Comparative link prediction of disease comorbidity regarding COVID19 using Knowledge Graph

Thesis to be submitted in partial fulfillment of the requirements for the degree

of

Master of Technology

by

Aditya Anand
20CS60R24

Under the guidance of

Prof. K. Sreenivasa Rao & Sir Saikat Biswas

COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

DECLARATION

I certify that,

(a) The work contained in this report has been done by me under the guidance of my supervisor.

(b) The work has not been submitted to any other Institute for any degree or diploma.

(c) I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

(d) Whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references. Further, I have taken permission from the copyright owners of the sources, whenever necessary.

Aditya Anand
(20CS60R24)

Place: Kharagpur
Date: 10/11/2021

Department of Computer Science and
Engineering
Indian Institute of Technology,
Kharagpur
India - 721302

CERTIFICATE

This is to certify that we have examined the thesis entitled **Comparative link prediction of disease comorbidity regarding COVID19 using Knowledge Graph**, submitted by **Aditya Anand**(Roll Number: *20CS60R24*) a postgraduate student of **Department of Computer Science and Engineering** in partial fulfillment for the award of degree of Master of Technology. We hereby accord our approval of it as a study carried out and presented in a manner required for its acceptance in partial fulfillment for the Post Graduate Degree for which it has been submitted. The thesis has fulfilled all the requirements as per the regulations of the Institute and has reached the standard needed for submission.

Supervisor

Department of Computer Science and Engineering
Indian Institute of Technology,
Kharagpur

Place: Kharagpur
Date:

ACKNOWLEDGEMENTS

The successful completion of any task can't be complete without acknowledging the people who made it all possible and whose constant guidance and encouragement leads to success. I am grateful to **Prof. K. Sreenivasa Rao** and **Sir Saikat Biswas**, Computer Science and Engineering Department, IIT Kharagpur for the support. I owe a debt of gratitude to them for incorporating in me the idea for this research project, helping me undertake it and also for being there whenever I needed their assistance. I also place on record, our sense of gratitude to my fellow batch-mates and all, who directly or indirectly have lent a helping hand in this venture. I feel proud and privileged in expressing my deep sense of gratitude to all those who have helped me in this project.

Aditya Anand
IIT Kharagpur
Date:

ABSTRACT

In this work, we used the concept of Knowledge Graphs (KG) for exploiting relations between genes along with their ontologies, diseases with SNP variants, symptoms and Human Phenotype Ontologies. Also, Construction and completion of KGs, these days, are rapidly advancing and hence a proper COVID related KG is necessary and sufficient for predicting not only comorbidity in a better way, but much more. KG triplets were extracted from the KG, embedded and fed to an LSTM model. Here our goal is to train few models using different embedding techniques on the obtained KG triplets, such that one of the models that outperform the others can be saved and used as an instance of our built KG. The saved instance of KG in the form of a well-trained model can then be used to better predict the chances of disease comorbidity regarding COVID 19.

Table of Contents

1. Introduction

Coronavirus (COVID-19) caused by sever acute respiratory syndrome 2 (SARS-CoV-2) is a global epidemic that began in Wuhan, China, and spread rapidly, reaching more than 180 countries. As the novel coronavirus continues to emerge, there are still many limitations to our knowledge of what the virus really is that could have a major impact. Older adults and people of any age with lower medical conditions, such as high blood pressure and diabetes, showed worse prognosis [R6].

"Comparative link prediction of disease comorbidity regarding COVID19 using Knowledge Graph" can be seen as an attempt to better predict comorbidity being motivated from the fact that a patient suffering from co-morbid diseases possesses more mortality risk than with a disease alone [R1]. Comorbid disease condition refers to the simultaneous presence of one or more disease(s) along with a primary disease. Comorbidity is associated with worse health outcomes, more complex clinical management, and increased health care costs. Hence, better prediction of comorbid diseases regarding COVID19 is important considering the current pandemic situation.

