KIPS Trans. Softw. and Data Eng.
Vol.9, No.7 pp.221~228 pISSN: 2287-5905

https://doi.org/10.3745/KTSDE.2020.9.7.221

Detecting Errors in POS-Tagged Corpus on XGBoost and Cross Validation

Min-Seok Choi† · Chang-Hyun Kim†† · Ho-Min Park† · Min-Ah Cheon† · Ho Yoon† · Young Namgoong† · Jae-Kyun Kim††† · Jae-Hoon Kim††††

ABSTRACT

Part-of-Speech (POS) tagged corpus is a collection of electronic text in which each word is annotated with a tag as the corresponding POS and is widely used for various training data for natural language processing. The training data generally assumes that there are no errors, but in reality they include various types of errors, which cause performance degradation of systems trained using the data. To alleviate this problem, we propose a novel method for detecting errors in the existing POS tagged corpus using the classifier of XGBoost and cross-validation as evaluation techniques. We first train a classifier of a POS tagger using the POS-tagged corpus with some errors and then detect errors from the POS-tagged corpus using cross-validation, but the classifier cannot detect errors because there is no training data for detecting POS tagged errors. We thus detect errors by comparing the outputs (probabilities of POS) of the classifier, adjusting hyperparameters. The hyperparameters is estimated by a small scale error-tagged corpus, in which text is sampled from a POS-tagged corpus and which is marked up POS errors by experts. In this paper, we use recall and precision as evaluation metrics which are widely used in information retrieval. We have shown that the proposed method is valid by comparing two distributions of the sample (the error-tagged corpus) and the population (the POS-tagged corpus) because all detected errors cannot be checked. In the near future, we will apply the proposed method to a dependency tree-tagged corpus and a semantic role tagged corpus.

Keywords : Error Detection, POS-tagged Corpus, XGBoost, Cross-validation

XGBoost와 교차검증을 이용한 품사부착말뭉치에서의 오류 탐지

최 민 석† · 김 창 현†† · 박 호 민† · 천 민 아† · 윤 호† · 남 궁 영† · 김 재 균††† · 김 재 훈††††

요 약

품사부착말뭉치는 품사정보를 부착한 말뭉치를 말하며 자연언어처리 분야에서 다양한 학습말뭉치로 사용된다. 학습말뭉치는 일반적으로 오류가 없다고 가정하지만, 실상은 다양한 오류를 포함하고 있으며, 이러한 오류들은 학습된 시스템의 성능을 저하시키는 요인이 된다. 이러한 문제를 다소 완화시키기 위해서 본 논문에서는 XGBoost와 교차 검증을 이용하여 이미 구축된 품사부착말뭉치로부터 오류를 탐지하는 방법을 제안한다. 제안된 방법은 먼저 오류가 포함된 품사부착말뭉치와 XGBoost를 사용해서 품사부착기를 학습하고, 교차검증을 이용해서 품사오류를 검출한다. 그러나 오류가 부착된 학습말뭉치가 존재하지 않으므로 일반적인 분류기로서 오류를 검출할 수 없다. 따라서 본 논문에서는 매개변수를 조절하면서 학습된 품사부착기의 출력을 비교함으로써 오류를 검출한다. 매개변수를 조절하기 위해서 본 논문에서는 작은 규모의 오류부착말뭉치를 이용한다. 이 말뭉치는 오류 검출 대상의 전체 말뭉치로부터 임의로 추출된 것을 전문가에 의해서 오류가 부착된 것이다. 본 논문에서는 성능 평가의 척도로 정보검색에서 널리 사용되는 정밀도와 재현율을 사용하였다. 또한 모집단의 모든 오류 후보를 수작업으로 확인할 수 없으므로 표본 집단과 모집단의 오류 분포를 비교하여 본 논문의 타당성을 보였다. 앞으로 의존구조부착 말뭉치와 의미역 부착말뭉치에서 적용할 계획이다.

키워드 : 오류 탐지, 품사부착말뭉치, XGBoost, 교차 검증

1. 서 론

말뭉치(corpus)란 자연언어 연구를 위해 특정 목적을 가지고 언어 표본을 추출한 집합을 의미한다. 그 중에서 품사 표지(POS tags)가 부착된 말뭉치를 품사부착말뭉치(POS-tagged corpus)라고 한다[1]. 이러한 말뭉치들은 오랜 기간 다양한 사람들이 제작하여, 다양한 오류를 포함하고 있다[2]. 이런 말뭉치는 학습말뭉치(training corpus)로 사용될 경우, 학습된 시스템의 성능을 저하시키는 요인이 되므로 오류의

※ 이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원(R7119-16-1001, 지식증강형 실시간 동시통역 원천기술 개발)과 2017년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원(NRF-2017M3C4A7068187, 한국어 정보처리 원천 기술 연구 개발)을 받아 수행된 연구임.
† 비 회 원 : 한국해양대학교 컴퓨터공학과 박사과정
†† 비 회 원 : 한국전자통신연구원 책임연구원
††† 비 회 원 : 한국해양대학교 컴퓨터공학과 석사과정
†††† 종신회원 : 한국해양대학교 컴퓨터공학과 교수
Manuscript Received : April 2, 2020
Accepted : April 25, 2020
* Corresponding Author : Jae-Hoon Kim(jhoon@kmou.ac.kr)

