한국어에 적합한 단어 임베딩 모델 및 파라미터 튜닝에 관한 연구

최상혁○, 설진석, 이상구
서울대학교, 연세대학교, 서울대학교
{sanghyuk, jamie, sglee}@europa.snu.ac.kr

On Word Embedding Models and Parameters Optimized for Korean

Sanghyuk Choi○, Jinseok Seol, Sang-goo Lee
Seoul National University, Yonsei University, Seoul National University

요 약

본 논문에서는 한국어에 최적화된 단어 임베딩을 학습하기 위한 방법을 소개한다. 단어 임베딩이란 각 단어가 분산된 의미를 지니도록 고정된 차원의 벡터공간에 대응 시키는 방법으로, 기계번역, 개체명 인식 등 많은 자연어처리 분야에서 활용되고 있다. 본 논문에서는 한국어에 대해 최적의 성능을 낼 수 있는 학습용 말뭉치와 임베딩 모델 및 적합한 하이퍼 파라미터를 실험적으로 찾고 그 결과를 분석한다.

주제어: 단어 임베딩, 한국어 Word2Vec, 한국어 말뭉치

1. 서론

단어 임베딩(word embedding)은 자연어로 이루어진 단어를 고정된 차원의 실수 벡터로 변환시키는 과정으로 분산 표현(distributed representation) 이라고도 한다. 이러한 벡터 표현은 다양한 응용의 자연언어처리 기반으로 활용된다[1-4]. 이를 활용하는 응용으로는 기계 번역[5-7], 문서 요약[8], 개체명 인식[9] 등이 있다.

한국어의 경우 교착어에 속하는 특성으로 인해 영어의 단어 임베딩을 형성하는 과정처럼 공백으로 구분된 어절을 그대로 입력으로 사용하는 것은 적합하지 않다. 다양한 조사, 어미로 인해[10] 다른 어절이 동일한 형태소를 포함하는 경우가 많으므로, 이를 고려한 단어 임베딩을 생성해야 한다. [11]의 경우, 형태소를 분리하지 않고 어절을 기본 단위로 단어 임베딩 생성하는 방법을 제안하고 있다. 하지만 이 경우 어근자체에 대한 단어 임베딩의 결과는 구할 수 없는 단점이 있다. 따라서 본 논문에서는 형태소를 기본 단위로 하는 한국어 단어 임베딩을 효과적으로 생성하는 방법을 제안한다.

단어 임베딩을 생성 할 때에는 일반적으로, 1)학습 말뭉치 구성 2)말뭉치의 형태소 분석 3)주요 파라미터 설정(벡터 차원 수, 주변 단어 수 윈도우 크기) 4)모델을 통한 학습 과 같은 단계로 이루어진다. 각각의 단계의 처리 방법이나 파라미터의 값에 따라 실제적인 단어 임베딩의 정확도가 차이가 생기기 때문에, 절적한 수치나 방법을 선택하는 것이 중요하다.

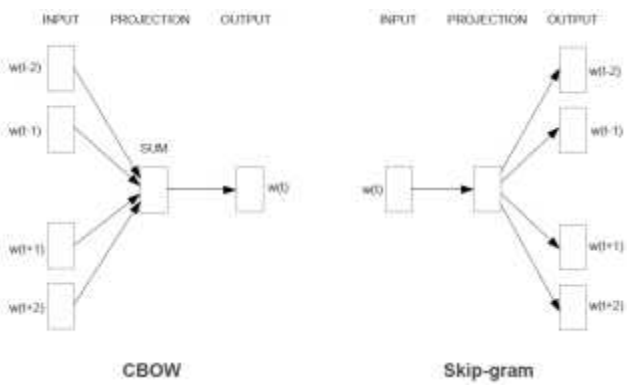
본 논문에서는 현대적인 언어생활을 반영하기 위해 인터넷에 공개된 말뭉치를 활용해 기존 세종 말뭉치보다 약 10배 정도 큰 규모의 말뭉치를 생성해 학습에 이용하였다. 이 말뭉치를 기반으로 GloVe[12], Word2Vec[4] 에서 제안하고 있는 학습 모델을 사용하였으며, 이때 형태소 분석, 파라미터 설정 등이 정확도에 어떤 영향을 주는지 분석하였다.

