skip-thought 벡터를 이용한 한국어 의미 표현

신창욱○, 차정원
창원대학교
papower1@changwon.ac.kr, jcha@changwon.ac.kr

Korean semantic representation using skip-thought vector

Chang-Uk Shin○, Jeong-Won Cha
Changwon National University

요 약

본 논문에서는 한국어 문장의 의미를 표현하는 새로운 방법으로 skip-thought를 도입하였다. 우리는 한국어 형태소 분석 코퍼스로 skip-thought 문장 분산 표현을 생성하고, 생성된 분산 표현을 감성분석과 문장분류, 함의 task의 자질로 사용하여 그 추론성능을 측정하였다. 자체 제작한 코퍼스에 대해 수행한 감성분석과 문장분류, 함의 문제에 대한 성능은 70.69%/72.17%/88.25%를 각각 기록하였다. 그간 형태소/음절 단위 분산 표현(embedding)을 생성한 후 그것을 이용한 자연어처리 task 해결은 많이 시도된 바 있다. skip-thought는 문장 단위의 분산 표현을 생성할 수 있어, 새로운 자질로써 활용 가능할 것이다.

1. 서 론

인공신경망을 이용한 자연어처리 분야에서 문장, 단어, 형태소 또는 음절 단위의 분산 표현 자질은 계속하여 연구되고 있다. 한국어에서는 많은 연구자들이 형태소 또는 음절 단위 분산 표현을 개선 및 적용하고 있다.

문장 또는 단어의 의미를 이해하기 위해 언어모델이라는 이름으로 많은 연구가 진행된 바 있다. 그 중 단어 분산 표현(word embedding)은 단어를 인공신경망에서 비교적 자유롭게 사용하기 위해 각 단어에 n차원 실수 벡터를 할당하는 방법으로, 그것을 생성하는 방법과 그를 여러 task에 이용하는 연구가 진행된 바 있다. 한국어에서는 영어권의 연구에서 분산 표현 단위를 형태소 또는 음절로 조정하여 적용하는 연구가 진행된 바 있다.

문장의 경우, 단어 분산 표현을 조합하여 문장의 분산 표현을 생성하는 지도학습 방법들이 연구된 바 있다. 반면에 skip-thought는 입력된 문장으로 주변의 문장을 표현해내는 방법으로 문장 분산 표현을 인코딩하여 비지도 학습으로 문장 분산 표현 생성을 시도하였다. 이는 [1]에서 단어 단위 skip-gram 또는 CBOW word embedding을 생성했던 방법과 유사하다.

우리는 한국어에서 비교적 연구가 진행되지 않은 문장 단위의 분산 표현 생성을 목표한다. 그리고 실험에 의해, 문장 분산 표현이 문장 분류 문제에 도움이 됨을 증명하였다. 그리고 본 논문에서 다루고 있는 skip-thought는 비지도 학습방법이기에 비교적 다른 task를 위한 추가적인 적용 소요가 적다.

2. 이전 연구

단어의 분산 표현을 생성하기 위해, 주변 문맥의 단어로 해당 단어의 분산 표현을 생성하는 연구가 진행된 바 있다. [1]은 그 단어의 표현을 생성하는 데에는 효과를 보였으나, 단어가 여러 의미를 가지는 경우에 그것을 표현할 수 없다는 제약이 있다. [2]는 기존의 단어 분산 표현 생성 기법들이 여러 의미를 가지는 단어에도 하나의 분산 표현을 할당함을 지적하며, 하나의 단어도 여러 의미를 가지는 경우에는 여러 개의 분산 표현을 가질 수 있도록 구분하는 방법을 제안하였다.

