형식형태소가 한국어 단어 벡터 생성에 미치는 영향

윤준영○, 김도원, 민태홍, 이재성
충북대학교 소프트웨어학과
junyoung292@cbnu.ac.kr, dowon.0914@daum.net, mintaehong@cbnu.ac.kr, jasonlee@cbnu.ac.kr

Grammatical morphemes' effect on Korean word vector generation

Junyoung Youn○, Dowon Kim, Tae Hong Min, Jae Sung Lee
Dept. of Computer Science, Chungbuk National University

요 약

단어 벡터는 단어 사이의 관계를 벡터 연산으로 가능하게 할 뿐 아니라, 상위의 신경망 프로그램의 사전 학습 데이터로 많이 활용되고 있다. 한국어 어절은 생산적인 조사나 어미 때문에 효율적인 단어 벡터 생성이 어려워 대개 실질형태소만을 사용하여 한국어 단어 벡터를 생성한다. 본 논문에서는 실질형태소와 형식형태소를 모두 사용하되, 형식형태소를 적절하게 분류하여 단어 벡터의 성능을 높이는 방법을 제안한다. 자체 구축한 단어 관계 테스트 집합으로 추출 성능을 평가해 본 결과, 제안한 방법으로 형식형태소를 사용할 경우, 성능이 향상되었다.

주제어: 단어 벡터 생성, 신경망 프로그램, 한국어 어절 표현, 형식형태소

1. 서론

단어 벡터 (word vector)는 자연어의 단어를 다차원의 실수 벡터로 압축하여 표현한 것으로, 단어들의 특징을 잘 표현하여, 각 단어 사이의 여러 가지 관계를 벡터 연산으로도 찾아 낼 수 있다[1-4]. 예를 들어 의미적 관계인 <king> - <man> + <woman> = <queen> 이라든지 문법적 관계인 <write> - <wrote> + <eat> = <ate> 등의 관계를 계산할 수 있다. 또한, 이러한 벡터 표현은 신경망 프로그램의 사전학습(pre-training)의 결과로 사용되어 보다 복잡한 자연어처리, 기계번역, 개체명인식 프로그램 등의 입력 속성으로 활용되고 있다. 이에 따라 품질 좋은 단어 벡터의 개발이 다른 응용프로그램의 성능 향상에 중요한 요소가 되고 있다[5-9].

한국어는 형태소 발달 언어(morphological rich language)로서 띄어쓰기 단위가 어절이며, 영어 등에서의 띄어쓰기 단위인 단어와는 다르게 여러 형태소를 함께 포함하고 있어 비교적 복잡하다. 이런 복잡성 때문에 한국어 어절 벡터를 한 단위로 계산하려면 영어보다는 훨씬 더 많은 학습데이터가 필요하다[10]. 뿐만아니라, 한국어 언어처리 응용프로그램에서도 어절 단위가 아닌 형태소 단위로 처리하는 프로그램들이 많다. 이런 이유로 한국어에 대한 단어 벡터는 어절을 먼저 형태소 단위로 분리한 후, 이를 벡터로 표현한 형태소 벡터를 주로 사용해 왔다[11, 12].

본 논문에서는 이러한 형태소 벡터 특히 내용어에 해당되는 형태소 벡터들을 효율적으로 학습하기 위한 형식형태소의 역할을 분석한다. 이를 위해 한국어 어절을 실질형태소와 형식형태소로 나눈 후, 다양한 형태로 변형하여 학습데이터로 만들었다. 이를 Word2Vec[13]의 학습데이터로 사용하여 단어 벡터를 생성하고, 평가는 유추관계를 평가할 수 있는 100개의 단어 유추 관계 쌍(1쌍은 4개의 단어로 구성)을 새로 구축하여 수행하였다.

2. 단어 벡터의 생성 방법

단어를 벡터로 표현하는 방법으로는 가장 간단한 방법은 one-hot 표기 방법이다. 이 방법은 단어 개수만큼의 차원을 두고, 그 단어에 해당되는 차원만을 1로 표현하고 나머지는 0으로 하는 방법이다. 이 방법은 매우 큰 차원의 벡터가 필요할 뿐만아니라, 각 단어 사이가 모두 독립적으로 처리되어 그 벡터로는 단어의 비교가 불가능하다. 또 다른 방법은 단어를 축소하여(혹은 확대할 수도 있음) 각 차원을 실수로 표현한 것이다. 이 논문에서는 전자는 "one-hot 표기 벡터", 후자를 "단어 벡터"라고 칭한다.

