JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 20, No. 4, pp. 825-833, Apr. 2019

한국어 단어 임베딩을 위한 Word2vec 모델의 최적화

강 형 석 · 양 장 훈*
서울미디어대학원대학교 뉴미디어학부

Optimization of Word2vec Models for Korean Word Embeddings

Hyungsuc Kang · Janghoon Yang*
Department of New Media, Seoul Media Institute of Technology, Seoul 07590, Korea

[요 약]

단어 임베딩 모델로 최근 인기를 끌고 있는 word2vec 모델을 한국어 처리에 적용하는 사례가 늘고 있다. Word2vec 모델에 대한 표준적인 성능 검증 방식은 유추 검사이지만, 최근까지 한국어에 적합한 유추 검사는 개발되지 않았다. 이런 이유로 한국어 word2vec 모델에 대한 하이퍼파라미터 최적화는 보통 유사도 검사를 통해 이루어졌다. 본 논문에서는 기존의 유사도 검사뿐만 아니라, 한국어의 언어학적 특성을 반영한 유추 검사를 이용해서 하이퍼파라미터 최적화를 시도했다. 그 결과, 학습 알고리즘으로는 skip-gram 방식이 CBOW보다 우수하고, 단어 벡터의 크기는 300 차원이 적절하며, 문맥 윈도의 크기는 5에서 10 사이가 적절함을 발견하였다. 또한, 말뭉치의 크기에 따라서 학습될 어휘 수를 적절하게 제한하는 데 사용되는 최소 출현빈도 값은 총 어휘 수가 100만개 이하일 경우에는 1로 설정하여 가급적 학습될 어휘 수를 적정 수준으로 유지하는 것이 중요함을 확인하였다.

[Abstract]

In Korean language processing, there are more and more cases of applying word2vec models, which are recently gaining popularity as word embedding models. Analogy tests are used as standard evaluation methods for word2vec models; however, no analogy test suitable for Korean has been developed yet. For this reason, similarity tests have been employed in optimizing hyperparameters for Korean word2vec models. This paper attempts to optimize some of these hyperparameters through the existing similarity test as well as a new analogy test that reflects certain features intrinsic to the Korean language. It turns out that the training algorithm of skip-gram is better than that of CBOW, the optimal dimension of word vectors is 300 and the optimal size of the context window lies between 5 and 10. It is also found that keeping the size of vocabulary trained in the corpus at a reasonable level is critical, which result in setting the hyperparameter of minimum count as 1 for the size of vocabulary less than one million.

색인어 : 유추 검사, 하이퍼파라미터, 유사도 검사, 단어 임베딩, Word2vec

Key word : Analogy test, Hyperparameter, Similarity test, Word embedding, Word2vec

http://dx.doi.org/10.9728/dcs.2019.20.4.825

Received 11 March 2019 ; **Revised** 25 April 2019
Accepted 26 April 2019

***Corresponding Author; Janghoon Yang**

Tel: +82-2-6393-3237
E-mail: jhyang@smit.ac.kr

Ⅰ. 서 론

최근에 딥러닝 등 인공지능 관련 기반 기술들이 발전하면서 다양한 영역에서 다양한 형태의 자연어 처리(NLP; Natural Language Processing) 기술의 활용이 급속히 증가하고 있다. 언어를 신호 공간에 매핑하는 단어 임베딩(word embedding)은 자연어 처리의 가장 기초적인 단계이며 현재까지 여러 방식이 제안되어 왔다. 이중 word2vec모델[1, 2]이 많은 분야에서 활용되고 있다.

다른 모델들과 마찬가지로 word2vec 모델의 성능도 어떻게 파라미터를 설정하는지에 따라 크게 달라진다. Word2vec 모델은 학습에 의해서 진화하는 모델이고, 학습을 시작하기 전에 결정해야 하는 파라미터인 하이퍼파라미터(hyperparameter)를 적절히 선택하는 것이 중요하다[3]. 하지만 최적의 하이퍼파라미터 조합을 찾는 것은 쉽지 않은데, 그 이유는 복수의 하이퍼파라미터들이 상호 독립적이지 않고 서로 영향을 주고받을 수 있기 때문이다. 보통의 경우, 예상되는 최적의 조합을 중심으로 각 하이퍼파라미터의 값을 조정해 가면서 해당 모델의 성능을 평가하는 방식으로 하이퍼파라미터를 최적화(튜닝)한다. 이런 최적화가 끝나면, 본격적으로 대량의 기계 학습을 시작할 수 있다.

Word2vec 모델의 하이퍼파라미터는 단어 벡터(word vector)의 크기와 문맥 윈도(context window)의 크기 등이다. 하지만 한국어 고유의 특성을 반영한 word2vec 모델 검증 방식이 거의 존재하지 않아서, 한국어 word2vec 모델의 하이퍼파라미터를 최적화하는 데는 한계가 있다.

본 연구에서는 한국어의 언어학적 특성을 반영한 성능 검증 방식을 이용해서, 한국어 word2vec 모델을 위한 최적의 하이퍼파라미터를 탐색하고자 한다. 기존 한국어 word2vec 최적화 연구에서는 유사도 검사(similarity test)를 기반으로 최적화 연구가 수행되었으나[10], 이 논문에서는 일반적인 영어 word2vec 모델의 성능 검증 방식 중 하나인 유추 검사(analogy test)를 한국어에 적합하도록 적절히 변형한 검사 방법과 기존의 유사도 검사를 모두 고려하여. 한국어 word2vec 모델의 하이퍼파라미터 최적화하고 기존 연구의 결과와 비교할 것이다.

