|  |
| --- |
| **Word embeddings** |
|  |
| **TP3**  Geneviève Chafouleas - 20144781  Philippe Gagné - 20315 |

Abstract

The purpose of this assignment was to train word embedding models on the same corpus of blogs used for TP2 and Homework 1. We used our simple normalized data to train different word embedding models using genism. We then evaluated the most related words using the models compared to a simple list of words to then we annotated by hand based on specific rules set by us.

Introduction

This assignment consisted of training word embeddings using the corpus of blogs used for the TP2 and Homework 2. These raw data from the corpus has a lot of noise and badly written words. We used the basic normalized corpuses from TP2 to train the word embeddings. We trained 4 different words embeddings changing different parameters from the genism library. We then used these trained word embedding models to load using the spacy library and evaluate word similarities for each of the different models. Then we performed a manual annotation of the similar words using our own annotation rules described further in the report.

Data

The data used to train the word embeddings is a list of blogs used for both the TP2 and Homework 2. The blogs are filled with “dirty” data. By “dirty” we mean a lot of punctuation, spelling mistake, duplicate letters. The task of normalizing the data was done in TP2 and therefore we used one of our normalized Corpus from TP2. We used a simple tokenization and removal of both stop words and punctuation for the training of the word embeddings. We removed the punctuation and the stop words because we wanted to test the word embeddings only on full words to validate how well they are correlated with each other. The same corpus was used for each model training using the genism library.

Word embedding training

Using the normalized corpus described in the previous section we trained different word embeddings models using the genism library. We played with the parameters of the genism library in order to see the difference in train time, vocabulary size, number of encoded words and size on disk. We used the Word2Vec function from genism and change the following parameters (min\_count, size) to record the differences during training. The following tables contains the data from the different training runs of the word embeddings.

|  |  |  |  |
| --- | --- | --- | --- |
| **Train time (s)** | **Vocabulary size** | **Size on disk (MB)** | **Number embedding vectors** |
| 764.6 | 123907 | 138 | 123907 |

**Table 1 – Model 1 (workers = 4)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Train time (s)** | **Vocabulary size** | **Size on disk (MB)** | **Number embedding vectors** |
| 810.4 | 82356 | 91.1 | 82356 |

**Table 2 – Model 2 (workers = 4, min\_count=10)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Train time (s)** | **Vocabulary size** | **Size on disk (MB)** | **Number embedding vectors** |
| 784.9 | 123907 | 280 | 123907 |

**Table 3 – Model 3 (workers = 4, size =200)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Train time (s)** | **Vocabulary size** | **Size on disk (MB)** | **Number embedding vectors** |
| 773.6 | 82356 | 183 | 82356 |

**Table 4 – Model 4 (workers = 4, size =200, min\_count=10)**

The two properties modified for each train run as mentioned were the min\_count and the size. The min\_count removes all words that have a count in all the entries of the corpus that are lower than. The default value from the genism library is 5. We trained two models using min\_count set to 10. As we can see from Table 2 and 4 it has reduced the size of the vocabulary and the number of embedded vectors generated by the model. The size of the vocabulary is reduced by almost 16% of the original vocabulary size. It also as expected requires a lot less space compared to the other models as there are fewer embedding vectors.

The size property deals with the size of the embedding being generated by the model being trained. The default value is set to 100. We trained two models using a size of 200.The larger the value the more information can be encoded in the word embedding vector but at a cost of space. As we can see from Table 3 and 4 this requires a lot more space on disk. It requires almost twice the amount of space to save the vectors when setting the size of the embedding to 200 compared to 100 if we compare the Table 2 and 4.

In the next section we will compare the models to evaluate how the word embeddings generated by the models can be used to evaluate word similarities and compare results found.

Word similarities

In this section we will evaluate the different models on the task of finding the similarity of different words using the models trained in the previous section.

We used the spacy library from python to load each model. The use of the similarity function from the spacy model to evaluate the similarity score between two tokens. We decided to keep all the words that got a similarity score higher than 0.5. We used the first 1000 words of the vocabulary to evaluate the similarity scores and save them in a text file. For each of the 1000 words chosen we went through the whole vocabulary and calculated the similarity score between each token. We only saved the 20 words with the highest similarity score.

Examples of the generated similarity between words can be seen in the annex section for each of the models trained.

