CSE499A Capstone
in
Building COVID Knowledge Base from Medical Journals

SENIOR DESIGN PROJECT

May 14, 2021

Md. Jubaer Khan
ID:1721616042
Walidul Alam Ehab
ID:1631214042
Department of Electrical and Computer Engineering
North South University

1 Abstract

Natural Language Processing (NLP), a subfield of linguistics, computer science and artificial intelligence, is concerned with the interactions between computers and human language, in particular with how computers are programmed to process and interpret large quantities of natural language data. The outcome is a machine that can "understand" the content of documents, including the qualitative complexities of the language inside them. The program will then reliably extract data and observations found in the documents and categorize and organize the documents themselves. Our project aims to utilize the benefits of NLP, as we use NLP to extract information about COVID-19 and other deadly diseases from medical papers and help make a system that understands the essence of diseases.

Contents

2 Introduction

World is in a state of setback and injury due to present pandemic situation. Hospitals are rushed, diagnosis is slow and certainty for safe physical conduct cannot be guaranteed. Everyone and everything is subjected towards a new form of tension and environmental pressure, and the usual system of conducting affairs cannot cope up. Almost every field has adapted or are trying to adapt to new ways to keep pace with the new challenges. It is necessary, and even more so for the field of medicine. Even though the rise of global pandemic has led to increase in the number of hospitals and the degree and form of care, these measures are not consistent throughout the globe. There are numerous places where efforts to tackle the threat are coming up short, mainly due to less hospitals, limited number of medical professionals or those with updated knowledge about the recent findings about this deadly form of disease. This puts the lives of people living in these areas at risk, and demands better counter measures. Preparing medical personnel with the latest of information about COVID or other diseases is a lengthy process, since there are thousands of people around the globe conducting research about different aspects about the virus and, therefore, there are thousands of papers. It seems by the time professionals are up-to-date with the papers released till now, the virus may assume different mutations or there would be new findings which contradict what is known about the virus till now. With the added pressure of limited hospitals and staff, the world is in a dire need of a technological advancement that accelerates the process. We aim to create an NLP-based model which prevails to gain essence of COVID and other deadly viruses from thousands of recent papers and can further help diagnose people from the comfort of their homes. We would use our model as in integration to a sort of online platform where patients would get accurate diagnosis or explanation as to what he/she might be going through.

3 Background

As the COVID-19 pandemic start people from all over the world start finding the solution for fighting this unpredictable situation. Scientists from all over the world tried to develop new and possible effective ways of fighting this pandemic. There are lots of medical journal articles that are available. People start to find out how to get those diseases related information with more automation system.

3.1 Informative keyword finding from documents

In [1] they proposed a method where used a graph-based model for extracting the critical information from available research articles. To train model, they used over 10,683 article's abstracts. Their motive is to find essential details connected with COVID-19. They tried to achieve this goal by dividing the main graph into three significant subgraphs. Those three subgraphs are transmission, drug types, and genome research that directly related

to coronavirus. And each of the subgraphs also contains many vital keywords that are connected with them. This graphical connection helps them to find related disease, the drug that can be effective for the virus type etc. In the paper of INFORMATION MINING FOR COVID-19 RESEARCH [1] they took three different necessary steps for analysing the text. Mainly keywords searching [2], classification [3] and lastly topic modeling [4] are used for basic text analysis. But for finding the connection among the words in text graph can be a handy approach. It also helps to create a connection between the words and syntactic ordering [5]. This is why they chose a graph-based model for important information mining related to the desired topic. The graphical model helps to find the distance between nodes (nodes are representing words here). Connecting edges helps to find the selectivity and neighbouring nodes helps to determine the node's weight. Through the node's weight, words are ranked. This model helps visualise the complex interconnected network of words and extract interactive and informative information from documents [6]. They chose to select the article's abstract of the COVID-19 dataset, where most of the information is briefly discussed. For this graph-based model, they came with a directed graph where each node represents the unique word, and each edge connects the contiguous words based on their position in the sentence. Through this model, they gain a sequence-based text graph from dataset's 10,683 articles. The reason behind this choice is most of the vital information of a research article contained by abstract. Firstly they remove those words that have no connection with keywords(e.g. to, for, where) and stop words. They have also found a list of 2796 stop words. This step helps them to reduce graph complexity. For measuring the word importance, they applied an algorithm called betweenness centrality also known as BC measurement [7]. They also divide their information mining into some subgraphs'. They targeted the coronavirus as their main word of the global graph. All these words are chosen through BC measurement. Under it, they selected some subgraphs with words like Drug, Transmission and gene. Those are selected because of their log BC values. Their generated each sub-graphs have some topic section which contains related words of that topic. For example, the gene subgraph has three cases disease, host and biomolecular. Diseases names('Malaria', 'Rabies', 'Dengue', 'Chikungunya') related to transmission subgraph. All of the words contain a BC measurement which indicates the importance of the word with related topic. Through this process of information extraction, their model can recommend informative drug types, pathogens etc. for concerned researchers.

3.2 COVID-19 Literature Clustering

In this work, Maksim Ekin Eren suggests a method to help health professionals get new coronavirus information. Clustering for labelling can reduce dimensionality to improve data visualisation. He uses a scatter plot to show literature. This will help to share highly similar label between research articles that plotted near each other. Natural Language Processing (NLP) is used for parsing the text of documents [8]. Frequency–inverse Document Frequency (TF-IDF) to convert the distance into the feature vector [9]. t-Distributed Stochastic

Neighbor Embedding (t-SNE) is used for clustering similar research articles into a two-dimensional plane [10]. After that, he used Principal Component Analysis (PCA)[11] for dimension projection of the dimensions of plane X. And kept those dimensions with .95 variance for reducing noise. Also used Stochastic Gradient Descent (SGD) for classification of the data.