단어 벡터의 학습은 여러 가지가 있을 수 있지만, 현재 Word2Vec[1-3],이나 GloVe[4]가 많이 쓰이고 있다. Word2Vec의 경우, 대상 단어를 매우 큰 차원의 one-hot 벡터로 표기하고, 그 단어의 문맥 단어들을 다시 one-hot 벡터의 합으로 표기한 후, 연산을 통해 그 벡터와 일치할 수 있도록 계산하는 과정에서 단어 벡터를 만들어 낸다. 그림 1은 Word2Vec의 CBOW모델로, 문맥 단어들의 one-hot벡터들을 더하여 신경망의 입력 벡터를 만들고, 변환 과정을 거친 후 다시 대상 단어의 one-hot 벡터를 생성해내는 과정을 나타낸다. 변환 과정은 입력 벡터에 대해 α행렬의 곱으로 선형변환 후 차원이 축소된 은닉층 벡터를 만들고, 이를 다시 β행렬의 곱으로

선형변환하는 과정을 나타낸다. 이를 수식으로 나타내면 (1)과 같다. 여기에서 w는 단어, C는 문맥을 나타낸다 [14].

$$\hat{y} = softmax_{\beta}(\sum_{w \in C} \alpha_w) \quad (1)$$

이 과정에서 만들어진 은닉층은 필요에 따라 여러 차원으로 만들어 낼 수 있으며, 대개 차원이 축소되어 표현되고, 이 벡터를 이용하여 단어 연산을 할 경우, 단어의 의미적 관계나 문법적 관계를 찾아 낼 수 있다[1-3].

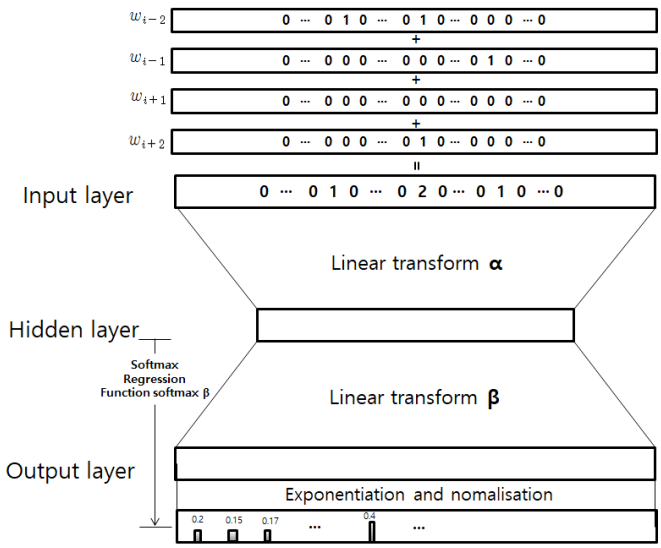

그림 1. Word2Vec의 CBOW모델 학습과정[14]

3. 한국어 단어 벡터 생성

앞 절에서 설명했듯이, Word2Vec의 입력으로는 one-hot 표기의 벡터가 사용된다. 영어의 경우, 비교적 단어 변형이 적어 충분히 많은 어휘에 대해 one-hot 벡터로 표현이 가능하고, 그 외 자주 나타나지 않는 단어에 대해서는 미등록어(unk)의 한 one-hot 벡터로 표기하여 처리한다[1-3].

한국어의 경우, 띄어쓰기 단위인 어절이 매우 생산적이어서, 어절의 종류가 영어 단어보다 훨씬 많아 큰 차원의 one-hot 벡터 표기가 필요하다. 이를 계산하는데는 많은 비용(메모리, 시간)이 들어 현재로서는 실용성이 적다. 따라서, 현재 대부분의 한국어 단어 벡터는 어절 단위보다는 형태소 단위로 분리한 후, 주로 실질형태소만을 취하여 벡터를 생성하고 있다.

