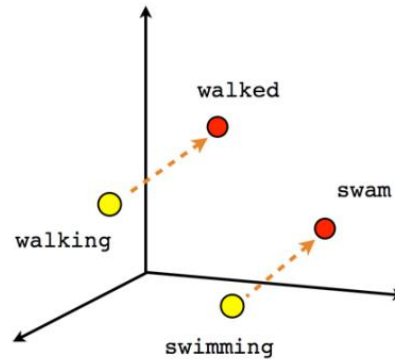
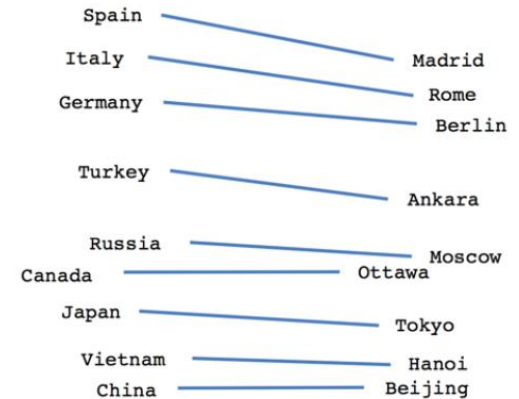


Male-Female



Verb tense



Country-Capital

Lecture 5: Text Representation II

Distributed Representations QA

Pilsung Kang

School of Industrial Management Engineering

Korea University

Q1 (안시후)

3월 16일 화요일 ▾



안시후 오후 9:21

Text Representation 강의를 보며 이상 데이터(이 경우 Text 관련이므로 특히 오타)를 해결해주는 경우에도 사용 가능하지 않을까 싶어 질문 남깁니다.

검색을 통해 얻은 지식으로, 이상 데이터의 경우 단순히 삭제를 하거나 다른 값으로 대체를 Preprocessing 과정에서 해결하는 것이 일반적이라고 알게 되었습니다. 하지만 데이터의 양이 많아지면 Rule base만으로는 한계가 있을 것이라고 생각하여 논문 서치를 해보았으나 큰 성과가 있지는 않았고, 검색어를 조금 변경해서 Spelling(맞춤법) 관련으로는 딥러닝 관련 논문이 여럿 있었으며, 맞춤법 검사기 등으로 실제 사용되고 있다는 사실을 알게 되었습니다.

이에 관하여, 오타 등을 Preprocessing 하는 과정에서 Text Representation을 활용하여 다른 값으로 대체하는 방식이 실제로 사용 되는지 궁금하며 실제로 사용되지 않는다면 다른 방식의 어떤 점이 더 좋은지 알고 싶습니다.



2개의 답글 5일 전 마지막 답글

AI-I (최정우)



최정우 11일 전

안녕하세요

좋은 질문 해주심에 감사드립니다.

질문을 읽다 보니 저도 궁금하여 몇 가지 추가로 찾아본 내용 공유 드립니다.

Character 단위 데이터를 encoding 하여 Input 값으로 받는 방법이 말씀하신 오타 처리 관점에서 효과적일 수 있다는 사실이 흥미롭다고 생각합니다.

찾아본 논문 중에 오타를 교정하기 위한 preprocessing은 아니지만, Text Representation 방법으로

Word(단어)가 아닌 Character(글자) 단위로 Input 값을 받아서 ConvNet을 통하여 Text classification 방법이 있습니다.

단어가 아닌 문자 단위의 Input값을 통하여 학습하면, 오타와 Spelling 교정 처리가 잘 된다는 것이 흥미롭다고 생각하여 아래 논문 공유 드립니다.

Character-level Convolutional Network for Text Classification

<https://arxiv.org/pdf/1509.01626.pdf>

추가로, 한글을 음절 단위 Multi-hot 벡터로 표현하여 Seq2Seq denosing autoencoder모델로 한글 오류를 보정한 사례도 같이 공유 드립니다.

Sequence-to-Sequence Autoencoder based Korean Text Error Correction using Syllable-level Multi-hot Vector Representation

[https://www.researchgate.net/publication/331987503_Sequence-to-](https://www.researchgate.net/publication/331987503_Sequence-to-Sequence_Autoencoder_based_Korean_Text_Error_Correction_using_Syllable-level_Multi-hot_Vector_Representation)

[Sequence_Autoencoder_based_Korean_Text_Error_Correction_using_Syllable-level_Multi-hot_Vector_Representation](https://www.researchgate.net/publication/331987503_Sequence-to-Sequence_Autoencoder_based_Korean_Text_Error_Correction_using_Syllable-level_Multi-hot_Vector_Representation)

[내용추가]

교수님께서 강의 중에 말씀해주신 Character level에 Input값으로 n-gram representation 하는 FastText도 동일하게 해당된다고 생각합니다.

비록 오타가 발생하여도 유사도가 높은 Vector로 Representation 예시 아래 같이 첨부 드립니다.

관련 link : <https://www.haptik.ai/tech/extract-spelling-mistakes-fasttext/> (편집됨)



Extract Spelling Mistakes Using Fasttext

In this blog, we showcase how we have tried to tackle spelling errors on our conversational AI platform using fastText. (63kB) ▼

AI-I (최정우)

NOV 3, 2018 6:44:00 PM / CHIRAG JAIN

EXTRACT SPELLING MISTAKES USING FASTTEXT

TAG

MACHINE LEARNING

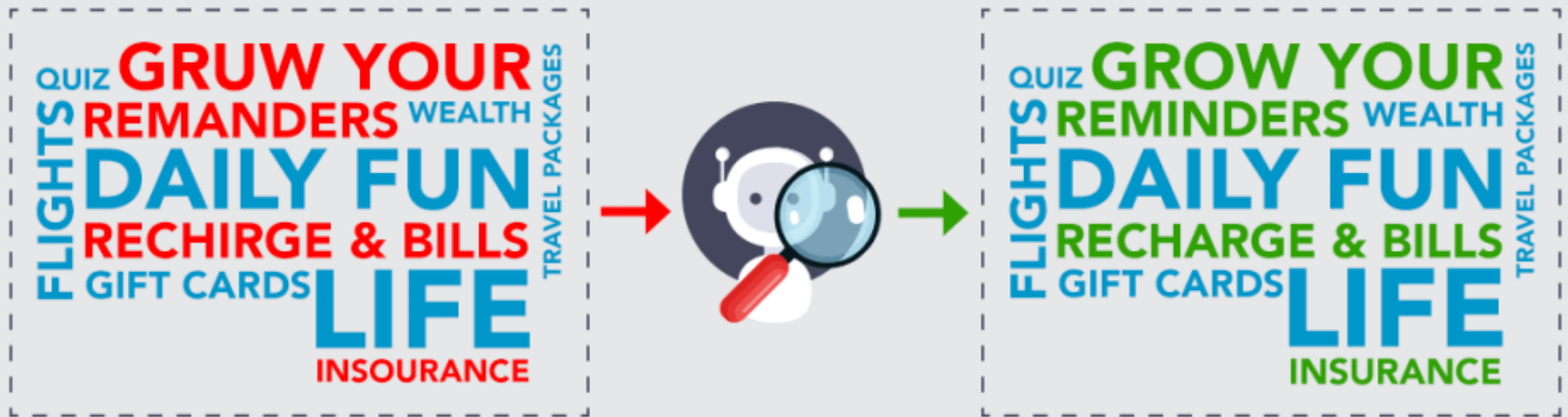
At [Haptik](#), we deal with a lot of noisy conversational data. This [conversational](#) data is generated from our 100+ chatbots that are live across various platforms like [mobile apps](#), web-SDKs etc.