4 Methodology

4.1 Basic Workflow

Our workflow includes somewhat familiar approaches to that of other machine learning NLP tasks. Our first step is acquiring a machine-readable dataset. We would then move on to pre-process the data by making sure the data-set does not contain any duplicates, irrelevant data, and does not contain sections which are almost entirely empty. Once we are confident with our pre-processed data, we would then prepare our data for word embedding. Once we embed the words, we will then visualize the word vectors in low dimensionality to access the quality of embeddings and also to find interesting patterns or information regarding COVID.

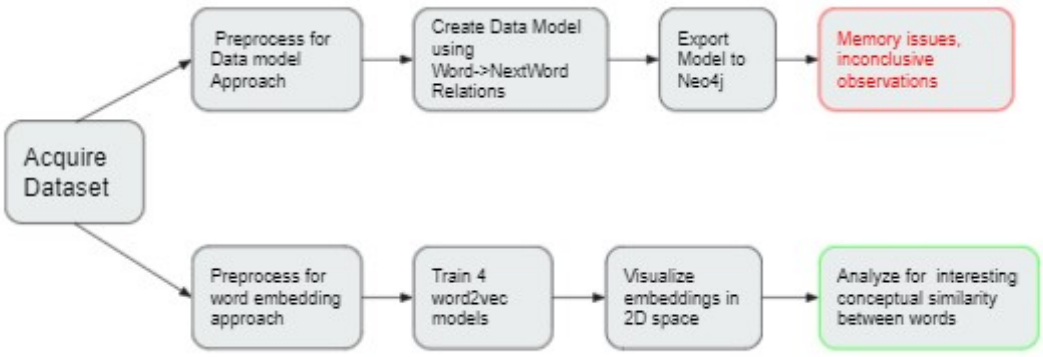

Figure 1: Architecture: Overall architecture of our proposed model.

4.2 CORD-19 Dataset

The White House and a consortium of leading study organizations have prepared the COVID-19 Open Research Dataset in reaction to the COVID-19 pandemic (CORD-19). CORD-19 is a database for over 200,000 scientific publications on COVID-19, SARS-CoV-2, and associated coronaviruses, including over 100,000 full text articles. The global research community is provided with this publicly accessible dataset to apply recent developments

in the processing of natural language and other AI techniques to generate new insights in support of the ongoing battle against this infectious disease.

4.2.1 Inside the CORD-19 Dataset

As of this time, the dataset includes information about 200,000 journals, with access to about 100,000 full text. The dataset contains metadata information about every journal present. These metadata files include properties about the journals such as the abstract portion, publication medical id, doi, authors, etc. The dataset also contains full text information about more than half of its PDF and PMC journals. There about 100,000 full text information about the PDF and PMC papers each and about 400,000 metadata information. More detailed information about the datasets are shown in Figure 2.

Total metadata row	414020
PDF	147686 JSONs
PMC	109662 JSONs
Full-text available	257348

- Cord_uid
- Sha
- Source_x
- Title
- Doi
- Pmcid
- Pubmed_id
- License
- Url
- s2_id
- Abstract
- Publish_time
- Authors
- Journal
- Mag_id
- Who_convenience_id
- Arxiv_id
- Pdf_json_files
- Pmc_json_files

Figure 2: Metadata information of COVID-19 dataset.

4.3 Pre-processing Data

We decided to analyze the abstract sections of each journals, as we believed that the abstract section captures the summary of a paper. Since the metadata file contained abstract portions of all the journals, analyzing the metadata was sufficient.

4.3.1 Analyzing Metadata

We extracted the title and abstract of each journal from the metadata file into a Dataframe where each row represents a single journal. A screenshot of what the dataframe looks like is given Figure 3.

```
df.head()
```

	title	abstract
0	Clinical features of culture-proven Mycoplasma...	OBJECTIVE: This retrospective chart review des...
1	Nitric oxide: a pro-inflammatory mediator in l...	Inflammatory diseases of the respiratory tract...
2	Surfactant protein-D and pulmonary host defense	Surfactant protein-D (SP-D) participates in th...
3	Role of endothelin-1 in lung disease	Endothelin-1 (ET-1) is a 21 amino acid peptide...
4	Gene expression in epithelial cells in respons...	Respiratory syncytial virus (RSV) and pneumoni...

```
df.shape
```

```
(482221, 2)
```

Figure 3: Metadata file shown as Dataframe.