수정이 필요하다. 일반적으로 오류 수정에는 말뭉치 구축만큼의 많은 인력과 시간과 비용이 소요될 뿐 아니라 여전히 일관성과 신뢰성 문제를 가지고 있다[3]. 따라서 많은 연구자들[3-7]이 말뭉치의 오류를 검출하고 수정하는 연구를 진행하였다. 최근 기계학습 기술이 발전하고 많은 분류시스템들(classification system)의 성능이 상당히 개선되었지만, 오류를 검출을 위해서 적용하는 것은 여전히 한계가 있다. 왜냐하면 오류를 검출하기 위한 학습말뭉치가 존재하지 않을 뿐 아니라 작업자들마다 오류가 다르므로 일반적인 분류 알고리즘을 적용하기에는 많은 문제를 가지고 있다.

이러한 문제를 다소 완화시키기 위해서 본 논문에서는 교차검증(cross validation)과 XGBoost를 이용하여 이미 구축된 품사부착말뭉치로부터 오류를 탐지하는 방법을 제안한다. 먼저 오류가 포함된 말뭉치와 XGBoost를 이용해서 품사부착기(POS tagger)를 학습하고 교차검증을 이용해서 품사오류를 검출한다. 그러나 오류가 부착된 학습말뭉치가 존재하지 않으므로 일반적인 분류문제로 해결할 수 없다. 따라서 본 논문에서는 매개변수를 조절하면서 학습된 품사부착기의 출력을 비교함으로써 오류를 검출한다. 매개변수를 조절하기 위해서 본 논문에서는 작은 규모의 오류부착말뭉치를 이용한다. 이 말뭉치는 오류 검출 대상의 전체 말뭉치로부터 임의로 추출된 것을 전문가에 의해서 오류가 부착된 것이다.

제안된 오류 검출 시스템은 크게 문맥표상의 표현(context embedding representation)과 품사확률(probability of POS)의 예측, 기준값의 설정(threshold setting), 오류 후보의 선택(error candidate selection)으로 구성된다. 문맥표상은 오류 주변의 문맥을 벡터로 표현한 것이며, 여기서 문맥은 주어진 형태소 주변의 형태소와 품사를 말한다. 기준값을 설정하기 위해 전체 말뭉치에서 1,000 문장을 임의 추출하여 오류를 부착한다. 이 오류가 최대한 검출될 수 있도록 기준값을 설정된다. 마지막으로 오류 후보는 설정된 기준값과 교차검증을 이용해서 최종 오류 후보로 선택한다.

본 논문의 구성은 다음과 같다. 2장에서 오류 검출을 소개하고, 3장에서는 오류 후보 탐지 방법을 기술한다. 4장에서는 실험의 환경과 기준값 및 실험을 통한 오류 탐지 성능을 분석하고, 5장에서 결론 및 향후 연구 방향을 기술한다.

2. 관련 연구: 오류 검출

본 장에서는 일반적인 오류 검출과 품사부착말뭉치에서 오류 검출에 대해서 간략히 기술할 것이다.

2.1 오류 검출과 XGBoost

오류 검출이란 전체 데이터에서 다른 형태의 데이터를 찾는 것을 말하며[8], 오류 검출(anomaly detection or error detection) 방법은 NN(nearest neighbor) 기반 방법[9], 스펙트럴(spectual) 기반 방법[10], 군집화(clustering) 기반 방법[11] 등이 있으며 최근에는 앙상블 방법[12]도 연구되고 있다. 본 연구에서 사용될 XGBoost(eXtreme Gradient Boosting)[13]는 앙상블 방법의 일종이다. 앙상블(ensemble)이란 여러 개의 모델을 학습하여 다음 결과 예측 시 여러 모델의 결과를 종합하여 사용하는 방법이다[14]. 이러한 방식은 크게 배깅(bootstrap aggregation)과 부스팅(boosting)으로 나눌 수 있다. 배깅의 대표적인 방법으로 랜덤포레스트(random forest)가 있다. 랜덤포리스트는 여러 개의 의사 결정 나무를 사용하여 평균을 내거나 다수결의 원칙으로 하나의 최종 결과를 예측하는 방법이다[15]. 부스팅의 대표적인 예로 XGBoost가 있다. XGBoot의 기본적인 개념은 약한 분류기(week classifier)를 묶어서 강한 분류기(strong classifier)를 만드는 것이다[13]. XGBoost는 Equation (1)과 같이 정의되며, Equation (1)은 모델 M이 x에 대해서 정확히 Y를 예측할 확률이다. 만약 모델 M의 오류 함수(error function) $M_e(x)$을 정확히 예측할 수 있는 모델 G가 존재한다면, $M_e(x)$는 Equation (2)와 같이 정의된다. 같은 방법으로 모델 G의 오류 함수 $G_e(x)$에 대하여 모델 H가 존재한다면, $G_e(x)$는 Equation (3)과 같이 정의된다. Equation (1)~(3)을 바탕으로 Equation (4)를 도출할 수 있으며, 각 모델에 가중치 w_i를 주면, Equation (5)와 같다.

$$Y = M(x) + M_e(x) \tag{1}$$

$$M_e(x) = G(x) + G_e(x) \tag{2}$$

$$G_e(x) = H(x) + H_e(x) \tag{3}$$

$$Y = M(x) + G(x) + H(x) + e(x) \tag{4}$$

$$Y = w_1 M(x) + w_2 G(x) + w_3 H(x) + e(x) \tag{5}$$

결과적으로 XGBoost는 모델 M, G, H와 가중치 w_1, w_2, w_3를 학습하여 모델을 구하는 방법이다.