Even though **12.5% of India** can speak English, **it ranks at 27th on the English Proficiency Index**. This means any kind of textual data obtained from Internet users like ours should not only expect multilingual data but also a lot of noise in the form of grammatical errors, internet slang, spelling mistakes, etc. Noise can create significant problems both during training and testing bots. During training time, a lot of noise can prevent machine learning algorithms from learning quickly, while at test time unseen noise can lead to wrong predictions.

Rare words like spelling mistakes often end up with not very useful vectors during the training process. This is because they occur so rarely that their randomly assigned vector hardly moves. Usually, rare words are marked as unknown words and discarded from the training process. If we can correct spellings accurately, we can keep more of our training data as well as reduce the total number of unique words which impact the training time and memory.

In this post, we will try tackling **spelling errors**. Given that we have a moderately sized corpus of text data, we will extract a dictionary of spelling mistakes and their possible corrections from the data in an unsupervised way.

AI-I (최정우)



AI-1 (최정우)

- 일정 조건을 만족하는 token들에 대해서만 spelling check를 수행

```
word_to_mistakes = collections.defaultdict(list)
nonalphabetic = re.compile(r'^a-zA-Z')

for word, freq in vocab.items():
    if freq < 500 or len(word) <= 3 or nonalphabetic.search(word) is not None:
        # To keep this task simple, we will not try finding
        # spelling mistakes for words that occur less than 500 times
        # or have length less than equal to 3 characters
        # or have anything other than English alphabets
        continue

    # Query the fasttext model for 50 closest neighbors to the word
    similar_words = model.wv.most_similar(word, topn=50)
    for similar_word in results:
        if include_spell_mistake(word, similar_word, similarity_score):
            word_to_mistakes[word].append(similar_word)
```

AI-1 (최정우+강필성)

- 일정 조건을 만족하면 FastText represnetation 상에서 가장 유사한 단어로 대체

```
enchant_us = enchant.Dict('en_US')
spell_mistake_min_frequency = 5
fasttext_min_similarity = 0.96
def include_spell_mistake(word, similar_word, score):
    """
    Check if similar word passes some rules to be considered a spelling mistake

    Rules:
    1. Similarity score should be greater than a threshold
    2. Length of the word with spelling error should be greater than 3.
    3. spelling mistake must occur at least some N times in the corpus
    4. Must not be a correct English word.
    5. First character of both correct spelling and wrong spelling should be same.
    6. Has edit distance less than 2
    """
    edit_distance_threshold = 1 if len(word) <= 4 else 2
    return (score > fasttext_min_similarity
            and len(similar_word) > 3
            and vocab[similar_word] >= spell_mistake_min_frequency
            and not enchant_us.check(similar_word)
            and word[0] == similar_word[0]
            and nltk.edit_distance(word, similar_word) <= edit_distance_threshold)
```

AI-1 (최정우+강필성)

- Word-superiority effect

According to a research team at Cambridge University, it doesn't matter in what order the letters in a word are, the only important thing is that the first and last letter be in the right place. The rest can be a total mess and you can still read it without a problem. This is because the human mind does not read every letter by itself, but the word as a whole.

캠릿브지 대학의 연결구과에 따르면, 한 단어 안에서 글자가 어떤 순서로 배치되어 있지는는 중요하지 않고, 첫 번째와 마지막 글자가 올바른 위치에 있는 것이 중하다고 한다. 나머지 글자들은 완전히 엉진망창의 순서로 되어 있라을지도 당신은 아무 문제 없이 이것을 읽을 수 있다. 왜냐하면, 인간의 두뇌는 모든 글자를 하하나나 읽는 것이 아니라 단어 하나를 전체로 인식하기 때문이다.

Psycholinguistic evidence on scrambled letters in reading

1) according to a research at Cambridge University... According to a research (sic) at Cambridge University

There are a number of groups in Cambridge, UK doing research on language. There is the group where I work ([Cognition and Brain Sciences Unit](#)), there are also groups in the [Department of Experimental Psychology](#) most notably the [Centre for Speech and Language](#) (where I used to work). There are also language researchers in Phonetics, the Research Centre for English and Applied Linguistics, and at Anglia Ruskin University.

To my knowledge, there's no-one in Cambridge UK who is currently doing research on this topic. There may be people in Cambridge, MA, USA who are responsible for this research, but I don't know of them. If you know different, please let me know [matt.davis@mrc-cbu.cam.ac.uk?subject=Cmabrigde].

AI-1 (최정우+강필성)

- A good language model should understand the following two sentences
 - ✓ Why do the hockey players satke around for so long between fights?
 - ✓ He holds a 51% sktae in the firm.
- ✓ Why do the hockey players **skate** around for so long between fights?
- ✓ He holds a 51% **stake** in the firm.

AI-2 (천주영)



천주영 5일 전

안녕하세요

우선 좋은 질문 감사드립니다

오타 교정과 관련하여 흥미로운 두가지 알고리즘을 찾아 공유드립니다.

첫번째로는 Peter Norving 알고리즘입니다. 이는 오타자의 키워드가 Input으로 주어졌을 때, 이를 교정한 단어를 Output으로 만들어주는 알고리즘입니다.

원리를 간단하게 말씀드리면 크게 1) 단어 사전 구축 2)키워드 교정입니다.

1)에서는 교정 사전으로 쓰일 텍스트를 모아 사전을 구축합니다.

2)에서는 검색 키워드를 오타 교정 함수에 전달하여 만일 단어 사전에 있는 키워드일 경우 그대로 반환합니다. 만일 키워드를 사전에서 찾지 못한 경우 다음과 같은 단계를 거쳐 단어를 반환합니다.

--

검색 키워드의 edit distance가 1인 키워드 목록을 만든다.

이 키워드가 단어사전에 있는지 확인한다.

단어 사전에 포함 되어있는 경우, 해당 단어를 반환한다.

--

만일 찾지 못한다면 edit distance가 2인 키워드 목록을 만들어 위의 단계를 반복합니다. 만일 이번에도 단어가 없다면 원본 키워드를 반환합니다.

edit은 deletes, transposes, replaces, inserts의 4가지 방식을 사용하여 진행됩니다.

edit distance란 edit을 적용한 횟수로 이해했습니다.

something의 경우 edit distance 1은 {'seething', 'smoothing', 'something', 'soothing'}이고

edit distance 2는 {'loathing', 'nothing', 'scathing', 'seething', 'smoothing', 'something', 'soothing', 'sorting'}가 될 것입니다.

AI-2 (천주영)

두번째 알고리즘은 Symspell 알고리즘입니다.

오타 후보군을 사전에 계산하고 DELETE만 사용하여 후보군을 계산합니다. 따라서 속도가 빠릅니다. 함수를 실행하기 전에 미리 단어사전 파일을 읽고, delete 후보군을 계산해 놓습니다.

peter Norvig 알고리즘이 입력 키워드의 후보군을 계산하여 단어 사전에 있는지를 확인하는 것과 달리 단어 사전의 단어들의 후보군을 계산해서 저장한 뒤 오타 교정을 할 키워드가 입력되면 이를 후보군과 비교하는 방식입니다.

단어 사전에서 키워드를 찾고

단어 사전의 Delete후보군에서 키워드를 찾고

단어 사전에서 키워드의 delete후보군을 찾고

단어 사전의 Delete 후보군에서 키워드의 Delete후보군을 찾는 방식입니다.

해당 내용이 담긴 참고할 만한 사이트들입니다.

<https://norvig.com/spell-correct.html>


<https://wolfgarbe.medium.com/1000x-faster-spelling-correction-algorithm-2012-8701fcd87a5f>

 Medium

1000x Faster Spelling Correction algorithm (2012)

Q2 (조한샘)

3월 17일 수요일 ▾

 조한샘 오후 5:12
Topic 5 Text Representation 2: Distributed Representation 중 GloVe에 대한 질문입니다.