Calling pandas "Dataframe.shape" function revealed that there were 482221 initial data rows.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 144695 entries, 0 to 144694
Data columns (total 4 columns):
 #   Column     Non-Null Count   Dtype
---  ------     --------------   -----
 0   paper_id   144695 non-null  object
 1   title      144695 non-null  object
 2   abstract   144695 non-null  object
 3   body_text  144695 non-null  object
```

Figure 4: Dataframe's information.

4.3.2 Removing Duplicate Data

It was crucial to remove duplicate data before we could feed the data to a word embedding model to prevent bias by the embedding model. We looked for entries consisting of duplicate abstracts or titles or body text, and eliminated any entries which appeared more than once. This required lowercasing the title and abstract column and removing any punctuation, since some journals were almost identical except that they differed by irrelevant uppercasing and punctuations. The processed titles and abstracts are stored as new columns as seen in the figure below. After removing duplicate data, number of dataframe entries were reduced to 246797 entries.

4.4 Word2Vec

Word2vec is a natural language processing technique. The word2vec algorithm learns word associations from a wide corpus of text using a neural network model. Once learned, a model like this can detect synonyms and recommend additional terms for a sentence. As the name suggests, word2vec associates each distinct word with a specific set of numbers known as a vector. The vectors are carefully selected such that a basic mathematical function (cosine similarity between the vectors) can be used to determine the degree of semantic similarity between the terms represented by those vectors.

Word2vec's aim and utility is to group vectors of related words together in vectorspace. That is, it uses mathematics to detect similarities. Word2vec generates vectors, which are distributed numerical representations of word features including meaning of individual terms. It accomplishes this without the need for human interference.

Word2vec can make highly accurate guesses about a word's meaning based on previous appearances if given enough data, usage, and contexts. These guesses may be used to determine a word's connection with other words (for example, "man" is to "boy" what "woman" is to "girl"), or to group and categorize documents by subject.In fields as diverse as science research, legal exploration, e-commerce, and customer relationship management, these clusters can be used to power search, sentiment analysis, and recommendations.

Related terms will be clustered together in that space by a well-trained collection of word vectors. Battle, dispute, and strife may huddle together in one corner, while oak, elm, and birch may huddle together in another.
Other associations can be found by querying a Word2vec model. There isn't always a need for two analogies that are mirror images of each other.

- Geopolitics: Iraq - Violence = Jordan
- Distinction: Human - Animal = Ethics

- President - Power = Prime Minister
- Library - Books = Hall
- Analogy: Stock Market Thermometer

We can progress beyond hard tokens to a simpler and more smoother sense of meaning by developing a sense of one word's proximity to other related words that do not actually include the same letters.
Traditional Knowledge Graph requires manual data modelling, specifying relations and extracting named entities, which is very time-consuming. Word2vec, on the other hand, learns these underlying relations between words without being exposed to any sort of knowledge graph or specified relations. We chose to use neural word embeddings, especially word2vec, for this very reason.

4.4.1 Gensim Word2Vec Library

We were using python as our programming language, and therefore looked for libraries with python implementation of word2vec.
Gensim Library is one of such libraries, and a very popular one. We went for Gensim because the library had a lot of resources, functions and implementation examples. Noteable features include:

- Function to find out most similar words relative to one oor more words.
- Distance between two words on a scale of 0 to 1, 1 being the closest.
- Vector Arithmetic operations between words.
- Options to save and load models.

4.4.2 Cleaning Text and Input

Before preparing data for input into our word embedding model, we need to make sure our corpus do not have any irrelevant information that may deviate from the actual information. Embedding models usually tend to find underlying relations between words based on how often and how far apart two words are in a context. Symbols, punctuation marks and numbers are not interpretable or useful to an embedding model because these do not have a contextual meaning in a text. We removed symbols and numbers by specifying a tokenizer with regular expressions to ignore. Gensim Word2Vec model requires an array of sentences, where each sentence is itself in a tokenized form. The following is the code we used to clean and prepare text for input.

```
tokenizer = nltk. RegexpTokenizer(r"\w+")
words = []
for sent in sents:
    sent = ''.join(word.lower() for word in sent if not word.isdigit())
    sent = tokenizer.tokenize(sent)
    words.append(sent)
```

Figure 5: Code for cleaning text

```
[['a', 'case', 'of', 'the', 'absorption', 'of', 'corona', 'virus', 'disease', 'covid', 'promoted', 'by', 'professor', 'xu', 'zo
u', 's', 'acupuncture', 'technique', 'for', 'benefiting', 'kidney', 'and', 'strengthening', 'anti', 'pathogenic', 'qi', 'is',
'introduced'], ['a', 'female', 'patient', 'suffered', 'from', 'covid', 'years', 'old', 'had', 'been', 'treated', 'with', 'acupu
ncture', 'and', 'chinese', 'herb', 'granules', 'for', 'days', 'on', 'the', 'base', 'of', 'the', 'oral', 'administration', 'of',
'moxifloxacin'], ['in', 'the', 're', 'examination', 'the', 'chest', 'ct', 'image', 'indicated', 'that', 'the', 'absorption', 'o
f', 'covid', 'was', 'obvious', 'as', 'compared', 'with', 'before', 'the', 'nucleic', 'acid', 'test', 'of', 'novel', 'corona',
'virus', 'was', 'negative', 'and', 'the', 'patient', 'narrated', 'no', 'obvious', 'discomfort'], ['acupuncture', 'therapy', 'pl
ays', 'its', 'active', 'adjuvant', 'effect', 'in', 'the', 'whole', 'process', 'of', 'the', 'treatment', 'of', 'covid'], ['the',
'covid', 'pandemic', 'forced', 'health', 'authorities', 'around', 'the', 'world', 'to', 'introduce', 'public', 'health', 'measu
res', 'to', 'contain', 'the', 'risks', 'of', 'contagion']]
```

Figure 6: Preview of part of the input after cleaning text

4.5 Training Word2Vec models

Word2Vec models can be trained as a skip-gram or CBOW(Bag of Words) model, where skip-gram tries to predict context words related to the input word, while CBOW tries to predict a target word given a context of words. The number of context words or the size of the context words depend on the window size specified. A window size of 5 means the model looks at 2 words before and after the target word as context.
We trained our model as a skip-gram, because given a word we wanted to figure out every other words in that context. To check which setting of model would be able to capture the essence of COVID, we trained 3 models, with varying window sizes of 5(default), 7 and 10. We chose a vector size of 300, which meant that each word in the trained word2vec vocabulary will e represented by 300 features. The model configurations are provided in Figure 7, 8 and 9.

The training phase involves initializing a word2vec model, building vocabulary from the input words, and finally training the model to create the word embeddings. For each model, we ran the training 10 times to ensure the model learns precisely.

```
w2v_model4 = Word2Vec(
                     vector_size=300,
                     workers=cores-1)
```

Figure 7: Default model

```
w2v_modelw7 = Word2Vec(
                      window=7,
                      vector_size=300,
                      workers=cores)
```

Figure 8: model with window 7

4.6 Visualization

The word embeddings that the word2vec models created were of 300 dimensions. This means that each word was a vector of 300 columns. Hence visualization in 2/3-space was not possible.

4.6.1 Principal Component Analysis

Principal Component Analysis, or PCA, is a dimensionality-reduction technique for reducing the dimensionality of large data sets by translating a large set of variables into a smaller one that retains the majority of the information in the large set.
Naturally, reducing the number of variables in a data set reduces accuracy; however, the trick to dimensionality reduction is to trade some accuracy for simplicity. Since smaller data sets are simpler to explore and imagine, and because machine learning algorithms can analyze data more easily and quickly without having to deal with extraneous variables.
We used PCA to reduce the dimensionality of word vectors from 300 to 2. This would allow us to visualize the vectors in two-space.

