ISSN 2383-630X(Print) / ISSN 2383-6296(Online)
Journal of KIISE, Vol. 43, No. 6, pp. 636-644, 2016. 6
http://dx.doi.org/10.5626/JOK.2016.43.6.636

커널 Ripple-Down Rule을 이용한 태깅 말뭉치 오류 자동 수정

(Automatic Correction of Errors in Annotated Corpus Using Kernel Ripple-Down Rules)

박 태 호 †　　차 정 원 ††
(Tae-Ho Park)　　(Jeong-Won Cha)

요 약 자연어처리에서 기계학습을 위한 학습 말뭉치는 매우 중요하다. 정제된 대량의 말뭉치는 자연어처리 시스템에 직접 영향을 준다. 본 논문에서는 대량의 말뭉치 오류를 자동으로 수정하는 새로운 방법을 제안한다. 오류 말뭉치와 정답 말뭉치에서 사람이 태깅한 문서의 특성을 반영한 수정 규칙을 자동으로 생성하였다. 수정 규칙은 RDR(Ripple-Down Rules)을 사용하여 표현하였다. 수정 방법의 가치를 보이기 위해 품사 부착 말뭉치와 개체명 부착 말뭉치에 대해서 실험하였으며 두 분야에서 유의미한 결과를 보였다. 이 방법은 대량의 말뭉치를 제작할 때 오류를 최소화하는 방법으로 사용이 가능하다.

키워드: 형태 분석 말뭉치, 오류 수정, 커널 RDR, 자연어처리

Abstract Annotated Corpus is important to understand natural language using machine learning method. In this paper, we propose a new method to automate error reduction of annotated corpora. We use the Ripple-Down Rules(RDR) for reducing errors and Kernel to extend RDR for NLP. We applied our system to the Korean Wikipedia and blog corpus errors to find the annotated corpora error type. Experimental results with various views from the Korean Wikipedia and blog are reported to evaluate the effectiveness and efficiency of our proposed approach. The proposed approach can be used to reduce errors of large corpora.

Keywords: morphological annotation corpora, error correction, kernel RDR, natural language processing

· 이 논문은 2015~2016년도 창원대학교 자율연구과제 연구비 지원으로 수행된 연구결과임

† 비 회 원 : 창원대학교 컴퓨터공학과
taehope@changwon.ac.kr
†† 종신회원 : 창원대학교 컴퓨터공학과 교수
(Changwon National Univ.)
jcha@changwon.ac.kr
(Corresponding author임)

논문접수 : 2015년 10월 6일
(Received 6 October 2015)
논문수정 : 2016년 3월 29일
(Revised 29 March 2016)
심사완료 : 2016년 3월 30일
(Accepted 30 March 2016)

1. 서 론

기계학습을 이용한 문제해결이 점점 더 많은 분야로 확대되고 있다. 기계학습에서 예측 성능은 학습 데이터의 오류에 영향을 받는다. 따라서 학습 데이터의 오류를 줄이기 위한 방법에 대해서 많은 연구들이 있었다[1-9].

자연어처리에서도 통계 정보에 기반하는 방법과 기계학습을 이용하는 방법이 주류를 이루고 있다. 이 두 방법에서는 학습을 위한 말뭉치가 중요한 역할을 담당한다. 하지만 학습 말뭉치는 작성하는데 시간과 비용이 많이 요구된다. 이러한 이유로 인해 지도학습(supervised learning)을 대체하는 비지도학습(unsupervised learning)이나 반지도학습(semi-supervised learning)에 대한 연구가 많이 진행되었다.

비지도학습과 반지도학습의 성공적인 결과에도 불구하고 학습을 위한 정보부착 말뭉치의 중요성은 줄어들

지 않고 있다. 왜냐하면 정보부착 말뭉치를 가공하여 활용할 수 있는 분야가 점점 증가하고 있기 때문이다.

하지만 대량의 말뭉치를 제작하려고 하면 다수의 사람들이 작업을 하기 때문에 일관성 있는 말뭉치를 제작하기가 매우 어렵다. 세종 말뭉치에 대한 다수의 오류 수정 논문들이 이를 증명한다[1,2].

기계학습에서 학습 데이터의 오류 수정에 관한 논문들은 다음과 같다. [3]은 학습 데이터의 오류를 다음과 같이 크게 두 가지로 분류하였다. 1) 속성 오류(attribute noise), 2) 범주 오류(class noise). 속성 오류는 속성값을 입력하는 중에 발생하는 오류이다. 여기에는 속성이 없거나 중복된 값이 있는 경우가 포함된다. 범주 오류는 다시 1)일관성 오류와 2)분류 오류로 나누어진다. 일관성 오류는 같은 데이터가 다른 범주로 분류된 경우이다. 분류 오류는 범주가 잘 못 할당된 경우이다.

[4,5]는 대용량, 분산 데이터에 초점을 맞추었다. 그들은 학습을 수행하는 데이터를 분류 알고리즘이 처리 가능한 보다 작은 크기로 분할(partition)하였다. 분할된 데이터 집합 각각에 대해서 분류기를 생성하였다. 전체 데이터 집합에서 오류로 판명되면 빈도수를 증가시키게 되고 높은 확률을 가지는 개별 항목은 최종 오류가 된다. 저자는 오류 항목을 발견하고 삭제하기 위해 최대치(majority)와 일치(non-objection) 전략을 사용하였다.

[6]에서는 최대 정보량 기준을 사용하였다. [7]은 포화 필터(saturation filter)라고 불리는 방법을 사용하였다. [4,8]은 C4.5를 사용하여 잠재적으로 오류가 될 항목을 구별하였다. [9]는 인공 신경망을 사용하였다.

이들은 모두 현재 학습 데이터에서 오류로 인식된 부분을 제거하여 학습 데이터의 일관성을 유지하는 것에 초점이 맞춰져 있다. 본 논문에서는 오류들을 최대한 수정하여 데이터를 확보할 수 있는 방법을 제안한다.

