Lab1 - Word embeddings training

Jiangnan Huang and You Zuo

January 14, 2021

Instructor: Sahar Ghannay

Abstract

This paper aims to make a discussion about the first lab exercise of word embedding. The objective of this lab session is to build and compare different word embeddings aproaches using the gensim and fasttext libraries. These embeddings will be trained on two different corpora: a small medical corpus (QUAERO_FrenchMed) and a large non-medical corpus (QUAERO_FrenchPress).

Keywords: Word2Vec, CBow, SkipGram, FastText

1 Introduction

1.1 Dataset

We used two dataset in the given file, **QUAERO_FrenchMed_traindev.ospl** and **QUAERO_FrenchPress_traindev.ospl**, they both contain corpora in "one sentence per line" format, with segmentation of tokens which are separated by spaces. The first dataset is a medical-related corpus, while the latter is a general-topic corpus.

1.2 Models

During our experiments, we implemented three different embedding models:

- **CBow**
  It learns to predict a target word leveraging all words in its neighborhood. The sum of the context vectors are used to predict the target word. The neighboring words taken into consideration is determined by a pre-defined window size surrounding the target word. The context is determined by a pre-defined window size surrounding the target word.

- **Skipgram**
  It learns to predict a word based on a neighboring word. To put it simply, given a word, it learns to predict another word in it's context.

- **FastText**
  It is built based on the SkipGram idea, it used a bag of character n-grams (also known as subwords) to represent a word, and each word is represented by the sum of its n-gram vectors. So it helps the embeddings understand suffixes and prefixes and can even cope with unknown words. But it usually takes a longer time to train compared with the previous two.

2 Experiments and Results

Before inputting the speaking corpus into our model, we tokenized the sentences, removed all the punctuations, and converted all uppercase letters to lowercase. Finally, we read 3022 sentences from the medical-related corpus and 38547 sentences from the non-medical-related corpus.

For each model, we trained them on respectively these two corpora, so in the end, we got a total of six models. Regarding the hyper-parameters, we set the hyper-parameters all identical for each model to better make a comparison among them.

- min count (word with frequency lower than this will be ignored) = 1
- length of word embedding = 100
- windom size (to determine the context) = 10
- number of epochs = 20

2.1 Evaluation on the same corpus

The words we are going to evaluate are: patient, traitement, maladie, solution, jaune. To evaluate the similarities among word representation vectors, we applied the cosine similarity to retrieve the most context-related word in each embedding model.

```
cbow_medical = Word2Vec.load('cbow_medical.model')
skipgram_medical = Word2Vec.load('skipgram_medical.model')
fasttext_medical = FastText.load('fasttext_medical.model')

for word in test_words:
    print(word+':')
    print(cbow_medical.wv.most_similar(word)[0])
    print(skipgram_medical.wv.most_similar(word)[0])
    print(fasttext_medical.wv.most_similar(word)[0],'\n')
```

```
10
11 # output:
12 patient:
13 ('symptômes', 0.9948393106460571)
14 ('carte', 0.8920832872390747)
15 ('patiente', 0.9999991655349731)
16
17 traitement:
18 ('expérience', 0.9797210693359375)
19 ('confirmé', 0.7372004985809326)
20 ('taaitement', 0.9999992251396179)
21
22 maladie:
23 ('infection', 0.986403226852417)
24 ('liée', 0.8259382247924805)
25 ('maladies', 0.9999959468841553)
26
27 solution:
28 ('conditionnés', 0.9839076399803162)
29 ('réchauffer', 0.910857617855072)
30 ('dissolution', 0.9999986886978149)
31
32 jaune:
33 ('capsule', 0.9976205229759216)
34 ('blanc', 0.878764271736145)
35 ('une', 0.9999926090240479)
```