```
w2v_modelw10 = Word2Vec(
                        window=10,
                        vector_size=300,
                        workers=cores)
```

Figure 9: model with window 10

4.6.2 MATPLOTLIB PyPlot

matplotlib.pyplot is a collection of functions that make *matplotlib* work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc. We used Pyplot Library of MatPlotLib to visualize the word vectors in two space.

4.6.3 Approach

We defined a function*pcascatterplot(model, [list of words], topN*, where *model* specifies the model we want to use, *list of words* are the word(s) whose similar words we want to analyze, and topN specifies the top *n* results we want to see in the graph. Depending on the number of words in *list of words*, the function plots words similar to the word or common to multiple words. Figure 10 and 11 shows function outputs when one word and two words are provided as parameters for *list of words* respectively.

5 Results and Discussion

With COVID-19 dataset, a massive text data is available. Lots of information which it contains that has no link with our work. After visualising the dataset, we have decided to remove unnecessary rows and columns for producing smaller data that is ready to feed our model. Basically, we tried to recreate the model from [1]. First, we tried to figure out the frequency of the words that contain by abstracts. But we find an unpleasant result shown in figure below.

5.1 Comparing the models

We began evaluating our work by comparing the trained models. As mentioned before, we trained three models based on window size. Since we did not remove stopwords from

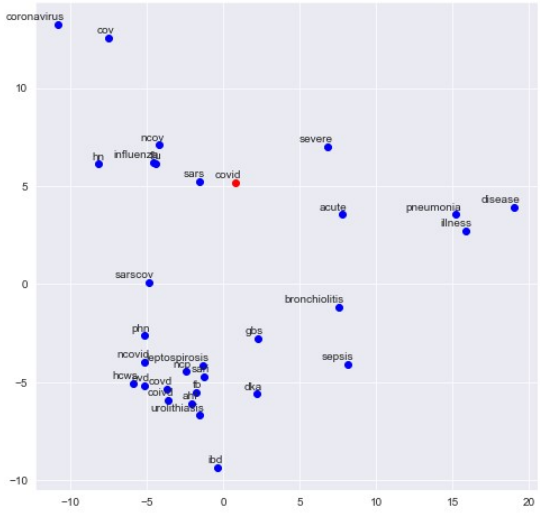

Figure 10: Closest 30 words to the word *covid*

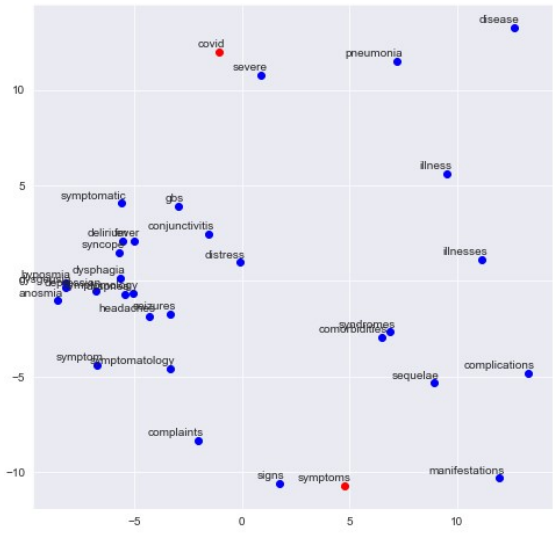

Figure 11: Closest 30 words related to two words, *covid* and *symptoms*

our text in concern that the model might not be able to capture sentence semantics with stopwords removed, we tried to check if varying window size has any effect on the embedding quality.

Since our models were domain-specific, there was no definite or defined way of evaluating the models. We had to exploit general knowledge about COVID in order to check if the models were able to capture those information. The hypothesis was that if the models were able to capture general knowledge about COVID properly, the models would also be able to capture other information which are not popular. The procedure for comparing models are also related to this hypothesis, i,e, the model which is more able to capture general semantics would also capture unpopular semantics somewhat accurately.

5.1.1 Using COVID general knowledge to evaluate word2vec models

We tested the models using a combination of the visualization function and raw text output. As an initial test, we checked to see the 20 closest words to the word 'covid' for each model and observed which model outputs the most relevant information.

In all the three models, the top five similar words are within the words ¡covd, sars, sarscov, coivd, ncov, ncp¿, which are either misspellings or alternate name for covid. Even though the words are different, they are used in the same context and therefore end up in top 5. Another observation is that as we increase window size, similarity scores for each word decrease, though this needs to be further verified. One possible reason for this could be