강의자료 11페이지에서 목적함수에 $f(x)$ 라는 함수를 도입하는 이유로 고반복 단어에 대해 학습의 가중치를 내리는 용도로 사용한다고 교수님께서 설명해 주셨습니다. 하지만 GloVe 논문에서는 $f(x)$ 라는 함수를 도입하는 이유로 저반복의 단어가 noisy하고 information이 적기 때문에 저반복에 대한 가중치를 낮추는 용도로 $f(x)$ 를 도입한다고 언급하고 있습니다. $f(x)$ 함수를 설계한 형태를 살펴보았을 때 논문에서 언급하는 이유가 더 적합한 것으로 생각되는데 다른분들의 의견이 궁금합니다.

GloVe 논문에서 관련된 부분 같이 첨부합니다.

Glove.PNG ▾

$$w_i^T \tilde{w}_j + b_i + \tilde{b}_j = \log(X_{ij}). \quad (7)$$

Eqn. (7) is a drastic simplification over Eqn. (1), but it is actually ill-defined since the logarithm diverges whenever its argument is zero. One resolution to this issue is to include an additive shift in the logarithm, $\log(X_{ij}) \rightarrow \log(1 + X_{ij})$, which maintains the sparsity of X while avoiding the divergences. The idea of factorizing the log of the co-occurrence matrix is closely related to LSA and we will use the resulting model as a baseline in our experiments. **A main drawback to this model is that it weighs all co-occurrences equally, even those that happen rarely or never. Such rare co-occurrences are noisy and carry less information than the more frequent ones — yet even just the zero entries account for 75–95% of the data in X , depending on the vocabulary size and corpus.**

We propose a new weighted least squares regression model that addresses these problems. Casting Eqn. (7) as a least squares problem and introducing a weighting function $f(X_{ij})$ into the cost function gives us the model

$$J = \sum_{i,j=1}^V f(X_{ij}) (w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij})^2, \quad (8)$$

our experiments. A main drawback to this model is that it weighs all co-occurrences equally, even those that happen rarely or never. Such rare co-occurrences are noisy and carry less information than the more frequent ones — yet even just the zero entries account for 75–95% of the data in X , depending on the vocabulary size and corpus.

We propose a new weighted least squares regression model that addresses these problems. Casting Eqn. (7) as a least squares problem and introducing a weighting function $f(X_{ij})$ into the cost function gives us the model

$$J = \sum_{i,j=1}^V f(X_{ij}) (w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij})^2, \quad (8)$$

where V is the size of the vocabulary. The weighting function should obey the following properties:

1. $f(0) = 0$. If f is viewed as a continuous function, it should vanish as $x \rightarrow 0$ fast enough that the $\lim_{x \rightarrow 0} f(x) \log^2 x$ is finite.
2. $f(x)$ should be non-decreasing so that rare co-occurrences are not overweighted.
3. $f(x)$ should be relatively small for large values of x , so that frequent co-occurrences are not overweighted.

A2-1 (윤훈상)

2개의 답글



윤훈상 10일 전

조한샘님. 먼저 논문 세부적으로 좋은 질문 해주셔서 감사합니다. 확실히 해당 텍스트만 보게 된다면 교수님의 의중과 정반대의 의견을 내포하는 것 같아 보입니다.

하지만 강의 슬라이드의 Objective Function이 존재하는 11 페이지나, 조한샘님께서 사진으로 보여주신 영역 다음을 보시면 Weighting Function의 3가지 조건이 나타납니다.

그 중 3번째 조건을 보게 된다면 (아래 메시지),

3. $f(x)$ 는 지나친 Overweight를 방지하기 위하여 큰 값에 대해서는 '비교적'으로 작아야 한다.

라고 서술하고 있습니다.

따라서 제 생각엔 저자의 의중은 노란색으로 표시한 문장을 달성하기 위하여

- 빈도가 적은 단어, Noisy Value를 Filtering 해야한다.
- 그 목적으로 가중치를 두어서, 빈도에 따라 Loss에 Weight를 부여한다.

의 방법을 사용했지만,

- 빈도에 비례하여 Weight를 주게 되면, 매우 높은 빈도의 단어들에 대하여 지나치게 큰 가중치를 주기 때문에
- Threshold를 넘으면 일정한 값을 주는 것으로 해결한다 (Threshold).

의 흐름을 따라가는 것 같습니다.

정리하자면, 논문에서 기술한 목적은 '높은 빈도의 단어들의 영향력을 제거하기 위하여' 라는 교수님의 수업에서 말씀하신 내용과 반대지만, 결과적으로 가중치를 형성하는 과정에서 그 목적이 달성되게 된다.

라고 할 수 있을 것 같습니다.

감사합니다. (편집됨)



A2-1 (윤훈상+강필성)

- Vocabulary size

We compare with the published results of a variety of state-of-the-art models, as well as with our own results produced using the `word2vec` tool and with several baselines using SVDs. With `word2vec`, we train the skip-gram (SG^\dagger) and continuous bag-of-words (CBOW^\dagger) models on the 6 billion token corpus (Wikipedia 2014 + Gigaword 5) with a vocabulary of the top 400,000 most frequent words and a context window size of 10. We used 10 negative samples, which we show in Section 4.6 to be a good choice for this corpus.

For the SVD baselines, we generate a truncated matrix X_{trunc} which retains the information of how frequently each word occurs with only the top 10,000 most frequent words. This step is typical of many matrix-factorization-based methods as the extra columns can contribute a disproportionate number of zero entries and the methods are otherwise computationally expensive.

Model	Size	WS353	MC	RG	SCWS	RW
SVD	6B	35.3	35.1	42.5	38.3	25.6
SVD-S	6B	56.5	71.5	71.0	53.6	34.7
SVD-L	6B	65.7	72.7	75.1	56.5	37.0
CBOW [†]	6B	57.2	65.6	68.2	57.0	32.5
SG [†]	6B	62.8	65.2	69.7	58.1	37.2
GloVe	6B	65.8	72.7	77.8	53.9	38.1
SVD-L	42B	74.0	76.4	74.1	58.3	39.9
GloVe	42B	75.9	83.6	82.9	59.6	47.8
CBOW*	100B	68.4	79.6	75.4	59.4	45.5

4.6 Model Analysis: Run-time

The total run-time is split between populating X and training the model. The former depends on many factors, including window size, vocabulary size, and corpus size. Though we did not do so, this step could easily be parallelized across multiple machines (see, e.g., Lebrete and Collobert (2014) for some benchmarks). Using a single thread of a dual 2.1GHz Intel Xeon E5-2658 machine, populating X with a 10 word symmetric context window, a 400,000 word vocabulary, and a 6 billion token corpus takes about 85 minutes. Given X , the time it takes to train the model depends on the vector size and the number of iterations. For 300-dimensional vectors with the above settings (and using all 32 cores of the above machine), a single iteration takes 14 minutes. See Fig. 4 for a plot of the learning curve.

Q3 (신동환)

신동환 오후 4:03

Topic5: Text_Representation: Distributed Representation의 Word2Vec에 관하여 질문드립니다.

픽스된 윈도우사이즈의 context word를 input으로 하여 특정 단어를 예측하는것이 CBOW이고, 반대로 하나의 word를 input으로 하여 픽스된 윈도우 사이즈의 context words를 예측하는 것이 Skip-gram이란 것을 알게되었습니다.

예를 들자면, 'the skinny cat sat on the table'에서, CBOW는 윈도우 사이즈를 각기 설정하여 context word를 input으로 하여 center word인 'cat'을 예측하고, 반대로, skip-gram에서는 'cat'을 input으로 하여 나머지 context word를 각기 예측합니다.