2.2 품사부착 오류 검출

1장에서 언급했듯이 수동으로 품사부착말뭉치에서 오류를 찾는 것을 품사부착말뭉치를 구축하는 것과 거의 동일한 시간과 비용이 소요된다. 따라서 많은 연구들은 오류수정 도구를 개발하여 오류를 찾고 수정하였다[3,16]. 특히 [3]은 오류패턴을 검출하고 검출된 오류를 수정하는 도구를 개발했으며 이 방법은 수동으로 일일이 수정하는 방법에 비해 9배 정도 빠르게 수정할 수 있었다. 또한 말뭉치를 구축하는 과정에서 오류를 최소화하려는 연구도 시도되었으며, 이 방법으로 널리 사용되는 방법은 여러 부착 시스템을 결과가 불일치할 경우 오류로 검출하는 방법이다[12]. 즉 완전한 학습말뭉치(gold standard corpus)보다는 여러 부착 시스템이 일치할 경우에도 오류가 포함되어 있을 수 있지만 오류가 없는 것으로 간주한다. 이렇게 구축된 말뭉치를 준완전 학습말뭉치(silver standards)[17]라고 한다. 그 밖에도 신경망을 이용한 오류 검출[5], 변형 n-그램(variation n-gram)[7]을 이용한 오류 검출 등이 있다.

3. 품사부착 오류 후보 검출 시스템

본 논문에서 제안된 오류 검출 시스템은 크게 세 단계(문맥 표상 표현과 품사확률 예측, 기준값 설정, 오류 후보 선택)로 구성된다. 이하의 절에서는 각 단계를 자세히 설명할 것이다.

3.1 문맥표상 표현과 품사확률 예측

주어진 단어의 품사는 문장 내에서 주변 단어에 의해서 결정된다. 즉, 주어진 단어의 품사가 오류인지를 판단하기 위해서는 주변의 문맥 정보를 충분히 활용해야 한다. 문맥이 많으면 많을수록 정확한 예측이 가능하지만, 본 논문에서는 기본적인 문맥은 주변의 두 단어(형태소)를 문맥으로 정의한다. Fig. 1은 세종형태분석말뭉치(이하 세종말뭉치)[18]에서 추출한 문장 "각 역들에서도 자전거 대여업을 겸하고 있다."의 품사부착 예이다.

```
각              각__01/MM
역들에서도      역__14/NNG+들__09/XSN+에서/JKB+도/JX
자전거          자전거/NNG
대여업을        대여업/NNG+을/JKO
겸하고          겸하/VV+고/EC
있다.           있__01/VX+다/EF+./SF
```

Fig. 1. An Example Sentence in Sejong POS-Tagged Corpus

Fig. 1의 형태소 '자전거'를 기준으로 문맥 표상을 정의하면 Fig. 2와 같다. 주어진 형태소 '자전거'의 문맥 표상은 Fig. 2에서 볼 수 있듯이 자신의 표상(회색 배경)을 포함하여 좌우 두 개의 형태소 표상(흰색 배경)과 품사 표상(점선 배경)을 연접하여(concatenate) 사용하며 이를 기본 문맥(default context)이라고 한다.

Fig. 2. An Example of Contextual Embedding for the Morpheme '자전거'

기본 문맥만으로 오류를 예측하는 것이 불충분하여 문맥을 추가적으로 확장할 것이다(4.2절 참조). 각 형태소와 품사의 표상은 FastText[19]를 이용해서 학습할 것이다. FastText는 주변 단어와 단어의 부분 단어(subword)를 이용하여 미등록어 문제에 좀 더 좋은 결과를 보여주는 장점이 있다. 이와 같은 방법으로 구해진 문맥표상은 XGBoost[13]의 입력으로 제공되어 주어진 문맥에 대한 각 품사의 확률을 구한다. 구해진 품사 확률을 바탕으로 3.2절에서 설명할 기준값을 설정할 것이다.

3.2 기준값 설정

분류기(classifier)를 이용해서 오류를 검출할 경우에는 주로 여러 분류기의 결과가 불일치할 경우에 오류로 간주한다[12,20]. 이 경우에는 여러 개의 분류기가 학습되어야 할 뿐 아니라 모든 분류기가 동시에 잘못된 결과를 출력할 경우에는 여전히 오류로 남아있을 것이다. 이런 이유로 앙상블 기반 불일치 방법(Ensemble-based disagreement)은 일반적으로 말뭉치를 구축하면서 발생되는 오류를 검출할 때 주로 사용된다.