Ⅱ. 관련 연구

2-1 Word2vec 모델의 개요

일반적인 단어 임베딩 모델과 마찬가지로 , word2vec 모델은 비슷한 의미를 갖는 단어는 비슷한 문맥에서 등장한다는 언어학의 분산 가설(distributional hypothesis)[4]에 근거한다. 이 모델은 CBOW(Continuous Bag-Of-Words)와 SG(Skip-Gram)라는 2가지 학습 알고리즘(training algorithm)을 제안하는데, 전자는 $2k$ 개의 주변 단어(흔히 문맥 윈도라고 불림)가 주어졌을 때 그

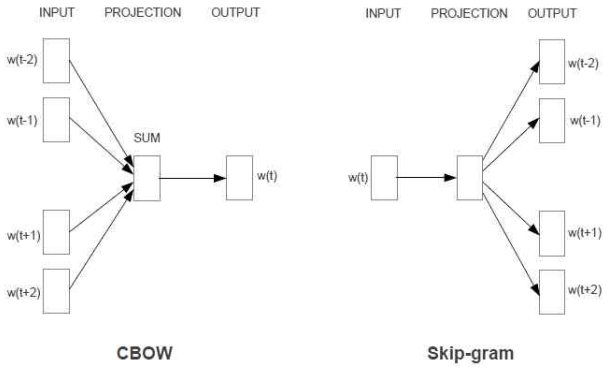


그림 1. Word2vec 모델링의 2가지 학습 알고리즘 (출처: [1])
Fig. 1. Two training algorithms of word2vec modeling (source: [1])

중심에 특정 단어가 나타날 조건부 확률을 계산하고, 후자는 중심 단어가 주어졌을 때 특정 조합의 주변 단어 $2k$ 개가 나타날 조건부 확률을 계산하는 알고리즘이다. Word2vec 모델의 학습 과정은 주어진 말뭉치(corpus)의 문장을 스캔해 가면서, 이런 조건부 확률이 최댓값이 되도록 단어 벡터(word vector)의 값을 업데이트하는 과정이다. 아래의 그림 1은 이런 2가지 학습 알고리즘($k = 2$)의 개요를 보여준다.

흔히 CBOW와 SG는 서로 반대되는 개념으로 설명되지만, 조건부 확률을 계산하는 데 근본적인 차이가 존재한다. CBOW는 k 값과 무관하게 중심 단어에 대해 한 번만 조건부 확률을 계산하지만, SG는 중심 단어에 대해 $2k$ 번의 조건부 확률을 계산한다. 다시 말해, $2k$ 개의 주변 단어가 주어졌을 때 CBOW에서는 중심 단어의 벡터가 한 번만 업데이트되지만, SG에서는 중심 단어의 벡터가 $2k$ 번 업데이트된다. 따라서 CBOW의 학습량은 k 값에 무관하지만, SG의 학습량은 k 값에 비례하게 된다 (흔히 문맥 윈도의 크기는 k 값으로 표시된다). 이런 학습량의 차이로 인해, 일반적으로 SG의 성능이 CBOW보다 나은 것으로 알려져 있다[1].

2-2 Word2vec 모델의 성능 검증 방식

다양한 word2vec 모델 검증 방식이 존재하지만[5-7], 그중 유사도 검사와 유추 검사가 대표적이다.

1) 유사도 검사

일반적인 단어 임베딩 모델의 성능 평가에서 자주 쓰이는 유사도 검사는 특정 단어 쌍의 유사도나 관련도(relatedness)가 임베딩 모델에 얼마나 잘 반영되었는지를 확인하는 검사이다. 영어 단어 임베딩 모델의 유사도 검사에서 가장 많이 사용되는 WordSim353 데이터세트[8]는 353개 영단어 쌍과 해당 쌍의 유사도/관련도를 소수(13명 또는 16명)의 영어 화자가 평가한 점수(1~10)의 평균으로 구성된다(e.g. tiger–cat–7.35). 이 데이터세트를 이용한 실제 유사도 검사는 각 단어 쌍에 해당하는 2가지 단어 벡터의 코사인 유사도(cosine similarity)의 분포와 이 유

사도/관련도 점수의 분포가 얼마나 유사한지를 평가하는 것이다. 즉, 각 단어 쌍에 대한 유사도/관련도 점수와 코사인 유사도의 분포를 상관분석해서 Pearson 계수를 구하고, 이 값이 클수록 해당 단어 임베딩 모델의 성능이 높은 것으로 간주한다.

하지만 이런 유사도 검사에는 한계가 있다. 우선 해당 데이터세트의 유사도/관련도 점수는 13명 또는 16명의 주관적인 판단에 근거한다. 즉, 이 점수가 영어 화자 전체의 의견을 대표한다고 보기에는 상당한 무리가 있다. 게다가, WordSim353 데이터세트는 문항 수가 한정적(353개)이고 대부분 명사로만 구성되어 있어서, 영어 단어 임베딩 모델의 극히 제한적인 측면만을 평가한다고 볼 수 있다. 따라서 유사도 검사가 단어 임베딩 모델의 대략적인 성능을 검증할 수는 있어도, 하이퍼파라미터의 최적화에 필요한 미세한 성능 비교에는 부적합할 수도 있다.

2) 유추 검사

Word2vec 모델과 함께 제안된 성능 검증 방식인 유추 검사는 벡터 공간에 임베딩된 단어 벡터들이 그림 2와 같은 의미론적/문법적 관계를 잘 반영하는지를 확인하는 검사이다. 좋은 성능의 단어 임베딩 모델은 그림 2(a)와 같은 남녀 관계에 있는 단어들의 벡터를 일정한 거리(그림에서 파란색 화살표)에 위치하도록 임베딩해야 한다. 마찬가지로 그림 2(b)와 같은 단수-복수 관계의 단어 벡터들도 일정한 거리(그림에서 빨간색 화살표)에 위치하도록 임베딩해야 한다. 일반적으로 유추 검사 세트는 해당 언어 고유의 다양한 의미론적/문법적 관계를 검사하는 문항으로 구성된다.