1.1 RDR(Ripple-Down Rules)

RDR(Ripple-Down Rules)은 1993년 Edwards와 Compton이 화학 병리학 보고서에 대한 병리학 유지 보수 시스템[10]에 처음 도입하였다. 이후 환자 관리에 도움을 주고자 RDR을 사용하여 화학 보고서에 주석을 작성하는 작업을 수행하였다[11]. 이 과정에서 RDR은 지식 기반 시스템을 구축하도록 수정되며 SCRDR(Single Classification Ripple Down Rules)과 MCRDR(Multiple Classification Ripple Down Rules), NRDR(Nested Ripple Down Rules) 등 다양한 형태의 RDR이 만들어졌다[12]. SCRDR은 입력된 값으로부터 하나의 결과를 출력하며, MCRDR은 한 개 이상의 결과를 출력한다. NRDR은 사용자가 정의한 임의의 조건에 따라 결과를 출력한다. 형태소 품사 태그 또는 개체명 태그 오류를 수정하는 작업을 수행하기 위해서는 입력된 오류 태그를 올바른 하나의 정답으로 수정할 수 있는 SCRDR이 적합하다. 따라서 본 실험에서는 SCRDR을 이용한 시스템을 구축하였다.

SCRDR은 최상위 루트 노드로부터 조건에 따라 “EXCEPT”와 “FALSE”가 발생하고 노드를 이동하며 분류가 결정된다. 본 논문에서 사용한 SCRDR의 규칙은 재귀적으로 정의되며 각 노드는 “EXCEPT”와 “FALSE”의 후속 노드와 연결되어 있다[13]. 관측대상은 루트 노드로부터 조건에 따라 노드를 이동하며 최종적으로 “ACCEPT”된 규칙에 의해 분류된다[14].

RDR과 의사결정트리는 각각 지도 학습의 한 방법으로 주어진 데이터를 분류하는 문제를 다룬다. 두 방법 모두 이미 알고 있는 정답셋을 이용하여 분류 방법을 생성한 후 주어진 대상의 정답을 예측하고, 최상위 루트 노드로부터 그 하위로 나뉘어 내려가며 노드를 생성한다. 하지만 두 방법에는 차이점이 존재한다. 의사결정트리는 데이터를 분류하기 위한 최적의 변수를 찾기 위해 확률 또는 통계 데이터를 이용하고 그 변수를 기준으로 트리를 생성하며 생성된 노드를 최적화하기 위해서 노드를 병합하는 과정을 거친다. 이와 다르게 RDR의 경우, 먼저 하나의 규칙이 생성되면 그 규칙이 가질 수 있는 예외, 즉, “EXCEPT”가 발생할 수 있는 경우에 대해서 예외적인 처리를 생성하고 더 이상의 예외가 발생하지 않고 새로운 규칙을 생성할 때, “FALSE”가 발생하여 새로운 예측변수에 대한 규칙을 생성한다. 또한 정답을 결정하는 과정에서 의사결정트리는 관측대상이 가진 변수들이 생성할 수 있는 확률 또는 통계값에 의해 정답이 결정되고, RDR의 경우에는 관측대상이 가진 변수를 다루는 모든 규칙을 확인하며 가장 마지막에 “ACCEPT”한 규칙에 의해 분류된다.

본 논문은 정답 말뭉치의 오류를 수정하여 학습 말뭉치의 성능을 높이고자 한다. 시스템 결과에서 나타나는 오류는 일정한 규칙을 띄고 높은 빈도로 나타난다. 따라서 시스템 결과 오류를 수정하기 위한 규칙을 쉽게 생성할 수 있다. 하지만 사람이 직접 태그를 부착한 말뭉치에서 나타나는 오류는 패턴이 불규칙하며 그 수가 적다. 본 실험은 사람이 직접 작성한 말뭉치에서 적은 수로 나타나는 패턴을 찾기 위해서 RDR 시스템에 커널이 동작할 수 있도록 하였으며, 커널만 교체하면 다양한 태그 부착 말뭉치에 동작할 수 있도록 설계하였다.

본 논문의 구성은 2장에서 제안 방법에 대한 설명을 하고 3장에서는 각 경우에 대한 실험 결과를 서술하고 분석한다. 마지막으로는 4장에서 결론을 내린다.

2. 제안방법

본 연구에서는 다수의 연구자들이 손으로 작성한 태

그 부착 말뭉치를 '초벌 말뭉치(Initial corpus)'라고 명명하고 이를 사람이 보완한 코퍼스를 '정답 코퍼스(Gold corpus)'라고 명명한다. 이 둘을 RDR로 학습하여 자동으로 태그 부착 코퍼스의 오류를 수정하는 방법을 제시한다.

2.1 실험 준비

기존의 RDR 학습 방법은 초벌 코퍼스와 정답 코퍼스를 비교하여 같은 위치에 서로 다른 태그가 부착되어 있는지 검토한다. 만약 태그가 다른 경우 오류가 정답으로 고쳐질 수 있도록 정답 말뭉치에서 이전 2개의 형태소와 다음 2개의 형태소의 패턴을 추출한다[15]. 하지만 영어 문서를 기준으로 만들어진 기존의 RDR은 한국어에서 사용하는 형태소와 어절 정보가 그대로 적용되기 힘든 점이 있다. 따라서 한국어 형태소 품사 태깅에서 나타나는 특징과 개체명 태깅에서 나타나는 특징을 분석하고 학습할 수 있도록 시스템을 변형하였다. 변형된 시스템은 입력된 한국어 문장을 학습과 평가할 수 있도록 형태소 단위의 분석이 가능하게 하였다. 또한 커널을 통해 한국어에 적합한 패턴을 추출할 수 있도록 수정하였다.