```
('covd', 0.5695934295654297)
('ncov', 0.4839536249637604)
('sars', 0.473135381937027)
('sarscov', 0.4658760130405426)
('coivd', 0.46002209186553955)
('ncp', 0.4552251100540161)
('disease', 0.4344501793384552)
('influenza', 0.41687262058258057)
('evd', 0.40332624316215515)
('cov', 0.40153035521507263)
('illness', 0.38518670201301575)
('phn', 0.3839412331581116)
('sepsis', 0.3801100254058838)
('leptospirosis', 0.37999865412712097)
('severe', 0.37891244888305664)
('hcws', 0.37355807423591614)
('flu', 0.36951884627342224)
('hn', 0.3691574037075043)
('acute', 0.3674499392509 4604)
('gbs', 0.3663898706436157)
```

Figure 12: window 5 model "covid" similarity list

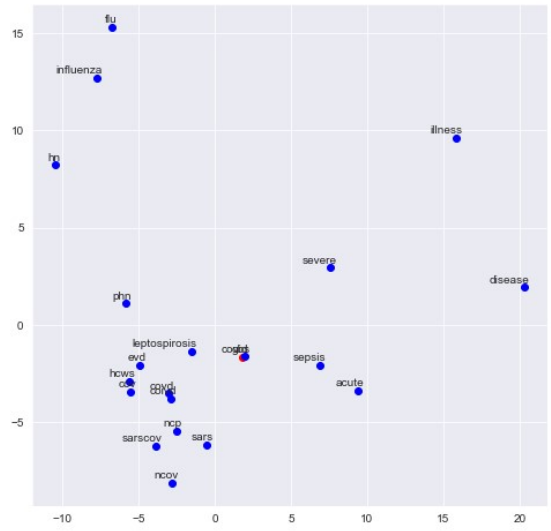

Figure 13: window 5 model "covid" similarity graph

that as window size increases, the model considers more words in context during training and therefore the similarity scores get distributed more into the new words in the larger window. Looking at the PCA reduced visualization, we can see clusters of words in case of all three models, where most of the clusters are similar names for covid. One interesting observation is that the word "evali" appears in the similar words list for window 7 and 10 model. Evali is reffered to as lung damage caused by e-cigarettes and vapes, and most of its symptoms are similar to that of covid, which is why it was difficult to diagnose E-vali since 2018. Reason why this information is interesting is because we can be sure our models are detecting context successfully. We now need to figure out a way to decide which model performs better. Even though "evali" is not present in the similarity scores by wondow 5 model, running similarity scores between window 5 word vectors and "evali" shows a 0.3284361 score, which is greater than the score by window 10 model. The reason why the word does not appear in the top 20 list of window 5 is because window 5 model lists more words within the fixed range of similarity scores than the other models. This is evident from the similarity score range within the top 20 words. The score range is between 0.569 and 0.366 for the window 5 model, 0.561 and 0.33 for window 7 model and 0.503 to 0.309 for window 10 model.

Close words for "disease" also show alternative words in the top 5, and all the other close

```
('covd', 0.5605820417404175)
('sarscov', 0.4647465944290161)
('coivd', 0.4622439444065094)
('ncov', 0.4497958719730377)
('ncp', 0.42070987820625305)
('disease', 0.4150412976741791)
('sars', 0.4106471836566925)
('obligación', 0.3636031150817871)
('illness', 0.36320793628692627)
('influenza', 0.3608083128929138)
('evali', 0.3581930100917816)
('ncovid', 0.35124003887176514)
('hcws', 0.34763428568840027)
('evd', 0.3431027829647064)
('phn', 0.34263870120048523)
('cov', 0.33659791946411133)
('ibd', 0.3344937264919281)
('bronchiolitis', 0.3314330577850342)
('psychiatric', 0.3309542238712311)
('severe', 0.3301689028739929)
```

Figure 14: window 7 model "covid" similarity list

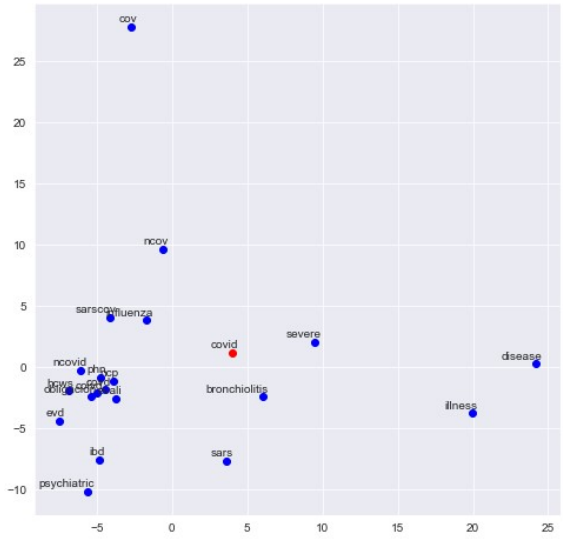

Figure 15: window 7 model "covid" similarity graph

words seem well within the context of the word. Window size did not seem to affect the scores much in the sense that all the listed words seems somewhat relevant to the target word. However, it was evident that smaller window size was creating close clusters for similar words, atleast when we were generating the first 20-100 close words.

5.1.2 Word2Vec Model Verdict

We chose to go with window 7 model because of its balance between clustering and showing relevant information. Even though the results were somewhat similar, we chose the model because the distribution of scores around similar words seemed more uniform and therefore it would be easier to visualize the results.

5.2 Interesting Findings

The next step was to find some interesting information related to COVID. We were mainly looking for morbidity causes and potential cures of the disease. We intended to achieve this either by generating similar words from a target word and generating similar words again, if need be, from the generated list until we find some concrete information, or using vector arithmetics to gain insights.