본 논문에서 완전한 하나의 분류기(XGBoost)가 있다고 가정한다. 이 분류기는 주어진 문맥에 대해 각 품사의 확률 $P(t|C)$를 출력하고, 이 확률을 내림차순으로 정렬하면 $p_1 = P(t_1|C)$, $p_2 = P(t_2|C)$, …, $p_n = P(t_n|C)$과 같다. 여기서 t는 품사이고, C는 문맥이고, n은 품사의 개수이다. 또한 부착된 품사 t_a의 확률이 $p_a = P(t_a|C)$일 때, 본 논문에서는 다음과 같은 두 가지 가정을 전제로 품사 t_a를 오류로 판단한다.

가정 1: $p_1 - p_a \rangle \theta_1$

실제로 부착된 품사의 확률 p_a과 분류기의 가장 높은 확률 p_1의 차이가 지정된 기준값 θ_1보다 클 경우, 오류로 간주한다 Fig. 3. 이 가정은 일반적으로 분류기의 출력과 부착된 품사가 다를 경우로서 일반적으로 흔히 사용하는 가정이다.

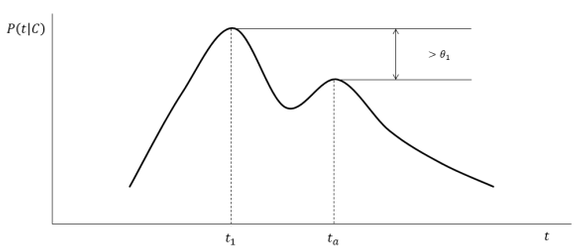


Fig. 3. Conceptual Graph for Assumption 1

가정 2: $p_1 - p_a \langle \theta_1$ and $p_1 - p_2 \langle \theta_2$:

가정 1의 조건을 만족하지 않더라고 분류기의 가장 높은 확률 p_1과 두 번째로 높은 확률 p_2의 차이가 작은 경우에도 오류로 간주한다(Fig. 4). 일반적으로 확률 분포가 균등할 경우, 그 시스템의 엔트로피(entropy)가 가장 높다. 따라서 엔트로피가 높을 경우에 시스템에 불안하여 오류가 발생된다. 따라서 p_1과 p_2가 거의 차이가 없다면 오류로 간주할 수 있다.

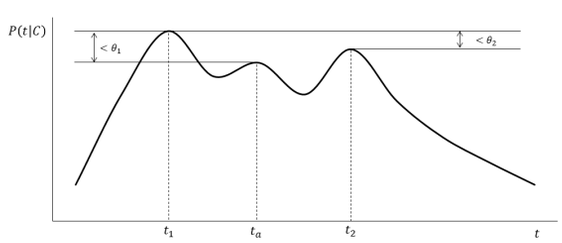


Fig. 4. Conceptual Graph for Assumption 2

여기서 기준값에 해당하는 θ_1과 θ_2을 매개변수(hyper-parameters)로서 실험을 통해 결정되며(4.2절 참조), θ_2는 θ_1에 비해 매우 작은 값이 될 것이다($\theta_2 \ll \theta_1$). 이 값들을 설정

하기 위해서는 오류가 부착된 말뭉치(error-tagged corpus)가 필요하다. 세종말뭉치로부터 임의로 1,000개의 문장을 추출하여 오류를 수동으로 부착하였다(4.1절 참조). 오류부착 말뭉치로부터 재현율(recall)이 최대가 되도록 θ_1과 θ_2를 설정한다.

3.3 오류 후보 선택

이미 구축된 말뭉치의 오류를 선정해야 하므로 모든 말뭉치를 대상으로 오류를 검출해야 한다. 본 논문에서는 3.2절에서 설정된 기준값을 바탕으로 말뭉치 전체를 교차검증(cross validation) 방법으로 오류 후보를 선택한다. Fig. 5에서 보는 바와 같이 교차 검증을 통한 오류 검출 방법은 전체 데이터를 N 등분하여 하나의 조각을 제외한 나머지 조각으로 학습하고 제외한 조각에서 오류를 검출하는 방법으로 이와 같은 과정을 N번 반복하면 전체 말뭉치에 대한 오류를 검출할 수 있다.

4. 실험 및 평가

4.1 오류 검출을 위한 대상 말뭉치

A	B	C
Train	Train	Error detection
Train	Error detection	Train
Error detection	Train	Train

Fig. 5. 3-fold Cross Validation for Error Detection

실험을 위해서 본 논문에서는 품사부착말뭉치로 널리 사용되는 세종말뭉치[18]을 사용하며, 이 말뭉치에는 다양한 형태의 오류들이 있다[21]. Table 1은 사용된 세종말뭉치와 3.2절에서 설명한 가준값을 설정하기 위한 오류부착말뭉치의 통계치이다. 오류부착말뭉치는 세종말뭉치로부터 임의로 추출한 1000개의 문장에 대해서 수동으로 오류를 부착한 말뭉치이다.

Table 1. The Statistics of Sejong POS-tagged Corpus and Error-annotated Corpus

Items	Sejong POS-tagged Corpus	Error-tagged Corpus
Sentence	700,000	1,000
Word(Eojeol)	7,006,941	10,182
Morphems	15,660,488	22,767

오류부착말뭉치에서 각 품사들의 오류분포는 4.3절의 Table 5에서 자세히 보여줄 것이며 약 8.54%의 오류가 포함되어 있었다. 꽤 많은 오류를 포함하고 있으며 이러한 오류는 영어권의 품사부착말뭉치의 오류율(약 3%)[22]에 비해 높은 편이라 어떤 형태로든 수정되어야 할 것이다.