[3]은 단어 분산 표현을 합성하여 구 또는 문장의 분산 표현을 생성하는 phrase vector 생성 방법이다. 이 때, 구 분산 표현(phrase vector)은 단어 분산 표현의 합으로 정의된다. [3]은 [1]의 C-BOW보다 높은 단어 분산 표현 성능을 보였으며, SICK 코퍼스[5] 등에 대해 수행한 문장 유사도 측정 문제에서도 높은 성능을 보였다. [3]과 같이 단어 분산 표현을 생성하고, 구 또는 문장의 분산 표현을 단어 분산 표현의 합성으로 생성하는 방법을 compositional learning이라 한다.

skip-thought는 [1]에서 제안한 skip-gram을 문장 단위로 확장한 방법이라고 볼 수 있다. skip-gram은 현재의 단어로 문맥의 단어를 추론하는데, skip-thought는 그 단위를 문장으로 변경하여, 현재의 문장으로 이전과 이후의 문장을 재생성함을 목적한다.

3. 제안 방법

우리는 [4]의 skip-thought로 한국어 문서를 학습하였다. 학습에 사용한 문서는 한국어 위키피디아 덤프[6]와 세종 형태분석 코퍼스[7], 그리고 자체 수집한 신문 텍스트이다.

학습에 사용한 코퍼스의 정보는 [표 1]과 같다.

[표 1] 학습 코퍼스의 통계

구 분	수 량	단 위
문장 수	7,629,639	문장
형태소 수	288,564,883	형태소

skip-thought는 GRU encoder-decoder로 문장을 인코딩하는 구조이다. 학습에 사용한 hyper-parameter는 [표 2]와 같다.

각 parameter의 의미는 다음과 같다. 먼저, epoch는 입력한 학습코퍼스를 학습하는 횟수이다. 예로 10을 입력한다면 모든 학습코퍼스를 총 10회 반복학습한다. batch는 batch training을 수행할 때의 batch의 크기이다. hidden node는 encoder-decoder의 hidden node의 수이고, 따라서 학습이 완료되고 나서의 skip-thought vector의 크기와 같다. cell type은 recurrent network의 cell의 종류인데, 현재 RNN, GRU[8], LSTM이 주로 사용되며, [4]에서는 GRU를 사용하였다. embedding size는 encoder-decoder에 입력으로 사용하는 단어 분산 표현의 크기이다. learning rate는 학습 중 발생한 정답과 추론값 사이의 오류를 네트워크에 얼마나 반영할지 결정하는 weight이다. decay rate는 학습이 수행됨에 따라 learning rate를 점차 줄여나가는 데 사용되며, decay가 적용된 경우 I번째 epoch에서의 learning rate는 식 (1)과 같이 정의된다.

$$lr_i = lr_{i-1} * decay \qquad (1)$$

마지막으로 grad clip은 학습 중 발생한 gradient가 grad clip을 넘어가는 경우에만 적용되는 threshold이다. 이는 기존 recurrent neural network의 exploding gradient problem을 해결하기 위해 [9]에서 제안되었다.

[표 2] 학습에 사용한 hyper-parameter 정보

구 분	값
epoch	10
batch	64
hidden node	512
cell type	GRU
embedding size	300
learning rate	0.0001
decay rate	0.99
grad clip	5.0

4. 실 험

4. 1. 실험 설정

skip-thought 모델은 문장을 입력으로 문장의 분산 표현을 생성한다. 우리는 생성된 분산 표현의 성능을 측정하기 위해, skip-thought vector를 아래 3가지 문제의 입력자질로 사용하여 각각의 문제에 대한 인공신경망을 학습하였다. 이렇게 함으로써, skip-thought vector의 생성부(encoder)와 검증부(decoder)를 나눌 수 있다.

우리가 생성한 skip-thought vector의 성능을 검증하기 위해, 현재 활발히 연구되고 있는 감성분석 문제와 문장분류 문제, 그리고 함의 문제의 성능을 측정하였다. 자질로는 skip-thought vector만을 사용하고, 분류기로 CNN을 사용하였다.

