GMM을 이용한 품사 부착 말뭉치의 오류 탐지

최민석[○†], 김창현[‡], 천민아[†], 박호민[†], 윤호[†], 남궁영[†], 김재균[†], 김재훈[†]
한국해양대학교[†], 한국전자통신연구원[‡]
ehdgus5136@naver.com, chkim@etri.re.kr, minah2018@kmou.ac.kr, homin@hanmail.net,
4169615@naver.com, young_ng@kmou.ac.kr, jgk20000@naver.com, jhoon@kmou.ac.kr

Detecting errors on Korean POS tagged corpus using GMM

Min-Seok Choi[○†], Chang-Hyun Kim[‡], Min-Ah Cheon[†], Ho-Min Park[†],
Ho Yoon[†], Young Namgoong[†], Jae-Kyun Kim[†], Jae-Hoon Kim[†]
Korea Maritime and Ocean University[†], Electronics and Telecommunications Research Institute[‡]

요 약

품사 부착 말뭉치란 문장에 포함된 각 단어에 품사 표지를 부착한 말뭉치를 말한다. 이런 말뭉치에는 다양한 형태의 오류들이 포함되어 있으며, 오류가 포함된 말뭉치를 학습 자료로 사용하는 자연언어처리 시스템의 좋은 성능을 기대할 수 없다. 따라서 말뭉치의 일관성이나 정확도는 자연언어처리 시스템의 성능에 많은 영향을 준다. 하지만 말뭉치 구축 과정에서 작업자의 실수가 발생하고 여러 작업자가 작업을 수행하다 보니 일관성을 유지하기가 쉽지 않다. 본 논문에서는 이러한 문제를 해결하기 위해서 GMM을 이용한 군집화를 수행하여 오류 후보를 추출한다. 이를 통해서 말뭉치 구축 과정에서 작업자의 실수를 방지하고 일관성을 유지하고자 한다. 세종품사부착 말뭉치를 대상으로 임의로 오류를 유발시켜 실험한 결과, 재현율 84.74%의 성능으로 오류를 탐지하였다. 향후에 좀 더 높은 재현율을 위해서 자질 확장이나 회귀분석 방법 등을 추진할 계획이다.

주제어: 품사 부착 말뭉치, 오류 탐지, GMM

1. 서론

말뭉치란 자연언어 연구를 위해 특정 목적을 가지고 언어 표본을 추출한 집합을 의미하며, 그 중에서 품사 표지가 부착된 말뭉치를 품사 부착 말뭉치라고 한다. 한국어에서도 다양한 품사 부착 말뭉치가 구축되었다[1-3]. 이 중에 다양한 분야에서 널리 이용되는 말뭉치는 세종말뭉치이다[3]. 세종 말뭉치는 오랜 기간 다양한 사람들이 제작하다 보니 여러 오류를 포함하고 있다[4]. 이런 오류들이 많이 포함된 말뭉치를 사용할 경우 자연언어처리 시스템의 성능 저하가 우려된다. 따라서 성능 향상을 위해서는 오류 수정이 필요하다. 하지만 이런 오류를 수정하기 위해서는 많은 인력과 시간이 필요하기 때문에 비용이 많이 들게 된다. 또한 많은 인력이 수작업을 통해 오류를 수정하기 때문에 일관성을 유지하기가 쉽지 않다.

이와 같은 문제점을 해결하기 위해서 본 논문에서는 품사를 부착하는 과정에서 발생할 수 있는 오류의 가능성을 줄이고 효율적인 오류 수정에 도움을 주기 위해 품사 부착 오류 탐지 방법을 제안한다. 제안 방법은 형태소 말뭉치 구축을 위한 수작업 시 군집화 알고리즘을 이용하여 오류 가능성이 있는 품사 태그를 탐지하여 재확인하여 수정할 수 있도록 한다. 본 논문에서는 GMM을 이용한 오류 검출 방법을 제안한다. 제안된 방법의 평가를 위해서 세종말뭉치를 사용한다. 세종품사부착말뭉치에서는 불필요한 단어가 포함되는 경우, 단어가 삭제된 경우, 단어의 분리가 잘못된 경우, 그리고 형태소에 품사가 잘못 부착된 경우 등 다양한 형태의 오류를 포함하고 있다[4,5]. 그 중 본 논문에서는 형태소에 부착된 품사의 오류를 검출하여 제안된 방법을 평가한다. 그 결과 84.74%의 재현율을 보였다. 그러나 아직 여전히 약 15%의 오류를 검출하지 못하였으므로 좀 더 깊은 연구가 필요한 실정이다.