In this work, we used the concept of Knowledge Graphs (KG) for exploiting relations between genes along with their ontologies, diseases with SNP variants, symptoms and Human Phenotype Ontologies. Also, Construction and completion of KGs, these days, are rapidly advancing and hence a proper COVID related KG is necessary and sufficient for predicting not only comorbidity in a better way, but also answering queries like which gene/symptom/SNP-variant/HPO lead to comorbidity, which gene leads to which type of symptoms/phonotypes/diseases, etc..

A KG is a multi-relational graph, **KG (E, R)**, with nodes as entities and edges between the nodes as relation between two entities, where **E** refers the set of entities/nodes and **R** refers to the set of different types of relations/edges. Every edge in the **KG** is represented as a triplet, composed of head entity, relation and tail entity. Web-based KGs provide a structured representation regarding world knowledge, such as Google Knowledge Vault. Many real-world applications namely semantic parsing, named entity disambiguation and information extraction tasks have been performed from a large number of popular KGs such as Freebase, Wordnet, DBpedia, YAGO and NELL. As these KGs are very logical and robust structures to make data machine understandable and process able, they seemed to be quite successful in biology and biomedicine as well.

In this work, we are trying to develop a standard KG that can better predict the chances of disease comorbidity. Here we constructed 6 subgraphs, namely, Gene Interaction Subgraph (including COVID GENE), Gene Ontologies Subgraph, Disease Association Subgraph, Overlapping Variants Subgraph, Overlapping Symptoms Subgraphs and Overlapping Human Phenotype Ontology Subgraph. These 6 subgraphs were merged into a single graph [4.2] that is referred here to as our built KG. Now, KG triplets were extracted from the KG [4.3]. These triplets were then embedded using word2vec embedding technique and fed to an LSTM model [4.4].

Here our goal is to train few models using different embedding techniques on the obtained KG triplets, such that one of the models that outperform the others can be saved and used as an instance of our built KG. The saved instance of KG in the form of a well-trained model can then be used to better predict the chances of disease comorbidity regarding COVID 19.

2. Literature Review

2.1. Relation Prediction of Co-morbid Diseases Using Knowledge Graph Completion. – IEEE 2019

Author(s): Saikat Biswas, Pabitra Mitra, and Krothapalli Sreenivasa Rao.

About: Here, a tensor factorization based approach was proposed using complex-valued embedding on biological knowledge graphs. Here, a knowledge graph was shown with disease-gene associations and their corresponding background information to predict the association between prevalent diseases, by exploiting Complex embedding based tensor decomposition method. Besides, they show new prevalent disease pairs using the MCL algorithm in a disease-gene-gene network and their corresponding inter-relations using edge predictions.

2.2. Analysis of SARS-CoV-2 mutations in the United States suggests presence of four substrains and novel variants – NATURE 2021

Author(s): Rui Wang, Jiahui Chen, Kaifu Gao, Yuta Hozumi, Changchuan Yin & Guo-Wei Wei.

About: This paper presents the analysis of approximately 50, 000 SARS-COV-2 Genome Sequence to understand their mutations. They observed that some mutations strongly prevailed while other faded out. They also showed that female immune systems are more active than those of males in responding to SARS-CoV-2 infections due to unusual strong gender dependence.

2.3. Knowledge Graph Embedding by Translating on Hyperplanes – AAAI Conference 2014

Author(s): Zhen Wang, Jianwen Zhang, Jianlin Feng, Zheng Chen.

About: Here, they deal with embedding a large scale knowledge graph composed of entities and relations into a continuous vector space. TransE is a promising method proposed recently, which is very efficient while achieving state-of-the-art predictive performance. Here they also discuss other relation mapping features that should be considered in embedding, such as reflexive, one-to-many, one-to-one, and one-to-many. Note that TransE does not perform well in dealing with these structures. Some sophisticated models are able to maintain these map properties but sacrifice efficiency in the process. To make a good tradeoff between model capacity and efficiency, in this paper they proposed a "TransH" Method that reflects relationships as a hyperplane and a translation function in it. In this way, the above map properties of the almost complex and complex relationships of the same TransE model can be preserved.

