Mikolov. et al, ‘Efficient Estimation of Word Representations in Vector Space’ (2013)

정재영

* 일러두기: 발제자주는 [ ] 안에 기술했다.

본고는 큰 데이터셋에서 단어의 연속적인 벡터 표현을 계산해주는 새로운 2 가지 모델 아키텍처를 제안한다. 그 모델은 단어 간의 유사성 측정에 의해 품질이 평가되고, 이전의 최고 성능을 지닌 기술들과 비교될 것이다. 제안된 이 모델은 또한 연산적인 저비용으로 향상된 정확도를 보이고, 더욱이 단어 간의 의미론적, 구문론적 유사성을 측정하는 테스트셋에서 최신 성능을 제공한다.

1. Intro

- 기존에 있던 기술 요약 (Vocab 내의 인덱스로 word를 표현하는 경우의 한계 (유사성 측정 x), N-gram model)

- 단순히 다량의 데이터를 가지고 학습하는 “simple tech”는 한계가 있다. 이것은 진정한 발전이 아니다.
- 최근 연구 소개: **단어의 분산 표현.** 이것으로 다량의 데이터셋에서 더 복잡한 모델을 학습시키는 것이 가능해졌다.

1.1 Goal

본고는 다량의 데이터셋에서 높은 품질의 단어 벡터를 학습시키는 기술을 소개하고자 한다.

[1] 여기서 최근에 제안된 기술 — (1) 유의어는 서로 가깝다, (2) 이 단어들(유의어)은 또한 다양한 유사성 multiple degrees of similarity[1]을 가질 수 있다. — 을 사용하여 산출된 벡터 표현의 품질을 측정한다.

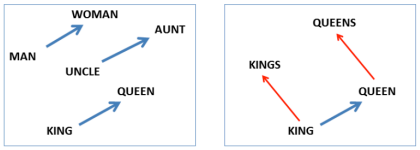


Figure 2: Left panel shows vector offsets for three word pairs illustrating the gender relation. Right panel shows a different projection, and the singular/plural relation for two words. In high-dimensional space, multiple relations can be embedded for a single word.

[예시][2] 이것은 굴절어 inflectional language 문맥에서 먼저 관찰되었다. 가령, 독일어의 명사는 성, 수, 상태에 따라 어미의 형태가 달라져서, 여러 개의 어미를 가질 수 있다. 만약 우리가 original 벡터 공간의 부분 공간에서 유사한 단어(이 경우 명사)를 찾는다면, 우리는 “유사한 어미를 가진 단어”를 찾을 수 있다. 예를 들자면, 독일어 명사 국가 ‘Staaten’와 유사한 단어를 찾고자 한다면, 이것과 유사한 어미 ‘-en’을 가진 다른 단어 소년 ‘Jungen’을 찾을 수 있다.

---

[1] Mikolov et al, ‘Linguistic Regularities in Continuous Space Word Representations’ (2013)

[2] Mikolov et al, ’Neural network based langage models for highly inflective languages’ (2009)

[2] 단어 표현의 유사성은 구문론적 규칙성 밖에서도 발견된다:

$vector('King') - vector('man') + vector('woman')$ 는 단어 'Queen'의 벡터 표현과 **가장 가까운** 벡터를 도출한다. (즉 위 연산을 통해 그 결과가 'Queen'임을 계산해낸다.)

1.2 이전 작업과의 연관

Word vector는 NLP의 여러 분야에서 유용한데, 이 word vector 자체에 대한 평가estimation는 다양한 아키텍처에 의해 수행된다. 하지만 이 아키텍처들은 트레이닝 시에 연산적으로 비싸다고 여겨진다.