논문의 구성은 다음과 같다. 2장에서 말뭉치 오류 검출 방법 및 군집화 알고리즘을 설명하고, 3장에서 품사 태그 오류 후보를 추출하는 방법을 제시한다. 4장에서는 제시한 방법에 대하여 분석 및 평가를 수행한다. 마지막 5장에서 결론 및 향후 연구 방안으로 마무리한다.

2. 관련 연구

이 장에서는 말뭉치의 오류 검출 방법에 대해서 간단하게 소개하고 군집화 알고리즘을 간단하게 소개하고자 한다.

2.1 말뭉치 오류 검출

이전의 세종 말뭉치 오류 검출 연구는 오류 검출 기법과 기분석 형태소 분석기를 이용하여 오류 수정 도구를 개발한 [6]이 있다. 또한 [5]는 빈도 정보를 활용한 오류 수정 방법을 제안하였다. 그 외에 세종 말뭉치는 아니지만 형태소분석 말뭉치의 오류 수정 방법에 대한 연구도 이뤄졌다. [7]은 패턴과 같은 규칙을 생성하여 분석 결과의 오류를 해결하는 방법을 제안하였고, 패턴이 아닌 통계 확률 모델을 이용한 오류 수정 방식은 [8]에서 연구되었다.

2.2 군집화 알고리즘

군집화란 데이터를 여러 범주로 그룹화하는 것을 의미한다. 군집화는 비지도 학습 방법이며 데이터 분석을 위한 많은 분야에서 사용되는 방법이다. 이러한 군집화를 수행하면 비슷한 속성 및 특징을 가지는 데이터 요소를 특정 그룹으로 분류 할 수 있다. 또한 비지도 학습의 한 종류인 만큼 학습을 수행하는데 정답 라벨이 필요하지 않다. 사람이 수작업으로 정답을 판별해야 하는 오류 수정 작업을 생각하면 효율과 비용 감소에 효과적이다. 뿐만 아니라 초기 말뭉치를 구축할 때 발생될 수 있는 오류의 검출에 매우 유용할 것이다. 대표적인 군집화 알고리즘은 K-Means Clustering, Mean-Shift Clustering, Density-Based Spatial Clustering of Applications with Noise(DBSCAN), Gaussian Mixture Model(GMM) 등이 있다. 본 논문에서는 GMM을 사용하여 군집화 한다.

2.2.1 GMM

GMM은 복잡한 형태의 확률 분포를 그림 1과 같이 k개의 가우시안 분포를 혼합하여 표현하는 것이다.

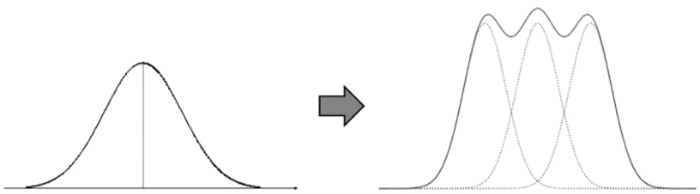

그림 1 혼합 가우시안 분포

k 범주의 확률변수 Z가 있다고 가정하면 확률분포함수는 식 (1)과 같다.

$$p(z=k)=\pi_{(k)} \tag{1}$$

실수 값을 출력하는 확률변수 X는 확률변수 Z의 표본값 k에 따라 기댓값 μ_k, 분산 Σ_k이 달라진다. 이를 수식으로 나타내면 식 (2)로 표현된다.

$$p(x|z)=N(x|\mu_k,\Sigma_k) \tag{2}$$

식 (1)과 (2)를 결합하면 식 (3)으로 정리된다.

$$p(x)=\sum_Z p(z)p(x|z)=\sum_{k=1}^{K}\pi_k N(x|\mu_k,\Sigma_k) \tag{3}$$

단, GMM에서 Z의 값을 알 수가 없으므로, 관측되지 않는다고 가정한다. 따라서 내부에 숨겨진 확률 변수를 포함하는 잠재변수모형이다.