4.2 기준값 설정

1) 기본 문맥의 기준값 설정

3.3절에서 언급한 θ_1과 θ_2를 설정하기 위해서는 4.1절에서 언급한 오류부착말뭉치를 사용한다. 오류 검출은 가능한 모든 오류를 검출해서 수작으로 수정할 수 있도록 제시하는 것이 매우 중요하므로 재현율이 최대가 되도록 설정한다. Table 2는 기본 문맥을 사용해서 θ_1과 θ_2의 변화에 따른 정밀도(precision)과 재현율(recall)의 변화를 보이고 있다.

Table 2. Precision and Recall According to the Change of θ_1 and θ_2

θ_1	θ_2	Precision	Recall
0.010	0.003	0.47	0.76
0.010	0.005	0.50	0.79
0.010	0.007	0.50	0.81
0.015	0.003	0.70	0.87
0.015	0.005	0.69	0.88
0.015	**0.007**	**0.70**	**0.90**
0.020	0.003	0.59	0.83
0.020	0.005	0.60	0.85
0.020	0.007	0.67	0.87

Table 2에서 볼 수 있듯이 θ_1과 θ_2가 각각 0.015와 0.007일 때, 가장 좋은 결과를 보였다. 전체 오류에 90%를 검출할 수 있어서 아직도 10%의 오류가 되지 않았다. Table 3은 아직도 검출되지 않은 몇 가지의 예를 보이고 있다.

Table 3. Examples of Undetected Errors

Eojeol	Wrong analysis	Correct analysis
안다.	안/VV+다/EF +./SF	알/VV+ㄴ다/EF+./SF
떠난	떠나/VV +ㄹ/ETM	떠나/VV +ㄴ/ETM
사이에는	사이/NNG +에/JKB+는/JX	사이/NNG +에는/JKB

Table 3의 '안다.'는 "여자친구를 안다"와 같은 품에 안다의 뜻과 "그 친구를 안다"와 같은 정보를 갖추다의 뜻을 가지는 '안다.'의 기본형 오류이고, '떠난'은 오탈자 오류이다. 한편 '사이에는'은 과분석 오류로서 조사 '에는'이 격조사 '에'와 보조사 '는'이 결합된 말로 표준국어대사전에 등재되어 '에는'과 같이 분석되어야 한다[1].

2) 문맥 확장을 통한 기준값 설정

앞 절에서 분석된 이런 오류들은 문맥 정보가 충분이 반영되지 않아서 미검출되었다. 따라서 본 논문에서는 아래와 같은 방법으로 기본 문맥을 확장한다.

확장표상A : $E(m) + P(t|m)$

확장표상B : 확장표상A + $E(w_i)$

1) 주석자에 따라서 '에는'으로 분석할 수도 있다.

확장표상C : 확장표상B + $E(w_{i-1})$
확장표상D : 확장표상C + $E(w_{i+1})$

여기서 m은 오류 검출 대상이 되는 형태소이고, $E(\cdot)$는 표상 함수(embedding function)이고, $P(t|m)$는 주어지 형태소 m에 대한 품사 t의 분포이다. 또한 w_i는 형태소 m이 포함된 어절을 의미하고 w_{i-1}과 w_{i+1}은 각각 w_i의 이전 어절과 이후 어절을 의미한다. 확장표상A는 품사의 오류를 탐지하기 위하여 형태소에 대한 품사분포를 추가했고 확장표상B는 오탈자 오류를 탐지하기 위하여 기본 문맥에 포함하는 어절의 표상을 추가했다. 확장표상C와 확장표상D는 과분석 오류를 탐지하기 위하여 앞 어절의 표상과 뒤 어절의 표상을 추가하였다. 모든 표상는 기본 문맥 표상과 같이 FastText를 이용해서 학습하였으며, 각 표상의 크기는 Table 4와 같다.

Table 4. The Size of Each Embedding

Embedding	Size	Remarks
Default	700	100 per morpheme, 50 per tag
Extended A	743	$\|P(t\|m)\|$ = 43
Extended B	893	150 per Eojeol
Extended C	1,043	
Extended D	1,193	

Table 5는 4.2.1절에서 설명한 방법과 같은 방법으로 확장표상에 대해서 기준값을 설정하였다. Table 5를 바탕으로 확장표상D일 때, θ_1과 θ_2는 각각 0.015와 0.005로 설정하여 오류를 검출할 것이다.