하지만, 한국어에서 형식형태소는 단어의 의미를 명확히 해주는 역할을 하고 있어, 이를 반영하면, 더 정확한 실질형태소 기반의 단어 벡터를 생성할 것이다. 이 논문에서는 이를 검증하기 위해 형식형태소를 포함하여 단어 벡터를 생성하고, 그 효과를 측정한다. 한국어 어절을 형태소로 분리하고 띄어쓰기를 고려하여 다음과 같은 방법을 제안한다.

어절은 미리 형태소분석이 되어 있다고 가정하고, 형태소와 태그 정보를 이용하여 분리한다. 형태소는 실질형태소(어휘형태소, Lexical morpheme: Lmor)와 형식형태소(문법형태소, Grammatical morpheme: Gmor)의 두 그룹으로 처리하고, 그 두 그룹의 형태소에 해당되는 태그를 각각의 태그에 대해 실질형태소 태그(Ltag), 형식형태소 태그(Gtag)로 표시한다. 또, 형식형태소를 보다 분포 의미적으로 분석하기 위해 그 단어 벡터를 클러스터링한 번호(C(X))를 사용한다. 이 내용을 정리한 것이 표 1이다.

표 1. 어절 표기에 사용된 약칭

Lmor	실질형태소 (명사, 형용사, 동사)-부사제외
Gmor	형식형태소 (어미, 조사)
Ltag	실질형태소 품사태그
Gtag	형식형태소 품사태그
C(X)	X에 대한 단어 벡터 클러스터 번호
<bln>	띄어쓰기 표시

본 논문에서는 4가지 방법으로 어절을 표기하고 그 성능을 측정한다. 각 방법을 표 1의 표기법에 따라 예와 함께 표현하면 다음과 같다.

예: 나는 학교에서

Mbase: 실질형태소만으로 생성 (기준모델)
단위 형식: Lmor/Ltag
예: 나/np 학교/nng

Mall: 기준모델에 형식형태소 및 태그 포함
단위 형식: Lmor/Ltag Gmor/Gtag
예: 나/np 는/jx <bln> 학교/nng 에서/jkb

Mtag: 기준모델에 형식형태소 태그 포함
단위 형식: Lmor/Ltag Gtag
예: 나/np jx <bln> 학교/nng jkb

Mcls: 기준모델에 형식형태소 클러스터 번호 포함
단위 형식: Lmor/Ltag C(Gmor/Gtag)
예: 나/np C12 <bln> 학교/nng C20

Mcls에서는 형식형태소의 클러스터 번호를 이용하는데, 이는 Mall의 "형식형태소 및 태그" 보다는 덜 세밀하고, Mtag의 "형식형태소 태그" 보다는 좀 더 세밀한 정보를 포함하기 위한 것이다. Mcls의 클러스터는 초기의 연구로서 비교적 간단한 사전 학습을 통해 구하였다. 즉, Word2Vec의 학습데이터를 사전학습 모델인 다음

과 같은 Mpre 모델 형식으로 구성한 후, 클러스터를 구하였다.

Mpre: 실질 형태소 태그와 형식형태소 및 태그
단위 형식: Ltag Gmor/Gtag
예: np 는/jx <bln> nng 에서/jkb

이 클러스터 결과를 이용하여 Mcls 모델의 C(Gmor/Gtag) 번호를 생성한다.

3. 실험

3.1 실험 데이터 및 평가 함수

학습에 사용한 데이터는 세종 형태소 품사분석 말뭉치로 약 1000만 어절이다[15]. 전처리로 ETRI 형태소 분석기를 이용하여 미리 형태소 분석한 결과를 사용하였다. 평가를 위해 같은 유추 관계에 있는 단어 쌍 100개를 수집하여 평가 셋으로 사용하였다. 표2는 평가 데이터의 일부 예이다.

단어 벡터의 평가 방법은 주로 단어유사도 평가와 유추관계 평가가 사용되고 있다[10, 11]. 단어유사도 평가는 의미유사도, 단어관련도 등으로 세분화해서 평가하고, 유추관계는 의미유추와 문법유추의 두 가지로 세분화하여 평가한다. 본 논문에서는 평가 데이터를 새로 구축해야 하는 관계로 단어유추관계 평가만을 시행하였다[1]. 또, 이 논문은 형태소분석을 수행한 후, 실질형태소만을 평가하는 것이므로 문법유추는 제외하고 의미유추만을 사용하여 평가하였다.