또한, skip-thought의 효용성을 입증하기 위해, 그 비교군으로 [10]에서 제안된 fastText 모델과 형태소 단위 분산 표현을 입력으로 취하는 CNN 모델을 함께 실험하였다. fastText의 실험은 learning_rate를 0.1~0.001까지, epoch를 500~5000까지, n-gram을 1~4까지 바꾸어가며 비교실험하여 최고 성능을 실었고, CNN모델은 [11]의 모델을 [표 3]의 parameter로 고정하여 실험하였다.

[표 3] CNN hyper-parameter 정보

구 분	값
epoch	10
batch	5
num filter	32
filter size	3, 4, 5
embedding dim	50
l2 reg lambda	0.0
dropout keep probability	0.5

감성 분석 실험 코퍼스로는 여행 리뷰 문서를 수집하여, 문장 당 긍정, 부정, 중립을 직접 주석하였고, 형태소 분석을 수행하였다. [표 4]에 그 정보를 정리하였다. 문장분류 실험 코퍼스로는 대화 문서를 형태소분석한 후 사용하였다. 문장의 분류 class는 두 대화자가 대화하는 제품의 분류이다. [표 5]에 그 정보를 정리하였다. 함의 문제는 앞의 두 문제와 달리 두 문장을 입력으로 한다. 우리는 함의를 판단할 두 문장을 띄어쓰기 단위로 연결하여 모델에 입력하였고, 함의 여부를 classification하도록 학습하였다. [표 6]에 코퍼스 정보를 정리하였다. 모든 실험은 전체 코퍼스의 80%를 학습에, 나머지 20%를 성능평가에 사용하였다.

[표 4] 감성분석 코퍼스의 통계

구 분	수 량	단 위
문장	14,490	문장
긍정문장	7,805	문장
부정문장	1,963	문장
중립문장	4,722	문장

[표 5] 문장분류 코퍼스의 통계

구 분	수 량
문장 수	73,821
label 수	101
최다 문장이 분류된 label	25,000(None)
최소 문장이 분류된 label	1

[표 6] 함의 코퍼스의 통계

구 분	수 량
총 문장 쌍	20,000
함의 문장 쌍	10,000(50%)
문장당 평균 형태소 수	12.5

4. 2. 결과 분석

[표 7] 감성분석 문제 성능 비교

구 분	Accuracy(%)
fastText[10]	78.70
CNN	79.92
skip-thought vector+ CNN	70.69

[표 8] 문장분류 문제 성능 비교

구 분	Accuracy(%)
fastText[10]	95.15
CNN	97.97
skip-thought vector+ CNN	72.17

[표 9] 함의 문제 성능 비교

구 분	Accuracy(%)
fastText[10]	54.93
CNN	89.90
skip-thought vector+ CNN	88.25

skip-thought 성능이 fastText나 CNN 모델에 미치지 못하는 것으로 Skip-thought vector만으로는 문장의 분류를 수행하는 데 부족함이 있음을 알 수 있다. 그러나, skip-thought vector만을 사용하였음에도 70.69%, 72.17%, 88.25%의 성능을 달성한 것을 보건데, skip-thought가 감성분석을 포함한 문장 분류 문제에 도움을 줄 수 있음을 알 수 있다.

감성분석과 문장분류 그리고 함의 문제에 대한 성능을 비교하면, 문장분류에서 가장 낮은 성능을 보이고 있다. 이는 우리가 실험에 사용한 말뭉치의 특성이 반영된 것이라고 볼 수 있다. 콜센터 코퍼스는 문장에 직접적으로 해당 제품명이 나타났을 때에 그 제품명을 class로 사용한 것이기 때문이다. 이로부터 skip-thought가 문장 내 개체명 추출 등 특정 단어에 집중해야 하는 문제에는 비교적 적합하지 않음을 알 수 있다.

