DD1418/DD2418 Language Engineering
Johan Boye, Dmytro Kalpakchi
2018-11-22

Assignment 5

Readings: Read slides from the lecture 9 and/or consult this paper on Random Indexing.

Code: The skeleton code can be downloaded from Canvas or from

`http://www.csc.kth.se/~jboye/teaching/language_engineering/a05/RandomIndexing.zip`

Unzip the code in your home directory. Go to the folder `RandomIndexing` and type:

```
pip install -r requirements.txt
```

Now everything needed for the assignment should be installed.

Problems:
This problem set is dedicated to the exploration of the distributional word embeddings, a.k.a. word vectors, which became extremely popular after publishing the *word2vec* paper. A couple of years ago there were a number of mass media publications (here is one of them) highlighting that word vectors capture semantics of words via simple operations in the vector space, for instance,

king - man + woman = queen.

In this assignment you will try to demystify word vectors by exploring a much simpler approach than *word2vec*, yet comparably well-performing technique, called *Random Indexing*. Your main task is to **extend the random_indexing.py script to make it create distributional word embeddings from the 7 books about Harry Potter**.

Effectively your assignment consists of the following three tasks:

1. **Clean the raw text**. The books about Harry Potter are provided as plain unformated text files, which should not be used directly for training any word vector model. For example, the text can be as unformated as the folowing snippets.

```
1
HARRY POTTER
AND THE CHAMBER OF SECRETS
by
J. K. Rowling
(this is BOOK 2 in the Harry Potter series)
Original Scanned/OCR: Friday, April 07, 2000
v1.0
(edit where needed, change version number by 0.1)
C H A P T E O N E
THE WORST BIRTHDAY
```

Having a word embeddings generation problem in mind, one can identify multiple problems with this text:

- there is some unrelated text like `Original Scanned/OCR`;
- there are some formatting artefacts, like the word `C H A P T E O N E` or sentences being broken by the newline character;
- the punctuation is glued together with words, like `needed,` (why is it a problem?).

For this assignment we will disregard the first two problems (what does it mean for the created word vectors?). Your task is to implement a `clean_line` function returning a line without punctuation and numeric symbols.

To pass: You need to run `check_cleaned_text.sh` and get a "Success!" message.

2. **Create word vectors**. Write the code creating word vectors using the Random Indexing technique. This would involve two steps: building a vocabulary of words which you are to embed and using Random Indexing to create word embeddings.

   For the purpose of this assignment the vocabulary should contain every word present in any of the 7 provided Harry Potter books. **When creating word vectors, assume that the left context of the first word and the right context of the last word is empty.**

   To pass: You should be able to call `get_word_vector` function and get a word vector for the word if it exists in the vocabulary and `None` otherwise.

3. **Find the closest words**. Write the code finding the closest words to the given ones in the induced vector space using a simple k-nearest neighbours algorithm (we suggest using `scikit-learn's` implementation of kNN algorithm).

   To pass: You should be able to call `find_nearest` function with a list of words of interest and get a list of the 5 nearest words with similarities. You should also be able to answer the following questions. What similarity metrics can you use in your algorithm? Which one would you prefer to use? Why?

   Try to find nearest neighbours for the following words:

   ```
   Harry, Gryffindor, chair, wand, good, enter, on, school
   ```

   To give an example, our implementation returns the following 5 nearest neighbors for the word *Harry*: *Hagrid, Snape, Dumbledore, Hermione.*

   Experiment by changing various hyper-parameters of the Random Indexing algorithm, for instance:

   - change the dimensionality of the vectors to 10 with 8 non-zero elements (try different dimensionalities and number of non-zero elements);
   - change window sizes to the values of 2, 3, 10 making left and right windows both symmetric and asymmetric.

   How did the result change with the various modifications?