AdvNLP Lab 4: Linguistic Regularities in Word Embeddings

Julie Weeds

February 17, 2021

1 Getting started

This week you are going to be investigating linguistic regularities in pre-built word embeddings. You should have linked or downloaded the embeddings in week 1 (see Canvas), uncompressed them and made them accessible in your working directory. As before, you can load them in to python using the following code (it may take a while to run this because the embeddings file is very large (1.5G)).

```
from gensim.models import KeyedVectors
filename="GoogleNews-vectors-negative300.bin"
mymodel = KeyedVectors.load_word2vec_format(filename, binary=True)
```

You can now query the model with calls to methods of `mymodel` such as:

```
mymodel.similarity('man','woman')
mymodel.most_similar(positive=['man'])
mymodel['man']
```

Mikolov et al. (2013) propose the use of an offset vector method in order to answer analogy questions. For example, if you want to find the concept X which satisfies the analogy *"X is to China as London is to England"*, you need to find the concept closest to the point X in the vector space:

$$X = vector_{China} - (vector_{England} - vector_{London})$$

$$X = vector_{China} + vector_{London} - vector_{England}$$

You can do this with gensim using the following code:

```
mymodel.most_similar(positive=['China','London'],negative=['England')
```

The file `relations.json` contains lists of pairs of words which satisfy some syntactic or semantic relation. You can load it into a dictionary using the following code:

```
import json
with open('relations.json', 'r') as fp:
    testtuples=json.load(fp)
print(testtuples)
```

2 Tasks

1. Write a function which when given one *(capital city, country)* training pair can predict the capital of the other countries in the *capital-common-countries* list in `testtuples`
2. Use the correct answers, also given, to evaluate how accurate your capital-predictor is. You should calculate the average accuracy over all possible training pairs.
3. Looking at your predictions, can you think of an easy way to improve performance?
4. Adapt your code so that you can predict the country of which a city is capital. Is performance the same, higher or lower this way round?
5. Adapt your code so that you can consider any of the relationships in `testtuples`. Rank the relationships in order of easiness to predict. Why do you think some are easier than others?

6. A critic might say that the evaluation carried out in Mikolov et al. (2013) does not test the importance of the direction of the vector offset. *London* is close to *England*, so the vector difference is very small. Therefore, the method might do as well if it predicted the nearest neighbour of *China* as its capital. Implement this naïve baseline which predicts the closest neighbour of the test item. Evaluate it for the different relationships in `testtuples`. Does it come close to doing as well as the vector offset method for any of the relationships?

References

Tomas Mikolov, Wen tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of NAACL-HCT*.