의미유추 평가를 위해, 평가 셋에 나타난 단어 관계식(A-B+C=D)을 계산하고, 계산된 단어(D) 벡터와 가장 유사한 단어를 모든 단어 벡터와 비교하여 유사도 순으로 상위 10개를 나열하고, 정답이 나온 순위를 점수로 계산하였다. 즉, 각 단어 i에 대한 순위(0부터 9)를 $rank_i$라 하면 n개의 데이터에 대한 점수(score)는 다음 식(2)와 같다. 즉, 각 순위점수의 평균에 10을 곱한 것으로 최고점은 100점이 된다.

$$score = 10 \times \frac{1}{n} \times \sum_{i=1}^{n} (10 - rank_i) \qquad (2)$$

표 2. 평가 셋의 일부 예

A	B	C	D
원	한국	미국	**달러**
아들	남자	여자	**딸**
할머니	여자	남자	**할아버지**
병원	의사	경찰	**경찰서**
농구공	농구	야구	**야구공**
알파벳	미국	한국	**한글**
하늘	위	아래	**땅**
공자	유교	불교	**석가모니**
왕	남자	여자	**여왕**
서울	한국	일본	**도쿄**

3.2 실험 결과

실험을 위해 Word2Vec[13]의 하이퍼 파라미터를 조절하였으며, 윈도우 사이즈는 11, 최소 빈도 컷(cutoff)은 5에서 높은 성능을 보였으며, CBOW와 Skip-gram을 비교해 본 결과 Skip-gram이 우수하여 이 파라미터 값을 사용하였다. 또, 클러스터를 구하기 위한 사전 학습 모델인 Mpre 모델은 클러스터 크기를 200으로 하여 학습하였고, 이 결과를 Mcls 모델의 클러스터 번호 함수로 사용하였다.

실험 결과는 표 3과 같고, 이를 그래프로 표현한 것이 그림 1이다. 결과에서 보듯이 전체적인 성능은 40점에서 45점 사이로 다른 한국어 연구(유추관계 약 67%[10], 유사도 약 61점[11])에 비해 그리 높지 않은 편이다. 그 이유는 [10]은 약 6억7천여 단어, [11]는 약 5억7천여 단어를 사용하여 본 연구의 데이터보다 학습데이터가 더 크기 때문이며, 단어 벡터 생성 방법의 차이(GloVe[4], Skip-gram[1-3]), 유사도 계산 공식의 차이(cosine add, cosine multiply)[16] 등에서 기인한 것으로 판단된다. 하지만, 본 논문은 주로 실질형태소와 형식형태소의 역할을 상대적으로 비교하기 위한 것이므로, 다른 연구 결과와의 자세한 성능 비교 및 분석은 생략한다.

각 모델을 최고 성능을 보인 차원(표3의 굵은 글씨)을 중심으로 비교해 보면, 일반적으로 많이 사용하는 Mbase 모델을 기준으로 보아 Mall모델은 매우 성능이 낮았다. 이 Mall모델은 특별한 고려 없이 형식형태소를 형식형태소 태그와 함께 사용한 경우로 형식형태소의 추가가 오히려 성능을 하락시켰다. 형식형태소 태그만 추가한 Mtag모델은 Mbase와 거의 비슷한 성능을 보였다. 형식형태소를 적절히 클러스터링하여 추가한 Mcls 모델은 기준 모델 Mbase보다 성능이 향상되어 최고점에서 1.0점 더 좋은 성능을 보였다. 이는 적절하게 형식형태소를 분류하여 사용할 경우, 한국어 단어 벡터 생성에 효과적임을 보여준다.

1) 다른 연구에서 한국어 단어 유추 평가 데이터가 개발되어 있었지만, 실험 시 평가 데이터를 얻을 수 없어, 본 연구에서 새로 구축한 데이터를 사용하였다.