By looking the list of similar words generated by each model in the annex we notice that the train parameters have quite an effect on the order of the word similarity. For example, if we look at the word Australia, we can see that the similar word order changes between one model to the next.

Also, we can notice that some words are captured by certain models but not others. If we look at the Model 2 and Model 4 results we can see that the number of similar words for each of the chosen words is smaller due to the fact that the vocabulary is reduced during training and therefore would have a smaller amount of vectors to chose from.

If we look at the Model 3 and 4, we can notice that the words chosen as similar seem to have a better similarity. For example, if we take the word craving in the Model 3 the top similar words are as follows; cravings, hankering, crave compared to Model 2 which is as follows; cravings, appetite, eating. We can see that they both have the first word as being cravings which is a good words as it is the plural of craving but the second word with highest similarity for Model 3 is hankering which is a very good synonym to craving compared to appetite which is related but not a synonym. This better granularity is because Model 3 and 4 have a larger embedding size for the vectors which allows the model to learn better subtleties between words. We increase the models embedding size by 2 compared to Model 1 and 2.

Some words did not seem to have a great impact on the change of the training parameters set during training. For example, nuts seemed get more of less the same words when looking for similar words in the vocabulary. Another word was hello it captured mostly the same similar words from one model to the next. This is probably due to the fact that theses words did not have many variations in the corpus reducing the different options possible and yielding similar word embedding vectors from one training to the next.

An interesting observation from the results of each model is the one found by looking at the word nuts. The similar words that were found using the trained embeddings are as follows; insane, crazy, overboard, bonkers, batshit. We can see that the embeddings learned must do more with the meaning of nuts being crazy. This is very interesting since this word can have two meanings one being someone is nuts and one that is the food nuts. We can see that the word embeddings learned can vary depending on the corpus used to train. We can see how the corpus can have a big impact on how well it will perform on specific tasks. If we use this embedding on a task that would have the word nuts used in the context of food, it would cause the task to perform very poorly. Therefore, if it is possible to train your embeddings on your corpus or add embeddings from your corpus to bonify pretrained embeddings this will help when training on a specific task using those embeddings.

From simply comparing the similar words in the annex we can see that the Model 4 seems to yield words that are more similar than the other models. This is simply at a high level as we are not comparing the actual annotation of each word chosen as being similar. In the next section we will annotate each similar word to get a better idea how this model performed.

Word annotation

In this section we explore the annotation of the similar words generated from section 4. The annotation will be performed manually using the following rules to tag each word.

**Cohyponym [1]:** [Words](https://www.thoughtco.com/word-english-language-1692612) that are hyponyms of the same broader term (that is, a [hypernym](https://www.thoughtco.com/hypernym-words-term-1690943)) are called *co-hyponyms*

**Hyponym [1]:** term used to designate a particular member of a broader class

**Synonym [3]:** a word or phrase that means exactly or nearly the same as another word or phrase in the same language

**Related:** connected words

**Part of speech:** In traditional grammar, a part of speech is a category of words that have similar grammatical properties

**Not related:** words that are not related to the word in question.

**Polysemy [3]:** the coexistence of many possible meanings for a word or phrase

**Hypernym [4]:** is a word whose meaning includes the meanings of other words

**Homonym [2]:** one of two or more words spelled and pronounced alike but different in meaning (such as the noun *quail* and the verb *quail*)

**Associated:** This would be when the word is usually employed with the other word in question but is not related in a semantic way.

**Antonym [3]:** a word opposite in meaning to another

**Conjugation:** same word but conjugated or put to plural.

We used these following definitions to annotate manually the similar word generated by Model 3. This task was quite hard as certain words can be ambiguous to determine the true nature of their annotation and could pertain to multiple buckets.

In the annex you can find some examples of the annotated words.

Even with the definition of each annotation it can be quite hard to annotate and can be quite a subjective task. I would have been expected the firsts words in the list of similar words would be either Hyponym, cohyponym or synonym as these would lead to show a better training on the embeddings since this would show that the model learned important correlations between words like when we see Hypernym.

But it was noticed that some words performed extremely well where all the words found as being similar were cohyponyms. For example, in the annex is we look at the word Australia we can see that the large majority of the words that were similar were tagged as cohyponyms of Australia as they were almost al countries.

