정보검색

과제 #1

Due: 9월 24일 23시

1 서론

본 과제의 목적은 정보 검색의 자연어 처리의 시작으로, word2vec을 이해하고 익숙해지기 위함이다.

2 Word2vec

본 과제에서 수업시간에 배운 word2vec을 실제로 활용하여 그 구조와 가능한 연 산들에 대해 알아보자. 이를 위해서 다음과 같은 준비물이 필요하다.

2.1 Gensim

자연어 처리 관련 python library.

https://radimrehurek.com/gensim/
https://radimrehurek.com/gensim/models/word2vec.html

다음과 같이 gensim을 install하자.

> pip install gensim

2.2 Text8

우리는 또한 embedding matrix의 학습을 위한 입력자료가 필요하다. 이를 위해 text8 (http://mattmahoney.net/dc/textdata) corpus를 이용하자. Text8 corpus는 위키피디아의 자료를 수집한 후, 정리한 자료이다. 웹브라우저나 다른 도구를 이용하여 다음 url에서 이 자료를 먼저 적당한 디렉토리에 다운로드 받는다.

http://mattmahoney.net/dc/text8.zip

예를 들면,
> wget http://mattmahoney.net/dc/text8.zip

> unzip text8.zip

2.3 Wordnet

우리가 하고자 하는 것은 단어의 의미 및 연관성을 찾는 일이므로, 사전을 참고하는 것이 도움이 될 수 있다. 구할 수 있는 사전에는 여러가지가 있지만, 여기서 우리 는 wordnet을 이용하기로 하자 (Fig. 1). (https://wordnet.princeton.edu/)

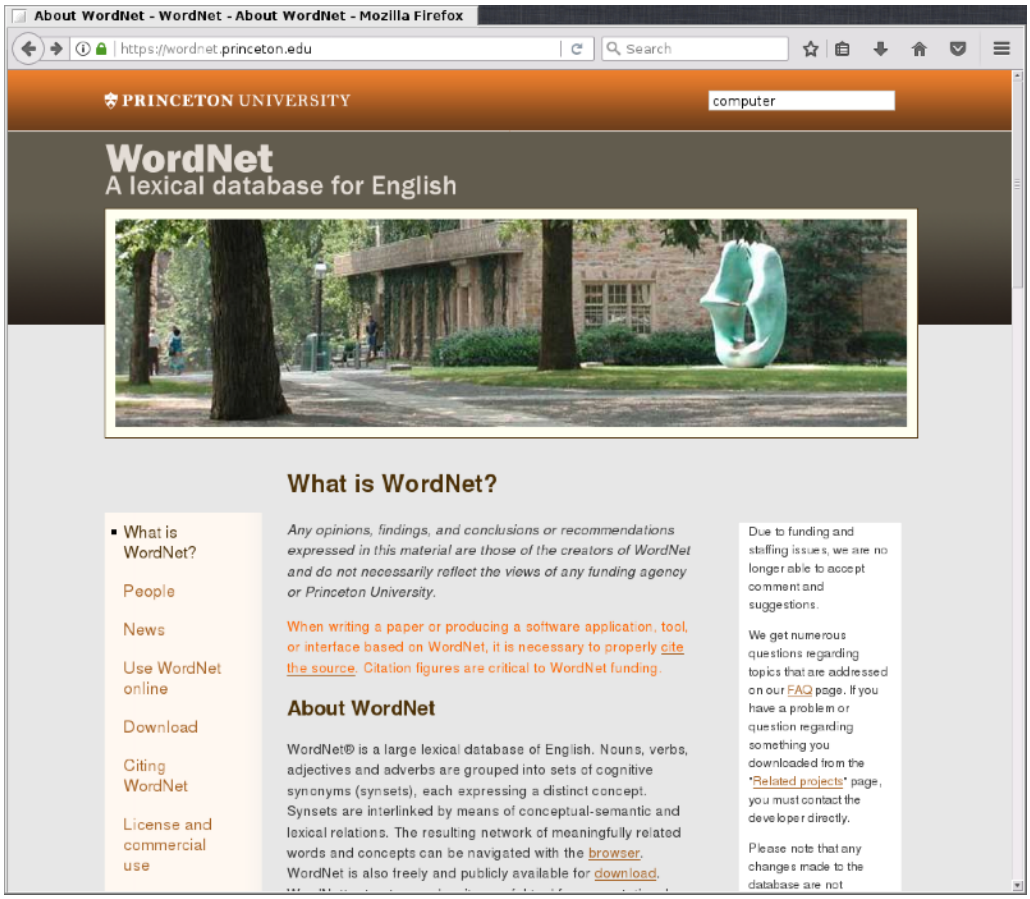

Figure 1: Wordnet.

