Designed report by Abdelrahman Rezk

COVID-19-Arabic-Tweets-Dataset

We have collected more than 3, 000, 000 tweets from twitter API, besides cleaning these tweets and we have make some analysis to get what is behind these tweets.

- Eng: Ayman Mahgoub
- Researcher at electronics research institute
- E-mail: Ayman Mhgb

- Eng: Abdelrahman Rezk
- Teaching Assistant at Arab Open University & NLP Engineer
- E-mail: Abdelrahman Rezk

Corresponding Code Files

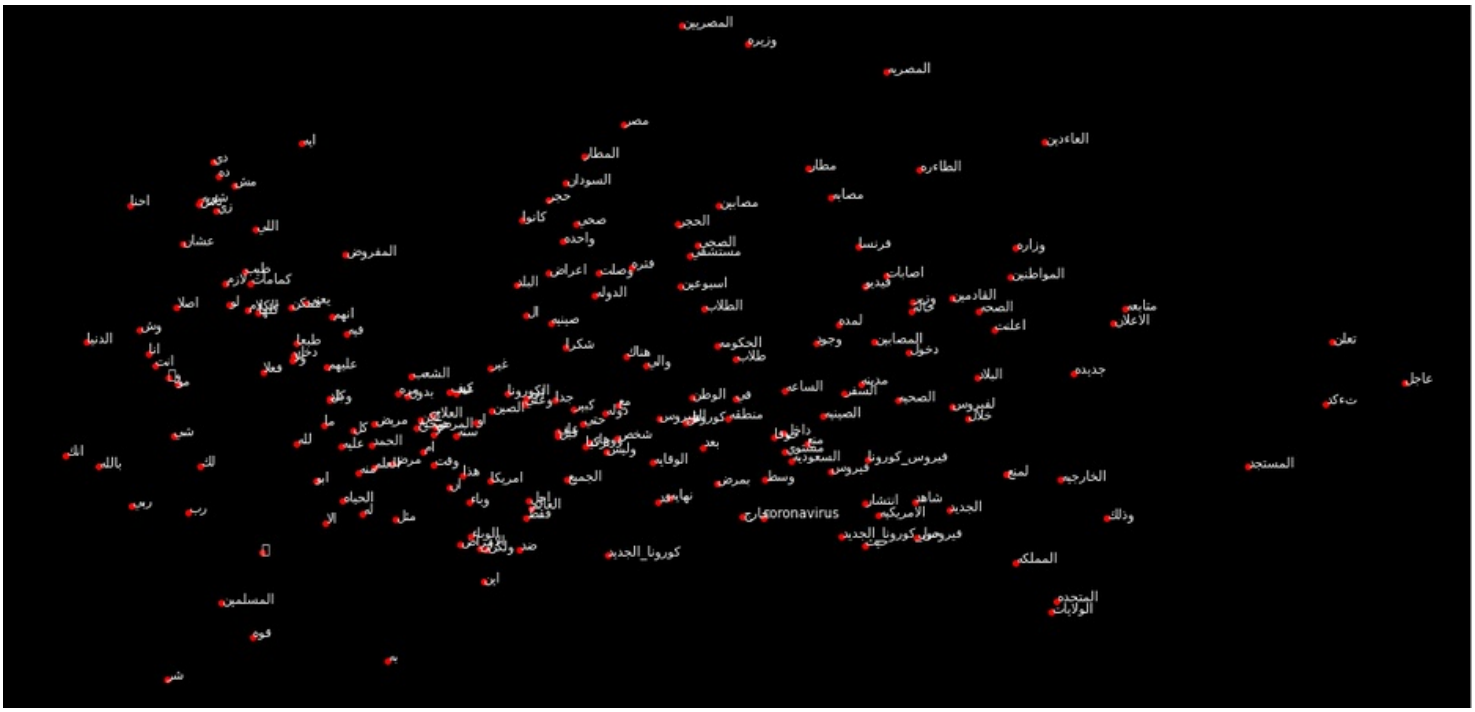
Files included in direction:

- config_files

Files Name:

- word2vec_anaysis.py
- features_engineering.py

----------word2vec_anaysis File----------



Word Embedding

Word embeddings are a modern approach for representing text in natural language processing.