본 논문의 구성은 2장에서 학습 모델에 관련한 연구들과 이를 형태소 단위로 적용하는 방법을 서술하고, 3장에서는 실제적인 말뭉치 구성과 파라미터 설정이 단어 임베딩의 성능에 어떻게 영향을 주는지 실험을 통해 보인다. 4장에서는 결론과 더불어 향후 연구에 대해 논의한다.

2. 관련 연구

2.1 학습 모델

단어 임베딩 방법으로는 LSI(Latent Semantic Indexing)[13]등 랭킹 및 검색을 위한 모델, 그리고 Word2Vec[4], GloVe[12] 와 같이 자연어처리를 위해 분산 시맨틱 가정에 기반하여 단어 임베딩을 학습하는 모델 등이 있다. 본 연구에서는 후자를 대상으로 실험을 진행하였으며, [14]에 따르면 영어를 비롯한 라틴문자 기반의 언어는 Word2Vec 모델이 전반적으로 가장 높은 성능을 보이고 있다.



<그림 1: CBOW 및 skip-gram 모델[4]>

Word2vec 모델은 continuous bag-of-words(CBOW) 모델과 skip-gram 모델로 나눠진다. CBOW 모델은 모든 투사 레이어(projection layer)가 입력으로 들어오는 단어들 공유하는 피드포워드 신경망 언어 모델 이다. 이 모델은 문장에서 특정 단어를 제외한 앞뒤 주변 단어들이 입력으로 주어지고, 해당 단어들로부터 특정 단어의 정보를 예측하도록 학습된다. 반면 skip-gram 모델은 문장에서 입력으로 들어오는 단어의 주변 단어를 예측하도록 학습되는 모델로, CBOW와는 정 반대되는 구조를 지닌다.

GloVe 모델은 말뭉치 전체에 등장하는 단어의 수가 V 일 때, 한 문장에 같이 등장하는 단어들을 모두 세어둔 행렬(co-occurrence matrix) $X \in R^{V \times V}$를 계산 한 뒤, 각각의 단어 $i,j,...$에 대한 임의의 차원 d 크기의 벡터 $w_i,w_j,... \in R^d$ 를 만들어 w_i와 w_j의 내적이 $\log X_{ij}$와 최대한 유사해지도록 단어 벡터들을 학습하는 모델이다.

본 연구에서는 위 세 가지 모델을 실험에 적용하였다.

2.2 형태소 분석

일반적으로, 영어 단어 임베딩 학습의 입력으로 사용되는 단위는 띄어쓰기 단위로서, 충분히 크기가 큰 말뭉치에서 등장하는 단어의 수가 약 18만개 정도이다[15].

한국어의 경우 교착어에 속하는 특성으로 인해 문장에서 어근이 독립적으로 존재하지 않는다. 또한 60여 가지에 이르는 조사 및 어미로 인해[10] 다른 어절이 동일한 형태소를 포함하는 경우가 많다. 예를 들어, “집에서”와 “집으로”는 동일한 “집” 이라는 형태소를 지니고 있지만 각각 다른 조사가 뒤에 붙음으로서 다른 형태의 어절로 나타난다. 따라서 영어와 같이 띄어쓰기 단위를 한국어 단어 임베딩 학습의 입력으로 사용한다면, 하나의 어근에 대한 너무 다양한 형태가 존재하여 어근 자체가 지니는 임베딩을 학습하기 어렵다. 또한 많은 형태에 따라 비례하여 늘어나는 단어 수는 단어 당 학습에 사용되는 컨텍스트(context words)의 부족으로 이어져 학습 성능을 떨어뜨리는 요인이 된다.