2.1.1 한국어 형태소 품사 태깅의 성질

한국어 형태소 품사 태깅은 영어와 다르게 하나의 어절이 2개 이상의 형태소로 분석될 수 있다. 따라서 기존의 RDR 학습 방법을 그대로 적용할 경우, 오류 문장과 정답 문장 간에 형태소 수가 달라지는 문제가 발생한다. 잘못 분석된 형태소로 인해 오류 문장과 정답 문장의 형태소 수가 다를 경우 이를 학습하기 위해서 학습되는 문장의 형태를 변형하여 학습할 필요가 있다. 표 1[1]에서처럼 "닌텐도"라는 명사가 "닌텐"과 "도"로 분리되어 분석된 오류가 나타날 경우, 해당 어절의 형태소들을 하나의 어절로 묶어 "닌텐/NNG+도/JX"로 학습할 수 있도록 시스템이 변형하여 정보를 분석한다. 또한 학습을 통해 생성된 규칙이 단어는 다르지만 같은 유형의 오류에 적용될 수 있도록 의미형태소[2]를 삭제하고 품사 태그만을 이용하는 방법을 사용하였다.

표 1 형태소 수가 다를 때, 학습 방법

Table 1 Corpus modification when the number of morphemes is different

	Error morp	Correct morp
Before Modified	닌텐/NNG	닌텐도/NNP
	도/JX	
Modified	닌텐/NNG+도/JX	닌텐도/NNP
Training	NNG+도/JX ⇨ NNP	

1) NNG:일반명사, NNP:고유명사, JX:보조사

2) 어휘적 의미가 있는 형태소로 일반적으로 명사, 동사, 형용사 등이 있다.

2.1.2 개체명 태깅의 성질

개체명 태깅 오류의 유형은 형태소의 오류 유형과는 다른 형태를 보인다. 형태소 오류는 동일한 형태소에 다른 품사가 부착된 오류이다. 이는 일반적으로 오류 유형을 삽입, 삭제, 치환으로 분류할 때에 치환에 해당되는 오류 유형이다. 하지만 개체명 태그 오류의 경우에는 삽입, 삭제, 치환의 모든 오류 유형이 나타난다. 삽입은 개체명이 아니지만 잘못 인식한 개체명, 삭제는 개체명 미인식, 치환은 개체명으로 인식은 했지만 잘못된 태그를 부착하거나 잘못된 경계를 가지는 경우로 분류된다. 따라서 개체명 태그 오류의 유형을 표 2와 같이 분류하였다.

유형 1은 개체명을 인식했으나 잘못된 개체명 태그를 부착해 발생한 오류이고, 유형 2는 개체명이 아닌 단어를 개체명으로 인식한 오류이다. 유형 1과 2는 개체명 태그 분류의 오류라고 정의한다. 그리고 유형 3은 개체명인 단어를 인식하지 못해 발생하는 오류이고, 유형 4는 개체명의 일부분만을 인식해 발생하는 오류이다.

본 논문에서 구축한 RDR 시스템은 이미 부착된 태그 정보를 수정하는 기능만을 수행하기 때문에 개체명의 미인식 오류와 인식 범위 오류 유형인 유형 3과 4는 실험에서 제외하였다. 초벌 말뭉치에서 나타나는 개체명 오류 유형별 오류 수는 표 3과 같다.

표 2 개체명 말뭉치에서 나타나는 오류 유형

Table 2 Error types of NE-tagged corpus

type	explain
1	Different named entity with same word
2	Named entity in initial corpus
3	Named entity in gold corpus
4	Named entity with different boundary

표 3 개체명 초벌 말뭉치의 오류

Table 3 Errors of initial test corpus annotated with NEs

type	# of errors
1	72
2	88
3	47
4	11

2.2 커널 RDR(Kernel RDR)

본 논문에서는 RDR에 커널 시스템을 추가로 구현하였고 학습 말뭉치에 따라 커널을 변경해 사용할 수 있도록 하여 범용성을 높였다. 또한 한국어의 형태소와 어절의 특성에 맞게 학습을 할 수 있도록 시스템을 구축하였다. RDR학습 시, 커널 시스템으로 입력한 정보에 따라 패턴을 분석하고 규칙을 생성한다. 본 시스템은 커널을 통해 패턴을 분석할 수 있도록 하였으며, 형태소

표 4 형태소 품사 태그 커널과 개체명 태그 커널
Table 4 Kernels according to template

	Windows		Kernel
Morp Feature	Morpheme	-3	morpheme/POS tag
		-2	morpheme/POS tag
		-1	morpheme/POS tag
		0	morpheme/POS tag
		+1	morpheme/POS tag
		+2	morpheme/POS tag
		+3	morpheme/POS tag
	Eojeol	-1	First POS morpheme in eojeol/ First POS tag in eojeol
		-1	Last POS morpheme in eojeol/ Last POS tag in eojeol
		+1	First POS morpheme in eojeol/ First POS tag in eojeol
		+1	Last POS morpheme in eojeol/ Last POS tag in eojeol
Neamd Entity Feature	Morpheme	-2	morpheme/POS tag
		-1	morpheme/POS tag
		0	morpheme/POS tag
		1	morpheme/POS tag
		2	morpheme/POS tag
	Eojeol	-1	eojeol morpheme
		+1	eojeol morpheme
	NE	0	Named-Entity belonging to the current morpheme

품사 태깅 말뭉치와 개체명 태깅 말뭉치 각각에 맞는 커널을 생성하였다(표 4).

형태소 품사 태깅 말뭉치 학습에 사용되는 커널은 형태소와 형태소 품사 태그가 있다. 이를 커널로 사용하기 위해 커널을 추출하는 기준을 형태소 단위와 어절 단위 모두 사용하였다. 형태소는 이전 어절의 첫 형태소, 마지막 형태소와 다음 어절의 첫 형태소, 마지막 형태소를 사용하였다. 또한 어절을 무시하고 이전의 3개 형태소와 다음의 3개 형태소를 사용하였다. 품사는 형태소와 동일하게 사용하였다.

개체명 품사 태깅 말뭉치 학습에 사용되는 커널은 이전 어절의 형태소와 다음 어절의 형태소를 사용하였다. 또한 어절을 무시하고 이전의 2개의 형태소와 다음의 2개의 형태소를 사용하였다.