```
('covd', 0.5030574202537537)
('sarscov', 0.43567827343940735)
('ncov', 0.4236910343170166)
('ncp', 0.37444615364074707)
('coivd', 0.3698664903640747)
('disease', 0.35761481523513794)
('sars', 0.35688942670822144)
('ncovid', 0.3409271240234375)
('accompagnant', 0.3389323055744171)
('hcws', 0.3282232880592346)
('evd', 0.3282163143157959)
('phn', 0.32605352997779846)
('psychiatric', 0.32394224405288696)
('evali', 0.3198174238204956)
('influenza', 0.31792494654655457)
('numb', 0.3178403079509735)
('tb', 0.3137878477573395)
('illness', 0.31169670820236206)
('panic', 0.31002935767173767)
('situation', 0.30923375487327576)
```

Figure 16: window 10 model "covid" similarity list

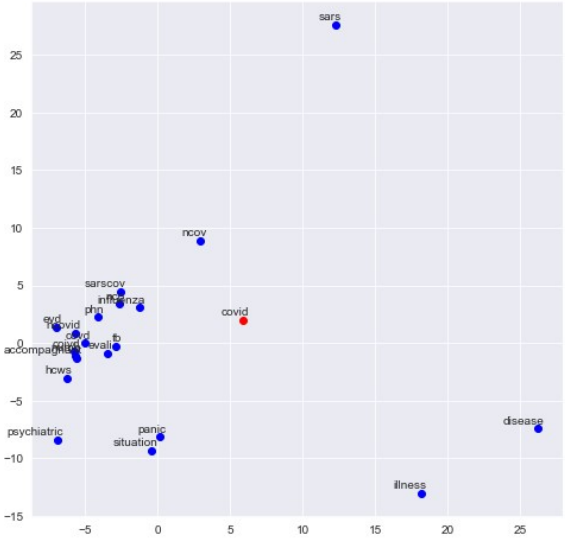

Figure 17: window 10 model "covid" similarity graph

5.2.1 Potential Cures

We found some interesting finds regarding cures or measures to suppress COVID symptoms. Word2vec allows vector arithmetics which in turn generates ontological results. For example, "King - Man + Woman" would most likely generate "Queen". This can be achieved by generating close words to the resulting vector of the arithmetic. We carried out a simple "Covid + Cure" arithmetic and generated top 100 close words to the resulting vector. We generated 100 to ensure we have a large option to consider. The results are given below. We shortlisted **dexamethasone**, **hydroxychloroquine** and **tocilizumab**.

Dexamethasone is a corticosteroid with anti-inflammatory and immunosuppressive properties that is used to treat a variety of conditions. In the UK's national clinical trial RECOVERY, it was studied in hospitalized patients with COVID-19 and found to have advantages for critically ill patients.

In COVID-19, drugs used to treat other diseases were tested, including **chloroquine**, which is used to treat malaria, and **hydroxychloroquine**, which is used to treat rheumatic diseases including rheumatoid arthritis and systemic lupus erythematosus. Evidence of

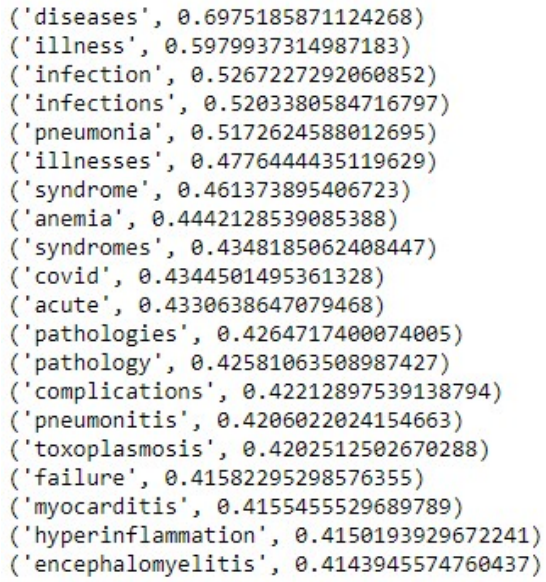

Figure 18: window 5 model "disease" similarity list

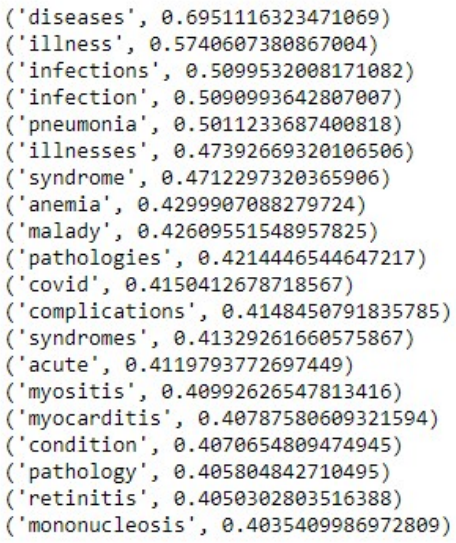

Figure 19: window 7 model "disease" similarity list

the effects of these drugs in treating people who were sick with the disease, preventing the disease in people who were at risk of having it, such as health workers, and preventing the disease in people who were exposed to the virus developing the disease were sought.

Tocilizumab, in addition to quality of treatment (i.e. steroids), is recommended by the Infectious Disease Society of America for hospitalized adults with COVID-19.

5.2.2 Morbidity or Increased Risk Factors

We tried to generate words based on "Lifestyle and COVID", "COVID and morbidity" and also took a look at the top 50 generated closed words for the word "covid" to find some relations.

We found some interesting information about morbidity risk factors. "Covid + Lifestyle" revealed words related to "obesity", "elderly" and so did the arithmetic "COVID + Morbidity". This is an indicator of the fact that COVID tends to affect more in elderly patients and even more so in those who are obese. Top similar words of "COVID" also lists "pregnant", which further strengthens recent findings that COVID affects pregnant women more than non-pregnant women.

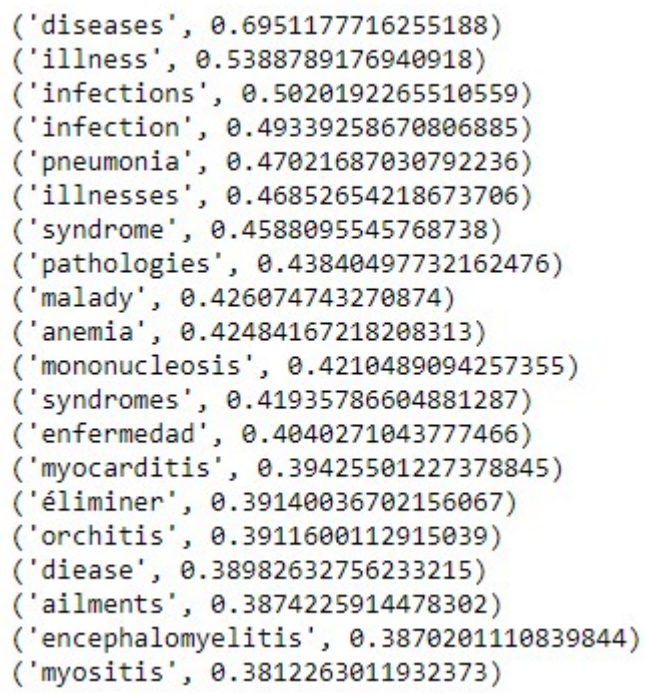

Figure 20: window 7 model "disease" similarity list

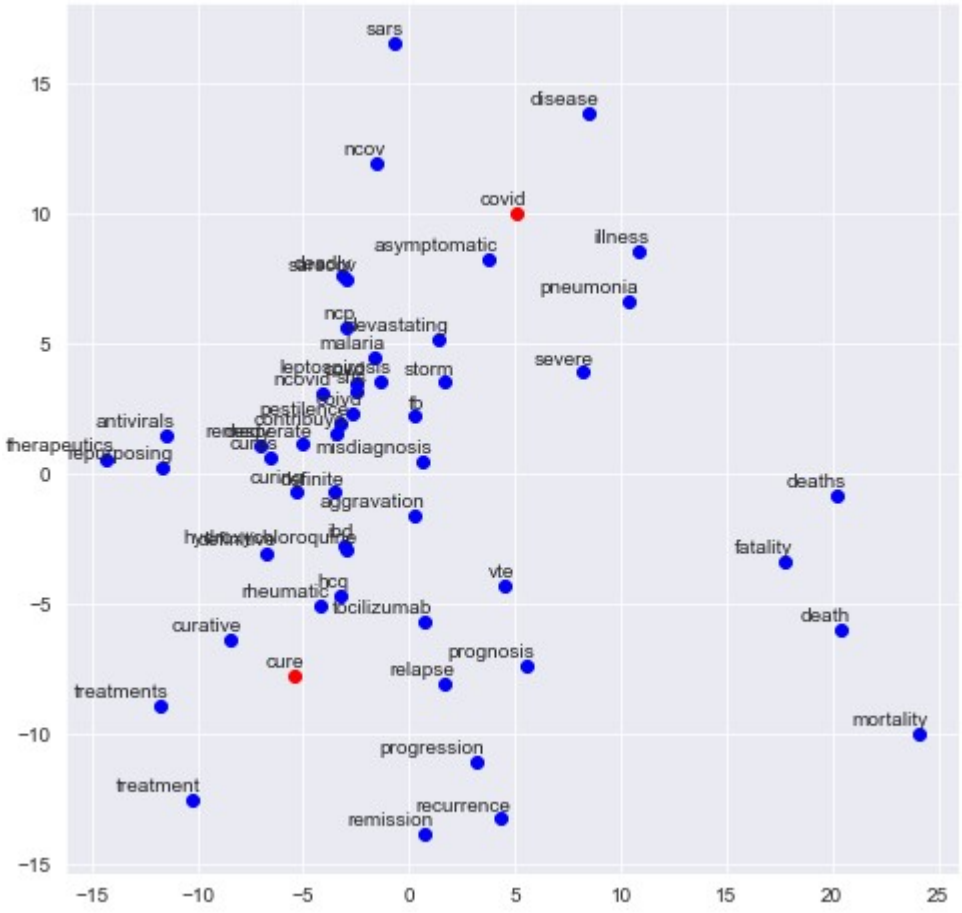

Figure 21: "covid"-"cure" similarity graph