형태소 분석기는 주어진 문장을 형태소단위로 나누어 어근과 조사, 어미 등을 분리해주고 어근 역시 명사 및 동사, 형용사 등으로 구분시켜준다. 본 연구에서는 트위터 형태소 분석기[16]와 꼬꼬마 형태소 분석기[17]를 이용하여 말뭉치에 등장하는 모든 문장에서 독립적으로 의미를 지니지 못하는 형식형태소를 제거하고, 체언과 용언 등 실질적인 의미를 지니는 단위를 단어 임베딩의 입력단위로 사용하였다. 학습에 사용한 품사 태그는 <표 1>과 같다.

<표 1: 학습에 사용한 형태소 품사 태그>

분류	태그		분류	태그	
	꼬꼬마	트위터		꼬꼬마	트위터
일반명사	NNG	Noun	형용사	VA	Adjective
고유명사	NNP				
의존명사	NNB		명사추정	UN	Unknown
동사	VV	Verb	어근	XR	-

3. 실험 및 결과

3.1 평가 방법

성능 평가로는 단어 임베딩 평가에 가장 널리 쓰이는 WS353 테스트셋을 사용하였다. WS353에는 관련도(WS353-R)와 유사도(WS353-S)를 평가하기 위한 353개의 단어 짝이 있으며, 각 단어 짝의 관련도와 유사도의 정도를 0~1 사이의 값으로 나타내고 있다. 각 점수는 여러 전문가들에 의해 매겨진 점수의 평균이다[14]. 원본은 영어로 되어있으며, 본 연구에서는 이를 한국어로 번역하여 사용하였다. 번역 결과, 한국어로 표현되기 어려운 단어들이 약 8% 존재하였으며 해당 단어들은 최대한 적합한 한국어 단어들을 찾아서 대체하였다.

평가지표로는 각 테스트셋에 있는 단어 짝들에 대해, 단어 임베딩을 시행한 후 얻은 두 벡터들 사이의 코사인 유사도 값과 실제 점수와의 피어슨 상관계수를 이용하였다. WS353 데이터의 예시는 <표 2>와 같다.

<표 2: WS353-R과 WS353-S 데이터 예시 (4짝씩)>

WS353-R			WS353-S		
단어1	단어2	관련도	단어1	단어2	유사도
컴퓨터	소프트웨어	0.850	마술사	마법사	0.902
호텔	예약	0.803	호랑이	고양이	0.800
화성	물	0.294	컴퓨터	뉴스	0.447
음료수	귀	0.131	주식	생명	0.092

3.2 말뭉치

학습에 사용된 말뭉치는 나무위키[18], 한국어 위키백과[19] 그리고 인터넷에서 크롤링(crawling)을 통해 얻은 뉴스 기사이다. 이 말뭉치들은 기존 연구에서 사용된 말뭉치[11]보다 좀 더 현대적인 단어의 용법을 포함하고 있으며 말뭉치의 크기가 더 크다는 장점을 지닌다. 본 연구에서는 수집된 말뭉치들이 단어 임베딩 학습에 사용될 수 있도록 URL, 외국어, 각종 특수문자 등 한국어와 관련 없는 표현을 모두 제거하였으며, 뉴스 기사 말뭉치에서는 본문 외에 광고성 문구를 삭제하고 중복 기사를 제거하는 전처리(pre-processing) 과정을 추가로 수행하였다. 각 말뭉치에 대한 자세한 정보는 <표 3>와 같으며, 표에 나와 있는 값들은 전처리 과정을 모두 마친 후에 집계된 통계이다.

<표 3: 학습에 사용된 말뭉치 분석>

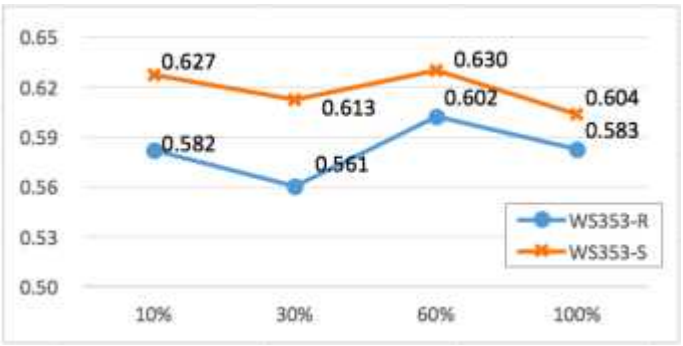
말뭉치	덤프 시점	단어 수	어휘 수	문장당 평균 단어 수
나무위키	2016/05/26	2.51억	440만	12.82개
한국어 위키백과	2016/05/26	0.97억	285만	12.62개
뉴스 기사	2012/06/02~ 2014/11/29	2.26억	232만	13.85개