영어 word2vec 모델의 유사도 검사에 쓰이는 세트는 흔히 GATS(Google Analogy Test Set)라고 불리고[9], 15개 섹션(e.g. 수도와 국가)으로 나누어진 총 19,544개의 문항(e.g. Athens-Greece-Cairo-Egypt)으로 구성되어 있다. 유사도 검사에 사용된 WordSim353 데이터세트에 비해, GATS는 다양한 품사로 구성된 훨씬 더 많은 검사 세트를 제공한다는 점에 주목할 필요가 있다. 이런 면에서, 하이퍼파라미터 최적화와 같은 미세한 성능 비교에는 유사도 검사보다 유추 검사가 더 적합하다고 볼 수 있다.

GATS의 각 문항은 동일한 의미론적/문법적 관계를 갖는 두 쌍의 단어로 구성된다(e.g. boy-girl-dad-mom). 예를 들어, 학습이 완료된 word2vec 모델에서 단어 boy, girl, dad, mom의 벡터를 각각 *v(boy), v(girl), v(dad), v(mom)*이라고 하고, 학습된 전체 단어 벡터의 집합을 *W*라고 하자. 그러면 boy-girl-dad-mom이라는 문항에 대한 유추 검사는 다음과 같은 벡터 관계식이 성립하는지를 확인하는 것이다.

$$v(mom) = \arg\min_{x \in W} |v(boy) - v(girl) - v(dad) + x| \quad (1)$$

즉, $|v(boy) - v(girl) - v(dad) + x|$의 값을 최소로 만드는 단어 벡터 *x*가 *v(mom)*과 같으면 해당 문항은 정답으로 처리하고, 그 외에는 오답으로 처리한다. 유추 검사는 모든 문항에 대해 (1)과 같은 벡터 관계식이 성립하는지 확인하는 과정이다. 흔히 유추 검사의 결과는 전체 문항에 대한 정답률(accuracy)로 요약되고, 이를 통해 특정 하이퍼파라미터 조합의 word2vec 성능을 다른 하이퍼파라미터 조합의 성능과 정량적으로 비교할 수 있다.

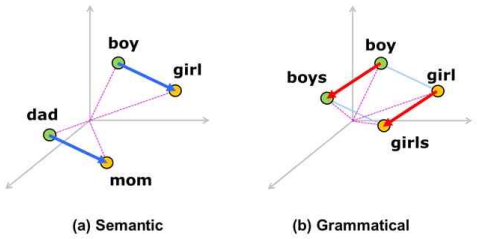


그림 2. 단어 벡터의 의미론적/문법적 관계
Fig. 2. Semantic/grammatical relationships of word vectors

2-3 한국어 Word2vec 모델의 최적화

학습 알고리즘(CBOW 또는 SG)의 선택과 함께, word2vec 모델에서 사전에 결정해야 하는 주요 하이퍼파라미터는 단어 벡터의 크기, 문맥 윈도의 크기 및 최소 출현빈도(min_count)이다. 단어 벡터의 크기는 단어가 임베딩되는 벡터 공간의 차원을 의미하고, 문맥 윈도의 크기는 word2vec 모델 특유의 하이퍼파라미터로서 흔히 주변 단어 개수($2k$ 개)의 절반에 해당하는 k 값으로 표기한다. 그리고 최소 출현빈도는 말뭉치 전체의 어휘 중 출현빈도가 낮은 어휘를 모델링에서 제외시키는 기준이 되는데, 이를 통해 학습되는 어휘의 수를 적절히 조절할 수 있다. 따라서 word2vec 모델의 최적화는 이런 4가지 하이퍼파라미터(이후 학습 알고리즘도 편의상 하이퍼파라미터에 포함시키겠다)에 대한 최적의 조합을 찾는 과정이다.

한국어 word2vec 모델의 하이퍼파라미터 최적화에 관한 기존 연구[10]에 따르면, 2가지 학습 알고리즘에 따른 성능 차이는 크지 않고, 벡터 크기는 300 그리고 윈도 크기는 5~7 사이가 적절하며, 최소 출현빈도 제한은 말뭉치 크기에 따라 적절히 큰 값으로 설정하는 것이 좋다고 한다. 하지만 이 연구의 결과는 영어 word2vec 모델에 적용되는 WordSim353 데이터세트를 한국어로 번역한 유사도 검사를 근거로 한 것이다. 즉 일반적으로 사용되는 영어 word2vec 모델의 유추 검사를 사용하지 않았다는 한계가 있다.

게다가, [10]에서는 한국어 고유의 특성을 드러내는 조사와 어미 같은 문법 형태소(grammatical morpheme)를 학습에서 제외시켰는데, 이는 한국어 NLP의 목적에 따라 문제가 될 수 있다. 예를 들어, 학습된 한국어 word2vec 모델링의 결과가 주제 분류(topic classification)에 쓰인다면, 이런 문법 형태소를 학습되지 않아도 큰 문제가 되지는 않을 것이다. 반면에, 감성 분류(sentiment classification)에 적용된다면, 문장의 의미를 결정하는 데 큰 역할을 하는 조사와 어미가 학습되지 않은 단어 임베딩 모델은 효율이 떨어질 가능성이 있다.