2.3 임계치(Threshold) 결정

RDR에서 학습을 통해 규칙을 생성할 때, 사용자가 지정한 임계치(Threshold)에 따라 규칙이 다르게 생성된다. RDR에는 ImproveThresold와 MatchThreshold, 두 가지의 임계치가 있다. ImproveThreshold는 어떠한 태그가 다른 태그로 수정되는 수에 대한 임계치이며, MatchThreshold는 ImproveThreshold를 만족하는 오류 쌍 중에 동일하게 나타나는 패턴에 대한 임계치이다. 예를들어 ImproveThresold가 3이고, MatchThreshold가 2일 경우, A라는 태그가 B라는 태그로 수정될 때, 1번 패턴에 의해서 수정되는 수가 4번, 2번 패턴에 의해서 수정되는 수가 1번이라고 하자. 이 경우 수정되는 총 수는 5번이므로 ImproveThreshold 이상이고 1번 패턴이 MatchThreshold 이상이므로 1번 패턴이 학습된다.

결정된 임계치는 말뭉치에 존재하는 다양한 오류에 동일하게 작용되기 때문에 오류를 수정하기에 가장 적합한 임계치를 찾는 작업이 중요하다. 따라서 본 실험은 오류 수정에 적합한 임계치를 결정할 수 있도록 다양한 임계치를 적용하여 실험하였다.

2.4 학습 조건

기존의 RDR 학습 방법은 초벌 코퍼스와 정답 코퍼스를 비교하여 태그가 다른 경우에만 학습하였다. 이는 오류가 발생하는 위치에서 규칙을 학습하는 원리이다.

하지만 임계치가 낮으면 오류를 수정하는데 부적절한 규칙이 학습될 수도 있다. 이는 오류가 발생했을 때만 나타나는 패턴이 아닌 일반적으로 나타나는 패턴을 학습하기 때문이다. 이러한 규칙은 오류가 아닌 상황에도 적용되어 시스템이 오류를 발생시키는 문제가 있다.

하지만 문서 전체를 학습하게 되면 오류를 발생하는 규칙을 다시 정답으로 되돌리는 규칙이 학습될 수 있다. 따라서 본 논문에서는 문서 전체를 학습하도록 하여 규칙을 생성하였다.

3. 실험 및 토의

제안된 방법의 효용성을 보이기 위해서 다양한 실험을 진행하였다. 먼저 오류만 학습하는 방법과 정답을 포함한 학습의 차이점을 확인하였다. 이후로는 다양한 임계치(threshold)를 학습량에 따라 실험을 진행하였다.

3.1 실험 환경

본 논문에서는 두 개의 문서 그룹을 만들어 실험하였다. 이것은 태깅 그룹 내에서 수정 성능과 태깅 그룹 간의 수정 성능을 조사하기 위해서이다.

표 5와 같이 한국어 위키피디아 문서는 작업 그룹A에서 태깅했고, 블로그 문서는 작업 그룹B에서 태깅하였다. 그룹A와 그룹B의 작업자가 발생시키는 오류의 유형은 서로 다르다. 따라서 초벌 말뭉치인 위키피디아(A) 문서와 블로그(B)문서는 서로 다른 오류를 포함하고 있다. 문서를 두 가지로 구분함으로써 학습량과 임계치의 변화에 대한 결과를 각각 확인할 수 있으며, 학습된 규칙을 교차 검증할 수도 있다. 3.3과 3.4에서는 그룹내의 오류 수정과 그룹간의 오류 수정에 대한 실험을 진행하였다. 형태소 품사 부착 말뭉치는 세종 태그셋을 따르며, 개체명 말뭉치는 ETRI 태그셋을 따른다.

표 5 작업 그룹과 말뭉치
Table 5 Tagging group and corpus

	Init corpus	Gold corpus
Tagging Group A	Wikipedia (A)	Wikipedia (B)
Tagging Group B	Blog (C)	Blog (D)

표 6 형태소 품사 태그 평가 코퍼스의 성능
Table 6 Init performance of POS tag to the test corpus

Init Corpus	sentence	# of morphemes	# of errors	Accuracy (%)
Wikipedia	10,000	260,561	8,776	96.63
Blog	4,000	102,388	1,931	98.11

표 7 개체명 평가 코퍼스와 성능
Table 7 Init performance of NE to the test corpus

Init Corpus	sentence	# of NEs	# of errors	Accuracy (%)
Blog	4,000	7,709	160	97.74

학습을 위해서 형태소 품사 태깅 말뭉치는 한국어 위키피디아 문서 중 학습을 위해서 4,000문장을 사용하고 평가를 위해서 10,000문장을 사용하였다. 블로그 문서는 학습을 위해서 총 4,000문장을 사용하고 평가를 위해서 4,000문장을 사용하였다. 평가에 사용한 위키 10,000문장과 블로그 4,000문장의 오류수와 형태소 단위 성능은 표 6과 같다.

개체명 태깅 말뭉치는 유형 1과 2에 대한 오류에 대해서만 개체명 태그 수정을 하였다. 표 7은 평가 말뭉치로 사용하는 블로그 문서에서 추출한 4,000문장에 대한 성능표이다. 평가 말뭉치에서는 유형 1은 72개, 유형 2는 88개의 오류를 각각 포함해 모두 160개의 오류를 포함하고 있다(표 3).

3.2 학습 조건 선택 실험

본 실험에서는 오류만을 학습하는 경우(Err)와 정답을 포함한 모든 내용을 학습하는 경우(All)에 성능에서 어떤 차이가 있는지 확인하는 실험을 진행하였다.