말뭉치 종류별 단어 임베딩의 성능은 <그림 2>와 같으며, 나무위키와 뉴스 기사를 합친 말뭉치가 가장 높은 결과를 보였다. 영어 단어 임베딩에서 나타나는 최고 성능[20]과 비교해 보았을 때는 낮은 수치지만, 3.1에서 밝혔듯이 평가에 사용된 한국어 WS353 테스트 셋이 영어 기반의 단어를 단순 번역한 테스트 셋임을 감안하였을 때, 온전한 성능비교는 불가능하다.

<그림 2: 말뭉치 종류별 성능 비교>

본 연구에서 사용된 말뭉치의 전체 크기는 약 5Gb 이며, 이 중 10%, 30%, 60%만을 임의로 추출하여 학습하여 평가한 말뭉치 크기에 따른 성능은 <그림 3>과 같다.

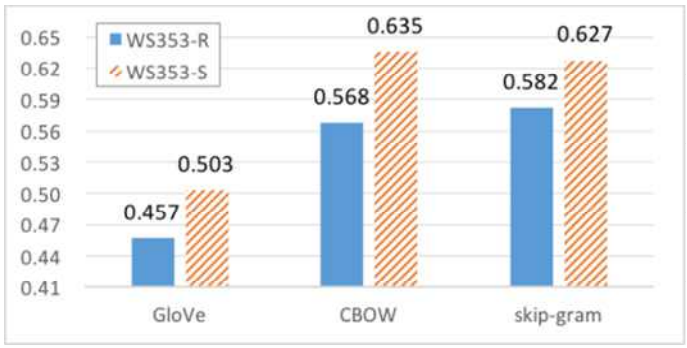
평가결과, 말뭉치의 크기와 성능이 비례해서 증가하지는 않는 것으로 확인되었다. 이는 말뭉치의 크기가 커짐에 따라 동사, 형용사와 같은 품사의 어휘 종류 수는 어느 정도 수렴하는 반면, 명사의 경우 고유명사 등 새로운 어휘가 계속 추가되기 때문에 학습에 사용되는 단어당 해당 단어를 포함하는 문장 수가 거의 증가 하지 않고, 오히려 새로 추가된 어휘들이 노이즈로 작용하는 경우도 존재하기 때문인 것으로 판단된다.



<그림 3: 말뭉치 크기에 따른 성능 비교>

3.3 학습 모델과 형태소 분석

단어 임베딩 모델 중 널리 사용되며 성능이 가장 좋은 것으로 알려진 모델로는 GloVe[12]와 CBOW 기반의 Word2Vec, 그리고 skip-gram 기반의 Word2Vec[4, 14]이 있다. 모델별 성능 비교는 <그림 4>와 같으며, [11]과는 달리 GloVe보다 Word2Vec이 더 높은 성능을 내는 것으로 나타났다. 영어의 경우 skip-gram 기반의 Word2Vec이 가장 좋은 성능을 가진 것으로 알려져 있다[20].