표 3. 각 차원에서의 모델의 성능

차원	100	200	300	400
Mbase	42.4	**44.0**	39.0	41.6
Mall	40.4	**42.1**	40.7	42.1
Mtag	**43.8**	42.5	42.2	41.1
Mcls	44.2	**45.0**	39.5	41.2

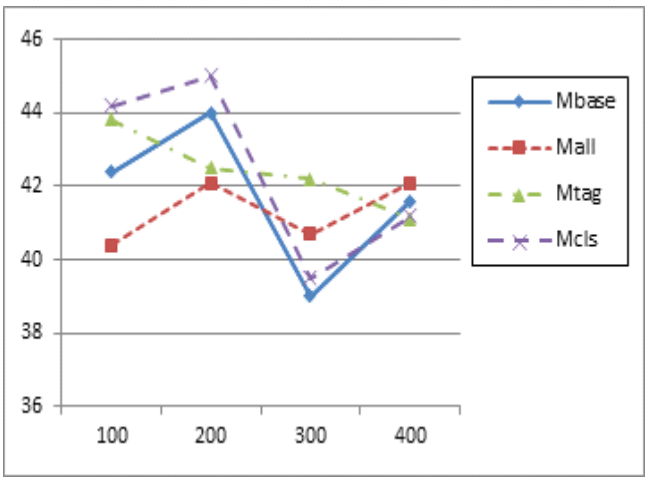

그림 2. 차원별 각 모델의 성능 변화

5. 결론

한국어 단어 벡터 생성 시 주로 한국어 어절 중 실질 형태소만을 분리하여 단어 벡터 학습에 사용한다. 본 논문에서는 한국어 단어 벡터 학습시 형식형태소를 추가하여 그 효과를 분석하였다. 즉, “형식형태소 및 태그”, “형식형태소 태그”, “형식형태소의 클러스터”를 각각 추가하여 학습하였고, 형식형태소를 적절히 클러스터링하여 그 정보를 사용할 경우, 단어 벡터의 품질이 높아짐을 알 수 있었다. 향후에는 충분히 큰 학습데이터와 평가 데이터를 구축하여 실험하고, 더 다양한 계산 방식으로 이를 평가하여, 품질 좋은 한국어 단어 벡터를 생성하기 위한 방법을 연구할 계획이다.

사사(Acknowledgement)

이 논문은 2017년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No. 2017R1D1A3B03035676)

참고문헌

[1] Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean, "Efficient estimation of word representations in vector space," ICLR workshop, 2013.

[2] Mikolov, Tomas, Wen-tau Yih, and Geoffrey Zweig, "Linguistic regularities in continuous space word representations," In Proceedings of HLT-NAACL, 2013.

[3] Mikolov, Tomas and J. Dean, “Distributed representations of words and phrases and their compositionality.” In proceedings of NIPS, 2013.

[4] Pennington, Jeffrey, Richard Socher, and Christopher D. Manning, “GloVe: Global vectors for word representation,” In Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 1532-1543, 2014.

[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] Siencnik, Scharolta Katharina, “Adapting word2vec to named entity recognition,” In proceedings of NODALIDA 2015, Vilnius, Lithuania, no. 109, pp. 239-243, 2015.

[10] 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 proceedings of KSSS, Vol 7, no. 4, 2015.

[11] 최상혁, 설진석, 이상구, “한국어에 적합한 단어 임베딩 모델 및 파라미터 튜닝에 관한 연구”, 제28회 한글 및 한국어 정보처리 학술대회 논문집, 2016.

[12] Kang, Myung Yun, Bogyum Kim, and Jae Sung Lee, “Word Sense Disambiguation Using Embedded Word Space,” Journal of Computing Science and Engineering, Vol 11, no. 1, 2017.

[13] Word2Vec web page, https://code.google.com/p/word2vec/

[14] Wilson, Benjamin, “An overview of word2vec,” presentation file, Berlin ML Meetup, 2014.

[15] 국립국어원, 21세기 세종계획 최종성과물 (2011년 12월 수정판), 2011.

[16] Levy, Omer and Yoav Goldberg, “Linguistic

regularities in sparse and explicit word representations," In proceedings of the eighteenth conference on computational natural language learning, 2014.