두 실험에 대한 검증은 형태소 품사 태깅 말뭉치를 사용하여 ImproveThreshold가 0에서부터 3까지, matchThreshold가 1일 때의 성능을 비교하였다. 표 8은 위키

표 8 형태소의 오류 학습과 전체 학습 성능 비교
Table 8 Performance comparison over the errors learning only and overall learning (Part-Of-Speech)

Learning \ Threshold		0-1	1-1	2-1	3-1
wiki 4000	Err	64.55	64.55	66.98	67.64
	All	**97.23**	**97.23**	**97.41**	**97.76**

표 9 개체명의 오류 학습과 전체 학습 성능 비교
Table 9 Performance comparison over the errors learning only and overall learning (Named Entity)

Learning \ Threshold		0-1	1-1	2-1	3-1
Blog	Err	47.99	47.99	48.00	49.21
	All	**98.50**	**98.50**	**98.51**	**97.06**

피디아 문서로 실험하였고, 표 9는 블로그 문서로 실험하였다. 두 실험 결과로 알 수 있듯 오류만 학습하는 경우(Err)보다 정답을 포함하여 학습하는 경우(All)가 성능이 좋게 나타났다. 그 이유는 임계치가 낮을 경우, 오류를 수정하는 특수한 패턴이 아닌 일반적으로 나타나는 패턴을 학습하기 때문이다.

따라서 본 실험에서는 형태소 품사 태그와 개체명 태그 오류 수정을 위해 모든 실험을 정답을 포함하는 방법(All)으로 학습하였다.

3.3 학습량과 임계치에 따른 성능 평가

본 연구는 소량의 학습으로도 학습 말뭉치를 수정할 수 있는지 또한 실험하였다. 이를 확인하기 위해 RDR 학습에 사용되는 말뭉치의 양을 다르게 하여 각각의 성능을 비교하였다. 이와 함께 임계치를 다양하게 적용하여 각 임계치별로 성능이 어떻게 다른지 또한 확인하였다.

RDR을 이용한 한국어 형태소 품사 태그 오류 수정에서 학습량에 따른 성능을 확인하기 위해서 위키피디아 문서와 블로그 문서를 각각 1,000문장에서 4,000문장까지 1,000문장 단위로 나누어 학습하였다. 또한 학습 문서에서 최적의 규칙을 생성할 수 있는 임계치를 찾기위해 임계치를 다양하게 변경하여 실험하였다. 표 10-14에서 각 행 처음의 숫자는 임계치를 나타내며 '-' 기호 앞의 숫자는 ImproveThreshold를 나타내고, 뒤의 숫자는 MatchThreshold를 나타낸다. 실험 결과 위키피디아 문서를 학습하여 같은 위키피디아 문서를 평가했을 경우, 1,000문장을 학습한 규칙은 임계치에 따라 최대 5,121개의 오류를 감소시켰고, 2,000문장과 3,000문장을 학습한 경우에는 최대 5,349개의 오류를 수정시켰으며, 4,000문장에서는 5,482개의 오류를 수정하였다(표 10). 블로그 문서에서는 1,000문장을 학습한 규칙은 최대 851개의 오류를 감소시켰고, 2,000문장에서는 839개, 3,000문장에서는 840개, 4,000문장에서는 853개의 오류를 감소시켰다(표 11).

개체명 태그 오류 수정의 RDR 학습에는 2,000문장에서 4,000문장까지 1,000문장 단위로 나누어 학습하였다. 2,000문장과 3,000문장, 4,000문장을 학습하여 각각 35개, 40개, 50개의 규칙이 생성되었다. 2,000문장 학습에서는 최대 56개의 오류를 감소시켰고, 3,000문장에서는

표 10 위키피디아 말뭉치 학습량에 따른 형태소 품사 태그 오류 수정 성능

Table 10 Experimental results of 10,000 wiki sentences; learning 1,000; 2,000; 3,000 and 4,000 wiki sentences. (Part-Of-Speech)

Where are r: # of reduce, e: # of error.

POS tag Corpus	Wiki1000		Wiki2000		Wiki3000		Wiki4000	
	r/e*100	r	r/e*100	r	r/e*100	r	r/e*100	r
0-0	42.06	3,691	46.78	4,105	46.78	4,105	50.15	4,401
1-0	42.06	3,691	46.78	4,105	46.78	4,105	50.17	4,403
2-1	52.54	4,611	57.29	5,028	57.29	5,028	55.61	4,880
3-1	52.04	4,567	59.22	5,197	59.22	5,197	58.32	5,118
4-1	54.47	4,780	59.70	5,239	59.70	5,239	60.00	5,266
4-2	54.42	4,776	59.65	5,235	59.65	5,235	59.98	5,264
5-1	58.10	5,099	60.53	5,312	60.53	5,312	60.79	5,335
5-2	58.10	5,099	60.51	5,310	60.51	5,310	60.79	5,335
6-1	58.35	5,121	60.95	5,349	60.95	5,349	61.66	5,411
6-2	58.35	5,121	60.95	5,349	60.95	5,349	61.68	5,413
7-1	58.27	5,114	60.34	5,295	60.48	5,308	62.44	5,480
7-2	58.26	5,113	60.34	5,295	60.46	5,306	62.47	5,482
8-1	56.70	4,976	58.28	5,115	60.02	5,267	60.55	5,314
8-2	56.70	4,976	58.26	5,113	60.02	5,267	60.51	5,310

표 11 블로그 말뭉치 학습량에 따른 형태소 품사 태그 오류 수정 성능

Table 11 Experimental results of 4,000 blog sentences; learning 1,000; 2,000; 3,000 and 4,000 blog sentences. (Part-Of-Speech)

Where are r: # of reduce, e: # of error.