```
('coivd', 0.41446471214294434)
('obesity', 0.4077340364456177)
('covd', 0.4042387306690216)
('burdens', 0.38978779315948486)
('deaths', 0.38973361253738403)
('risks', 0.38780757784843445)
('sequelae', 0.3843066394329071)
('illnesses', 0.38203418254852295)
('risk', 0.37814784049987793)
('worldwide', 0.37533581256866455)
('morbi', 0.3732417821884155)
('comorbidities', 0.37217193841934204)
('elderly', 0.37130385637283325)
('sepsis', 0.36874786019325256)
('infections', 0.36516696214675903)
('disability', 0.3651624023914337)
('hospitalizations', 0.3649965822696686)
('coagulopathy', 0.36470600962638855)
```

Figure 22: "covid" + "morbidity" similarity list

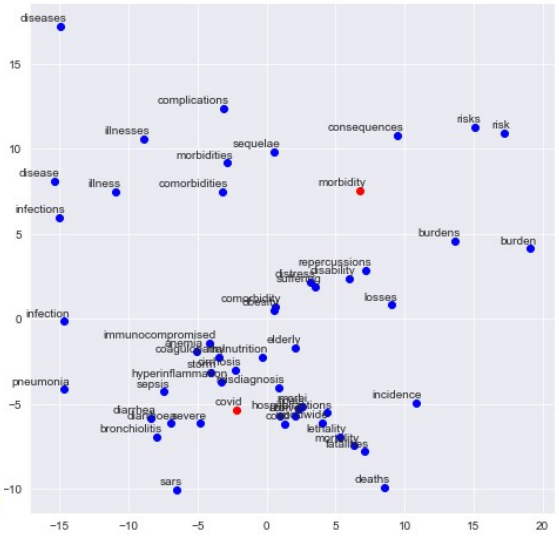

Figure 23: "covid" + "morbidity" similarity graph

5.3 Discussion

So far, we have been able to identify potential cures and few morbidity risk factors of COVID. These information agrees strongly with mostly recent published findings. However, there are limitations and inefficiency in the proposed approach. The data we collected were from the abstract portions of journals, and even though abstract section provides a summarized overview of the paper, importing body text could further strenthen the word2vec model. The models were made with a vocabulary size of around 100000 unique words. Increasing the vocabulary to a million could provide us with a more robust model. There were certain issues with the word2vec models. Altering window size did not help much in choosing a word2vec model, even though the window sizes chosen were not too small or too large to understand the difference. It was difficult to search for relevant words manually from a list of 100000 words.

6 Conclusion and Recommendation

So far, we have been able to identify potential cures and few morbidity risk factors of COVID. The results obtained were strong in the sense that they were backed up by recent findings. Word2vec does help in capturing underlying information, which would be exhaustive to search for by reading journals. The embedding system also has the potential