But on the contrary we had words where many of the words chosen as being similar where not even related to the actual word in question. We can notice this when we look at the word “be”. We can see what all the words tagged as being similar are not really related or even close in semantics to “be”. This brings up the question of why this word was similar by the model as for us this is without a doubt not a related word. Was it because it is seen very often next to the word in question? This might be the case and would give a good clarification as to why since the models trained with genism have a default window\_size value of 5. This means that they can keep 5 words in history for the training phase. So, it might be a possibility that the unrelated word in terms of meaning was found very often next to the word.

Another interesting observation was that in many cases the words being found as similar were words that are not related in terms of semantics but more words that are very highly correlated when used together. This again could be because the window\_size used during training was quite large and may have contributed to these being highlighted as being similar. These associative words were not found as the words with the highest similarity score, but it is still interesting to note that the model picked up on the subtlety of not only related words but words that are highly correlated in a sentence building point of view.

Conclusion

In conclusion, we trained 4 models using the genism library on a large corpus contains blogs from different age groups. We modified the training parameters in order to compare these models on the task of identifying similar words to a specified word.

These models were used to generate a list of similar words set in order of most to less similar using the spacy library. The threshold used to determine if the similarity test passed was >0.5. Using the generated words an annotation of the output of Model 4 was used using the following annotation criteria’s: cohyponym, hyponym, hypernym, synonym, related, part of speech, polysemy, homonym, not related, associated, antonym, conjugation.

From very high lever observations of the similar words generated by each model we determined in a qualitative way that Mode 4 seemed to infer in general the best words compared to the other models. This is because the min\_count was increase therefore reducing the words with less count in the vocabulary and increasing the embedding size which would increase how much can be encoded in the embedding therefore carrying more information compared to a smaller embedding.

In order to evaluate fully the effects of each of the training parameters that can be modified using the genism library we could have played also with the window\_size which we believe would have made an effect on how the embeddings are learned. Also, the normalization used to train the model stayed fix it would have been interesting to test the same model on different forms of the normalized data and see how the embeddings differ from one model to the other.

As was shown word embeddings can be quite good at encoding important correlations between words adding much more richness compared to the bag of word vectorized form of a sentence. It encodes a lot more information in each word which can further help in training model on specific task related to language.

**REFERENCE**

[1]Hyponym definition : <https://www.thoughtco.com/hyponym-words-term-1690946>

[2]Homonym definition: <https://www.merriam-webster.com/dictionary/homonym>

[3] Oxford dictionary

[4] Hypernym definition: <https://www.thoughtco.com/hypernym-words-term-1690943>

**ANNEX**

Some examples of word embeddings.

**Model 1:**

**australia 176** thailand,finland,japan,europe,germany,philippines,taiwan,canada,italy,ireland,norway,nz,asia,russia,china,scotland,indonesia,switzerland,spain,Poland

**hello 9** hi,hey,greetings,hullo,howdy,goodbye,hiya,welcome,hellow

**craving 94** cravings,appetite,eating,binge,yogurt,hankering,caffiene,caffeine,consumed,munchies,fattening,hungry,crave,snacking,caffeinated,caffine,mmmmm,tasted,mmmm,nicotine

**clinic 155** hospital,checkup,pharmacy,icu,oncologist,specialist,dermatologist,doctor,appt,checkups,pediatric,doctors,vet,outpatient,orthopedic,prenatal,gyn,surgeon,pediatrician,dental

**nuts 5**

crazy,insane,bonkers,mad,batshit

**Model 2:**

**australia 183** thailand,philippines,finland,canada,taiwan,japan,norway,germany,europe,spain,asia,sweden,indonesia,china,malaysia,ireland,scotland,mainland,italy,brazil

**hello 9** hi,hey,greetings,hullo,howdy,goodbye,hiya,welcome,hellow

**craving 84** cravings,appetite,hankering,eating,snacking,fattening,yogurt,caffiene,consumed,crave,hungry,caffeine,indulged,mmmmm,munchies,carbs,binge,mmmm,caffine,digesting

**clinic 152** hospital,pediatric,oncologist,checkup,pharmacy,rehab,nurses,specialist,icu,outpatient,appt,dermatologist,psychiatric,orthopedic,hospice,chiropractor,nurse,doctors,veterinary,vet

**nuts 4** insane,crazy,bonkers,batshit

**Model 3:**

**australia 90** japan,thailand,canada,europe,germany,finland,asia,ireland,sweden,taiwan,nz,switzerland,france,spain,norway,indonesia,philippines,iceland,poland,bangladesh