X가 주어졌을 때의 조건부 확률 $p(z|x)$를 정의해 보면 식 (4)처럼 정리할 수 있다.

$$\begin{aligned}\pi_{ik} &\equiv p(z_i=k|x_i)\\ &=\frac{p(z_i=k)p(x_i|z_i=k)}{\sum_{k=1}^{K}p(z_i=k)p(x_i|z_i=k)}\\ &=\frac{\pi_k N(x_i|\mu_k,\Sigma_k)}{\sum_{k=1}^{K}\pi_k N(x_i|\mu_k,\Sigma_k)}\end{aligned} \tag{4}$$

π_{ik}는 i번째 데이터 x_i가 k 범주에서 만들어졌을 확률을 나타내고 k에 대한 조건부확률이라고 한다.

이제 GMM의 모수 추정을 하여야 한다. N개의 데이터에 대한 X의 우도는 식 (5)로 정의된다.

$$\begin{aligned}p(x)&=\prod_{i=1}^{N}p(x_i)\\ &=\prod_{i=1}^{N}\sum_{z_i}p(z_i)p(x_i|z_i)\\ &=\prod_{i=1}^{N}\sum_{k=1}^{K}\pi_k N(x_i|\mu_k,\Sigma_k)\end{aligned} \tag{5}$$

계산의 편의를 위해 식 (5)에 로그를 취하면(로그-우드) 식 (6)으로 정리된다.

$$\log p(x)=\sum_{i=1}^{N}\log\left(\sum_{k=1}^{K}\pi_k N(x_i|\mu_k,\Sigma_k)\right) \tag{6}$$

만약, x_i가 어떤 범주 z_i에 포함되는지 확인할 수 있다면, 범주 확률분포 π_k와 정규분포의 모수 μ_k, Σ_k도 확인할 수 있다. 하지만 실제로는 z_i를 확인할 수 없기 때문에 확률분포함수 $p(x)$를 최대화하는 π_k, μ_k, Σ_k를 비선형 최적화를 통해 구해야 한다.

로그-우드를 μ_k로 미분하여 0이 되도록 방정식을 만들면 식 (7)과 (8)과 같이 정리할 수 있다.

$$0=-\sum_{i=1}^{N}\frac{p(z_i=k)p(x_i|z_i=k)}{\sum_{k=1}^{K}p(z_i=k)p(x_i|z_i=k)}\Sigma_k(x_i-\mu_k) \tag{7}$$

$$\begin{aligned}\mu_k&=\frac{1}{N_k}\sum_{i=1}^{N}\pi_{ik}x_k\\ N_k&=\sum_{i=1}^{N}\pi_{ik}\end{aligned} \tag{8}$$

μ_k는 데이터 전체의 평균값을 취한다는 의미를 나타낸다. 마찬가지로 로그-우드를 Σ_k로 미분하여 계산하면 식 (9)로 정리된다.

$$\Sigma_k=\frac{1}{N_k}\sum_{i=1}^{N}\pi_{ik}(x_i-\mu_k)(x_i-\mu_k)^T \tag{9}$$

마지막으로 로그-우도를 π_k로 미분하여 계산하여야 하는데, π_k는 혼합 계수이므로 추가적인 제약이 존재한다. 따라서 이러한 제약을 추가하여 라그랑주 승수법따라서

이러한 제약을 추가하여 라그랑주 승수법(Lagrange multiplier method)을 사용하여 처리한다. 제약을 추가한 수식은 아래 수식 (10)으로 정의되고, 이 수식을 미분하여 0이 되는 값을 찾으면 수식 (11)로 정리된다.

$$\log p(x) + \lambda\left(\sum_{k=1}^{K}\pi_k - 1\right) \quad (10)$$

$$\pi_k = \frac{N_k}{N} \quad (11)$$

수식 (8),(9),(11)은 모두 조건부확률에 영향을 받기 때문에 자명한(closed-form)해를 가지지 못한다. 따라서 해를 구하기 위해 반복적인 방식의 Expectation Maximization(EM) 알고리즘을 사용하여 구한다.