So we have used one of these techniques which is Word2Vec to knowing what is the most words describe our dataset based on the context of the words that appear in it.

Unlike the usage of CountVectorizer or TF-idf models,which depends on the word count or the rare words and they depends on the Bag-Of-Words approach at the end. Word2Vec come to get an intuition about the words itself how it's related to other words from its category like words of [man, women], they have described different gender but have some of the features attach each other together, like the same context they appear in and the meaning is in one category of Gender and others like juices [Orange, Apple and so on].

We have impletent some of the function in this file and others we have used from previous work:

- dir_max_all_tweets
- word_to_vec
- save_words
- Features Reduction Graph using PCA

For Features Extraction function

- tfidf_vectorizer_for_supervised_models
- load_tfidf_vectorizer_for_supervised_models

dir_max_all_tweets

To build our Word2Vec model we need to know which review has the max number of words, this required to train the model for future work, but as for this function also we return the tweets of each file as a list of tweets each of these tweets is a list of words, as we can see ['كورونا', 'شخص', ' 1000 ', 'علي', 'الصحي', 'الحجر', 'تعلن', 'الامريكيه', 'المتحده', 'الولايات', 'الرئيس'] this is required by word2vec model.

word_to_vec

Moved from what model require to work with, to the model itself which also take these parameters:

number_of_features

The number of features we need for each word, and the number of features is that word represented in number of dimensions spaces.

window_size

Then we will use window of 7 words as we know from the language model there is **Bigram** and others N-gram so we use 7-grams as our window which helps to detect for each word which context appears in as well as knowledge from other words beside this words which we have here 7 words before and 7 after.

min_words_count

Here we tell the model to consider words that have appear a lot in the data and we have considered that the minimum count is 500 as our data have more than 200, 000 tweets in the first direction and more than 150, 000 tweets in the second direction.

Note 1

We can see how models understand the words related to each other are in have most of the same features which represented in have a close location on the dimensions of the graph above.

Also, we can notice that by:

```
In [9]: word_to_vec_model.wv.most_similar('بالفيروس')

Out[9]: [('0.5693107843399048', 'بفيروس'),
         ('0.5517096519470215', 'بكورونا'),
         ('0.45561763644218445', 'بالكورونا'),
         ('0.4381934702396393', 'بفايروس'),
         ('0.4354102611541748', 'حاله'),
         ('0.4131368100643158', 'مءكده'),
         ('0.4125266671180725', 'وفاه'),
         ('0.4038786292076111', 'الفيروس'),
         ('0.3810560703277588', 'بمرض'),
         ('0.38051560521125793', 'فيروس_كورونا_الجديد')]
```