<그림 4: 모델별 성능비교>

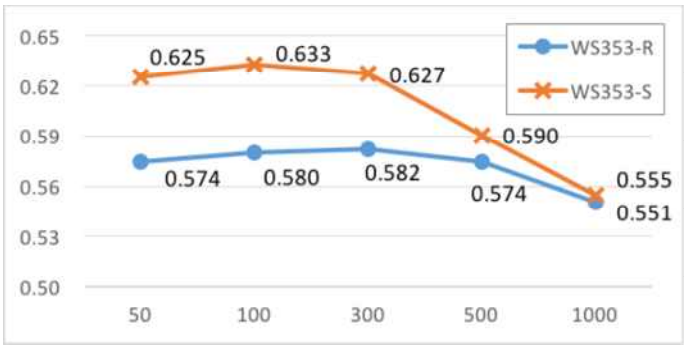
단어 임베딩 학습에 앞서 형태소 분석기를 사용하여 품사 분류(POS Tagging)를 하였다. 이는 형태소 분석을 하지 않은 [11]과는 다른 접근이며, 동일한 글자로 구성된 단어라 할지도 품사가 다르다면 다른 의미를 가진 단어로 취급하기 위해 필요한 처리이다. 또한 조사와 같은 어미를 분리해야 온전한 명사와 동사, 형용사 등을 구분지을 수 있고 어근만을 이용하여야 WS353을 사용한 평가가 가능해지기 때문이다. 형태소 분석에는 트위터 형태소 분석기와 꼬꼬마 형태소 분석기를 사용하였으며, 이에 따른 성능 비교는 <그림 5>와 같다.

<그림 5: 형태소 분석기별 성능 비교>

3.4 학습 파라미터

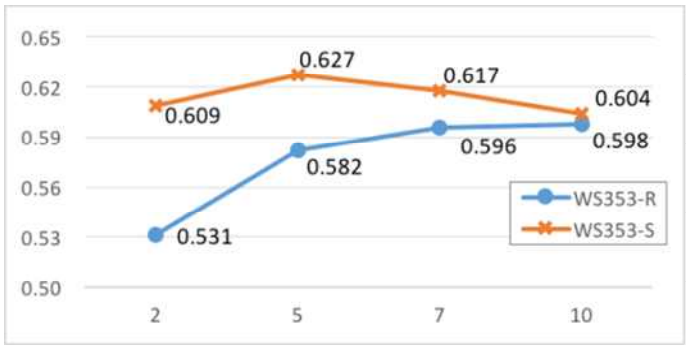
파라미터 실험에서는 한국어에 맞는 단어 임베딩 모델을 학습하기 위한 최적의 파라미터를 추정하였다. 기준이 되는 파라미터 설정은 나무위키와 뉴스 기사를 사용한 말뭉치, skip-gram 기반의 Word2Vec 모델, 꼬꼬마 형태소 분석기 사용, 300 차원, 윈도우 크기 5, 최소 단어 출현 수 제한 50이다.

임베딩 대상이 되는 차원의 수에 따른 성능 비교는 <그림 6>과 같다. 50 이상, 300 이하의 차원에서는 비슷한 성능을 내고 있으며, 300 이상의 차원에서는 차원의 크기가 커질수록 성능이 하락하는 모습이 나타났다. 전체 어휘 수에 따라서 결과는 달라질 수 있지만, 본 연구에서 사용한 말뭉치 크기 수준에서는 100~300차원에서 가장 높은 성능을 보이고 있다.