**craving 22** cravings,hankering,crave,appetite,eating,munchies,caffeine,indulged,snacking,binge,yogurt,mmmmm,consumed,caffiene,mmmm,munching,fattening,carbs,caffine,eaten

**hello 3**

hi,hey,greetings

**clinic 90** hospital,pediatric,checkup,icu,outpatient,specialist,pharmacy,gyn,orthopedic,oncologist,psychiatric,surgeon,doctor,chiropractor,nurses,nurse,appt,dermatologist,psychiatrist,rehab

**nuts 3**

crazy,insane,bonkers

**Model 4:**

**australia 167** japan,europe,philippines,thailand,germany,canada,france,spain,finland,ireland,hawaii,taiwan,melbourne,italy,montreal,china,norway,sweden,russia,asia

**hello 9** hi,hey,goodbye,howdy,goodnight,greetings,hiya,heya,welcome

**craving 29** cravings,hankering,munchies,crave,appetite,consumed,snacking,craved,indulged,eating,binge,addiction,caffiene,caffeinated,yogurt,caffeine,delicious,hungry,nicotine,fattening

**clinic 85** hospital,specialist,pediatric,checkup,dermatologist,pharmacy,rehab,spca,triage,orthopedic,outpatient,icu,hospice,checkups,vet,veterinary,appt,doctor,psychiatric,nurses

**nuts 5**

insane,crazy,overboard,bonkers,batshit

Examples of word annotations

**boolean** 740 parameter[COHYPO],numeric[RELATED],tcp[RELATED],specification[RELATED],runtime[RELATED],interfaces[COHYPO],constructor[RELATED],identifier[RELATED],filename[RELATED],binaries[COHYPO],namespace[COHYPO],ejb[?],ioexception[COHYPO],versioning[COHYPO],variable[COHYPO],protocol[RELATED],dataset[COHYPO],metadata[COHYPO],clr[?],subset[RELATED]

**perpetuates** 782 perpetuating[CONG],patriarchal[NOT\_RELATED],propagated[SYNO],pervading[SYNO],perpetuation[CONG],reinforces[SYNO],undermines[NOT\_RELATED],reinforcing[SYNO],obscures[NOT\_RELATED],fallacious[NOT\_RELATED],exposes[NOT\_RELATED],perpetuate[CONG],antithetical[NOT\_RELATED],characterised[NOT\_RELATED],inescapable[RELATED],idolatry[NOT\_RELATED],predominant[NOT\_RELATED],affirms[RELATED],espouses[NOT\_RELATED],eugenics[ASSO]

**classics** 79 selections[NOT\_RELATED],novels[ASSO],soundtracks[ASSO],anthology[ASSO],compilation[ASSO],nonfiction[?],classic[CONG],classical[CONG],hardcover[ASSO],composers[ASSO],genre[ASSO],greats[RELATED],biographies[RELATED],novelists[ASSO],genres[HYPER],faves[ASSO],screenplays[RELATED],composer[ASSO],paperback[RELATED],masterpiece[ASSO]

**clinic** 92 hospital[COHYPO],checkup[RELATED],pharmacy[COHYPO],outpatient[RELATED],pediatric[COHYPO],icu[COHYPO],specialist[RELATED],appt,dermatologist[RELATED],rehab[RELATED],oncologist[RELATED],nurses[RELATED],orthopedic[RELATED],nurse[RELATED],pediatrician[RELATED],doctor[RELATED],clinics[CONG],nicu[COHYPO],psychiatric[RELATED],chiropractor[RELATED]

**australia** 96 thailand[COHYPO],canada[COHYPO],japan[COHYPO],europe[RELATED],finland[COHYPO],taiwan[COHYPO],switzerland[COHYPO],nz[COHYPO],germany[COHYPO],norway[COHYPO],sweden[COHYPO],philippines[COHYPO],ireland[COHYPO],iceland[COHYPO],melbourne[RELATED],bangladesh[COHYPO],asia[RELATED],france[COHYPO],spain[COHYPO],indonesia[COHYPO]

**be** 17 knw[?],yall[NO\_RELETED],shes[NO\_RELATED],tts[?],thats[?],isnt[ANTO],reallie[?],knoe[?],tht[?],endin[NO\_RELATED],tats[?],anw[?],thot[?],mayb[?],hows[?],izzit[?],noe[?]