따라서 문법 형태소까지 포함하는 경우, 한국어 word2vec 모

델의 최적 하이퍼파라미터가 달라지는지를 확인할 필요가 있다. 일반적으로 단어 벡터의 크기는 학습되는 총 어휘 수에 따라 결정될 것이고 추가되는 문법 형태소의 수는 전체 어휘 수에 비해 제한적일 것이므로(대부분의 언어에서 문법 형태소의 개수는 제한적이다), 최적의 벡터 크기는 크게 달라지지 않을 것이다. 하지만 학습 알고리즘, 문맥 윈도의 크기 및 최소 출현빈도에 대해서는 좀 더 고려해 볼 필요가 있다. 우선 영어 word2vec 모델의 경우, 유추 검사를 기준으로 SG 알고리즘이 CBOW보다 더 우수하고 알려져 있다[1]. 그리고 [10]의 경우, 말뭉치에서 조사와 어미 같은 문법 형태소를 제거했기 때문에, 이런 문법 형태소까지 포함한다면 최적의 문맥 윈도 크기가 5~7 사이 값보다 커질 가능성이 있다. 마지막으로 말뭉치의 크기에 따라 최소 출현빈도로 어떤 값이 적절한지도 확인해 볼 필요가 있다.

특히, 형태소 분석기(morphological analyzer)를 전처리 단계에서 적용한 한국어 word2vec 모델의 경우는 최적의 윈도 크기가 달라질 가능성이 크다. 왜냐하면, 기본적으로 체언(명사, 대명사, 수사)과 결합된 조사가 분리되고, 특히 용언(동사, 형용사, 서술격 조사)은 2개 이상의 형태소로 분리될 가능성이 크다 (e.g. 들어오셨던 = 들어오 + 시 + 었 + 던). 즉, 한 문장을 어절(띄어쓰기) 단위가 아니라 형태소 단위로 나누면 단어의 개수가 증가하게 된다. 게다가, 한국어 문장에서 일반적으로 주어는 맨 앞에 그리고 서술어는 맨 마지막에 배치되는 경우가 많으므로, 주어와 서술어 사이의 거리가 상당히 멀어질 수 있다. 한국어 문장에서 주어와 서술어의 호응[1]이 중요하기 때문에, 문맥 윈도의 크기를 예를 들어 10 정도로 증가시켜야 할 수도 있다.

Ⅲ. 실험 데이터 및 방법

3-1 모델링 방법

본 연구에서 한국어 word2vec 모델링에 사용한 말뭉치는 나무위키(https://namu.wiki/)와 위키백과(https://ko.wikipedia.org/)의 대용량 덤프 파일을 이용해서 구성했다. 수집된 대용량 파일에서 문장 부호, 수식, 외국어, 특수 문자, URL 등 한국어와 무관한 표현은 제거되었다. 이렇게 정제된 텍스트 파일 기준으로 전체 말뭉치의 용량은 3.52 GB이다. 이 텍스트 파일의 토큰화(tokenization)를 위해 사용한 형태소 분석기는 mecab 품사 태거(POS tagger)이다. 오픈 소스(open source)로 공개된 한국어 형태소 분석기/품사 태거 중 mecab 태거의 성능이 가장 우수함을 이전 연구에서 확인한 바 있다[11, 12]. Mecab 태거로 분석된 말뭉치는 약 6.6억 개의 형태소(토큰)로 구성되어 있고, 총 어휘 수는 형태소 기준으로 935,548개이다.

1) 언어학에서 호응(呼應)은 앞에 어떤 말이 오면 거기에 대응하는 말이 따라오는 현상을 일컫는다. 예를 들어, '결코'라는 단어 뒤에는 부정의 뜻을 갖는 서술어가 따라온다.

형태소 분석, word2vec 모델링 및 성능 검증은 파이썬(Python) 코드로 구현되었다. 형태소 분석에는 한국어 정보처리를 위한 파이썬 패키지 KoNLPy[13, 14]를 그리고 word2vec 모델링에는 NLP 파이썬 패키지인 Gensim[15, 16]을 사용했다.

3-2 모델링의 성능 검증 방식

WordSim353 데이터세트에는 한국어로 적절히 번역될 수 없거나 시대적 혹은 지역적 특색이 강한 단어 쌍이 포함되어 있다. 예를 들어, 해당 데이터세트에는 체스 용어인 'king'과 'rook' 쌍이 포함되어 있는데, 이를 단순히 '왕'과 '성장(城將)'으로 번역하는 것은 의미가 없을 것이다. 그리고 80년대와 90년대 미국 언론에서 자주 함께 언급되었을 'Arafat'와 'terror'라는 단어 쌍도 포함되어 있는데, 이런 단어들은 제외되는 것이 바람직할 것이다. 따라서 한국어 word2vec 모델링 결과에 대한 유사도 검사를 위해, 이런 단어 쌍 60 개를 제외한 293개의 단어 쌍을 한국어로 번역한 데이터세트[2]를 사용했다.

한국어 word2vec 모델을 검증하기 위해 이런 식으로 WordSim353 데이터세트를 번역해서 적용한 경우가 많지만 [10, 17], 해당 데이터세트는 영어 화자가 생각하는 각 단어 쌍의 유사도/관련도이므로 한국어 단어 임베딩의 평가에 사용하기에는 근본적인 한계가 있음을 주의해야 한다.

그리고 한국어 word2vec 모델에 대한 유추 검사를 위해서, 이전 연구에서 개발한 한국어 유추 검사 세트(KATS; Korean Analogy Test Set)[12]를 사용했다. 이 유추 검사 세트는 한국어의 문법적 유추 관계까지 검증할 수 있도록, 기존의 유추 검사 세트인 GATS 및 BATS(Bigger Analogy Test Set)[18]를 확장한 것이다. KATS는 다양한 한국어 품사로 구성된 다수의 문항(4,620개)으로 구성되어 있다.