2. 모델 아키텍처

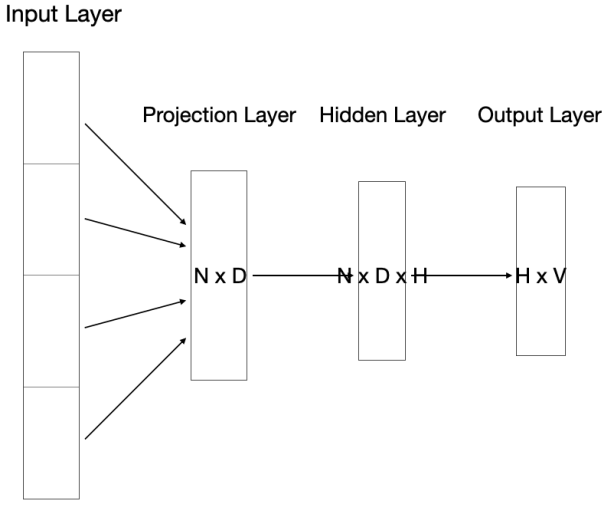
본고에서 우리는 NN에 의해 학습된 words의 분산 표현에 집중한다. 이것은 words 간의 선형 규칙성을 보존하는 LSA 보다 더 좋은 수행능력을 보여준다.

(1) 모델 아키텍처들 간의 비교를 위해, 모델의 연산적 복합성을 "모델을 완전히 학습시킬 때 요구되는 매개변수의 개수"라고 정의하자.

(2) 연산적 복합도↓ $\Longmapsto$ 정확성 ↑

: 학습training 복잡도 O = E (학습 epoch의 개수) x T (트레이닝 셋의 단어 개수) x Q (각 모델의 아키텍처에 따라 추가로 정의됨)

2.1 NNLM



각 트레이닝 견본example 당 연산적 복잡도

$$Q = (N \times D) + (N \times D \times H) + (H \times V)$$[3]

[3] N: 투사층에 투사되는 입력층에 있는 단어들의 개수, V: 단어 집합의 사이즈, D: word representations [임베딩 크기를 말하고자 하는 것 같다], H: 은닉층 개수. 여기서 투사층이란 "활성화 함수가 존재하지 않으며 룩업 테이블이라는 연산을 담당하는 층"을 가리킨다.

위 식은 NNLM의 경우에 상대적으로 연산이 비싸다는 것을 함의한다. 대개의 복잡도는 N x D x H 항에 기인한다.

2.1.1 이 곤경을 해소할 2 가지 방안

(1) 계층적 소프트맥스 hierarchical softmax

여기서 단어 집합vocabulary은 Huffman binary tree로 표현된다. 이것은 단어의 빈도수가 신경망 언어 모델에서 성립하는 클래스에 대해 잘 작동한다는 이전의 관찰로부터 나왔다. 단어 집합의 이진 트리 표현을 사용하면 output unit의 개수를 $log_2(V)$ 근처까지 낮출 수 있다.

- 그 중에서, 허프만 이진 트리[4]는 <짧은 이진 코드>를 <빈번하게 나타나는 단어 frequent words>에 할당해주어 output units의 개수를 감소시키는데, 이것은 단지 $log_2(Unigram\ perplexity(V))$ 정도로 작은 복잡도만을 요구한다.

- 반면에, 균형 이진 트리 balanced binary tree[5]는 $log_2(V)$ 의 복잡도를 요구한다.

단어 집합의 사이즈가 1 백만 개의 단어로 이루어져 있을 때, 위 복잡도들 간의 차이는 측정 시 2 배의 속도 차이를 만들어낸다. 이 결과는 NNLM의 연산적 병목 현상bottleneck이 N x D x H 항에 있을 때 중요한 속도 향상은 아니다. 하지만 우리는 나중에 은닉층이 없고, 소프트맥스 정규화normalization[6]의 효율성에 매우 의존하는 아키텍처를 제안할 것이다. 여기서는 은닉층이 없기 때문에 복잡도가 N x D x H에 기인하지 않고, 대신에 H x V 항에 기인할 것이다. (아래의 3장에 있는 CBOW, Skip-Gram에서 이것을 확인해보라.)

2.2 Recurrent Neural Net Language Model (RNNLM)

RNN 모델은 투사층이 없다. 이것은 오직 입력층, 은닉층, 출력층으로만 구성되어 있을 뿐이다. 이것의 학습 견본 당 복잡도는 다음과 같다:

Q = (H x H) + (H x V)

여기서 (NNLM)에 있던 D는 [*N x D를 말하는 것 같다] 은닉층의 개수 H의 차원과 동일하다. 우리는 이 복잡도에서 다시 계층적 softmax를 사용하여 H x V 항을 H x $log_2(V)$ 로 낮출 수 있다. 이 모델에서의 대부분의 복잡도는 H x H 항에서 기인한다.