POS tag Corpus	Blog1000		Blog2000		Blog3000		Blog4000	
	r/e*100	r	r/e*100	r	r/e*100	r	r/e*100	r
0-0	28.90	558	21.60	417	15.02	290	10.25	198
1-0	28.90	558	21.60	417	15.02	290	10.25	198
2-1	37.91	732	36.77	710	29.93	578	28.28	546
3-1	44.07	851	38.43	742	37.75	729	32.94	636
4-1	43.29	836	43.45	839	42.72	825	38.27	739
4-2	43.24	835	43.29	836	42.47	820	41.27	797
5-1	43.03	831	42.67	824	43.50	840	44.17	853
5-2	42.98	830	42.67	824	43.50	840	44.12	852
6-1	38.43	742	41.27	797	43.03	831	44.07	851
6-2	38.43	742	41.17	795	42.98	830	44.02	850
7-1	37.39	722	37.75	729	41.27	797	43.50	840
7-2	37.39	722	37.75	729	41.27	797	43.50	840
8-1	32.94	636	36.61	707	38.27	739	37.65	727
8-2	32.94	636	36.61	707	38.27	739	37.65	727

55개, 4,000문장에서는 56개의 오류를 감소시켰다(표 12). 문장수를 변경하며 학습하고 평가하였을 때 각각의 결과는 큰 차이를 보이지 않았다. 이는 형태소 말뭉치와 비교하여 개체명은 주변 형태소 또는 어절 정보에 영향을 적게 받기 때문에 낮은 임계치에서 패턴을 찾기 어렵기 때문이라고 분석된다. 따라서 여러 번 등장하는 오류는 대부분 수정되었지만 규칙에 적용되지 않은 새로운 단어나 주변 정보를 가지는 개체명은 학습량을 늘려도 규칙에 적용되지 않는 것으로 분석된다.

실험 결과 표에서 볼 수 있는 'r: # of reduce'는 오류 수정으로 감소된 오류 수이고, 'e: # of error'는 전체 오류 수이다. 또한 'r/e * 100'으로 오류 수정률을 표현하였다.

3.4 작업그룹 간 성능 평가

형태소 품사 태그 오류 수정은 학습한 결과가 다른 그룹의 문서에 얼마나 적용되는지 확인하기 위한 실험을 진행하였다. 이를 위해 학습한 규칙을 서로 다른 그룹의 문서에 적용하여 성능을 확인하였다. 실험에 사용한 임계치는 각 작업그룹 실험에서 상위의 성능을 보여준 임계치를 선택하였다. 블로그 문서를 학습한 규칙은 [(3-1), (3-2), (4-1), (4-2), (5-1), (5-2), (6-1), (6-2)]의 임계치를 사용하였다. 위키피디아 문서를 학습한 규칙은 [(5-1), (5-2), (6-1), (6-2), (7-1), (7-2), (8-1),

표 12 블로그 말뭉치 학습량에 따른 개체명 태그 오류 수정 성능

Table 12 Experimental results of 4,000 blog sentences; learning 2,000; 3,000 and 4,000 blog sentences. (Named Entity)

Where are r: # of reduce, e: # of error.

NE tag Corpus	Blog2000		Blog3000		Blog4000	
	r/e*100	r	r/e*100	r	r/e*100	r
0-0	34.38	55	34.38	55	35.00	56
1-0	34.38	55	34.38	55	35.00	56
1-1	34.38	55	34.38	55	35.00	56
2-0	35.00	56	33.75	54	33.75	54
2-1	35.00	56	33.75	54	33.75	54
2-2	35.00	56	33.75	54	33.75	54
3-0	28.13	45	33.75	54	33.75	54
3-1	28.13	45	33.75	54	33.75	54
3-2	28.13	45	33.75	54	33.75	54
3-3	28.13	45	33.75	54	33.75	54

표 13 블로그 학습으로 위키피디아 말뭉치 형태소 품사 태그 오류 수정 성능

Table 13 Experimental results of 10,000 wiki sentences; learning 1,000; 2,000; 3,000 and 4,000 blog sentences. (Part-Of-Speech)

Where are r: # of reduce, e: # of error.

POS tag Corpus	Blog 1,000		Blog 2,000		Blog 3,000		Blog 4,000	
	r/e*100	r	r/e*100	r	r/e*100	r	r/e*100	r
3-1	30.64	2,689	30.31	2,660	28.73	2,521	29.81	2,616
3-2	30.63	2,688	30.30	2,659	28.68	2,517	29.48	2,587
4-1	32.69	2,869	31.05	2,725	30.54	2,680	29.09	2,553
4-2	32.70	2,870	31.04	2,724	30.54	2,680	29.06	2,550
5-1	34.74	3,049	32.01	2,809	31.57	2,771	26.33	2,311
5-2	34.74	3,049	32.01	2,809	31.56	2,770	26.32	2,310
6-1	31.14	2,733	30.34	2,663	27.55	2,418	23.58	2,069
6-2	31.11	2,730	30.34	2,663	27.52	2,415	23.59	2,070

표 14 위키피디아 학습으로 블로그 말뭉치 형태소 품사 태그 오류 수정 성능

Table 14 Experimental results of 4,000 blog sentences; learning 1,000; 2,000; 3,000 and 4,000 wiki sentences. (Part-Of-Speech)

Where are r: # of reduce, e: # of error.

POS tag Corpus	Wiki1000		Wiki2000		Wiki3000		Wiki4000	
	r/e*100	r	r/e*100	r	r/e*100	r	r/e*100	r
5-1	46.82	904	41.22	796	30.55	590	17.56	339
5-2	46.82	904	40.50	782	30.45	588	17.56	339
6-1	46.35	895	22.53	435	19.68	380	8.13	157
6-2	46.35	895	22.37	432	19.52	377	8.08	156
7-1	33.35	644	15.95	308	11.39	220	5.08	98
7-2	33.35	644	15.95	308	11.34	219	5.02	97
8-1	23.20	448	6.01	116	5.02	97	3.31	64
8-2	23.15	447	5.96	115	4.97	96	3.21	62

