Information Retrieval - Assignment 1

May 13, 2019

Contents

Evaluate Existing Word Embeddings Models

Pretrained word embedding

I decided on using the `GoogleNews-vectors-negative300.bin` word embeddings. That word embedding got trained on Google news articles. It includes 3 million words over all put into 300-dimensional vectors.

5 nearest neighbors

For the nearest neighbor search I used the `most_similar` function from the `gensim` package. But that one technically uses only the cosine similarity for finding similar words, so the euclidean distance is missing.

SVD plot

In this case I only took the first 500 words from the model and outputted it with SVD as following plot. The Words get clustered according to their distance to each other, see in the plot for example numbers (three, four etc.) are relatively near to each other.

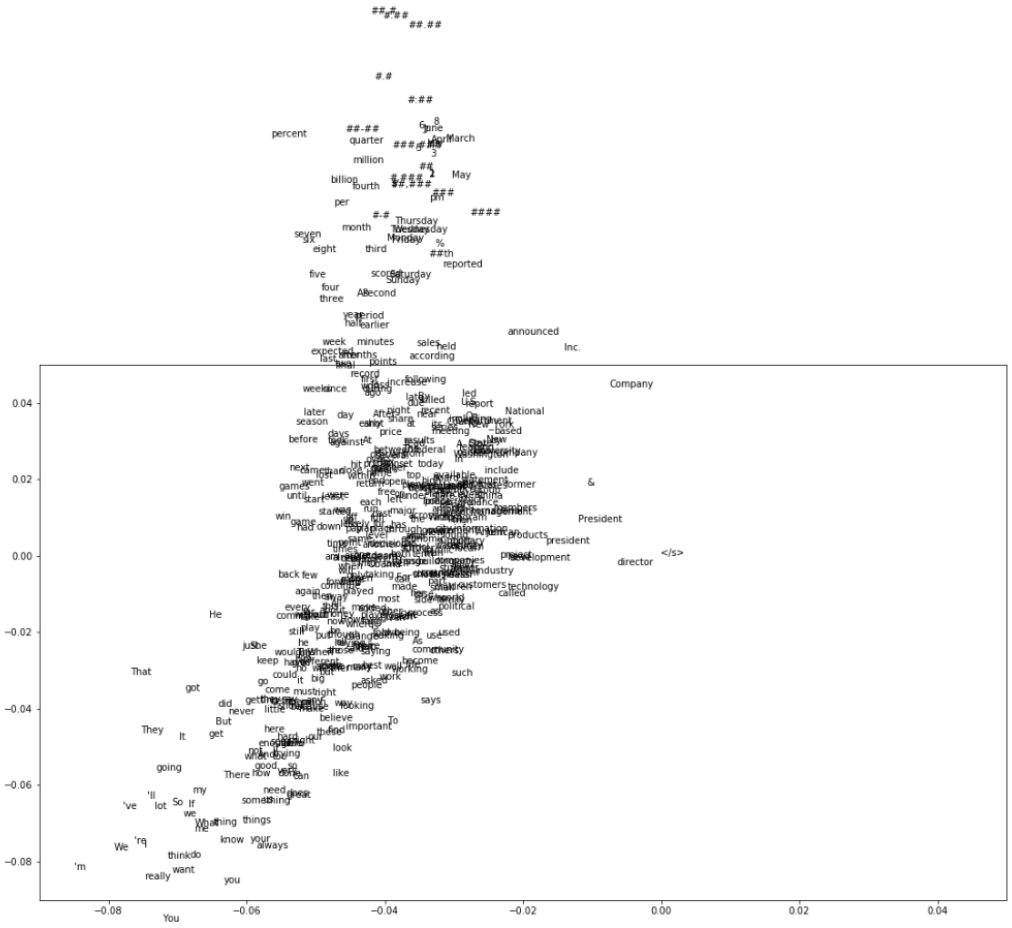

Figure 1: 500 words SVD plot

Main Task

Also for that task I used the `gensim` package again with the `most_similar` function. For getting the fourth word I pretty much only had to look at the distance between the first two words to be able to predict the fourth word. Example for predicting Iraq:

Athens Greece Baghdad X (Iraq)
X = Baghdad + Greece – Athens

Accuracy scores

Syntax	Semantics	Overall	Average
0.7400	0.7308	0.7358	0.7355

Word Embeddings for Text Classification

Also here using the embedding from to do the text classifications.

I went at it by only taking the vectors of the embeddings which are actually showing up in the texts (but needed to reduce the amount of used data for faster running time), those vectors I use then to train a `AdaBoostClassifier`. I create vectors for each document by taking the mean over all the words from the document to have a single vector for each document. Overall the model seems to perform rather well with an accuracy of `0.824` on the test data provided.

Contextually Propagated Term Weights for Document Representation

I only implemented the reading training, test data and transforming it to TF-IDF on which then I run a `NearestNeighbors` (as desribed in the paper) with `n-neighbors=1`. Which gives gives me a single neighbor on that I then can then predict the test data on from which I can get the following f1-scores.

F1	TD-IDF macro	TD-IDF micro	CPTW
Reuters	0.786	0.840	todo

Since I didn't directly know how to implement CPTW so it's not included.