---

[4] http://blog.skby.net/%ED%97%88%ED%94%84%EB%A7%8C-%EC%BD%94%EB%93%9C-huffman-code/

[5] "균형이진트리는 다음 그림과 같습니다. 모든 잎새노드의 깊이 차이가 많아야 1인 트리를 가리킵니다. 균형이진트리는 예측 가능한 깊이(predictable depth)를 가지며, 노드가 n개인 균형이진트리의 깊이는 log n을 내림한 값이 됩니다. 깊이 말고 *left subtree*와 *right subtree*의 노드 수를 기준으로 균형이진트리를 정의하는 경우도 있다."

[균형이진트리 사례 그림]

[6] 헷갈리던 번역어이다. 대한수학회에서는 normalization을 정규화, regularization을 정칙화라고 번역한다. http://www.kms.or.kr/mathdict/list.html?key=ename&keyword=regularization

3. 새로운 로그 - 선형 모델

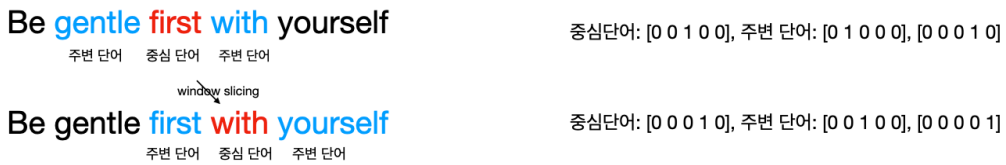
이번 절에서는 <연산 복잡도를 최소화하려는 목적을 지닌 단어의 분산 표현 학습>에 대한 2 개의 새로운 모델을 제안한다. 이전 절에서의 주요 관찰은 대개의 복잡도가 모델의 비선형적 은닉층에 기인한다는 것이었다.

3.1 연속적인 Bag-of-Words 모델 (CBOW)

이것은 NNLM과 유사한데, CBOW에는 비선형적 은닉층이 없고, 투사층이 모든 words에 대해 공유된다. 따라서 모든 words는 동일한 위치에 투사된다. (즉, 투사층의 해당 벡터는 각각의 입력 벡터들의 평균이다)

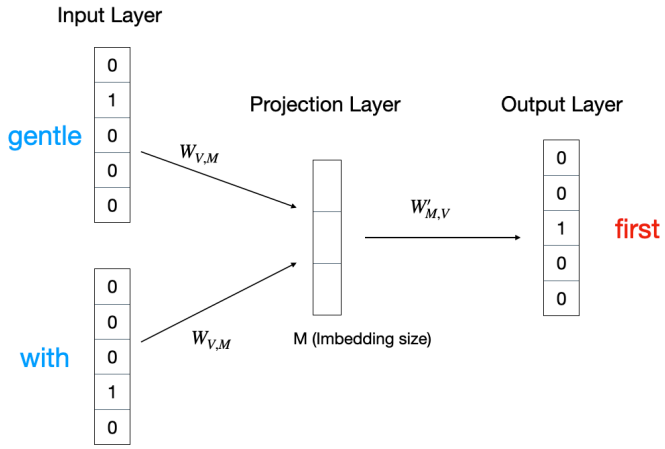
[예시][7]

다음 예문을 고려해보자: “Be gentle first with yourself”. 이 문장을 스플릿해주어 {“Be”, “gentle”, “first”, “with”, “yourself”}이 주어졌다고 해보자. CBOW를 가지고 우리가 수행할 일은 앞의 set 중 {“Be”, “gentle”, “with”, “yourself”}으로부터 “first” 를 예측하는 것이다. 그러면, CBOW 방법론에 따를 때 “first”는 중심 단어 center word이고, 나머지 앞의 set 내의 단어들은 주변 단어 context word라고 간주된다. 아래 예를 보자.



window = 1인 경우

CBOW에서 중심 단어 주변에 있는 단어를 앞, 뒤로 각각 몇 개의 단어를 참고할지를 나타내주는 것을 **윈도우** window라고 한다. 그리고 위 그림과 같이 중심 단어와 주변 단어를 오른쪽으로 한 칸씩 이동하며 학습을 위한 데이터 셋을 만드는 것을 **윈도우 슬라이싱**이라고 일컫는다. (아래 그림에서 주의할 점은 가중치 W, W’ 은 서로 전치 행렬 관계가 아니라는 것이다. 이들은 단지 서로 다른 랜덤 가중치일 뿐이다.)