함의 문제에서 fastText의 성능이 다소 낮게 나타난 것은 fastText가 형태소 단위 분산 표현과 형태소 단위 n-gram 자질만을 사용했기 때문이라고 보여진다. 실험에 입력으로 나란히 두 문장을 연결하여 입력하였고, 형태소 단위 n-gram은 멀리 떨어진 두 문장의 단어를 쉽게 인식할 수 없기 때문이다. Skip-thought는 그 문장의 형태소들을 보다 효율적으로 구성하였기 때문에, 그 성능이 fastText보다 높게 나타났다고 볼 수 있다.

실험결과를 보건데, 종합적으로 판단되는 skip-thought의 특성은 비교적 문장의 의미를 폭넓게 이해하여야만 해결할 수 있는 함의 등 문장 분류 문제에 적합하고, 특정 단어를 문장 내에서 추출하는 부류의 문제에는 기존 제안된 방법보다 성능이 낮음을 알 수 있었다. 하지만 오늘날 한국어 문장 단위 분류 문제의 상황이 미리 작성한 사전과 형태소 단위 분산 표현 외에 효율적으로 사용가능한 자질이 부족한 만큼 앞으로는 문장 또는 구 단위 분산 표현 자질을 사용하여 기존 성능을 더욱 개선할 수 있을 것이라 생각된다.

5. 결론 및 향후과제

자연어처리의 여러 subtask에서 인공신경망을 활용하는 연구가 계속 진행되고 있다. 그리고 그 연구 방향은 첫 번째, 자질 발굴 및 활용과 두 번째, 인공신경망의 구조 개선으로 구분할 수 있다.

우리의 이번 연구는 이 중 첫 번째에 해당한다고 볼 수 있다. skip-thought는 비지도학습이라는 점 등이 다른 방법과 차별적이며 이후의 적용에 큰 이점으로 작용할 것이다. 또한, 이전까지 사용되어 왔던 형태소 분산 표현 등과 함께 문장의 의미에 해당하는 분산 표현을 얻음으로써 여러 자연어처리 연구에 사용될 수 있을 것으로 예상된다.

영어권에서는 skip-thought vector의 검증을 위하여 2400~4800차원의 skip-thought vector를 실험하였다. 본 논문에서는 그 크기를 512로 조절하여 실험하였다. 이로 인해, 학습 시간에는 이득을 보았지만 여러 실험에서 나타난 성능은 더욱 개선될 수 있다. 향후, 모델의 구조 개선, 말뭉치의 확장 또는 벡터의 크기 조절 등으로 skip-thought의 성능을 더욱 개선하고 이와 다른 자질을 함께 사용하였을 때 이 자질이 실제 환경에서 얼마나 의미 있을지 검증해야 할 것이다.

참 고 문 헌

[1] T. Mikolov, K. Chen, G. Corraso et al., “Distributed Representations of Words and Phrases and their Compositionality”, NIPS, 2013.

[2] M. Pelevina, N. Arefyev, C. Biemann, A. Panchenko, “Making Sense of Word Embeddings”, Proceedings of the 1st Workshop on Representation Learning for NLP, 2016.

[3] N. Pham, G. Kruszewski, A. Lazaridou, M. Baroni, “Jointly optimizing word representations for lexical and sentential tasks with the C-PHRASE model”, ACL, 2015.

[4] R Kiros, Y. Zhu, R. Salakhutdinov et al., “Skip-Thought Vectors”, In Arxiv, 2015.

[5] SICK corpus, https://clic.cimec.unitn.it/composes/

[6] 위키미디어 덤프, https://dumps.wikimedia.org/kowiki/

[7] 국립국어원, https://ithub.korean.go.kr/

[8] J. Chung, C. Gulcehre, K. Cho, Y. Bengio, “Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling”, In Arxiv, 2014.

[9] R. Pascanu, T. Mikolov, Y. Bengio, “On the difficulty of training Recurrent Neural Networks”, Proceedings of The 30th International Conference on Machine Learning, 2012.

[10] A. Joulin, E. Grave, P. Bojanowski, T. Mikolov, “Bag of Tricks for Efficient Text Classification”, In Arxiv, 2016.

[11] https://github.com/jiegzhan/multi-class-text-classification-cnn