2.4. Knowledge Graph Construction Techniques - Journal of Computer Research and Development 2016

Author(s): Liu Qiao, Li Yang, Duan Hong, Liu Yao, Qin Zhi guang.

About: Google's knowledge graph technology has drawn a lot of research attentions in recent years. However, due to the limited public disclosure of technical details, people find it difficult to understand the connotation and value of this technology. In this paper, the key techniques involved in the construction of knowledge graph in a bottom-up way was introduced, starting from a clearly defined concept and a technical architecture of the knowledge graph. Firstly, the definition and connotation of the knowledge graph were discussed in detail, and the technical framework for knowledge graph construction was proposed.

2.5. COMOR: a software for disease comorbidity risk assessment - Journal of Clinical Bioinformatics 2014

Author(s): Mohammad Ali Moni & Pietro Liò.

About: The diagnosis of comorbidities, which refers to the coexistence of different acute and chronic diseases, is difficult due to the modern extreme specialisation of physicians. In this paper, they presented/ developed an R software 'comoR' to compute novel estimators of the disease comorbidity associations. Starting from an initial diagnosis, genetic and clinical data of a patient the software identifies the risk of disease comorbidity. Then it provides a pipeline with different causal inference packages (e.g. pcalg, qtlnet etc) to predict the causal relationship of diseases. It also provides a pipeline with network regression and survival analysis tools (e.g. Net-Cox, rbsurv etc) to predict more accurate survival probability of patients. The input of this software is the initial diagnosis for a patient and the output provides evidences of disease comorbidity mapping. The functions of the 'comoR' offer flexibility for diagnostic applications to predict disease comorbidities, and can be easily integrated to high–throughput and clinical data analysis pipelines.

2.6. Comorbidity and its Impact on Patients with COVID-19 – NATURE 2020

Author(s): Adekunle Sanyaolu, Chuku Okorie, Aleksandra Marinkovic.

About: This paper examines stressful situations, the progression of the disease, and the mortality rate of patients of all ages, who are infected with the ongoing COVID-19 disease. This literature review was conducted, and active data was then collected from peer-reviewed articles published from January to April 20, 2020. From what is currently known, patients with COVID-19 patients who are ill, such as hypertension or diabetes mellitus, we may have developed a more complex course and progression of the disease. In addition, older patients, especially those 65 years of age and older who have comorbidities and are infected, have an increased intensive care unit (ICU) and COVID-19 mortality. Patients with comorbidities should take all necessary safety precautions to avoid contracting SARS CoV-2, as they often have worse prognosis.

3. Proposed Methodology (For Current Semester Work)

To better predict disease comorbidity regarding COVID-19, we needed the interactions between COVID genes with other genes in general. Higher the interactions between two genes, higher are the chances of their respective diseases (related to the two genes) having something intersecting, hence higher chances of disease comorbidity. Hence, the base of our KG is a Gene-*Gene Interaction Subgraph* that includes COVID genes and *Diseases Association Subgraph* that include SARS-COV-2. To better understand the interactions between genes, their corresponding Ontologies/functions were required. Hence, we added a *Gene-Ontology subgraph*. We also added *symptoms subgraph* as overlapping symptoms may possibly lead to overlap in disease comorbidity. To further enhance/strengthen the KG, we added *Variants subgraph* and *Human Phenotype Ontology Subgraph.* The Variants Subgraph has interconnection of SNP Variants of genes and diseases. SNP Variant refers to the variants that have less 0.01% mutation. Similarly, HPO Subgraph is used to better understand gene-gene interactions. All the above mentioned 6 subgraphs were constructed [4.1] and then Merged [4.2] into a single graph, that we call it as our built KG.

Then, KG triplets were extracted based on some rules [4.3]. These triplets were then embedded using *word2vec* embedding and fed to an LSTM Model for training [4.4].