3-3 검증할 하이퍼파라미터의 조합

실험에서 검증하고자 하는 word2vec 모델의 하이퍼파라미터는 학습 알고리즘, 벡터 크기, 윈도 크기 및 최소 출현빈도이다. 우선 최소 출현빈도는 아래 그림 3에서처럼, 모델링에 포함되는 어휘 수를 전체, 50만 개, 10만 개, 5만 개 및 만 개 정도가 되도록 각각 1, 3, 110, 421 및 4923으로 정할 것이다.

그리고 최적의 하이퍼파라미터 조합은 학습 알고리즘 SG, 벡터 크기 300, 윈도 크기 10이라고 임시로 가정하고, 이 조합을 기준으로 아래 표 1과 같은 총 28개의 하이퍼파라미터 조합을 구성해서 그 성능을 비교할 것이다. 표 1에서 하이퍼파라미터 조합들의 첫 번째 범주는 벡터 크기와 윈도 크기를 각각 300과 10으로 고정하고 학습 알고리즘(TA; Training Algorithm)과 최소 출현빈도를 변경한 조합들이다. 비슷한 방식으로, 두 번째

2) 실제로 사용된 유사도 검사 세트는 다음의 구글 공유 문서를 참조하라.
https://drive.google.com/file/d/1FyTAp0G1sNYv-K76HyGMoFnDkNsTqkXI/view?usp=sharing

표 1. 실험에서 사용된 하이퍼파라미터 조합
Table 1. Combinations of hyperparameters used in the experiment

Cat.	Fixed hyperparameters	Varied hyperparameter	min_count
1	Vector size = 300 Window size = 10	TA = CBOW, SG	1, 3, 110, 421, 4923
2	TA = SG Window size = 10	Vector size = 100, 3000, 500, 700	3, 110, 421
3	TA = SG Vector size = 300	Window size = 5, 10, 30, 50	3, 110, 421

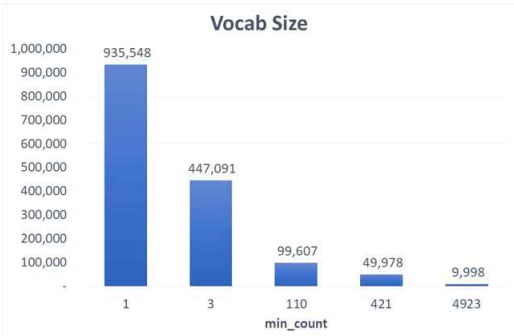


그림 3. 최소 출현빈도에 따른 어휘 수의 차이
Fig. 3. Differences in vocabulary size according to minimum counts

범주는 학습 알고리즘과 윈도 크기를 각각 SG와 10으로 고정한 채 벡터 크기와 최소 출현빈도에 변화를 준 조합들이고, 세 번째 범주는 학습 알고리즘과 벡터 크기를 각각 SG와 300으로 고정한 채 윈도 크기와 최소 출현빈도에 변화를 준 조합들이다.

Ⅳ. 실험 결과

4-1 모델 학습에 소요된 시간

위 그림 4는 각 하이퍼파라미터 조합으로 word2vec 모델링에 수행하는 데 걸린 시간을 정리한 도표이다. 그림 4에서 제일 먼저 확인할 수 있는 점은 최소 출현빈도(min_count)의 차이는 학습 시간에 큰 영향을 미치지 않는다는 점이다. 즉, 학습되는 어휘의 수가 학습 시간에 결정적인 영향을 미치지는 않는다고 볼 수 있다. 예를 들어, 총 어휘 수가 약 100분의 1로 줄어든 경우(min_count=4,932)에도 알고리즘 SG로 학습하는 데 걸린 시간은 7.88 시간에서 5.94 시간으로 약 2시간(약 25%)밖에 줄어들지 않는다. 따라서 학습 시간의 측면에서, 최소 출현빈도를 증가시켜서 총 어휘 수를 줄이는 것이 큰 이득을 주지는 않는다고 볼 수 있다.

그림 4. 하이퍼파라미터 조합에 따른 학습 시간의 차이
Fig. 4. Differences in training hours according to hyperparameter combinations

하지만 그림 4(a)에서 확인할 수 있듯이, CBOW에 비해 SG 학습 알고리즘의 경우 약 4배 이상의 학습 시간이 소요되었다. 그리고 그림 4(b)와 그림 4(c)에 따르면, 대략적으로 벡터 크기가 2배 증가하면 학습 시간은 약 0.75배 증가하고, 윈도 크기가 2배 증가하면 약 0.45배 증가한다. 따라서 학습 시간에 결정적인 영향을 미치는 것은 학습 알고리즘이라고 할 수 있다.

4-2 유사도 검사 결과

아래의 그림 5는 각 하이퍼파라미터 조합의 word2vec 모델링에 대한 유사도 검사의 결과(Pearson 계수)를 비교한 도표이다. 모든 하이퍼파라미터 조합 중 가장 큰 Pearson 계수를 보인 조합(그림 5(c)에서 초록색 테두리로 표시된 조합)은 TA=SG, Vector size=300, Window size=30, min_count=110의 조합이고, 그 값은 0.670이다.

학습 시간의 경우처럼, 전체적으로 최소 출현빈도에 따른 Pearson 계수의 차이는 그다지 크지 않다(단 min_count=4,932인 경우는 예외). 비록 미세한 차이이긴 하지만, 일반적으로 최소 출현빈도가 작을수록 Pearson 계수가 약간 더 높게 나온다. 즉 그림 5 전체에 따르면, 최소 출현빈도가 작은 경우가 단어 임베딩 성능이 더 낫다고 볼 수 있다. 최소 출현빈도에 대한 본 연구의 결과가 기존 연구[10]와 다른 이유는 말뭉치의 총 어휘 수의 차이에서 기인한 것으로 보인다. 본 연구에서 사용된 총 어휘 수는 약 100만 개이고, 기존 연구[10]의 어휘 수는 최소 400만 개다. 따라서 100만 개 이하의 어휘를 포함하는 작은 말뭉치의 경우, 최소 출현빈도를 증가시켜서 총 어휘 수를 줄이는 것은 오히려 성능을 약간 떨어뜨리는 것으로 보인다.