```
('lifestyles', 0.5391206741333008)
('psychological', 0.44340986013412476)
('habits', 0.43662720918655396)
('obesity', 0.4269326627254486)
('eating', 0.4132443070411682)
('unhealthy', 0.405137836933136)
('psychosocial', 0.3999299705028534)
('mental', 0.3990371823310852)
('behaviours', 0.3967192471027374)
('anxiety', 0.3919355571269989)
('covd', 0.38245323300361633)
('sarcopenia', 0.3742658197879791)
('sleep', 0.3734264373779297)
('depression', 0.37145692110061646)
('people', 0.36891648173332214)
('psychiatric', 0.364752858877182)
('malnutrition', 0.3591114580631256)
('dietary', 0.3591068983078003)
('smoking', 0.3574753999710083)
('nutritional', 0.35720208287239075)
('behaviors', 0.35643428564071655)
```

Figure 24: "covid" + "lifestyle" similarity list

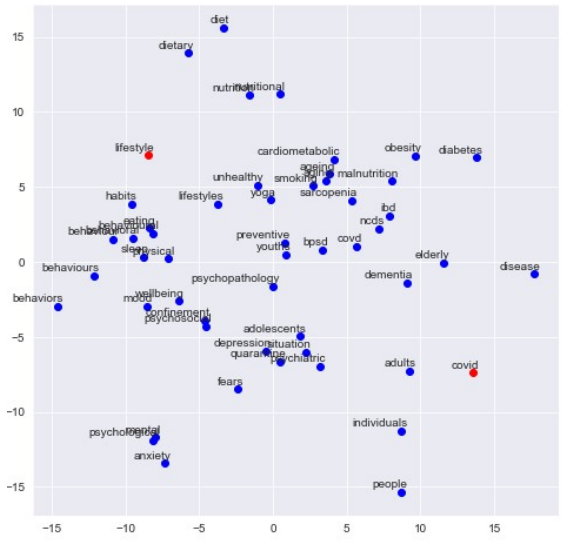

Figure 25: "covid" + "lifestyle" similarity graph

to detect patterns that cannot be found from one journal. But, at the end of the day, these findings does not conclude facts, but rather points to potential areas of research. The word embeddings can also be used to create automated knowledge retrieval system which automatically specifies relations between recognized entities. Also, this model once trained on a much larger dataset can be used to create medical diagnostics system.

7 Acknowledgment

References

[1] S. Ahamed and M. Samad, "Information mining for covid-19 research from a large volume of scientific literature," *arXiv preprint arXiv:2004.02085*, 2020.

[2] Á. Hernández-Castañeda, R. A. García-Hernández, Y. Ledeneva, and C. E. Millán-Hernández, "Extractive automatic text summarization based on lexical-semantic keywords," *IEEE Access*, vol. 8, pp. 49 896–49 907, 2020.

[3] A. Díaz-Manríquez, A. B. Ríos-Alvarado, J. H. Barrón-Zambrano, T. Y. Guerrero-Melendez, and J. C. Elizondo-Leal, "An automatic document classifier system based on genetic algorithm and taxonomy," *IEEE Access*, vol. 6, pp. 21 552–21 559, 2018.

[4] H. Jelodar, Y. Wang, C. Yuan, and X. Feng, "Latent dirichlet allocation (lda)

```
('diabetes', 0.6303427815437317)
('hypertension', 0.5921398997306824)
('dyslipidemia', 0.5861342549324036)
('adiposity', 0.5817481279373169)
('overweight', 0.5692273378372192)
('undernutrition', 0.5556676387786865)
('comorbidities', 0.5402392148971558)
('obese', 0.5361801385879517)
('cvd', 0.5194556713104248)
('malnutrition', 0.5188653469085693)
('hyperglycaemia', 0.5179166793823242)
('bmi', 0.5158200860023499)
('cardiometabolic', 0.5113181471824646)
('smoking', 0.5094902515411377)
```

Figure 26: "obese" similarity list

and topic modeling: Models, applications, a survey. arxiv, 2017," *arXiv preprint arXiv:1711.04305*.

[5] S. K. Biswas, M. Bordoloi, and J. Shreya, "A graph based keyword extraction model using collective node weight," *Expert Systems with Applications*, vol. 97, pp. 51–59, 2018.

[6] D. Rusu, B. Fortuna, D. Mladenic, M. Grobelnik, and R. Sipoš, "Document visualization based on semantic graphs," in *2009 13th International Conference Information Visualisation*. IEEE, 2009, pp. 292–297.

[7] L. C. Freeman, "A set of measures of centrality based on betweenness," *Sociometry*, pp. 35–41, 1977.

[8] A. Kao and S. R. Poteet, *Natural language processing and text mining*. Springer Science & Business Media, 2007.

[9] F. Shi and O. Institute, *Learn about Term Frequency–inverse Document Frequency in Text Analysis in Python with Data from how ISIS Uses Twitter Dataset (2016)*. SAGE Publications, Limited, 2019. [Online]. Available: https://books.google.com.bd/books?id=Rc7wxQEACAAJ

[10] C. Beecks, F. Borutta, P. Kröger, and T. Seidl, *Similarity Search and Applications: 10th International Conference, SISAP 2017, Munich, Germany, October 4-6, 2017, Proceedings*, ser. Lecture Notes in Computer Science. Springer International Publishing, 2017. [Online]. Available: https://books.google.com.bd/books?id=eZs3DwAAQBAJ

[11] I. Jolliffe, *Principal Component Analysis*, ser. Springer Series in Statistics. Springer New York, 2013. [Online]. Available: https://books.google.com.bd/books?id=-ongBwAAQBAJ