(8-2)]의 임계치를 사용하였다. 실험 결과는 블로그 문서를 학습한 규칙으로 위키피디아 문서를 수정했을 때, 같은 그룹의 문서인 위키피디아 문서를 학습한 결과보다는 성능이 떨어졌다. 하지만 결과적으로 최대 3,049개의 오류를 수정하여 약 1.1%의 성능이 향상되었다(표 13). 위키피디아 문서를 학습하여 블로그 문서에 적용한 결과는 같은 블로그 문서를 학습하여 평가한 결과보다 오히려 성능이 좋게 나타났다. 표 14를 보면 학습량 1,000문장에 임계치가 (5-1)일 때, 같은 블로그 문서를 평가한 실험에서 최대 853개의 오류를 수정한 결과보다 51개가 더 많은 904개의 오류를 수정하였다. 하지만 이 결과로는 위키피디아 학습 결과가 블로그 문서 수정에 적합하다고 판단하기 어렵다. 이는 위키피디아를 학습한 규칙 중 시스템 오류를 발생시킬 수 있는 규칙이 블로

그 문서에서 적용되지 않는 경우가 많아 이러한 결과가 나타났다고 분석되었다. 실험결과를 보면 일반적으로 학습량이 적을 때, 다른 작업 문서간의 실험결과가 학습량이 많을 때보다 상대적으로 성능이 좋게 나타났다. 이는 학습 문서의 양이 많을수록 해당 도메인에 적합한 규칙이 생성되고, 학습된 규칙은 다른 도메인의 문서를 수정하기에 부적합하다는 것을 알 수 있다.

3.5 토의

본 연구에서는 RDR을 통해 형태소 품사 태그 오류와 개체명 태그 오류를 수정하는 실험을 진행하였다. 실험을 통해 사람이 직접 구축한 정답 말뭉치에도 적지만 오류가 존재하고 RDR을 통해 패턴을 추출할 수 있음을 확인하였다. 형태소 품사 태깅 문서와 개체명 태깅 문서에서는 각각 표 15[3]와 표 16[4]과 같은 오류가 나타났다.

형태소 품사 태깅 문서에서 나타난 오류에는 단어의 마지막 음절이 조사와 동일할 경우, '명사'+'조사'로 분석하는 오류가 다양하게 포함되어 있었다. 또한 접속조사와 부사격조사 분석을 서로 반대로 한 오류도 있었다. 가장 많이 나타난 오류는 마침표와 가운데점에 대한 태깅이다. 소숫점이나 URL에 포함되는 가운데점과 문장 끝에 나타나는 마침표점은 서로 다른 태그를 부착하는데 이를 수작업자가 쉽게 오류를 범하는 것을 알 수 있었다.

개체명 태깅의 오류는 2.1.2.에서 설명하였듯이 4개의 오류 유형이 존재하고 본 실험에서는 오류 유형 1과 2만 수정하였다. 유형 1의 오류는 기업명과 기업에서 제공하는 서비스 시스템명이 동일 할 때 품사 부착에 애매성이 발생한 경우이다. 유형 2의 오류는 개체명이 아닌 단어에 품사를 부착한 경우이다.

오류만 학습하는 경우(Err)와 정답을 학습하는 경우(All)에 대한 실험으로 소량의 문서를 학습하여 임계치가 낮은 경우 모든 내용을 포함하여 학습하는 경우가 성능이 더 우수하다는 것을 알 수 있었다. 학습량과 임계치에 대한 실험으로는 학습량이 많을수록 학습 문서에 나타나는 오류의 수가 증가해 학습에 도움이 되는 것은 당연하지만 학습량이 적더라도 임계치에 따라 충분히 오류를 수정할 수 있다는 것을 실험결과를 통해 증명하였다. 이는 블로그 문서를 학습하고 평가한 실험인 표 11에서 1,000문장을 학습하고, 임계치가 (3-1)인 경우, 최대 성능과 오류 수정수가 2개 밖에 차이가 나지 않는 것을 통해 확인할 수 있다. 마지막으로 학습 결과를 다른 그룹 문서에 적용한 실험은 학습 문서량이 많아짐에 따라 규칙이 학습 도메인에 치우침으로 성능이 떨어지는 것을 확인하였다.

3) NNG:일반명사, NNP:고유명사, JX:보조사, JKS:주격조사, JC:접속조사 JKB:부사격조사, XR:어근, SN:숫자, SF:마침표, SP:소수점

4) TRM:전문 용어, OGR:기관/기업, CV:문명/문화 용어

표 15 형태소 품사 태깅의 오류 유형
Table 15 Error types in the POS tagging corpus

Error type	Error example	
	Initial Corpus	Gold Corpus
NNG + JX	경상남/NNG+도/JX	경상남도/NNP
NNG + JKS	호랑/NNG+이/JKS	호랑이/NNG
NNG + JKB	고속도/NNG+로/JKB	고속도로./NNG
JC & JKB	~와/JC	~와/JKB
NNG & XR	저렴/NNG	저렴/XR
SP & SF	2/SN+./SF+0/SN	2/SN+./SP+0/SN

표 16 개체명 태깅 오류 유형
Table 16 Error types in the Named Entity corpus

Error type	Error example	
	Initial Corpus	Gold Corpus
1	페이스북/TRM	페이스북/ORG
	네이버폰/ORG	네이버폰/TRM
2	컴퓨터/TRM	-
	모바일/TRM	-
3	-	아타리/TRM
	-	일러스트/CV
4	위키백/TRM	위키백과/TRM
	구글/ORG	구글 플레이/TRM

4. 결 론

본 연구에서는 품사 부착 코퍼스의 오류를 커널 RDR을 통해 자동으로 수정하는 새로운 방법을 제시하였다. 그 결과로 정답을 포함한 학습 방법으로 생성한 규칙은 같은 작업 그룹이 작성한 문서에 대한 실험에서 위키피디아 문서의 경우 표 10에서 확인할 수 있듯 최대 5,482개의 오류를 감소시켜 오류를 62% 감소시켰고 문서 성능을 2% 향상시켰다. 블로그 문서 또한 최대 853개의 오류를 감소시켜 오류를 44% 감소시켰고 문서 성능을 0.8% 향상시켰다(표 11). 이처럼 학습과 평가가 같은 그룹의 문서는 RDR 학습을 통한 코퍼스 오류 수정으로 코퍼스의 성능을 향상 시킬 수 있다는 결과를 얻었다. 실험을 통해 수정하지 못한 오류가 남아 있지만, 사람이 직접 구축하여 오류의 수가 적고, 규칙이 복잡하더라도 RDR 시스템을 통해 수정 가능하다는 것을 보였다.