Table 5. Threshold Setting for Extended Embedding

Embedding	θ_1	θ_2	Precision	Recall
Default	0.015	0.007	0.70	0.90
Extended A	0.015	0.005	0.74	0.93
Extended B	0.015	0.007	0.76	0.94
Extended C	0.015	0.005	0.79	0.94
Extended D	**0.015**	**0.005**	**0.80**	**0.96**

4.3 오류 검출

4.2절에서 설정된 θ_1과 θ_2을 바탕으로 세종말뭉치 전체에 대해서 오류 후보를 선택하였다. 오류 검출은 3.3절에서 설명한 교차 검증 방법으로 수행되며 그 결과는 Table 6과 같다. Table 6에서 세종말뭉치(Sejong POS-tagged)의 오류(Error)는 본 논문에서 제안된 오류 검출 시스템에 의해서 검출된 것이고, 오류부착말뭉치(Error-tagged)의 오류(Error)는 수동으로 표시된 것이다. 세종말뭉치에서 1,250,500개의 품사 오류를 검출하여 약 8%를 오류로 검출하였다. 검출된 오류를 일일이 확인하여 오류 여부를 판단해야 하지만 이를 수정하는 일은 너무나 많은 시간과 비용이 소요되어 검출된 오류 중에서 1,000개를 직접 분석해 보았다. 그 결과, 29개가 정답을 오류로 판단하여(false alarm) 오탐률(false positive rate)은 2.9%로 매우 정확하게 판단하고 있음을 알 수 있었다.

Table 6. The Results of Error Detection for Sejong POS-tagged Corpus and the Error-tagged Corpus Constructed Manually

Tag	Sejong POS-tagged		Error-tagged	
	Morph.	Error	Morph	Error
NNG	3,659,551	179,318	5,248	244
EC	955,694	48,740	1,416	68
VV	960,192	132,506	1,376	156
ETM	824,534	65,138	1,182	88
JX	636,587	32,466	944	52
JKB	643,841	26,397	907	44
SF	624,703	19,366	901	24
EF	602,881	36,776	870	48
JKO	543,776	33,170	796	44
NNB	479,595	51,796	754	72
XSV	465,645	50,290	673	80
SS	432,139	15,989	625	16
MAG	397,792	28,243	582	40
JKS	384,634	23,847	582	36
NNP	412,671	34,664	569	56
EP	382,651	15,689	539	20
JKG	338,983	17,288	514	24
VA	323,506	54,026	481	72
VCP	283,081	18,966	423	20
SN	264,513	12,432	396	12
VX	262,131	22,805	394	24
SP	256,383	14,870	355	12
NP	224,499	72,064	352	112
MM	187,934	67,092	304	104
XR	221,976	20,200	298	116
XSN	174,399	66,446	257	92
XSA	131,041	45,995	175	76
JC	97,394	8,863	143	52
ETN	75,484	6,114	102	40
SL	51,356	2,671	87	16
NR	60,272	4,882	79	16
SH	52,298	3,033	79	24
MAJ	53,210	3,246	73	8
JKC	33,997	4,352	54	12
VCN	28,252	2,288	41	4
SW	34,886	3,070	40	4
SE	24,553	859	35	0
SO	22,347	380	31	0
XPN	13,206	1,426	31	8
IC	18,311	1,300	27	4
JKQ	15,901	1,288	25	4
JKV	2,781	75	4	0
NA	908	70	3	0
Total	**15,660,488**	**1,250,500**	**22,767**	**1,944**

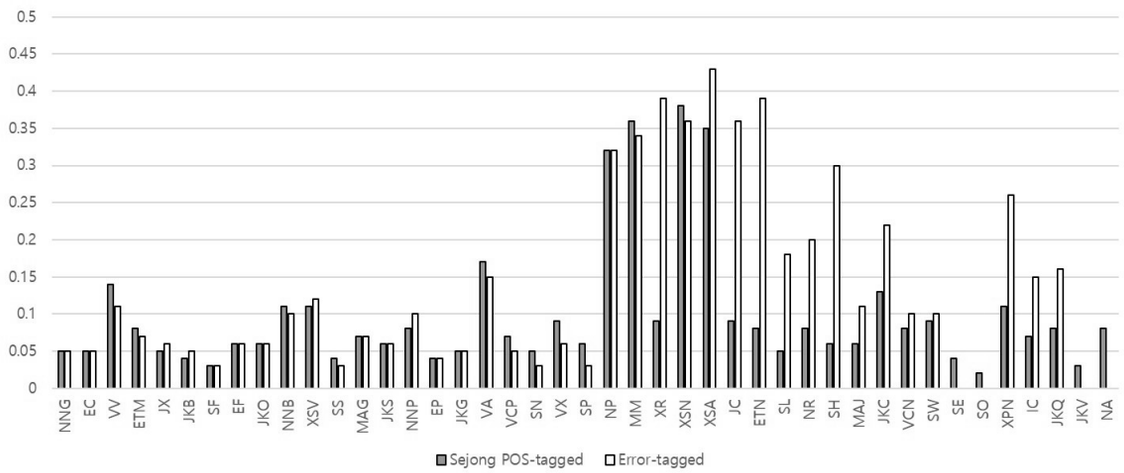


Fig. 6. The Graph for the Error Rate Per Tag

4.4 모집단과 표본집단의 오류 분석

본 논문에서 제안된 방법의 성능을 평가하기 위하여 표본집단(오류부착말뭉치)의 오류의 비율과 모집단(세종말뭉치)의 오류 비율을 비교해 보았다. Table 6은 그 빈도수를 보이며, Fig. 6은 형태소별 모집단과 표본 집단에서의 오류 비율을 비교한 그래프이다. 이를 정량적으로 분석하려고 Equation (6)과 같은 상대 엔트로피(Kullback-Leibler divergence, relative entropy)[23]를 한다.