또한, 수업을 통해 Skip-gram은 여러개의 gradient 값으로 backpropation을 통해 파라미터를 업데이트해주고, CBOW는 하나의 값으로 모두 동일하게 업데이트를 해주므로 Skip-gram이 더 **효율적**이다 라고 이해했습니다.

그러나, 여러개의 input이 있는 CBOW의 정확도가 더 높을 것이라 예상되어 논문을 찾아보니 아래와 같이 소요시간은 3배이지만 의미와 구문의 정확도가 Skip-gram에서 더 높은 결과가 나왔습니다.

결론적으로, CBOW에 비해 Skip-gram이 쓰이는 모델의 정확도가 왜 더 높은 것인지, 각 기법은 어떤 상황에서 더 적합한 지에 대해서 궁금합니다.

image.png ▼

Table 5: Comparison of models trained for three epochs on the same data and models trained for one epoch. Accuracy is reported on the full Semantic-Syntactic data set.

Model	Vector Dimensionality	Training words	Accuracy [%]			Training time [days]
			Semantic	Syntactic	Total	
3 epoch CBOW	300	783M	15.5	53.1	36.1	1
3 epoch Skip-gram	300	783M	50.0	55.9	53.3	3
1 epoch CBOW	300	783M	13.8	49.9	33.6	0.3
1 epoch CBOW	300	1.6B	16.1	52.6	36.1	0.6
1 epoch CBOW	600	783M	15.4	53.3	36.2	0.7
1 epoch Skip-gram	300	783M	45.6	52.2	49.2	1
1 epoch Skip-gram	300	1.6B	52.2	55.1	53.8	2
1 epoch Skip-gram	600	783M	56.7	54.5	55.5	2.5

A3-1 (차형주)

- 효율적 Vs 효과적?



차형주 10일 전

먼저, 좋은 질문 감사드립니다.

직관적으로 생각해보았을 때, Center word 주변의 단어로 center word를 맞추는 것이 성능이 좋아보일 수 있습니다. 저도 그렇게 처음에는 생각했구요. 하지만, 조금만 더 깊이 생각해보면, CBOW의 경우 center word는 단 한번의 업데이트 기회를 갖습니다. 반면에 skip gram은 일반적인 경우에 윈도우 크기가 만약 2라면 4번 업데이트 기회를 가질 수 있게 되는 것 이죠.

아래 테이블에서 보시면, "The boy is going to school" 이라는 문장에 대해 각각의 방법의 Input 과 output을 볼 수 있습니다. CBOW는 각 단어에 대해 한번만 학습이 되는 반면, Skip gram은 각 단어가 여러번 학습되는 것을 볼 수 있습니다.

(<https://yjjo.tistory.com/13>, <https://ratsgo.github.io/from%20frequency%20to%20semantics/2017/03/30/word2vec/>) (편집됨)

CBOW

Input	Output
boy, is, going	the
the, is, going, to	boy
the, boy, going, to school	is
the, boy, is, to, school	going
boy, is, going, school	to
is, going, to	school

the	1
boy	1
is	1
going	1
to	1
school	1

Skip-Gram

Input	Output
the	boy, is, going
boy	the, is, going, to
is	the, boy, going, to school
going	the, boy, is, to, school
to	boy, is, going, school
school	is, going, to

the	3
boy	4
is	5
going	5
to	5
school	3

A3-2 (강형원)

- 효율적 Vs 효과적?

형원

강형원 10일 전

안녕하세요.

질문에 대한 답변 남기도록 하겠습니다.

CBOW와 Skip-gram 중에서 Skip-gram이 왜 더 성능이 좋은지에 대해 질문을 하셨습니다.

이 부분에 대해서는 강의 중에 교수님께서 언급을 하셨지만

제가 다시 설명을 드리자면은

Center word의 window size만큼 주변 단어를 input으로 받아 center word를 예측하는 CBOW의 경우 질문자께서 남기신 댓글 처럼 한 단어(center word)를 통해 주변 단어들에 gradient를 한번에 흘리게 되면서 한 단어의 정보를 이용해서 여러 단어에 대해 모두를 업데이트 해야하는 상황입니다.

Skip-gram의 경우 주변 단어들을 통해서 한 단어(center word)에 gradient를 흘리게 되면서 주변 단어들의 정보를 모두 하나의 단어를 업데이트를 하는데 사용되기 때문에 더 많은 정보를 이용해서 업데이트 된다고 볼 수 있을 것이며, window size의 2배만큼 더 여러번 업데이트 된다고도 볼 수 있을 것 같습니다.

따라서 Skip-gram이 더 성능이 좋게 나타난다고 할 수 있으며, Word2Vec을 할 때는 Skip-gram을 주로 사용하여 임베딩 벡터를 구하는 것이 일반적이라고 합니다.

또한 두 기법을 어떤 상황에서 더 적합한지에 대해서 질문을 남기셨는데 두 기법을 임베딩 벡터를 구하는 Word2Vec의 접근 방법으로 알고 있고, Word2Vec 학습에서 생성되는 가중치 행렬인 W 를 구하는 것이기 목표이기 때문에 더 성능이 좋게 나타나는 Skip-gram을 사용하는게 일반적이라고 생각을 하며 상황에 따라 나뉘지는 방법은 아니라고 생각합니다.

이 부분에 대해서는 저도 명확하지 않습니다. 제가 생각하는 것이 틀리다면 이에 대해서 추가적으로 댓글 남겨주시면 감사하겠습니다.

A3-3 (정의석)

• 효율적 Vs 효과적?



정의석 8일 전

안녕하세요! 좋은 질문해주셔서 감사합니다 😊

앞에 두 분 모두 좋은 답변해주셨는데, 그거 말고도 직관적이고 흥미로운 의견을 찾아서 공유드리고 싶어서 이렇게 답변을 달게 되었습니다. 일단 질문자 분과 앞서 답해주신 답변자분들께서 말씀하셨던 것 처럼 CBOW는 context word를 input으로 하여 특정 단어를 예측하는 것이고, Skip-gram은 반대로 하나의 word를 input으로 하여 context words를 예측하는 것입니다.

CBOW의 경우, 단어 주변의 문맥을 확인하여 대상 단어가 나올 확률을 최대화하여 예측을 수행하게 되는데 이는 rare한 단어들을 예측하는 데는 매우 어려운 task이게 됩니다. 예시로, `yesterday was really delightful day` 라는 문장이 있다고 하겠습니다. 이때, `yesterday was really [...] day` 라고 context가 들어왔을 때 CBOW 모델은 주로 많이 사용되는 `beautiful` 또는 `nice` 라고 답을 내릴 것이지만, 실제로 우리가 원하는 단어는 `delightful` 이므로 틀리게 예측한 것이 되어버리고 맙니다. 즉, 다시 말해 CBOW는 주어진 `context` 만으로 가장 나올법한(most probable) 단어를 예측하기 때문에 적게 나온 단어에 대해서는 그만큼 적게 관심(less attention)을 주게 되어버리게 됩니다.

Skip-gram의 경우는 단어가 주어지면 해당 단어 주변의 context를 예측하는 것이기 때문에 `delightful` 이라는 단어가 들어오면 `yesterday was really [...] day` 또는 다른 유사한 `context` 를 제공하게 됩니다. 하지만, Skip-gram에서는 `delightful` 이 다른 고확률(고빈도) 단어들과 경쟁을 하지 않아도 되고(?), `delightful + context` (단어+context) 둘 다를 고려해주며 학습하게 되고, 앞 CBOW에서는 동일한 observation로 보았던 `beautiful + context`, `nice + context`, `delightful + context` 등으로 각각 다른 new observation으로 학습하게 됩니다. 이러한 이유 때문에 skip-gram은 더 많은 데이터가 학습시에 필요로 하게 되지만, 드문 단어들에 대해서도 높은 예측 성능을 띄게 됩니다.