그림 5(a)는 학습 알고리즘이 SG인 경우 모델링의 성능이 더 좋음을 보여 준다. 특히 최소 출현빈도가 작을수록 성능 차이가 더 크다. 그리고 그림 5(b)에 따르면, 벡터 크기는 300이 가장 적절해 보인다. 마지막으로 그림 5(c)에서 확인할 수 있는 바는 윈도 크기가 30 ~ 50인 경우 모델링 성능이 가장 우수하다는 것이다.

한국어 word2vec 모델의 하이퍼파라미터 최적화에 대한 기존 연구[10]와 본 연구의 유사도 검사 결과 사이에는 큰 차이가 존재한다. 즉, 최적의 벡터 크기를 제외하면 학습 알고리즘, 윈도 크기, 최소 출현빈도에 대해 서로 다른 결론을 내리고 있다. 이런 차이는 조사와 어미 같은 문법 형태소의 포함 여부에서 기인한 것으로 보인다. 문법 형태소가 포함된 한국어 word2vec 모델의 경우, 학습 알고리즘은 SG가 CBOW보다 더 우수하고, 윈도 크기는 30 이상인 경우가 더 우수하며, 최소 출현빈도가 작은 경우가 약간 더 우수한 것으로 보인다.

하지만 이 유사도 검사의 결과를 최적의 하이퍼파라미터라고 결론 내리기에 충분하지 않다. 앞서 언급한 바 있는 유사도 검사 자체의 한계(명사로만 이루어진 제한적인 문항 수 및 소수의 영어 화자에 의한 유사도/관련도 평가) 및 영어와 한국어의 차이를 감안하면, KATS에 의한 유추 검사의 결과까지 고려해서 최적의 하이퍼파라미터를 결정하는 것이 타당할 것이다.

4-3 유추 검사 결과

그림 6은 각 하이퍼파라미터 조합의 word2vec 모델링에 대한 유추 검사의 결과(정답률)를 비교한 도표이다. 모든 하이퍼파라미터 조합 중 가장 높은 정답률을 보인 조합(그림 6(a)에서 초록색 테두리로 표시된 조합)은 TA=SG, Vector size=300, Window size=10, min_count=1의 조합이고, 그 값은 39.3%이다.

최소 출현빈도가 4,923인 경우(어휘 수가 만 개 이하인 경우로, KATS 문항에 포함된 상당수의 어휘가 학습되지 않아서 정답률이 현저히 낮은 것으로 보인다)를 제외하면, 최소 출현빈도에 따른 정답률의 차이는 역시 그다지 크지 않다. 유사도 검사의 결과와 마찬가지로, 일반적으로 최소 출현빈도가 작을수록 정답률이 약간 더 높게 나온다.

그림 5. 하이퍼파라미터 조합에 따른 Pearson 계수의 차이
Fig. 5. Differences in Pearson coefficients according to hyperparameter combinations

유사도 결과를 보여주는 그림 5(a)와 마찬가지로, 그림 6(a)도 SG 알고리즘이 모델링 성능 면에서 CBOW보다 더 나음을 보여 준다. 그림 6(b)는 벡터 크기가 300 이상이면 모델링 성능이 대체로 동일함을 보여 주는데, 이는 최적의 벡터 크기가 300이라는 그림 5(b)의 결론과 상충되는 것은 아니다.

하지만 그림 5(c)의 유사도 검사 결과와 상충되는 부분은 그림 6(c)의 유추 검사 결과이다. 즉, 그림 5(c)가 보여 주는 추세와 달리, 그림 6(c)에 따르면 윈도 크기가 커질수록 정답률이 조금씩 떨어진다. 좀 더 엄밀히 살펴보면, min_count=3인 경우는 window size=10인 경우가 window size=5인 경우보다 미세하게 더 우수하고, 나머지 경우는 window size=5인 경우가 window size=10인 경우보다 약간 더 우수하다. 따라서 유추 검사의 결과에 근거한 최적의 윈도 크기는 5에서 10 사이의 범위에 존재하는 것으로 보인다.

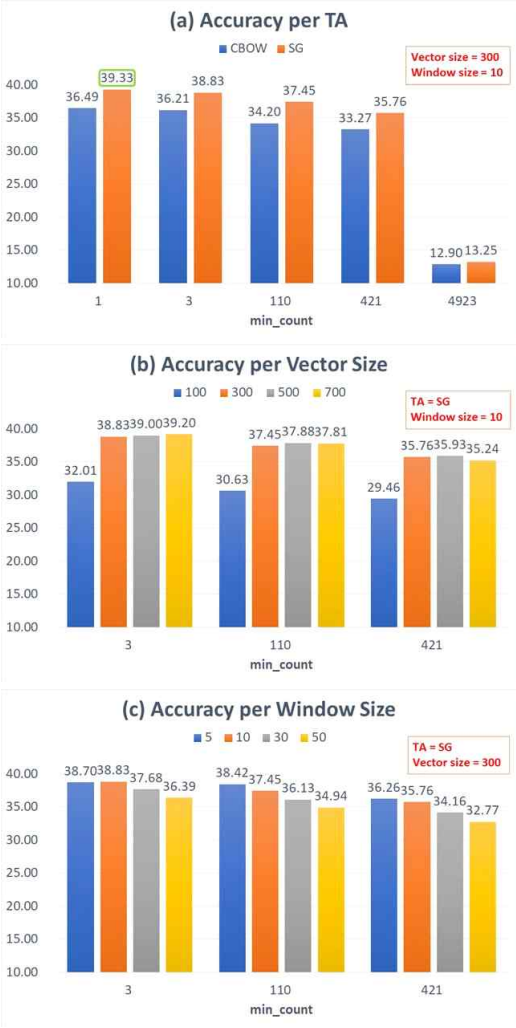


그림 6. 하이퍼파라미터 조합에 따른 정답률의 차이
Fig. 6. Differences in accuracy according to hyperparameter combinations