EM 알고리즘은 조건부확률을 추정하는 E 단계와 모수를 추정하는 M 단계로 이루어지며 이를 번갈아 추정하여 정확도를 높이는 방법이다.

E 단계는 모수가 정확하다고 가정하고 이를 바탕으로 조건부확률을 추정한다.

$$(\pi_k, \mu_k, \Sigma_k) \Rightarrow \pi_{ik}$$

M 단계는 조건부확률이 정확하다고 가정하고 이를 사용하여 모수를 추정한다.

$$\pi_{ik} \Rightarrow (\pi_k, \mu_k, \Sigma_k)$$

위에 두 단계를 반복하면 모수와 조건부확률을 점진적으로 개선할 수 있다.

3. 오류 후보 검출 시스템

그림 2은 본 논문에서 제안한 오류 검출 시스템의 구성도이며 문맥 표상(contextual embedding), 문맥 표상의 차원 축소(dimensionality reduction), 품사별 단어 군집화(clustering), 품사 오류 검출(error detection) 순으로 진행된다. 이하에서 각 과정에 대해서 자세히 기술한다.

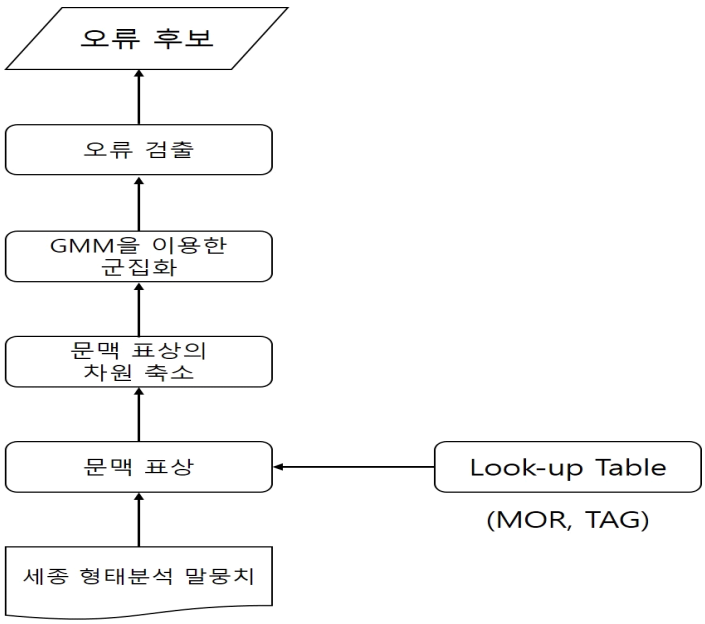

그림 2 오류 후보 검출 시스템 구조도

3.1 Look-up Table 생성

말뭉치의 오류 후보를 탐지하기 위해서는 형태소의 의미를 파악하고 동일 형태소라도 의미에 맞는 태그가 부착되어 있는지를 파악해야 한다. 이를 위해, 형태소의 의미를 효과적으로 표현할 수 있는 형태소 표상이 필요하다. 이런 표상에 대표적인 것은 Word2Vec[10]이 있다. 이 Word2Vec과 세종 형태분석 말뭉치의 1,200만 문장을 사용하여 20차원 크기의 형태소 Look-up Table을 생성하였다. 또한 같은 방법으로 태그 Look-up Table을 생성하였다. 그리하여 2개의 Look-up Table을 생성하여 사전 준비를 수행하였다.

3.2 문맥 표상

형태소의 의미는 독립적이지 않고, 다른 형태소들의 영향을 받는다. 따라서 형태소 하나만으로는 문맥에서의 의미를 파악하기 어려우므로 앞, 뒤의 형태소 정보를 반영하였다. 포함되는 정보는 앞 형태소, 앞 형태소의 태그, 현재 형태소, 뒤 형태소, 뒤 형태소의 태그이며, 총 5개를 합쳐서 100차원의 표상을 생성한다. 현재 형태소가 같더라도 앞, 뒤의 정보가 각 문장마다 다르기 때문에 문장에 따라 해당 형태소의 표상은 바뀌게 된다. 그림 3은 5개의 표상을 합쳐 하나의 문맥 표상을 만드는 것을 표현한 것이다.