$$KL(p||q) = \sum_{x \in T} p(x)\log(\frac{p(x)}{q(x)}) \quad (6)$$

여기서 T는 품사집합이고, $p(x)$와 $q(x)$는 각각 표본집단(오류부착말뭉치)과 모집단(세종말뭉치)에서 품사 x의 오류율이다. $p(x)$가 0인 경우, 상대 엔트로피를 0으로 하였으며, 그 결과, $KL(p||q)$는 0.0579로 매우 낮으므로 표본 집단과 모집단이 유사함을 알 수 있었다. 따라서 두 집단의 오류율의 거의 유사하여 제안된 방법이 타당한 방법임을 알 수 있다.

5. 결 론

본 논문에서는 XGBoost와 교차 검증을 이용하여 이미 구축된 품사부착말뭉치로부터 오류를 탐지하는 방법을 제안한다. 제안된 방법은 먼저 오류가 포함된 품사부착말뭉치와 XGBoost를 사용해서 품사부착기를 학습하고, 교차검증을 이용해서 품사오류를 검출한다. 그러나 오류가 부착된 학습 말뭉치가 존재하지 않으므로 일반적인 분류문제로 해결할 수 없다. 따라서 본 논문에서는 매개변수를 조절하면서 학습된 품사부착기의 출력을 비교함으로써 오류를 검출한다. 매개변수를 조절하기 위해서 본 논문에서는 작은 규모의 오류부착말뭉치를 이용한다. 이 말뭉치는 오류 검출 대상의 전체 말뭉치로부터 임의로 추출된 것을 전문가에 의해서 오류가 부착된 것이다. 본 논문에서는 성능 평가의 척도로 정보검색에서 널리 사용되는 정밀도와 재현율을 사용하였다. 또한 모집단의 모든 오류 후보를 수작업으로 확인할 수 없으므로 표본 집단과 모집단의 오류 분포를 비교하여 본 논문의 타당성을 보였다.

앞으로 의존구조부착 말뭉치와 의미역 부착말뭉치에서 적용할 계획이며 반지도학습 방법으로 말뭉치를 구축할 때도 같은 방법으로 적용할 계획이다.

References

[1] J. Kim and G. Kim, Building a Korean Part-of-speech Tagged Corpus: KAIST Corpus, CS-TR-95-99, 1995. (in Korean).

[2] M. Lee, H. Jung, W. Sung, and D. Park, "Verification of POS Tagged Corpus," in *Proceedings. of the 31th Annual Conference on Human and Cognitive Language Technology*, pp.145-150, 2005. (in Korean).

[3] 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.1, pp.227-235, 2013 (in Korean).

[4] E. Eskin, "Detecting Errors Within a Corpus using Anomaly Detection," in *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference*, pp.148-153, 2000.

[5] Q. Ma, B. Lu, M. Murata, M. Ichikawa, and H. Isahara, "On-line Error Detection of Annotated Corpus using Modular Neural Networks," *Lecture Notes in Computer Science*, Vol.2130, pp.1185-1195, 2001.

[6] T. Nakagawa and Y. Matsumoto, "Detecting Errors in Corpora using Support Vector Machines," in *Proceedings*

of the 19th International Conference on Computational Linguistics, pp.1-7, 2002.

[7] M. Dickinson, "Detection of Annotation Errors in Corpora," *Language and Linguistics Compass*, Vol.9, No.3, pp. 119-138, 2015.

[8] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly Detection: Survey," in *Proceedings of ACM Computing Surveys*, Vol.41, No.3, p.15, 2009.

[9] S. Bybers and A. E. Raftery, "Nearest-neighbor Clutter Removal for Estimating Features in Spatial Point," in *Proceedings Journal of the American Statistical Association*, Vol.93, No.442, pp.572-584, 1998.

[10] A. Agovic, A. Banerjee, A. R. Ganguly, and V. Protopescu, "Anomaly Detection in Transportation Corridors using Manifold Embedding," in *Proceedings of the 1st International Workshop on Knowledge Discovery from Sensor Data*, pp.435-455, 2007.

[11] D. Yu, G. Sheikholeslami, and A. Zhang, "Findout: Finding Outliers in Very Large Datasets," in *Proceedings of Knowledge and Information Systems*, Vol.4, No.4, pp. 387-412, 2002.

[12] I. Rehbein, "POS Error Detection in Automatically Annotated Corpora," in *Proceedings of the 8th Linguistic Annotation Workshop*, pp.20-28, 2014.

[13] C. Tianqi and G. Carlos, "XGBoost : A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Vol.16, pp.785-794, 2016.

[14] T. G. Thomas, "Ensemble Methods in Machine Learning," in *Proceedings of Multiple Classifier Systems. MCS 2000. Lecture Notes in Computer Science*, Vol. 1857, 2000.

[15] L. Breiman, "Random Forests," *Machine Learning*, Vol.45, pp.5-32, 2001.