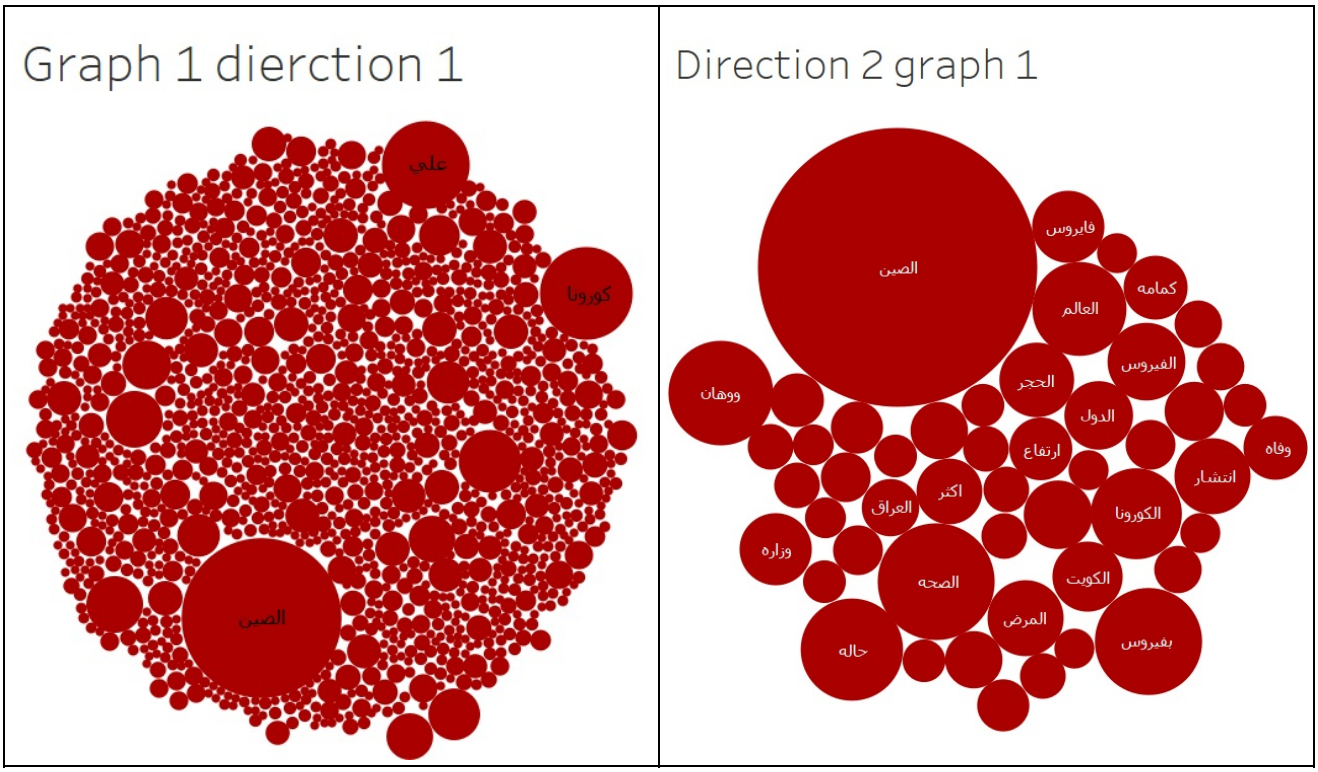
save_words

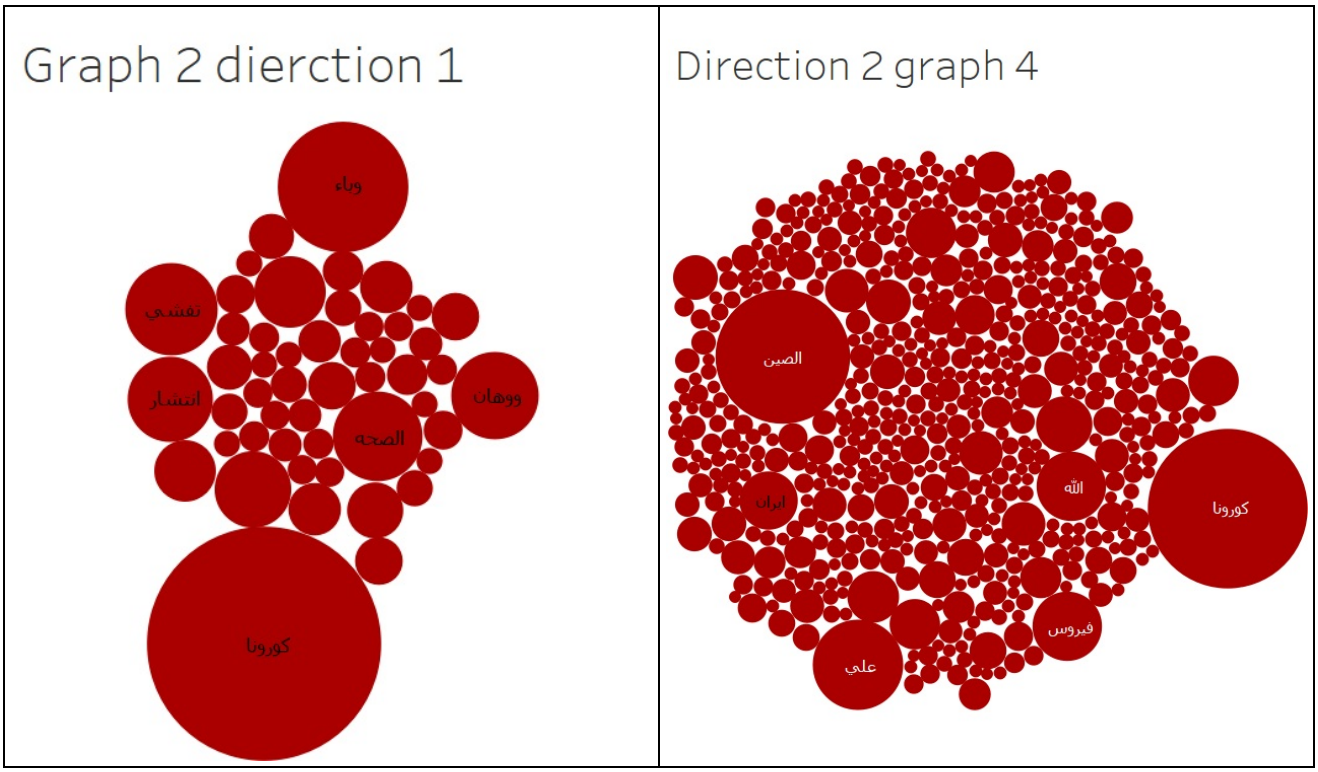
From this point and after we get the model work we need to save the words that appear a lot in our data in separated file to graph it with Tableau, besides of that we choose the most 50 words affect our dataset to graph it also, not just these words we consider also the frequency of these words.

Features Reduction Graph using PCA

From these words we get from the Word2Vec model we have graphed the first image in the report depends on the idea of Features Reduction as we have 300 features we can graph like these spaces so using Principal component analysis, we have reduced the features to just **2-feature**, and it will understand that the close features of different words will be looks like in one cluster as we can see in the graph above.

Snapshot of the Table after we get the most words affect the dataset





----------features_engineering File----------

The machine learning model can work with numbers not text data and for this point we need to provide the Models of machine learning the data in numbers and this process can be done by what is called features extraction and there are different Algorithms of these process one of them and the basic Algorithm is called One-Hot Vectorization and its work as for all of the words in the corps we used ordered the words using the alphabetic order and give each of the indexes from 0 to the last word in our available data and for this ordered the algorithm used to predict which words are similar to each other and thus lead to bad result because some words like [Orange & Apple] have the same meaning but the model will keep these words away of each other and make words like [Apple and again] to have similar meaning based on alphabetic order and for this we used another technique that helps us get a better result in training the machine learning algorithms like TF-IDF (term frequency & inverse term frequency).

tfidf_vectorizer_for_supervised_models

The first part is TF that the term frequency is the number of times a word appears in a document divided by the total number of words in the document. Every document has its own term frequency. The second part is IDF inverse document frequency is the log of the number of documents divided by the number of documents that contain the word w. Inverse data frequency determines the weight of rare words across all documents in the corpus. Also, the TF-IDF makes these rare words have more weights because its repeat is not a big as other words like [they are and others] that may have a lot of time in our text so it takes the inverse after counting the number of each term over its document and overall corpus.

load_tfidf_vectorizer_for_supervised_models

Just load the tf-idf model to be used.