. 결론

본 논문에서는 1차 실험으로 다양한 실험을 통해, 일반 문서의 구두점 대상 문장경계 모호성 해소를 위해서 최적화된 자질이 무엇인지 확인하고, 학습데이터에 따른 성능 차이가 있음을 확인하였고, 기계학습 기반 분류모델의 경우에는 모델 간 성능 차이가 크지 않음을 확인하였다.
2차 실험에서는 1차 실험을 통해 최적화된 기본 문장경계 인식기를 구두점 생략이 빈번한 웹 문서에 적용하였고, 실험 결과로부터 기본 모델을 웹 문서에 그대로 적용하기에는 무리가 있음을 알 수 있었다. 그러므로 웹 문서의 특징을 반영하도록 본 논문에서 제안한 개선된 모델이 웹 문서의 문장경계 인식에 효과적이었다.
향후 연구로는 현재의 10개 자질 외에 더욱 효과적인 자질을 찾아보고, 문장경계 인식에 더욱 최적화된 분류모델이 있는지 찾아보고, 웹 문서의 추가 학습데이터 구축 및 규칙 추가를 통한 성능 개선을 고려하고자 한다.

참고 문헌

[1] 박수혁,임해창, "기계학습 기법을 이용한 문장경계인식" 한국정보처리학회 춘계학술발표대회 논문집 제15권 제1호, pp.122-122, 2008.
[2] Changki Lee, and Myung-Gil Jang, "Fast Training of Structured SVM Using Fixed-Threshold Sequential Minimal Optimization," ETRI Journal, vol.31, no.2, Apr. pp.121-128, 2009.
[3]G.Grefenstette and P.Tapanainen, "What is a word, what is a sentence? problems of tokenization," In Proceedings of the 3rd International Conference on Computational Lexicography, pp. 79-87, 1994.
[4]John 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]Stamatatos,Fakotakis,and Kokkinakis, "Automatic extraction of rules for sentence boundary disambiguation, " In ACAI'99, 1999.
[6]Riley and Michael, "Some Applications of Tree-based Modeling to Speech and Language Indexing," In Proceedings of the DARPA speech and natural language workshop, pp. 339-352, 1989.
[7]D.D.Palmer and M.A.Hearst, "Adaptive sentence boundary disambiguation", Proceedings of the fourth conference on Applied natural language processing, 1994.
[8]D.D.Palmer and M.A.Hearst, "Adaptive Multilingual Sentence Boundary Disambiguation," Computational Linguistics, 23(2), pp. 241-267, 1997.
[9]J.C.Reynar and A.Ratnaparkhi,"A Maximum Entropy Approach to Identifying Sentence Boundaries," In Proceedings of the Fifth Conference on Applied Natural Language Processing, pp. 16-19, 1997.
[10]Mikheev, A. 2000. Tagging Sentence Boundaries. In NACL'2000 (Seattle) ACL, pp. 264 - 271, 2000.
[11]Haoyi Wang and Yang Huang, "Bondec -- A Sentence Boundary Detector", URL:http://nlp.stanford.edu/courses/cs224n/2003/fp/huangy/final_project.doc, 2003
[12]임희석, 한국희, "메모리 기반의 기계 학습을 이용한 한국어 문장 경계 인식", 한국콘텐츠학회논문지, Vol.4 No.4, 2004.
[13]Y.Liu, A.Stolcke, E.Shriberg, and M.Harper, "Using conditional random fields for sentence boundary detection in speech", ACL '05, 2005
[14]Tibor Kiss, and Jan Strunk, "Unsupervised Multilingual Sentence Boundary Detection", Computational Linguistics , Volume 32 Issue 4 , 2006.
[15]박수혁, 임해창, "기계학습 기법을 이용한 문장경계인식", 한국정보처리학회 춘계학술발표대회 논문집 제15권, 제1호, 2008.