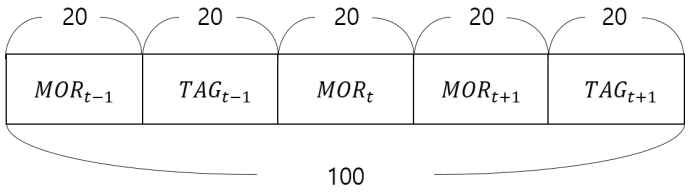

그림 3 문맥 표상의 구조

3.3 문맥 표상의 차원 축소

문맥 표상의 차원이 커질수록 오류 후보 추출을 수행하는데 시간이 오래 걸린다. 이러한 문제를 해결하기 위하여 문맥 표상의 차원 축소를 수행하였다. 차원 축소 방법은 여러 가지가 있지만 본 논문에서는 자기부호화기(AutoEncoder)의 부호부(Encoder)를 이용하여 차원 축소를 수행하였다. 그림 4는 이 논문에서 사용한 자기부호화기를 표현한 것이다. 자기부호화기의 입력과 출력은 동일하게 100차원이고 은닉층(hidden layer)은 하나의 문맥 반영 표상의 크기와 같은 20차원으로 설정하였다.

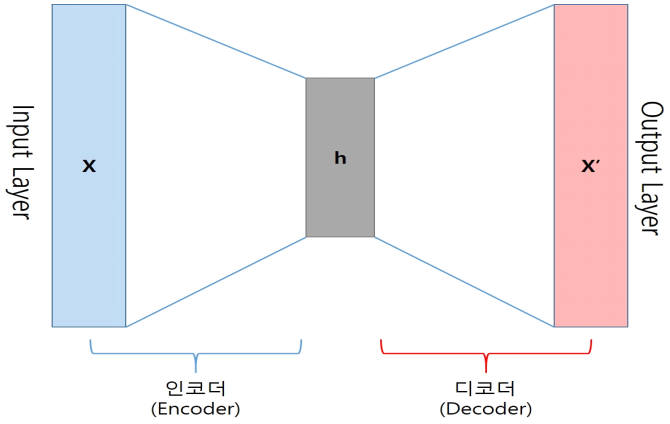

그림 4 자기부호화기의 구조

3.4 GMM을 이용한 품사별 단어 군집화

GMM을 이용하기 위해서는 각 태그별 k의 값을 사용자가 지정해야 한다. 따라서 세종 형태분석 말뭉치를 학습 데이터로 사용하여 실험을 통해 각 태그의 최적의 k의 값을 지정한다. k 값이 결정되면 차원 축소한 문맥 표상을 이용하여 각 태그별로 군집화를 수행한다. 군집화를 수행한 결과로 단어가 속한 가우시안 분포를 알 수 있다. 본 논문에서 실행한 실험에서는 k의 범위를 1 ~ 15로 지정하였다.

3.5 오류 검출

3.4의 방법으로 k값이 결정되고 군집화가 완료되면 이를 바탕으로 다음 단계인 오류 검출을 수행한다. 같은 군집 안에 속하는 단어들이라도 단어별로 특정 가우시안 분포에 속할 확률에는 차이가 존재한다. 본 논문에서는 이 확률에 기준값을 정해서 그 이하를 오류로 판단한다. 각 태그별로 기준값은 0.3부터 0.05씩 증가시켜 0.7까지 실험하였다. 각 실험별로 재현율과 정밀도를 측정 및 분석하여 각 태그의 확률 기준값을 정하였다.

4. 실험 및 분석

제안하는 방법의 성능을 검증하기 위해서 세종 형태분석 말뭉치로 오류 후보 검출을 수행하는 과정을 실험하였다. 세종 형태분석 말뭉치 내의 형태소는 '형태소/태그' 형태이며 형태소 사이에는 '+' 띄어쓰기는 '<SP>'로 제공된다. 상기의 데이터를 이용하여 제안하는 오류 후보 검출 방법의 성능을 측정하였다.