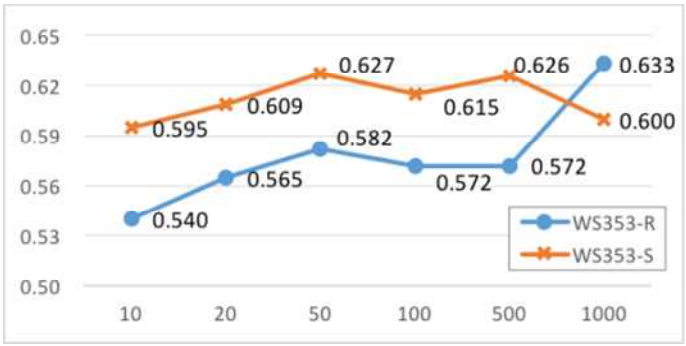
<그림 6 : 임베딩 차원 크기에 따른 성능 비교>

윈도우 크기란 CBOW 혹은 skip-gram 기반의 Word2Vec 모델에서, 한 단어를 기준으로 좌우에 있는 최대 몇개의 컨텍스트용 단어를 학습에 사용할 것인지를 결정하는 값이다. 영어의 경우, 이 값이 5인 경우가 최적으로 알려져 있다 [14, 20, 21]. 본 연구에서 사용한 말뭉치의 문장당 평균 단어 수는 <표 3>에 정리되어 있으며, 전체 말뭉치의 경우 평균 13개 정도의 단어가 한 문장에 나타나고 있다. 윈도우 크기별로 성능을 비교한 결과는 <그림 7>과 같다. 윈도우 크기가 커짐에 따라 WS353-R의 결과가 높아지는 것으로 보아, 앞뒤 단어를 많이 학습에 활용할수록 해당 단어와 관련성이 있는 단어를 잘 학습하는 것으로 보이고, WS353-S의 결과를 보면 해당 단어와 유사한 단어는 앞뒤 5개정도의 단어만을 이용하여 학습하였을 때 가장 잘 찾는 것으로 나타났다.



<그림 7 : 윈도우 크기에 따른 성능 비교>

출현수가 적은 단어의 경우, 해당 단어의 정보를 학습할 컨텍스트(context word)가 부족하게 되고, 다른 단어의 학습에 노이즈로 포함되어 결과적으로 학습을 방해하는 요인으로 작용할 수 있다. 최소 출현수가 일정 값 이하인 경우는 학습에 사용하지 않는 것으로 처리하고 학습 하였을 때의 결과는 <그림 8>과 같다. 실험적으로, 같은 어휘 크기(vocabulary size)를 가진 말뭉치에서 어휘당 평균 출현수는 영어가 한국어보다 약 4.9배 가량 많으며[20], 이는 곧 한국어 어휘가 학습할 컨텍스트가 부족하다는 의미이기 때문에 최소 출현수 제한이 높을수록 어휘당 평균 출현수는 더 올라가며, 결과적으로 성능이 향상되는 것으로 나타났다. 하지만 출현 수 제한은 해당 단어의 학습 배제를 의미하므로, 출현 빈도가 높지 않은 고유명사 등에 대한 임베딩을 얻기 위해서는 말뭉치 크기에 따른 적절한 수준의 최소 출현수를 유지 하여야 한다.



<그림 8 : 최소 출현수 제한 크기에 따른 성능 비교>

4. 결론 및 향후 연구

최근 분산 시멘틱 가정에 기초한 신경망 기반의 단어 임베딩 모델이 많은 자연어 처리 분야에 사용되고 있다. 기존의 많은 연구들이 영어에 특화된 학습모델 및 실험 방법들로 수행되어져 온 반면, 본 연구에서는 언어의 특성을 고려한 다양한 실험을 통해 한국어 처리에 가장 적합한 단어 임베딩 모델을 찾고 및 각각의 파라미터 설정이 성능에 미치는 영향을 분석하였으며, 최종적으로 가장 좋은 성능을 보이는 학습 방법을 도출하였다.

본 실험을 통해 최종적으로 한국어에 적합한 단어 임베딩 학습 방법은 CBOW 혹은 skip-gram 기반의 Word2Vec 모델을 사용하고, 300 차원 크기의 임베딩, 5~7 사이의 윈도우 크기를 설정하며, 최소 출현수 제한은 말뭉치 크기에 따라 적절히 큰 값으로 설정하는 것이 가장 좋은 성능을 보이는 것을 확인하였다. 모든 경우에 있어서 WS353-R과의 피어슨 상관계수가 가장 높았던 수치는 0.633이며, WS353-S의 경우는 0.638이다.

다만 본 연구에서 평가 자료로 사용한 WS353은 영어 단어를 기반으로 만들어진 자료로써 한국어 임베딩을 평가하기에는 부족한 점이 있으며, 관련 연구를 지속하기 위해서는 한국어 특성에 부합하는 임베딩 평가 방법 및 데이터를 구축해야 한다. 또한 기존 모델을 한국어에 적용하는 것에서 나아가 한국어 처리에 적합한 새로운 단어 임베딩 모델을 고안하는 연구가 필요하다.

사사(Acknowledgement)

* 이 논문은 2016년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임 (No. NRF-2016R1C1B2015528)

참고문헌

[1] Bengio, Yoshua, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. "A neural probabilistic language model." In JMLR 3, no. Feb, pp. 1137-1155, 2003.