제가 이해 및 번역하면서 잘못 해석한 부분이 있을수도 있으니 피드백은 언제나 환영입니다^^

참고자료:

<https://stats.stackexchange.com/questions/180548/why-is-skip-gram-better-for-infrequent-words-than-cbow>

+ 추신 : 해당 답변이 매우 높은 vote를 가지고 있고, 직관적이라서 답변으로 적었으나 이 의견에 반대하는 분의 의존 또한 있어서 수업 시간에 토의하기에는 매우 바람직한 주제라고 생각합니다. 해당 답변에 반대하고 있는 분은 아래와 같은 논문들에서 위 답변에 대한 반론들이 펼쳐진다고 합니다.

1. The Effects of Data Size and Frequency Range on Distributional Semantic Models (<https://arxiv.org/pdf/1609.08293>) : CBOW가 rare한 단어들에 대해 skip-gram에 비해 더 강력하다고 주장하고 있는 논문

A3-3 (정의석 + 강필성)

- Sahlgren, M., & Lenci, A. (2016). The effects of data size and frequency range on distributional semantic models. *arXiv preprint arXiv:1609.08293*.

DSM	HIGH	MEDIUM	LOW	MIXED
CO	32.61 (↑62.5, ↓04.6)	35.77 (↑66.6, ↓21.2)	12.57 (↑35.7, ↓00.0)	27.14 (↑56.6, ↓07.9)
PPMI	55.51 (↑75.3, ↓28.0)	57.83 (↑88.8, ↓18.7)	25.84 (↑50.0, ↓00.0)	47.73 (↑83.3, ↓27.1)
TSVD	50.52 (↑70.9, ↓23.2)	54.75 (↑77.9, ↓24.1)	17.85 (↑50.0, ↓00.0)	41.08 (↑56.6, ↓19.6)
ISVD	63.31 (↑87.5, ↓36.5)	69.25 (↑88.8, ↓46.3)	10.94 (↑16.0, ↓00.0)	57.24 (↑83.3, ↓33.0)
RI	53.11 (↑62.5, ↓30.1)	48.02 (↑72.2, ↓20.4)	23.29 (↑39.0, ↓00.0)	46.39 (↑66.6, ↓21.0)
SGNS	68.81 (↑87.5, ↓36.4)	62.00 (↑83.3, ↓27.4)	18.76 (↑42.8, ↓00.0)	56.93 (↑83.3, ↓30.2)
CBOW	62.73 (↑81.2, ↓31.9)	59.50 (↑83.3, ↓32.4)	27.13 (↑78.5, ↓00.0)	52.21 (↑76.6, ↓25.9)

Table 2: Average results for DSMs over four different frequency ranges for the items in the TOEFL, ESL, SL, MEN, and RW tests. All DSMs are trained on the 1 billion words data.

✓ Skip-gram보다 CBOW가 rare occurrence words에 보다 효과적이다.

A3-3 (정의식 + 강필성)

<https://wikidocs.net/69141>

- Skip-gram with negative sampling

앞서 배운 Skip-gram을 상기해봅시다.

중심 단어
↓
The fat **cat** sat on the mat
↑
주변 단어

Skip-gram은 중심 단어로부터 주변 단어를 예측하는 모델이었습니다. 위와 같은 문장이 있다고 한다면, Skip-gram은 중심 단어 cat으로부터 주변 단어 The, fat, sat, on을 예측합니다. 기존의 Skip-gram 모델을 일종의 주황 박스로 생각해보는다면, 아래의 그림과 같이 입력은 중심 단어, 모델의 예측은 주변 단어인 구조입니다.



하지만 네거티브 샘플링을 사용하는 Skip-gram(Skip-Gram with Negative Sampling, SGNS) 이하 SGNS는 이와는 다른 접근 방식을 취합니다. SGNS는 다음과 같이 중심 단어와 주변 단어가 모두 입력이 되고, 이 두 단어가 실제로 윈도우 크기 내에 존재하는 이웃 관계인지 그 확률을 예측합니다.

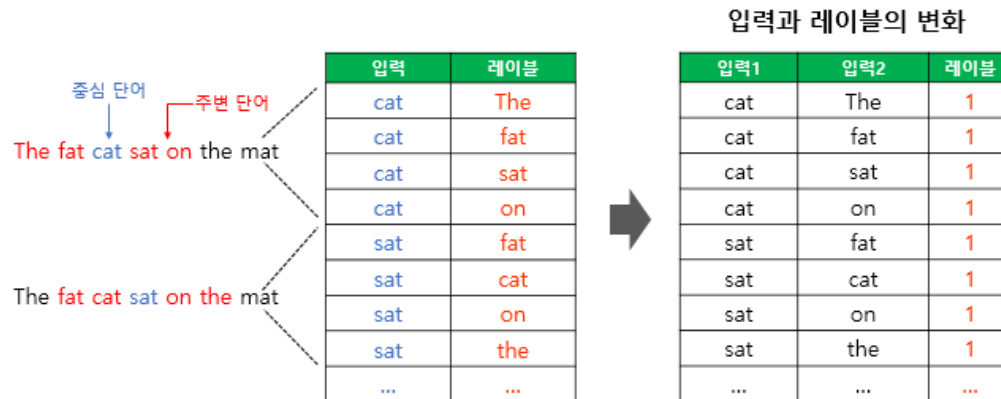


A3-3 (정의식 + 강필성)

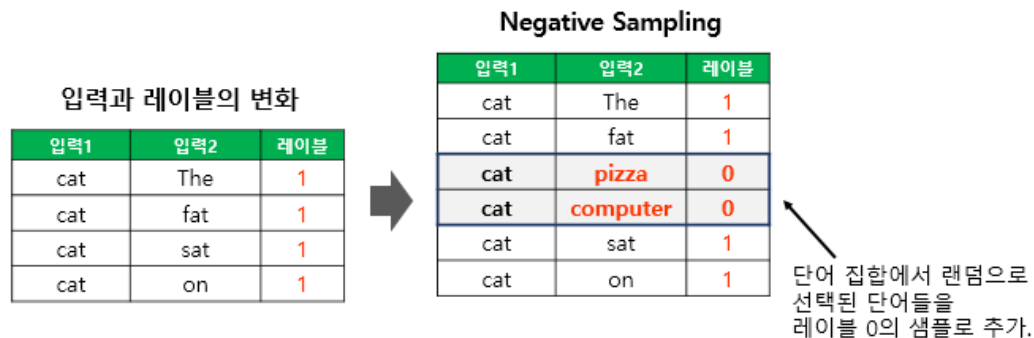
<https://wikidocs.net/69141>

• Skip-gram with negative sampling

기존의 Skip-gram 데이터셋을 SGNS의 데이터셋으로 바꾸는 과정을 봅시다.



위의 그림에서 좌측의 테이블은 기존의 Skip-gram을 학습하기 위한 데이터셋입니다. Skip-gram은 기본적으로 중심 단어를 입력, 주변 단어를 레이블로 합니다. 하지만 SGNS를 학습하고 싶다면, 이 데이터셋을 우측의 테이블과 같이 수정할 필요가 있습니다. 우선, 기존의 Skip-gram 데이터셋에서 중심 단어와 주변 단어를 각각 입력1, 입력2로 둡니다. 이 둘은 실제로 윈도우 크기 내에서 이웃 관계였으므로 레이블은 1로 합니다. 이제 레이블이 0인 샘플들을 준비할 차례입니다.



Q4 (허재혁) & Q7 (이윤승)



허재혁 오후 5:51

안녕하세요!

Topic5: Text_Representation: Distributed Representation 수강하면서 궁금한 점이 있어서 질문 올립니다.