WordNet is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept.

예를 들어, wordnet에서 'computer'를 검색하면, 다음과 같은 결과를 볼 수 있다 (Fig. 2).

a machine for performing calculations automatically

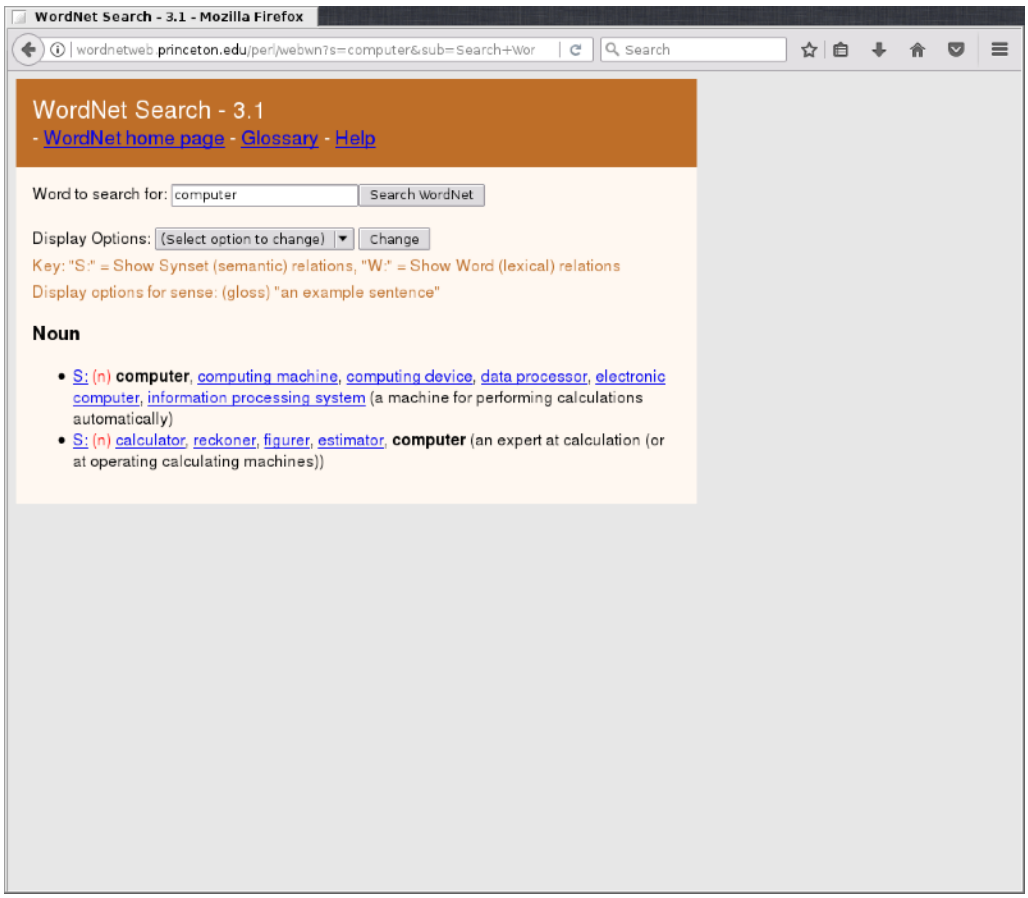

Figure 2: Wordnet에서 찾은 “computer”의 정의.

다만, wordnet을 웹에서 사용하는 것은 불편한 점이 있으므로, NLTK(http://www.nltk.org/)를 이용 하여 wordnet을 호출하자. 먼저, nltk를 설치하자.

> pip install nltk
각종 corpus나 사전 등의 자료는 nltk와 따로 배포된다. 다음과 같이 wordnet 의 자료를 내려받아 설치하자.

> python -m nltk.downloader -d ~/nltk_data wordnet

이때 사전이 설치될 디렉토리는 설명서(http://www.nltk.org/data.html)를 참조하여 적절히 수정한다. nltk를 통하여 wordnet을 호출할 수 있는지 확인해 보자.