4-4 결과의 종합

아래 표 2는 유사도 검사만을 이용한 기존 연구[10]의 하이퍼파라미터 최적화의 결과 및 유사도/유추 검사를 이용한 본 연구의 결과를 비교한 표이다. 최적의 벡터 크기 (300)에 대해서는 기존 연구와 동일한 결과를 갖지만 이외의 다른 하이퍼파라미터에 대해서는 여러 가지 다양한 이유로 차이가 다소 발생하였다.

기존 연구[10]에서는 학습 알고리즘에 대한 차이를 확인하지 못했지만, 본 연구의 유사도/유추 검사에 따르면 일관되게 SG 알고리즘이 우수하다는 결론을 얻었다. 이런 차이는 조사나 어미 같은 문법 형태소의 포함 여부에서 기인한 것으로 보인다. 즉, 문법 형태소가 word2vec 모델링에 포함되는 경우, SG 알고리즘이 CBOW보다 더 우수한 성능을 보인다고 할 수 있고, 이는 일반적인 영어 word2vec 모델의 경우와 동일한 결론이다.

표 2. 최적의 하이퍼파라미터에 대한 비교
Table 2. Comparison of optimal hyperparameters

		Previous Research [10]	This research	
Evaluation method		Similarity test	Similarity test	Analogy test
Optimal	TA	CBOW or SG	SG	
	V. dimension	300	300	
	Window size	5 ~ 7	30 ~ 50	5 ~ 10
	min_count	as big as possible	as small as possible	

표 2에서 확인할 수 있듯이, 최적의 윈도 크기는 기존 결과와 차이가 검사의 종류에 따라서 다르게 발생하고 있다. 기존 연구[10]에서는 윈도 크기를 2 ~ 10 사이의 값으로만 제한했고 문법 형태소를 제외했기 때문에, 최적의 윈도 크기가 5 ~ 7 사이라는 결론은 한계가 있는 것으로 예상된다. 또한, 본 연구에서 최적의 윈도 크기는 유사도/유추 검사 방식에 따라 다르게 나왔다. 하지만 유사도 검사가 유추 검사에 비해 훨씬 제한적인 범위(명사에 한정된 293개의 문항)에 대한 성능 평가라는 점을 고려하면, 다양한 품사와 4,620개의 문항으로 구성된 유추 검사 결과에 따라서 최적의 윈도 크기(5~10)를 선정하는 것이 타당할 것으로 예상된다. 따라서 앞서 2-3절에서 예상했던 것과는 달리, 형식 형태소를 포함하는 한국어 word2vec 모델링에서도 문맥 윈도를 크게 증가시킬 필요는 없어 보인다.

최소 출현빈도에 대한 기존 연구[10]와 본 연구의 차이는 사용된 말뭉치의 크기에서 비롯된 것으로 보인다. 본 연구에서 사용된 말뭉치의 총 어휘 수가 기존 연구에 배해 대략 1/4 수준이기 때문에, 최소 출현빈도를 크게 설정해서 총 어휘 수를 줄일 필요가 없어 보인다. 즉, 100만 개 정도의 어휘를 포함하는 작은 말뭉치의 경우, 가능한 모든 어휘를 학습하는 것이 word2vec 모델의 성능을 높이는 방법으로 보인다.

Ⅴ. 결 론

본 논문은 유사도 검사뿐만 아니라 word2vec 모델의 표준적인 성능 검증 방식인 유추 검사를 이용해서, 한국어 word2vec 모델의 하이퍼파라미터에 대한 최적화를 시도했다. 그 결과, 문법 형태소를 포함하는 한국어 word2vec 모델에서 학습 알고리즘은 SG, 벡터 크기는 300, 윈도 크기는 5~10 사이가 최적이다. 그리고 총 어휘 수가 100만 개 이하인 작은 말뭉치의 경우, 최소 출현빈도를 가능한 한 작게 설정해서 모든 어휘를 포함시키는 것이 바람직하다.

이 연구의 중요한 한계점 중에 하나는 영어를 기준으로 만들어진 유사도 검사 세트를 한국어로 번역해서 사용했다는 점이다. 게다가 사용된 유사도 검사 세트는 명사에만 국한된 293개의 문항으로만 이루어져 있다. 만약 KATS의 경우처럼, 유사도 검사 세트도 한국어 고유의 특성을 반영하고 다양한 품사와 더 많은 문항 수를 갖도록 변형한다면, 하이퍼파라미터의 최적화를 위한 기준으로 사용할 수 있을 것이다. 따라서 한국어를 기

반으로 확장된 유사도 검사 세트의 개발과 이를 기반으로 한 단어 임베딩 모델의 최적화에 대한 추가적인 연구가 요청된다.