4.1 실험 데이터

실험을 위하여 세종 형태분석 말뭉치의 태그가 모두 정확하다는 가정을 바탕으로 오류 후보 검출 실험을 하였다. 실험 대상은 세종 형태분석 말뭉치 중 임의로 10만개의 문장 총 4,400,000여 개의 형태소이다. 그 중 각 태그별로 1%를 다른 태그로 임의 변경하여 오류를 생성하였다. 표 1은 실제 문장에서 나타난 태그 중에 실험을 위해 정답 태그를 오류 태그로 변경한 개수이다.

표 1. 태그별 오류 생성 개수

태그명	개수	태그명	개수	태그명	개수
NNB	1533	MAG	768	JKG	789
MM	478	XSA	43	SE	88
VA	823	NNP	2403	NP	315
IC	43	VCN	95	JX	1613
SW	292	JKO	1688	SS	2193
NNG	10637	VV	3946	ETM	2194
JC	368	EC	2423	SP	681
SF	1032	XSB	40	XSN	572
JKS	1010	MAJ	68	XSV	76
VCP	547	NR	120	JKQ	374
JKV	13	SH	86	XPN	145
SO	74	EF	1483	SN	813
ETN	220	SL	179	JKC	125
VX	627	JKB	1985	EP	1105

4.2 실험 결과

제안 방법은 형태소 말뭉치 구축을 위한 수작업 시 형태소에 부착된 태그의 오류 여부를 판별한다. 형태소 말뭉치 구축을 위한 수작업 시 오류 발생 가능성을 낮추는 것이 목적이기 때문에 정밀도 보다 재현율을 우선으로 설정하였다. 표 2는 실험 방법에 따라 실험적으로 정해진 태그별 k의 값과 오류로 판단하는 확률 기준값을 나타낸 것이다. 표 3은 표2의 실험값과 제안한 방법으로 오류 검출을 한 결과를 가지고 각 태그별 재현율 및 정확률을 나타낸 것이다.

표 2. 각 태그별 k와 기준값

태그명	k	기준값
NNB	15	0.65
MM	15	0.70
VA	15	0.65
IC	15	0.65
SW	15	0.65
NNG	15	0.65
JC	15	0.65
SF	15	0.55
JKS	15	0.65
VCP	15	0.70
JKV	15	0.70
SO	15	0.70
ETN	15	0.80
VX	15	0.60
MAG	15	0.70
XSA	15	0.70
NNP	15	0.65
VCN	15	0.65
JKO	15	0.65
VV	15	0.70
EC	15	0.60
XSB	15	0.75
MAJ	15	0.65
NR	15	0.70
SH	15	0.6
EF	15	0.65
SL	15	0.70
JKB	15	0.70
JKG	15	0.65
SE	15	0.70
NP	9	0.70
JX	15	0.75
SS	15	0.70
ETM	15	0.70
SP	15	0.70
XSN	15	0.65
XSV	15	0.70
JKQ	15	0.65
XPN	15	0.70
SN	15	0.65
JKC	15	0.65
EP	15	0.65

표 3. 각 태그별 점수

태그명	Recall	Precision	F1-score
SE	0.88	0.70	0.80
SW	0.87	0.73	0.79
NNG	0.87	0.68	0.76
VV	0.87	0.62	0.72
SS	0.87	0.71	0.78
VCP	0.86	0.65	0.74
XSA	0.86	0.61	0.71
JKB	0.86	0.63	0.73
JKG	0.86	0.61	0.71
JX	0.86	0.62	0.72
JKQ	0.86	0.61	0.71
NNB	0.85	0.68	0.76
JKV	0.85	0.71	0.77
SH	0.85	0.69	0.76
MM	0.84	0.70	0.76
SF	0.84	0.74	0.79
VX	0.84	0.63	0.72
EF	0.84	0.64	0.73
SP	0.84	0.73	0.78
XSV	0.84	0.58	0.69
VA	0.83	0.71	0.77
JKS	0.83	0.69	0.75
NNP	0.83	0.62	0.71
NR	0.83	0.68	0.75
XPN	0.83	0.72	0.77
JC	0.82	0.68	0.74
MAG	0.82	0.62	0.70
JKO	0.82	0.70	0.76
XSN	0.82	0.61	0.70
JKC	0.82	0.68	0.74
SO	0.81	0.70	0.75
MAJ	0.81	0.61	0.70
NP	0.81	0.66	0.73
ETM	0.81	0.63	0.70
SN	0.81	0.70	0.75
EP	0.81	0.65	0.72
ETN	0.80	0.63	0.70
EC	0.80	0.72	0.76
SL	0.80	0.71	0.75
VCN	0.79	0.67	0.73
XSB	0.73	0.59	0.65
IC	0.67	0.68	0.67