다른 작업 그룹간 성능 평가에서는 학습을 통해 생성되는 규칙이 학습 문서와 긴밀하기 때문에 학습 문서량이 증가함에 따라 다른 그룹 문서에는 적합하지 않다는 것을 확인하였다.

표 17은 형태소 품사 태깅 오류 수정에서 확인한 각 실험별 최대 성능표이다.

본 실험에서는 학습하는 문서에 가장 알맞은 임계치를 찾기 위해 다양한 임계치를 적용하는 실험을 반복 진행하였다. 결정된 임계치는 오류의 유형에 따라 제각

표 17 학습 문서와 평가 문서에 따른 형태소 품사 태깅 오류 수정 최대 성능

Table 17 Best modified performance according to the corpus types

Train / Evaluation	Wiki		Blog	
	# of reduce	accuracy (%)	# of reduce	accuracy (%)
Wiki	5,482	98.73	3,049	97.80
Blog	904	98.99	853	98.94

기 다른 오류 수를 모두 만족시킬 수 없다. 이러한 문제점을 극복하기 위하여 오류 수나 유형에 따라 유동적인 임계치를 결정할 수 있는 시스템을 구축 가능하다면 더 나은 오류 수정률을 보일 수 있을 것으로 생각된다.

본 논문에서 제시된 방법은 실험을 통해 확인했듯이 품사 부착 말뭉치와 개체명 태그 부착 말뭉치뿐만 아니라 구문분석, 의미역 분석 등 레이블링 문제에 해당되는 문제에는 모두 적용이 가능하다. 우리는 이 방법을 구문분석 및 의미역 분석 말뭉치 구축에 적용할 예정이다.

References

[1] J. Hong, J. Cha, "Error Correction of Sejong Morphological Annotation Corpora using Part-of-Speech Tagger and Frequency Information," *Journal of KIISE. SA*, ISSN:1226-2285, Vol. 40, No. 7, pp. 417-428, 2013.

[2] M. Choi, H. Seo, H. Kwon and J. Kim, "Detecting and correcting errors in Korean POS-tagged corpora," *Journal of the Korean Society of Marine Engineering*, Vol. 37, No. 2, pp. 227-235, 2013.

[3] Wu. X., "Knowledge acquisition from database," Ablex Publishing Corp., USA, 1995.

[4] Zhu. X., Wu. X. and Chen Q., "Eliminating Class Noise in Large Datasets," *Proc. of the 20th ICML International Conference on Machine Learning (ICML 2003)*. Washington D. C., Vol. 3, pp. 920-927, 2003.

[5] Zhu. X., Wu. X. and Chen. Q., "Bridging Local and Gobal Data Cleansing: Identifying Class Noise in Large," *Distributed Data Datasets, Data Mining and Knowledge Discovery*, pp. 275-308, Dec. 2006.

[6] Guyon, Isabelle, Matic. N. and Vapnik. V., "Discovering informative patterns and data cleaning," *Advances in Knpwledge Discovery and Data Mining*, AAAI/MIT Press, pp. 181-203, 1996.

[7] Gamberger, Dragan, Lavrac. N. and Groselj. C., "Experiments with noise filtering in a medical domain," *Proc. of 16th ICML Conference*, pp. 143-151, San Francisco, CA, 1999.

[8] John, G. H., "Robust decision trees: Removing outliers from databases," *Proc. of the First International Conference on Knowledge Discovery and Data Mining*, pp. 174-179, AAAI Press, 1995.

[9] Zeng, Xinchuan and Martinez. T., "A noise filtering method using neural networks," *SCIMA 2003. IEEE International Workshop on Soft Computing Techniques in Instrumentation, Measurement and Related Applications*, pp. 26-31, 17, May 2003.

[10] Edwards, G., and Compton, P., "Peirs: A pathologist maintained expert system for the interpretation of chemical pathology reports," *Pathology*, Vol. 25, No. 1, pp. 27-34, 1993.

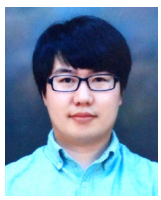
[11] Edwards, G., and Compton, P., "Experience with Ripple-Down Rules," *Knowledge-Based System Journal*, Vol. 19, Issue 5, pp. 356-362, 2006.

[12] Cao, T.M. and Compton, P. A., "Simulation Framework for Knowledge Acquisition Evaluation," *Twenty-Eighth Australasian Computer Science Conference ACSC2005*. Newcastle, Vol. 38, pp. 353-360, 2005.

[13] Ghassan Beydoun, PhD Thesis, "Incremental Knowledge Acquisition for Search Control Heuristics," *UNSW*, 2000.

[14] Edwards and Compton. (2007. May 09). [Online]. Available: http://www.cse.unsw.edu.au/~cs9416/06s1/lectures/rdr/RDR_slides.pdf(downloaded 2016. Apr. 7)

[15] Nguyen, D. Q., Nguyen, D. Q., Pham, D. D., & Pham, S. B., "RDRPOSTagger: A Ripple Down Rules-based Part-Of-Speech Tagger," *EACL'14*, pp. 17-20. 2014.



박 태 호

2013년 창원대학교 컴퓨터공학과(학사)
2015년 창원대학교 컴퓨터공학과(석사)
2016년~현재 창원대학교 친환경해양플랜트FEED공학과 박사과정. 관심분야는 자연어처리, 의미 분석

차 정 원

숭실대학교(학사). 포항공과대학교(석사, 박사). USC/ISI(박사후연수). 2004년~현재 창원대학교 컴퓨터공학과 교수. 관심분야는 자연어처리, 기계학습, 정보검색