큰 질문은 아래와 같고 이러한 질문을 떠올리게 된 계기로는 크게 두 가지가 있습니다.

Q. 여러 임베딩 방법들은 가지고 있는 데이터에서 방법별로 나눈 각 token(word, character 등)에 대해 학습을 진행하는데 어떤 방식으로 학습을 해야 하는가?

1. 검증(validation or dev) 데이터는 어떻게 구성해야 하는지
2. 가지고있는 데이터에 과적합(overfitting)을 해야 맞는지
 - a. 과적합을 해야한다면 loss를 0으로 가져가는데 맞는지
 - b. 아니라면 정해진 기준이 있을지

이러한 질문에 대해 떠올린 계기를 설명드리자면 첫 번째는 기본적으로 supervised learning의 경우 검증 데이터를 통해 모델이 과적합 되는 것을 방지합니다. 하지만 임베딩 모델은 가지고 있는 데이터의 representation을 학습하는 것이기 때문에 검증 데이터가 필요하지 않은 건가 라는 생각이 됩니다. 두 번째는 '따로 검증 데이터가 없다면 representation을 학습하기 위해 과적합을 하는게 맞을까' 입니다. 하지만 가지고 있는 데이터의 양이 충분히 크다면(상황에 따라 다르겠지만..) 과적합해도 새로운 데이터에 대해서 각 token이 의미를 잘 반영하고 있다고 생각되지만 그렇지 않다면 새로운 데이터에 대해 각 token이 의미를 잘 반영하지 못하지 않을까 싶습니다. 그렇다고 하기엔 정해진 '기준'에 맞게 학습하는해야 한다는 것인데 이러한 '기준' 따로 있는지 궁금합니다. (편집됨)



2개의 답글 8일 전 마지막 답글

Q4 (허재혁) & Q7 (이윤승)

3월 26일 금요일 ▾



yunseung_lee 오후 5:43

Topic 5: Text Representation II에서 word embedding 관련해서 전반적으로 질문이 있습니다. 강의에서는 단어의 임베딩이 잘 되었다/되지 않았다의 평가를 위해 단어 간 semantic relationship이 벡터 크기로서 잘 보존되는지 확인하는 과정이 여러 번 소개되었습니다. (ex. king-queen 벡터 크기와 man-woman 벡터 크기가 유사하다면, 단어 임베딩이 잘 되었다고 간주) 이 방법은 정성적인 평가기준이라고 생각되는데, 단어 임베딩이 잘 되었는지 확인하기 위한 정량적 평가 방법이 존재하는지 궁금합니다.



1개의 답글 2일 전

A4-I (김영석)



김영석 10일 전

안녕하세요. Word2Vec에 대해 다시 한 번 고민해볼 수 있는 좋은 질문 감사드립니다. 몇 가지 추가로 찾아보고 경험적인 생각을 말씀드립니다.

1. 검증 데이터는 어떻게 구성해야 하는지

검증 데이터는 구성하지 않는 것이 맞다고 생각합니다. Word2Vec은 supervised learning을 하는 알고리즘이지만, Word2Vec을 이용한 word embedding을 한다는 관점에서는 word2vec의 output이 아닌, 사이에 끼있는 projection layer(혹은 latent space 등)를 임베딩 벡터로 활용하므로 unsupervised learning의 형태를 띄게 됩니다. 따라서 검증 데이터를 구성하지 않는 것이 더 옳바르다고 생각합니다.

2. 가지고 있는 데이터에 과적합을 해야 맞는지

이 역시 word embedding 관점에서 과적합은 큰 의미가 없다고 생각합니다. 1번에 대한 답변에서 말씀드렸듯 projection layer의 경우 unsupervised learning이기 때문에 output을 가지고 loss를 계산하며 매우 오랫동안(하다못해 loss가 0이 될 때까지) 학습을 진행한다고 하더라도 projection layer(word embedding)가 더 잘 학습이 되냐는 다른 문제라고 생각합니다.

Word2Vec의 근본적인 문제인 unknown token에 대해서 취약하다는 단점을 극복하는 것이 더 좋은 word embedding 결과를 뽑아내는데 도움이 될 것이라고 생각합니다.

즉, 최대한 많은 수의 데이터와 많은 단어들을 확보하는 것이 학습을 오래하는 것 보다 좋은 embedding을 찾는데 도움이 될 것이라고 생각합니다. 또한 word embedding을 평가하는 기준은 주관적이라고 알고 있습니다. 유사도를 통해서 평가를 해볼 수도 있을 것이며 각 task의 목적에 맞게 평가지표를 설정하여 평가를 해볼 수 있을 것이라고 생각하고 정성적인 평가를 많이 사용하는 것으로 알고 있습니다.

Word2Vec의 평가에 관한 관련 링크를 공유드립니다. 아직 많이 부족하여 혹시나 잘못된 부분이 있다면 말씀해주시면 감사하겠습니다.

관련 link : <https://stackoverflow.com/questions/47018088/how-to-evaluate-word2vec-build-on-a-specific-context-files> (편집됨)

Stack Overflow

How to evaluate word2vec build on a specific context files

Using gensim word2vec, built a CBOW model with a bunch of litigation files for representation of word as vector in a Named-Entity-recognition problem, but I want to know how to evaluate my



Response to A4-I (허재혁)



허재혁 8일 전

답변 감사합니다!

아무래도 word embedding 모델에 학습을 하기위해서 iteration을 얼마나 적용해야하는지 에 대한 답변은 역시 어렵네요 $\pi\pi$ 말씀해주신 대로 downstream task에 적용해보거나 정성적인 평가를 해야하는 것 같습니다.

이후로 호기심이 생겨서 word embedding 어떻게 평가하는지 조금 더 알아보았는데요. 알아본 내용은 크게 두 가지로 먼저 word embedding에 대한 평가 방법에 대해 답변내용보다 더 구체적으로 찾아본 결과와 word embedding에서 dimensionality가 어느정도로 하는게 맞는가? 에 대해 찾아본 결과입니다.

[1. word embedding에 대한 평가 방법]

여기에 대해서는 "Evaluating Word Embedding Models: Methods and Experimental Results" 논문의 내용을 참고했고 추가로 각 데이터에 대해 조금 더 찾아본 내용과 혹시나 이후 실험을 위해 찾아본 데이터 다운로드 링크를 함께 추가했습니다.

지금까지 알아본 내용으로는 좋은 word embedding에 대해 평가할 수 있는 방법이 크게 두 가지가 있었습니다.

1. Intrinsic Evaluation
2. Extrinsic Evaluation

Intrinsic evaluation의 경우 word embedding 모델 자체의 성능을 평가하기 위한 방법이고 extrinsic evaluation은 말씀해주셨던 downstream task에 적용해서 평가하는 방법입니다.

[1.1 Intrinsic Evaluation]

Intrinsic evaluation을 위해 사용되는 평가기준과 그에 맞는 데이터는 아래와 같이 있습니다.

- Word Similarity
- Word Analogy
- Concept Categorization

[Word Similarity]

첫 번째로 word similarity 단어 간의 유사도를 평가하는 방법입니다. 예시로는 두 단어를 주고 그에 대한 유사도를 점수로 나타낸 정답과 word embedding을 통해 계산된 유사도를 비교하는 방법입니다. 이를 위한 데이터 셋으로는 대표적으로 WS-353, WS-353-SIM, WS-353-REL, MTurk-771, SimLex-999 그리고 SimVerb-3500 가 있습니다. 두 단어의 유사도를 어떻게 숫자로 표현했는지 저도 궁금해서 찾아보니 MTurk-771에 대해서만 말씀드리자면 정말 많은 데이터를 구성하는 AMT.. 에서 만든 데이터이고 한 사람에게 두 단어를 보여주면서 binary('right' or 'wrong')로 평가하게 해서 총 50명에게 이 과정을 반복하여 50점 척도로 나타냈다고 합니다.