In [1]: from nltk.corpus import wordnet as wn
In [2]: wn.synsets(’computer’)[0].definition()
Out[2]: ’a machine for performing calculations automatically’

3 과제

위에서 준비한 내용들을 바탕으로, 실제로 어떠한 일을 수행할 수 있는지 알아보자. 이를 위해, 다음 질문들에 대하여 적당한 코드를 작성하여 수행하고 결과를 제출하세요.

1. Gensim 의 Text8Corpus를 이용하여 text8을 로드하고, 다음과 같이 처음 10 개의 단어를 출력하시오. 다음은 하나의 예시이다.

In [54]: for sentence in sentences:

- ...: print(sentence[0:10])

- ...: break
  ['anarchism', 'originated', 'as', 'a', 'term', ...]

2. text8을 입력으로 하는 gensim word2vec model을 생성하시오.
   model = ...

3. 매번 word vector를 계산하는 것은 낭비이므로, gensim에서는 학습된 model 을 save, load할 수 있도록 지원한다. 학습된 모델을 save 하고 다시 load하는 code를 작성하시오.

4. model.wv.vocab의 type()을 확인해보자. 결과는?

5. model.wv.vocab을 print해본다. 다음과 같이 나오면 된다.
   In [24]: model.wv.vocab
   Out[24]:
   {'anarchism': <gensim.models.keyedvectors.Vocab at 0x7fc0dab94f98>,
   'originated': <gensim.models.keyedvectors.Vocab at 0x7fc0dab94080>, 'as':
   <gensim.models.keyedvectors.Vocab at 0x7fc0dab940b8>,
   'a': <gensim.models.keyedvectors.Vocab at 0x7fc0dab94128>,
   'term': <gensim.models.keyedvectors.Vocab at 0x7fc0dab94160>,
   'of': <gensim.models.keyedvectors.Vocab at 0x7fc0dab94780>, 'abuse':
   <gensim.models.keyedvectors.Vocab at 0x7fc0dab947f0>, 'first':
   <gensim.models.keyedvectors.Vocab at 0x7fc0dab948d0>, 'used':
   <gensim.models.keyedvectors.Vocab at 0x7fc0dab94a20>, 'against':
   <gensim.models.keyedvectors.Vocab at 0x7fc0dab94a90>,

6. word2vec 단어의 일부를 출력해보시오. 예를 들면 다음과 같은 결과가 나오 는지 확인해보시오.
   In [25]: print(list(model.wv.vocab.keys())[0:10]) ['anarchism', 'originated', 'as', 'a', 'term', 'of', ...]

7. 'computer'의 word vector를 구하시오.

8. (Sanity check 또는 identity 의 확인) 'computer'의 word vector로부터 가장 가까운 단어를 구하시오 (hint: similar by vector()를 이용하자.)

9. 이제, word vector들의 연산이 가능한지 알아보기 위해 다음 연산을 수행하고 결과를 말하시오.
   most similar = woman + king – man most similar는 무엇인가?

10. (Extra) Reverse dictionary는 주어진 설명으로부터 단어를 찾아내는 기법이 다 [1]. 예를 들어, 앞에서 우리는 'computer'의 정의가 다음과 같음을 알 수 있었다.
    a machine for performing calculations automatically

그렇다면, 반대로 이 설명으로부터 'computer'를 찾아낼 수 있을까? 이를 위해 몇가지 기법을 사용할 수 있다. 가장 단순한 모델은 설명을 이루는 각 단어들의 mean을 이용하는 것이다. 다음과 같이 질의를 만들어 보자.

```
In [4]: def definition(word):
...: return wn.synsets(word)[0].definition()

In [5]: definition('computer')
Out[5]: 'a machine for performing calculations automatically' In [6]: q = definition('computer')
In [20]: model.wv.most_similar(positive=q.split(), topn=50)
```

4 제출

• 결과물은 source code를 포함하여 보고서 양식으로 제출. file format은 pdf.

• due date: 9월 24일 (목) 23시

References

[1] Felix Hill, Kyunghyun Cho, Anna Korhonen, and Yoshua Bengio. Learning to understand phrases by embedding the dictionary. CoRR, abs/1504.00548, 2015.