[7] 명료한 이해를 위해 논문 밖의 예시를 첨가한다.

[1] CBOW의 과정:

(1) 먼저 입력값으로서 주변 단어들이 원핫 인코딩 된다.
(2) 그 뒤에 단어가 임베딩된 뒤에 만들어지는 **투사층 projection layer**이 있다.[8]
(3) 마지막으로, 출력값으로는 원핫 인코딩된 중심 단어가 나온다.

[2] 입력층 => 투사층

입력층에서 투사층으로 입력값들이 가중치를 곱해서 사상될 때, 투사층에 입력되는 값은 다음과 같다: 이 예에서는 윈도우 n = 1 이기 때문에, 각 투사층에 입력되는 값의 개수는 2n으로서 2 개이다. 즉, 투사층의 하나의 값은 $v = \frac{v_{gentle} + v_{with}}{2n}$ 이다.[9]

[3] 투사층 => 출력층

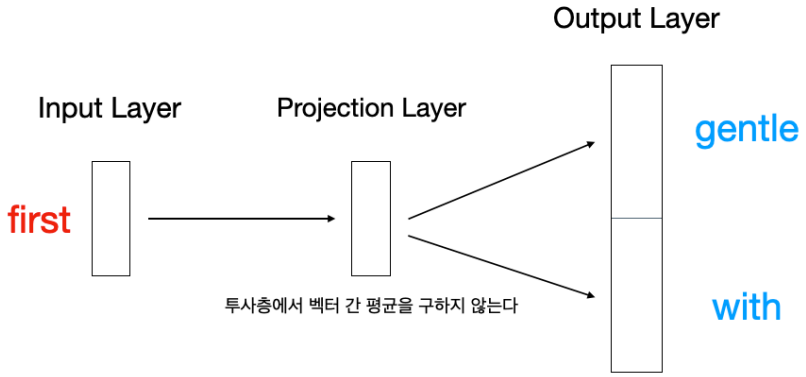
투사층의 벡터는 W'와 곱해짐으로써 다시 입력값의 크기와 동일한 벡터가 된다. 그런데 우리는 이것을 다시 원핫 벡터 형태로 만들어주기 위해, softmax를 사용한다.

[4] 역전파를 수행하여 W, W'를 학습시킨다.

[5] 트레이닝 복잡도: Q = N x D + D x $log_2(V)$

3.2 연속적인 Skip-gram 모델

이것은 한 문장 내의 다른 word(s)를 기초로 하여 하나의 word의 분류를 최대화하는 것을 목표로 한다. Skip-Gram의 메커니즘은 CBOW의 그것에 반대되는 형태를 지닌다. CBOW와 Skip-Gram 중 어느 것이 성능이 더 좋은지는 경우에 따라 다르다. 본고에서 수행된 실험에서는 구문론적 질문에 대해서는 CBOW가, 의미론적 질문에 대해서는 Skip-gram이 더 좋은 성능을 보였다. 스킵그램의 메커니즘은 아래 같이 도식화될 수 있다:



이 모델에서, window의 범위 ↑ ⟹ (1) Quality ↑ & (2) 계산 복잡도 ↑
트레이닝 복잡도: Q = C(words의 최대 거리) x (D + D x $log_2(V)$

---

[8] 이것은 은닉층과 구분되는데, 그 이유는 word2vec에서 이 투사층은 오직 하나이기 때문에 심층 신경망이 아닌 얕은 신경망 shallow NN이기 때문이고, 또한 word2vec에서는 활성화 함수가 존재하지 않고 대신 lookup table 연산을 담당하기 때문이다. 이러한 이유 때문에 은닉층과 구분하고자 투사층이라는 용어를 사용한다.

[9] 뒤에서 알아볼 skip-gram 방법에서는 입력되는 중심 단어가 하나이기 때문에 투사층에서 벡터의 평균을 구하지 않는다.