[Word Analogy]

두 번째로 word analogy는 word2vec 저자인 Tomas Mikolov가 word2vec에 대해 "Efficient Estimation of Word Representations in Vector Space"에서 평가 실험한 방법과 같은 맥락입니다. 먼저 relationship을 나타내는 두 쌍의 단어를 보여주고 다음으로 하나의 단어를 제시한 다음 앞서 언급한 relationship에 맞는 단어를 유추하는 방법입니다. 이에 대한 데이터로는 대표적으로 Google Semantic, Google Syntactic and MSR 가 있습니다.

ex) Relationship : France - Paris,

Example : Italy - Rome

[Concept Categorization]

Concept categorization은 유사한 단어를 하나의 그룹으로 묶어서 여러 그룹으로 구성된 데이터를 기준으로 word embedding을 사용하여 비교한 결과로 평가하는 방법입니다. 여기에 대한 데이터로는 AP, BLESS and BM 가 있습니다.

[Dataset for Intrinsic Evaluation]

Intrinsic 평가를 위한 데이터는 ACL wiki에도 benchmark별 리더보드와 함께 잘 정리가 되어있습니다.

https://aclweb.org/aclwiki/Main_Page

또한 의문의 vecto-ai곳에서 앞서 언급한 데이터들을 모두 github에 깔끔히 업로드하셔서 공유드립니다.

<https://github.com/vecto-ai/word-benchmarks>

[1.2 Extrinsic Evaluation]

Extrinsic 평가와 관련해서는 word embedding 모델을 NMT(Neural Machine Translation)과 같은 downstream task에 적용해서 평가하는 방법입니다. 이러한 방법으로 평가하는 것이 실제로 원하는 task에 맞게 학습할 수 있다는 장점이 있지만 그에 반해 downstream task를 위한 모델도 학습해야해서 연산 시간과 양이 늘어난다는 단점이 있습니다.

Response to A4-I (허재혁+강필성)

Don't Stop Pretraining: Adapt Language Models to Domains and Tasks

Suchin Gururangan[†] Ana Marasović[‡] Swabha Swayamdipta[†]
Kyle Lo[†] Iz Beltagy[†] Doug Downey[†] Noah A. Smith[‡]

[†]Allen Institute for Artificial Intelligence, Seattle, WA, USA

[‡]Paul G. Allen School of Computer Science & Engineering, University of Washington, Seattle, WA, USA
{suching, anam, swabhas, kylel, beltagy, dougd, noah}@allenai.org

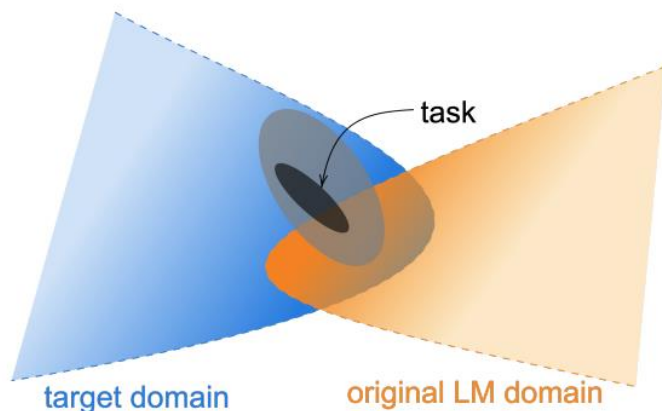


Figure 1: An illustration of data distributions. Task data is comprised of an observable task distribution, usually non-randomly sampled from a wider distribution (light grey ellipsis) within an even larger target domain, which is not necessarily one of the domains included in the original LM pretraining domain – though overlap is possible. We explore the benefits of continued pretraining on data from the task distribution and the domain distribution.

PT	100.0	54.1	34.5	27.3	19.2
News	54.1	100.0	40.0	24.9	17.3
Reviews	34.5	40.0	100.0	18.3	12.7
BioMed	27.3	24.9	18.3	100.0	21.4
CS	19.2	17.3	12.7	21.4	100.0
	PT	News	Reviews	BioMed	CS

Figure 2: Vocabulary overlap (%) between domains. PT denotes a sample from sources similar to ROBERTA’s pretraining corpus. Vocabularies for each domain are created by considering the top 10K most frequent words (excluding stopwords) in documents sampled from each domain.

Response to A4-I (허재혁+강필성)

Domain	Task	RoBERTa	Additional Pretraining Phases		
			DAPT	TAPT	DAPT + TAPT
BIO MED	CHEMPROT	81.9 _{1.0}	84.2 _{0.2}	82.6 _{0.4}	84.4 _{0.4}
	[†] RCT	87.2 _{0.1}	87.6 _{0.1}	87.7 _{0.1}	87.8 _{0.1}
CS	ACL-ARC	63.0 _{5.8}	75.4 _{2.5}	67.4 _{1.8}	75.6 _{3.8}
	SCI ERC	77.3 _{1.9}	80.8 _{1.5}	79.3 _{1.5}	81.3 _{1.8}
NEWS	HYPERPARTISAN	86.6 _{0.9}	88.2 _{5.9}	90.4 _{5.2}	90.0 _{6.6}
	[†] AGNEWS	93.9 _{0.2}	93.9 _{0.2}	94.5 _{0.1}	94.6 _{0.1}
REVIEWS	[†] HELPFULNESS	65.1 _{3.4}	66.5 _{1.4}	68.5 _{1.9}	68.7 _{1.8}
	[†] IMDB	95.0 _{0.2}	95.4 _{0.1}	95.5 _{0.1}	95.6 _{0.1}

Table 5: Results on different phases of adaptive pretraining compared to the baseline RoBERTa (col. 1). Our approaches are DAPT (col. 2, §3), TAPT (col. 3, §4), and a combination of both (col. 4). Reported results follow the same format as Table 3. State-of-the-art results we can compare to: CHEMPROT (84.6), RCT (92.9), ACL-ARC (71.0), SCI ERC (81.8), HYPERPARTISAN (94.8), AGNEWS (95.5), IMDB (96.2); references in §A.2.

Response to A4-1 (허재혁)

[2. word embedding에서 dimensionality가 어느정도로 하는게 맞는가?]

학습을 어떻게 해야 하는가를 고민하다 보니 자연스레 word embedding의 차원(dimension)은 어느정도가 좋은지에 대해서 궁금해졌습니다.

최근 방향에 대해서는 저도 잘 아는건 아니지만 word embedding에 대한 open question은 여전히 몇가지 남아 있는 것 같습니다. 그 중에 하나가 'word embedding model을 학습할 때 어느정도의 dimension이 좋은가?' 인데 여기에 대해서 연구한 "On the Dimensionality of Word Embedding"이라는 논문에 대해서 소개해 드립니다. 2018년 NIPS에 나왔던 논문으로 제안한 방법은 PIP loss이고 두 임베딩 모델 간의 차이를 정량적으로 표현하는 방법입니다.

이 논문에서 앞선 word embedding model의 open question으로 어느정도가 좋은것인지? 에 대해서 얘기하고 실제로 grid search를 통해 찾거나 downstream task를 통해 수행하게 되면 연산 시간이 너무 많이 든다는 문제점을 말합니다.

그리고 이에 대해서 언급한 방법이 바로 PIP loss 라는 정량적인 평가 방법이고 이 방법을 통해서 하나의 word embedding model의 dimension 간의 차이를 비교하여 최적의 dimension을 적은 비용으로 찾을 수 있다고 말하고 서로 다른 방법의 word embedding model에 대해서도 평가 가능하다고 합니다.