Listing 1: three embeddings trained on medical-related corpus

The three models in Listing 1 were trained based on the medical corpus. So as we can see, most of the words above are also medical terms and have very high similarities with the given words. However, for the same corpus, the most similar word of a certain word depends on the embedding strategies applied.

```
1 cbow_non_medical = Word2Vec.load('cbow_non-medical.model')
2 skipgram_non_medical = Word2Vec.load('skipgram_non-medical.model')
3 fasttext_non_medical = FastText.load('fasttext_non-medical.model')
4
5 for word in test_words:
6     print(word+':')
7     print(cbow_non_medical.wv.most_similar(word)[0])
8     print(skipgram_non_medical.wv.most_similar(word)[0])
9     print(fasttext_non_medical.wv.most_similar(word)[0],'\n')
10
11 # output:
12 patient:
13 ('mototaxi', 0.5873354077339172)
14 ('hospitalisé', 0.6993711590766907)
15 ('contient', 0.9885638356208801)
16
17 traitement:
18 ('système', 0.7244834899902344)
19 ('médicamenteux', 0.5807715654373169)
20 ('farouchement', 0.9810761213302612)
21
22 maladie:
```

```
('perte', 0.5813629627227783)
('neurologique', 0.6623693108558655)
('trilogie', 0.9433014988899231)

solution:
('alternative', 0.6969314813613892)
('lancinant', 0.6705160140991211)
('résolution', 0.9928999543190002)

jaune:
('maillot', 0.883590817451477)
('maillot', 0.8162438273429871)
('lune', 0.9673931002616882)
```

Listing 2: three embeddings trained on non-medical-related corpus

These three model above in the Listing 2 were trained based on the non-medical corpus. From the results we can see that the most similar word of a certain word still depends on the embedding strategies, and the resulting word are not necessarily medical terms.

2.2 Evaluation on different corpora

```
cbow_medical = Word2Vec.load('cbow_medical.model')
cbow_non_medical = Word2Vec.load('cbow_non-medical.model')

for word in test_words:
    print(word+':')
    print(cbow_medical.wv.most_similar(word)[0])
    print(cbow_non_medical.wv.most_similar(word)[0],'\n')

# output:
patient:
('symptômes', 0.9948393106460571)
('mototaxi', 0.5873354077339172)

traitement:
('expérience', 0.9797210693359375)
('système', 0.7244834899902344)

maladie:
('infection', 0.986403226852417)
('perte', 0.5813629627227783)

solution:
('conditionnés', 0.9839076399803162)
('alternative', 0.6969314813613892)

jaune:
('capsule', 0.9976205229759216)
('maillot', 0.883590817451477)
```

Listing 3: cbow trained on different corpora

```
skipgram_medical = Word2Vec.load('skipgram_medical.model')
skipgram_non_medical = Word2Vec.load('skipgram_non-medical.model')

for word in test_words:
    print(word+':')
    print(skipgram_medical.wv.most_similar(word)[0])
    print(skipgram_non_medical.wv.most_similar(word)[0],'\n')

# output:
patient:
('carte', 0.8920832872390747)
('hospitalisé', 0.6993711590766907)

traitement:
('confirmé', 0.7372004985809326)
('médicamenteux', 0.5807715654373169)

maladie:
('liée', 0.8259382247924805)
('neurologique', 0.6623693108558655)

solution:
('réchauffer', 0.910857617855072)
('lancinant', 0.6705160140991211)

jaune:
('blanc', 0.878764271736145)
('maillot', 0.8162438273429871)
```

Listing 4: skipgram trained on different copora

```
fasttext_medical = Word2Vec.load('fasttext_medical.model')
fasttext_non_medical = Word2Vec.load('fasttext_non-medical.model')

for word in test_words:
    print(word+':')
    print(fasttext_medical.wv.most_similar(word)[0])
    print(fasttext_non_medical.wv.most_similar(word)[0],'\n')

# output:
patient:
('patiente', 0.9999991655349731)
('contient', 0.9885638356208801)

traitement:
('taaitement', 0.9999992251396179)
('farouchement', 0.9810761213302612)

maladie:
('maladies', 0.9999959468841553)
('trilogie', 0.9433014988899231)

solution:
('dissolution', 0.9999986886978149)
('résolution', 0.9928999543190002)

jaune:
('une', 0.9999926090240479)
```

```
28 ('lune', 0.9673931002616882)
```

Listing 5: fasttext trained on different corpora

The Listing 2 - 5 show the resulting words obtained for a certain given word with the same embedding strategies but different corpus(medical and non-medical corpus). For example when we found the most similar word for 'jaune' with Cbow Word2Vector model, we got 'capsule' from the medical corpus but 'maillot' from the non-medical corpus.