[2] Collobert, Ronan, and Jason Weston. "A unified architecture for natural language processing: Deep neural networks with multitask learning." In ICML, pp. 160-167, ACM, 2008.

[3] Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. "Efficient estimation of word representations in vector space." arXiv preprint arXiv:1301.3781, 2013.

[4] Mikolov, T., and J. Dean. "Distributed representations of words and phrases and their compositionality." In NIPS, 2013.

[5] Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." arXiv preprint arXiv:1409.0473, 2014.

[6] Luong, Minh-Thang, Hieu Pham, and Christopher D. Manning. "Effective approaches to attention-based neural machine translation." arXiv preprint arXiv:1508.04025, 2015.

[7] Mikolov, Tomas, Quoc V. Le, and Ilya Sutskever. "Exploiting similarities among languages for machine translation." arXiv preprint arXiv:1309.4168, 2013.

[8] Rush, Alexander M., Sumit Chopra, and Jason Weston. "A neural attention model for abstractive sentence summarization." arXiv preprint arXiv:1509.00685, 2015.

[9] Sienčnik, Scharolta Katharina. "Adapting word2vec to named entity recognition." In NODALIDA2015, Vilnius, Lithuania, no. 109, pp. 239-243, 2015.

[10] 김한샘. 현대국어사용빈도조사. Vol. 2. 국립국어원, 2005.

[11] Yang, Hejung, Young-In Lee, Hyun-jung Lee, Sook Whan Cho, and Myoung-Wan Koo. "A Study on Word Vector Models for Representing Korean Semantic Information." In KSSS, Vol 7, no. 4, 2015.

[12] Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "Glove: Global Vectors for Word Representation." In EMNLP, vol. 14, pp. 1532-43. 2014.

[13] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. JASIS, 41:391-407, 1990.

[14] Siwei Lai, Kang Liu, Liheng Xu, and Jun Zhao. "How to Generate a Good Word Embedding?" arXiv preprint arXiv:1507.05523, 2015.

[15] Levy, Omer, Yoav Goldberg, and Ido Dagan. "Improving distributional similarity with lessons learned from word embeddings." In TACL 3, pp. 211-225, 2015.

[16] 트위터에서 만든 오픈소스 한국어 처리기, Github, twitter/twitter-korean-text, https://github.com/twitter/twitter-korean-text, 2016.

[17] 이동주, 연종흠, 황인범, and 이상구. "꼬꼬마: 관계형 데이터베이스를 활용한 세종 말뭉치 활용 도구." 정보과학회논문지: 컴퓨팅의 실제 및 레터 16, no. 11, pp. 1046-1050, 2010.

[18] 나무위키, https://namu.wiki, 2016.

[19] 한국어 위키백과, https://ko.wikipedia.org/, 2016.

[20] Levy, Omer, Yoav Goldberg, and Ido Dagan. "Improving distributional similarity with lessons learned from word embeddings." In TACL 3, pp. 211-225, 2015.

[21] Goldberg, Yoav, and Omer Levy. "word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method." arXiv preprint arXiv:1402.3722, 2014.

부록

부록에 제시된 표는 나무위키와 뉴스 기사를 사용한 말뭉치 전체, skip-gram 기반의 Word2Vec 모델, 꼬꼬마 형태소 분석기 사용, 300 차원, 윈도우 크기 5, 최소 단어 출현 수 제한 50으로 학습한 모델의 word analogy 평가 결과의 일부 이다. 벡터 사이의 관계는 두 단어 벡터의 차에 다른 단어 벡터를 더하여 얻어진다. 예를 들어, 서울-한국+일본 = 도쿄 이다. 이 모델을 활용한 word analogy 테스트는 http://virgon.snu.ac.kr:8000 에 공개하였다.

관계	
남자-왕	여자-왕비
남자-아버지	여자-어머니
한국-서울	일본-도쿄
한국-서울	프랑스-파리
미국-오바마	러시아-푸틴
이과-수학	문과-국어
맨유-리버풀	루니-수아레스
잠실-용인	롯데월드-에버랜드
컴퓨터과학-자연과학	인문학-문학
행복-사랑	불행-비극