4. 결과

[1] 단어 ‘프랑스’와 ‘이탈리아’가 유사하다는 것을 보이는 것은 쉽지만, 우리는 더 도전적인 유사성 관계에 대해 물을 수 있다. 가령, “ ‘biggist’와 ‘big’이 유사하다는 의미에서 ‘small’과 유사한 단어는 무엇인가?”를 묻는다고 해보자.

이 물음은 단어의 벡터 표현을 가지고 단순한 대수적 연산을 수행함으로써 답해질 수 있다:

$$X = vector('biggest') - vector('big') + ('small')$$

올바르게 학습된다면 이 계산의 결과 X는 — 벡터 공간 내에서 cosine 거리에 의해 측정된 X와 가장 가까운 단어를 찾기 때문에 — 올바른 답인 ‘smallest’를 가리킬 것이다.

[2] 또한, 우리는 다량의 데이터에서 고차원 word vectors를 학습시킬 때, 산출된 벡터는 — 가령, **도시**와 그 도시가 포함된 **나라** 같이 — 단어들 간의 매우 미묘한 의미론적 관계를 답하는 데 사용되어질 수 있다. 이 사용은 현존하는 여러 NLP 분야 — 가령 기계 번역, 정보 탐색, 질의응답 시스템 등 —에서 유용하게 쓰일 수 있을 것이다.

4.1 Task Decription

단어 벡터를 측정하기 위해, 5 유형의 의미론적 질문과, 9 유형의 구문론적 질문을 가진 포괄적인 테스트 셋을 정의해보자.

Table 1: *Examples of five types of semantic and nine types of syntactic questions in the Semantic-Syntactic Word Relationship test set.*

	Type of relationship	Word Pair 1		Word Pair 2	
5 유형의 의미론적 질문:	Common capital city	Athens	Greece	Oslo	Norway
	All capital cities	Astana	Kazakhstan	Harare	Zimbabwe
	Currency	Angola	kwanza	Iran	rial
	City-in-state	Chicago	Illinois	Stockton	California
	Man-Woman	brother	sister	grandson	granddaughter
9 유형의 구문론적 질문:	Adjective to adverb	apparent	apparently	rapid	rapidly
	Opposite	possibly	impossibly	ethical	unethical
	Comparative	great	greater	tough	tougher
	Superlative	easy	easiest	lucky	luckiest
	Present Participle	think	thinking	read	reading
	Nationality adjective	Switzerland	Swiss	Cambodia	Cambodian
	Past tense	walking	walked	swimming	swam
	Plural nouns	mouse	mice	dollar	dollars
	Plural verbs	work	works	speak	speaks

- 정답 조건: <위 2, 3 절에서 알아본 방법에 따라 계산된 벡터에 가장 가까운 단어> = <정답 단어> 이면, 질문에 올바르게 답한 것.

4.2 Maximization of Accuracy

[1] (단어 벡터의) 차원 수 & 훈련 단어의 개수 ↑ ⟹ 정확도 ↑

Table 2: *Accuracy on subset of the Semantic-Syntactic Word Relationship test set, using word vectors from the CBOW architecture with limited vocabulary. Only questions containing words from the most frequent 30k words are used.*

Dimensionality / Training words	24M	49M	98M	196M	391M	783M
50	13.4	15.7	18.6	19.1	22.5	23.2
100	19.4	23.1	27.8	28.7	33.4	32.2
300	23.2	29.2	35.3	38.6	43.7	45.9
600	24.0	30.1	36.5	40.8	46.6	50.4

[2] 의미론 질문 — Skip-gram, 구문론 질문 — CBOW

Table 3: *Comparison of architectures using models trained on the same data, with 640-dimensional word vectors. The accuracies are reported on our Semantic-Syntactic Word Relationship test set, and on the syntactic relationship test set of [20]*

Model	Semantic-Syntactic Word Relationship test set		MSR Word Relatedness
Architecture	Semantic Accuracy [%]	Syntactic Accuracy [%]	Test Set [20]
RNNLM	9	36	35
NNLM	23	53	47
CBOW	24	64	61
Skip-gram	55	59	56

References

[1] Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781* (2013)
[2] Mikolov et al, 'Linguistic Regularities in Continuous Space Word Representations' (2013)
[3] https://ratsgo.github.io/data%20structure&algorithm/2017/10/21/tree/ 각주 4
[4] https://wikidocs.net/22660