[3. 결론]

눈으로만 서치하다가 막상 찾아본 내용을 정리하며 쓰다보니 글이 길어졌는데.. 이렇거면 블로그에 작성해서 링크만 드리는게 더 보기 편하실 것 같았네요..

결론적으로 word embedding을 얼마나 training하는게 좋은지는 말씀해주신 내용대로 일반적인 supervised learning 처럼 early stopping을 통해 찾을 수 있기보다는 앞서 얘기했던 평가 방법들을 통해 찾는게 맞다고 생각합니다. 하지만 word embedding의 performance로 중요한건 역시 vocabulary size 겠제..?

좋은 답변 덕분에 많이 공부하고 갑니다..! 혹시나 제가 작성한 내용에 코멘트 있으시면 언제든 감사드립니다. 🙏 (편집됨)



[vecto-ai/word-benchmarks](https://github.com/vecto-ai/word-benchmarks)

Benchmarks for intrinsic word embeddings evaluation. - vecto-ai/word-benchmarks



Q5 (허종국)



허종국 오후 7:25

Topic 5-2 Word2Vec 에서 Negative Sampling 이 들어간 Loss function 에 대해 이해가 잘 되지 않는 부분이 있어 간단한 질문 드립니다.(강의 ppt 14p)

1. 해당 Loss에서 log 뒤에 있는 sigma 는 activation function인 것인가요?

1-1. Loss function J 를 전개하면 log 안에는 input word와 output word 에 대한 embedding 값 들의 단순한 내적만 들어가야 할 것 같은데 sigma가 왜 들어가는 것인지 궁금합니다(기존 Loss function의 어떠한 문제점을 개선하기 위한 테크닉이라면, 문헌 등의 자료를 공유해주시면 감사하겠습니다.)



2개의 답글 10일 전 마지막 답글

$$J_t(\theta) = \log \sigma(u_o^T v_c) + \sum_{i=1}^k E_{i \sim P(w)} [\log \sigma(-u_i^T v_c)]$$

A5-1 (김영석)



김영석 10일 전

안녕하세요. Negative Sampling에서 Loss function에 대해서 저도 궁금해서 알아본 부분이라 공유드립니다.

1. 해당 Loss에서 log 뒤에 있는 sigma 는 activation function인 것인가요?

제가 찾은 문헌을 기준으로 말씀드리면 sigma는 sigmoid function으로 보입니다.

Negative sampling은 정확하게 정의하자면 사용자가 지정한 윈도우 사이즈 내에 등장하지 않는 단어와 input word의 쌍에 대한 확률, 그리고 context 단어와 input word의 쌍에 대한 확률을 합쳐 전체 단어처럼 소프트맥스 확률을 구하는 것입니다. 올려드린 문헌을 참고하시면 $P(D=1 | (w,c))$ 은 corpus 내에서 함께 등장할 확률과 $P(D=0 | (w,c))$ 는 윈도우 사이즈 내에 등장하지 않는 단어에 대한 확률을 의미하며 이를 통해 Loss function을 정의해 나갑니다. 따라서 단순한 내적 만으로 Loss function을 표현하지 않기 때문에 sigma가 들어가게 됩니다.

Negative sampling을 적용한 loss function의 수식 전개 과정이 자세히 적혀있는 참고 논문과 개념과 관련된 링크를 올려드리겠습니다.

참고 논문 : <https://arxiv.org/pdf/1402.3722.pdf>

관련 링크 : <https://ratsgo.github.io/from%20frequency%20to%20semantics/2017/03/30/word2vec/> (편집됨)

A5-2 (윤훈상)

 윤훈상 10일 전

안녕하세요. 허종국님.

저 또한 Loss Function에서 Sigma의 출현 이유에 대해서 궁금했습니다. 제가 생각했을 때는 sigma는 sigmoid이며, Sigmoid가 왜 등장하는 지에 대한 이유는 Center의 Context이냐, 아니냐의 분류인 Positive / Negative를 Sampling하는 Negative Sampling과 연관이 있다고 생각합니다.

Positive / Negative를 Sampling한 후에 해당 Sampled Data를 사용해 Loss Function을 구성할 때, Gradient Ascent를 해야 하니 + 기준 좌측과 우측의 값 모두 값이 커야 합니다. 따라서 내적을 하여 유사도를 계산하며 이것을 Sigmoid Activation Function을 통해 확률값으로 나타내 줍니다. 좌측 값은 내적 값이 커야 확률이 증가하지만 우측 값은 내적 값이 작아야 Sigmoid 값이 비교적 커지게 됩니다. 즉, Positive Sample들은 내적값이 커 Sigmoid를 통해 높은 확률을 갖게 되고, Negative Sample은 내적값이 작고 이는 작은 음수로 표현되어 Sigmoid 내에서 비교적 큰 값을 갖게 됩니다.

따라서 정리하자면,

1. Loss Log 뒤에 있는 Sigma는 Sigmoid Activation이며,
2. Embedding 값들에 대한 단순한 내적은 확률로 표현할 수 없으므로 Sigma를 사용한다 라고 할 수 있을 것 같습니다.

A5-3 (김영석+윤훈상+강필성)

- Without Negative Sampling

- ✓ Maximize the probability of occurring the output words

$$p(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w=1}^W \exp(u_w^T v_c)}$$

- With Negative Sampling

- ✓ $P(Y=1 \mid c \text{ is one of the context words of the output word } o)$
- ✓ $P(Y=0 \mid l \text{ is not a context words of the output word } o)$

$$J_t(\theta) = \log \sigma(u_o^T v_c) + \sum_{i=1}^k E_{i \sim P(w)} [\log \sigma(-u_i^T v_c)]$$

Q6 (김태연)

3월 23일 화요일 ▾



김태연 오후 7:08

안녕하세요. Topic5: Text_Representation 2: Distributed Representation 중 GloVe에 대한 질문입니다.

해당 강의자료 10페이지에서 연산 과정중에서 $\log X_i$ 의 term을 b_i, b_k 인 상수항으로 치환하는 과정이 존재합니다.

X_i 라는 term은 주어진 Corpus 내에서 연산이 가능한 값으로 이해되는데 그렇다면 결국 $\log X_i$ 의 term은 동일 Corpus 내에서는 하나의 상수값으로 고정되어 있다고 생각됩니다.

그렇다면 Word Embedding의 결과인 w_i 를 구하는 학습 과정에서 하나의 고정되어 있는 값인 $\log X_i$ 의 항을 왜 b_i/b_k 의 두 항으로 분리하여 학습을 진행하는지에 대해 궁금합니다.



1개의 답글 5일 전

A6-1 (김수빈)

1개의 답글



김수빈 5일 전

안녕하세요. 우선 좋은 질문 감사합니다.

저도 그 부분이 궁금하여 찾아보았었는데요, Glove에 대한 Optional Video 강의에서 그 답을 찾을 수 있었습니다.

우선 $\log X_i$ term이 하나의 상수로 고정되어있다는 부분은 맞습니다.

그러나, 여기서 이 term이 b_i, b_k 두 부분의 상수항으로 치환되는 것은 Glove 모델이 가정하고 있는 "단어 벡터간 교환법칙의 성립" 때문입니다.

즉, center 와 context word의 co-occurrence는 symmetric해야하는데, 강의자료 부분의 수식을 보면 $\log X_i$ 부분에 대해서는 $i \leftrightarrow k$ 의 교환을 생각하였을 때, symmetric하지 않습니다 ($\log X_i \neq \log X_k$).

따라서 이 부분에 대해 보완하기 위해, b_i, b_k 두 상수항을 추가하여 교환법칙이 성립하도록 bias를 넣어주었다고 생각하시면 됩니다.

($b_i + b_k = b_k + b_i$ 로 교환법칙 성립) (편집됨)