표 3의 결과를 보면 각 태그별로 재현율이 비슷하게 나타난다. 이를 통해 특정 태그에 관계없이 오류 검출이 잘 된다고 볼 수 있다. 태그 전체의 재현율은 macro-average로 64.93%의 성능을 보였고, micro-average로는 84.74%의 성능을 보였다.

5. 결론

본 논문에서는 품사를 부착하는 과정에서의 오류 발생 가능성을 낮추고 효율적인 오류 수정을 수행하기 위한 방법을 제시한다. 본 논문에서는 GMM을 이용한 군집화를 통해 오류 후보를 검출한다. 이 방법은 군집화를 수행한 집단에서는 비슷한 특성을 보인다는 점을 이용하며, 부착한 품사가 기준값 이하의 확률로 군집화 될 경우 오류 후보로 판단한다. 제안된 방법을 이용하여 오류 후보 검출을 할 경우 84.74%의 재현율을 보이므로 효과적인 방법임을 확인할 수 있었다.

그러나, 표 2의 결과와 같이 대부분 k의 값이 15일 때 가장 좋은 성능을 보인다는 것은 아직 수렴이 완료되지 않았을 가능성이 있을 것으로 생각된다. 또한 재현율을 우선시하다 보니 정밀도가 재현율에 비해 낮다. 따라서 향후 연구로서 k의 범위 조정과 정밀도를 높일 방법을 연구할 계획이다.

감사의 글

이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원(R7119-16-1001, 지식증강형 실시간 동시통역 원천기술 개발)과 2017년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(NRF-2017M3C4A7068187, 한국어 정보처리 원천 기술 연구 개발)

참고문헌

[1] 김재훈, 김길창, “한국어에서의 품사 부착 말뭉치의 작성 요령: KAIST 말뭉치”, 기술문서 CS-TR-95-99, 한국과학기술원 전산학과, 1995.

[2] C.-H. Han and N.-R, Han, Part of Speech Tagging Guidelines for Penn Korean Treebank, Technical Report IRCS Report 01-09, Institute for Research in Cognitive Science, University of Pennsylvania, 2001.

[3] 김홍규, “21세기 세종계획 국어 기초자료 구축 최종연구보고서”, 국립국어원 연구보고서 2007-01-10, 국립국어원, 2007.

[4] 이미경, 정한민, 성원경, 박동인, “품사 표지 부착 말뭉치 검증”, 제17회 한글 및 한국어 정보처리 학술대회 논문집, pp.145-150, 2005.

[5] 최명길, 서형원, 권홍석, 김재훈, “한국어 품사 부착 말뭉치의 오류 검출 및 수정”, 한국마린엔지니어링학회, Vol.37, No. 2, pp.227-235, 2013.

[6] 홍진표, “품사 태거와 빈도 정보를 활용한 세종 형태 분석 말뭉치 오류 수정”, 정보과학회논문지, pp.417-428, 2013.

[7] 이정규, 이상주, 임희석, 임해창, “규칙 기반 한국어 품사 태깅을 위한 어휘 규칙 획득의 수작업 최소한 방안”, 한국정보과학회 학술발표논문집, Vol.24(1B), pp.479-482, 1997.

[8] 김영길, “형태소 어휘 문맥에 기반한 태깅 오류 정정”, 제15회 한글 및 한국어 정보처리 학술대회 논문집, pp.63-68, 2003.

[9] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representation in vector space”, workshop at ICLR, 2013.