[16] J.-H. Kim, H.-W. Seo, G.-H. Jeon, and M.-G. Choi, "Error Correction Methods for Sejong Corpus," in *Proceedings of the Joint Conference on Marine Engineering and Navigation and Port Research*, pp.435-436, 2010 (in Korean).

[17] N. Kang, E. M. van Mulligen, and J. A. Kors, "Training Text Chunkers on a Silver Standard Corpus: Can Silver Replace Gold?," *BMC Bioinformatics*, Vol.13, No.1, pp.17-22, 2012.

[18] CORPUS, Sejong, 21st Century Sejong Project, The National Institue of the Korean Language, 2010 (in Korean).

[19] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching Word Vectors with Subword Information," *Transactions of the Association for Computational Linguistics*, Vol.5, pp.135-146, 2017.

[20] M. Cheon, C. Kim, J. Kim, E. Noh, K. Sung, and M. Song, "Automated Scoring System for Korean Short-answer Question using Predictability and Unanimity," *KIPS Transaction Software and Data Engineering*, Vol.5, No.11, pp.527-534, 2016.

[21] J. Hong and J. Cha, "Error Correction of Sejong Morphological Annotation Corpora using Part-of-speech tagger and Frequency Information," *Journal of KISS : Software and Applications*, Vol.40, No.7, pp.417-428, 2013.

[22] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz, "Building a Large Annotated Corpus of English: The Penn Treebank," *Computational Linguistics*, Vol.19, No.2. pp. 313-330, 1993.

[23] S. Kullback, Information Theory and Statistics, Dover Publications, 1968.

최 민 석

https://orcid.org/0000-0002-6729-706X
e-mail : ehdgus5136@naver.com
2018년 한국해양대학교 컴퓨터공학과(학사)
2020년 한국해양대학교 컴퓨터공학과(석사)
2020년 ~ 현 재 한국해양대학교 컴퓨터공학과 박사과정
관심분야 : Natural Language Processing, Machine Learning

김 창 현

https://orcid.org/0000-0001-7692-0733
e-mail : chkim@etri.re.kr
1991년 홍익대학교 전계계산학과(학사)
1993년 한국과학기술원 전산학과(석사)
2001년 한국과학기술원 전산학과(박사수료)
2001년 ~ 현 재 한국전자통신연구원 책임연구원
관심분야 : Natural Language Processing, Machine Translation, Dialogue Processing

박 호 민

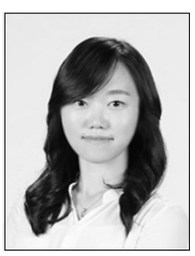
https://orcid.org/0000-0001-7324-387X
e-mail : homin2006@hanmail.net
2017년 한국해양대학교 컴퓨터정보공학과 (학사)
2019년 한국해양대학교 컴퓨터공학과(석사)
2019년 ~ 현 재 한국해양대학교 컴퓨터공학과 박사과정
관심분야 : Natural Language Processing, Information Retrieval, Sentiment Analysis

천 민 아

https://orcid.org/0000-0003-4745-6235
e-mail : minah.cheon@g.kmou.ac.kr
2014년 한국해양대학교 컴퓨터정보공학과 (학사)
2016년 한국해양대학교 컴퓨터공학과 (석사)
2016년 ~ 현 재 한국해양대학교 컴퓨터공학과 박사과정
관심분야 : Natural Language Processing, Named Entity Recognition, Sentence Generation

윤 호

https://orcid.org/0000-0001-9023-8204
e-mail : 4168615@naver.com
2018년 한국해양대학교 컴퓨터정보공학과 (학사)
2020년 한국해양대학교 컴퓨터공학과 (석사)
2020년 ~ 현 재 한국해양대학교 컴퓨터공학과 박사과정
관심분야 : Natural Language Processing, Named Entity Recognition, Word Embedding



남 궁 영

https://orcid.org/0000-0001-7405-0498
e-mail : ynamgoong@g.kmou.ac.kr
2015년 고려대학교 컴퓨터정보학과(학사)
2020년 한국해양대학교 컴퓨터공학과 (석사)
2020년 ~ 현 재 한국해양대학교 컴퓨터공학과 박사과정
관심분야 : Natural Language Processing, Chunking, Dependency Parsing

김 재 균

https://orcid.org/0000-0002-2506-9851
e-mail : jgk20000@naver.com
2018년 한국해양대학교 컴퓨터정보공학과 (학사)
2019년 ~ 현 재 한국해양대학교 컴퓨터공학과 석사과정
관심분야 : Natural Language Processing, Machine Learning, Sentence Generation

김 재 훈

https://orcid.org/0000-0001-8655-2591
e-mail : jhoon@kmou.ac.kr
1986년 계명대학교 전자계산학과(학사)
1988년 한국과학기술원 전산학과(석사)
1996년 한국과학기술원 전산학과(박사)
1988년~1997년 한국전자통신연구원 선임연구원
2001년~2002년 Information Sciences Institute USC 방문연구원
2007년~2008년 Beckman Institute UIUC 방문연구원
1997년 ~ 현 재 한국해양대학교 컴퓨터공학과 교수
관심분야 : Natural Language Processing, Information Retrieval, Corpus Linguistics, Sentiment Analysis