또 다른 추가 연구 과제로는 총 어휘 수가 100만 개보다 훨씬 많은 큰 말뭉치를 이용한 한국어 word2vec 모델의 하이퍼파라미터 최적화이다. 큰 말뭉치의 경우, 최소 출현빈도를 크게 설정해서 학습되는 총 어휘 수를 줄여야 할 필요가 있을 것이다. 이럴 경우 다른 하이퍼파라미터의 최적치가 달라질 수도 있으므로, 이에 대한 검증이 필요하다.

마지막으로, word2vec 이외의 다른 단어 임베딩 모델(e.g. GloVe[19] 및 fastText[20])의 하이퍼파라미터 최적화에 본 연구의 유사도 검사 및 유추 검사를 적용할 수 있다. 이를 통해, 전반적인 한국어 단어 임베딩 모델의 최적 하이퍼파라미터를 확인해서 각 단어 임베딩 모델의 장단점을 확인할 수도 있을 것이다.

감사의 글

이 논문은 2019년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업입니다(과제번호 : NRF-2017R1A2B4007398).

참고문헌

[1] T. Mikolov et al., “Efficient estimation of word representations in vector space,” *arXiv preprint arXiv:1301.3781*, 2013.

[2] T. Mikolov et al., “Distributed representations of words and phrases and their compositionality,” *Advances in neural information processing systems*, pp. 3111-3119, 2013.

[3] M. Claesen and B. De Moor, “Hyperparameter search in machine learning,” *arXiv preprint arXiv:1502.02127*, 2015.

[4] M. Sahlgren, “The distributional hypothesis,” *Italian Journal of Disability Studies*, Vol. 20, pp. 33-53, 2008.

[5] M. Baroni, G. Dinu and G. Kruszewski, “Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors,” Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vol. 1, pp. 238-247, 2014.

[6] T. Schnabel et al., “Evaluation methods for unsupervised word embeddings,” *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing,* pp. 298-307, 2015.

[7] Siwei Lai et al., “How to generate a good word embedding,” *IEEE Intelligent Systems*, Vol. 31, No. 6, pp. 5-14, 2016.

[8] L. Finkelstein et al., “Placing Search in Context: The Concept Revisited,” *ACM Transactions on Information Systems*, Vol. 20, No. 1, pp. 116-131, January 2002.

[9] ACL(Association for Computational Linguistics) Wiki. Analogy (State of the art) [Internet]. Available: https://aclweb.org/aclwiki/Analogy_(State_of_the_art).

[10] Sanghyuk Choi, Jinseok Seol and Sang-goo Lee, “On Word Embedding Models and Parameters Optimized for Korean,” *Proceedings of the 28th Annual Conference on Human and Cognitive Language Technology*, Busan Korea, pp. 252-256, 2016.

[11] Hyungsuc Kang and Janghoon Yang, “Selection of the Optimal Morphological Analyzer for a Korean Word2vec Model,” Proceedings of the KIPS Fall Conference 2018, Busan Korea, pp. 376-379, Nov. 2018.

[12] Hyungsuc Kang and Janghoon Yang, “The Analogy Test Set Suitable to Evaluate Word Embedding Models for Korean,” Journal of Digital Contents Society, pp. 1999-2008, 2018.

[13] Eunjeong L. Park and Sungzoon Cho, “KoNLPy: Korean natural language processing in Python,” *Proceedings of the 26th Annual Conference on Human and Cognitive Language Technology*, Chuncheon Korea, pp. 133-136, 2014.

[14] Eunjeong L. Park. KoNLPy: Korean NLP in Python [Internet]. Available: http://konlpy.org/ko/latest/.

[15] R. Rehurek and P. Sojka, “Software framework for topic modelling with large corpora,” *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, Valletta Malta, pp. 45-50, 2010.

[16] R. Rehurek. Gensim: Topic Modelling for Humans [Internet]. Available: https://radimrehurek.com/gensim/index.html.

[17] Dongjun Lee, Yubin Lim and Taekyoung Kwon, “Morpheme-based Efficient Korean Word Embedding,” *Journal of KIISE*, Vol. 45, No. 5, pp. 444-450, May 2018.

[18] A. Gladkova, A. Drozd and S. Matsuoka, “Analogy-based detection of morphological and semantic relations with word embeddings: what works and what doesn't,” *Proceedings of the NAACL Student Research Workshop*, pp. 8-15, 2016.

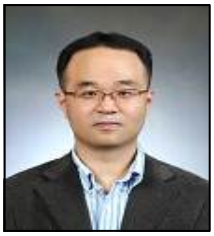
[19] J. Pennington, R. Socher and C. Manning, “GloVe: Global vectors for word representation,” Proceedings of the 2014 conference on empirical methods in natural language processing, pp. 1532-1543, 2014.

[20] P. Bojanowski et al., “Enriching word vectors with subword information,” Transactions of the Association for Computational Linguistics, pp. 135-146, 2017.

강형석(Hyungsuc Kang)

1996년 : 연세대학교 전파공학과 (공학사)
1998년 : 한국과학기술원 대학원 전기 및 전자공학과 (공학석사)
2013년 : 숙명여자대학교 TESOL 대학원 TESOL 전공 (TESOL 석사)

1998년～2003년: 삼성전자 네트워크사업부 선임연구원
2018년～현 재: 서울미디어대학원대학교 뉴미디어학부 재학
※관심분야 : 인공지능(Artificial Intelligence), 자연어처리(Natural Language Processing), 기계학습(Machine Learning) 등



양장훈(Janghoon Yang)

2001년 : University of Southern California (공학박사)

2001년～2006년: 삼성전자, 책임연구원
2006년～2010년: 연세대학교, 연구교수
2010년～현 재: 서울미디어대학원 뉴미디어학부, 부교수
※관심분야 : 중재 기술, 감성 공학, 간사이 공학, 정보이론, 이종 시스템 제어, 무선통신, 무선 네트워크, 뇌공학