4. My Contributions (For Current Semester Work)

4.1. Construction of Subgraphs

A. COVID -GENE INTERACTION SUBGRAPH

1. Collected 30 COVID Gene Interaction datasets (.tab3) from BioGrid.
2. All datasets were parsed for Interactors information, and combined into a single dataset [D1] consisting of 'Interactor A' and 'Interactor B' as columns.
3. An all Gene-Gene Interaction dataset was also collected from BioGrid, parsed it for taxonomy = '9096' (Homosapiens only) and appended it to [D1].
4. Now, we have most of the information about Interactions between genes (including COVID gene) in [D1].
5. Applied MCL to get clusters out of [D1]; interactions as edges and interactors as nodes.
6. Now, from approximately 3000 clusters, only two clusters had COVID gene in it (as a node).
    a. Along with 'Interactor A' and 'Interactor B', their corresponding Organism Names were stored; **'severe acute respiratory syndrome coronavirus 2'** as organism name was filtered and corresponding Interactor IDs were obtained.
    b. In [D1], only 11 distinct COVID genes Interactor ID were found.
7. Used NetworkX package to develop graphs out of clusters' edges.
8. Now, to retain most information at the same time to make the graph sparse (reduce complexity), we thought of a spanning tree approach.
    a. Here, we needed to get central nodes to start building a spanning tree.
        1. We applied Page Rank, Betweenness and HITS Centrality, on all the cluster graphs.
        2. Observations were to be made in the clusters containing covid, as our centrality measure should give us covid genes as the most central nodes.
        3. Betweenness centrality outperformed the other two measures, for the two covid-gene containing clusters, hence, applied to all the clusters and central nodes were stored in descending order of their centrality value.
    b. Now, what if the centrality values of two or more nodes are the same (or too close like 0.39999… and 0.39998…).
        1. We applied a concept of binning the nodes in cluster graphs based on the centrality values, with a bin size of 0.1.

           E.g.: Node X - 0.320000, Node Y - 0.31000, Node Z - 0.29000
           Bins: [0.3 to 0.4 - Node X and Node Y,
           0.2 to 0.3 - Node Z]

2. The top most bin (the bin with the highest closing range) contained the most central node as required.
   E.g. Node X and Node Y in the above case.

c. Based on the chosen central nodes, spanning trees were generated for each cluster graph.

9. After Getting all the spanning trees it was now time to join the clusters to form a covid gene interaction subgraph, that will act as the base of the Knowledge Graph. Since, the clusters may be overlapping, the joining of clusters can be divided into two cases:

   a. CASE A: When clusters are overlapping - Simply merge the clusters based on union of edges and union of nodes.

   b. CASE B: When clusters are disjoint - For every cluster spanning tree (G), clone it as G_C

      1. For each pair of nodes (u , v) such that u ∈ A and v ∈ B, for all clusters A and B, check if (u, v) is present in [D1].
      2. If (u, v) ∈ [D1], then add it to GC, else discard it.
      3. Out of the newly added edges, we computed the edge betweenness centrality in GC and most central edges were chosen based on the concept of binning as explained in [8.b.i].
      4. The most central edges were then added to G, and then G_C was discarded/deleted.

10. This is how, all the clusters were joined and a covid-gene interaction subgraph was formed.

B. GENE - ONTOLOGY SUBGRAPH

1. The dataset for Gene Names and their corresponding ontologies (GO) ID was collected from UniProtKb. After preprocessing, D2 has 20386 rows and 2 columns with schema as shown in Table 1[LOT].
2. GOs were stored with their Genes in a dictionary where GOs are keys having a list of genes they correspond to, as values.
   E.g.: From the above table, the dictionary will be generated as shown below:
   Dict = { GO1: [ABC, XYZ],
   GO2: [ABC],
   GO3: [ABC, XYZ],
   GO4: [XYZ] }
3. The above step was done to help us obtain some similarity between all pairs of GOs. So, we applied Jaccard Similarity between GOs based on the gene list in Dict.
   In the above Dict, number of distinct GO = 4.

Thus, total combinations = nC2 = 4C2 = 6, as shown below with their Jaccard Similarity Score.
{ (GO1, GO2): 0.5 (intersection = [ABC], union = [ABC, XYZ]) ,
(GO1, GO3): 1.0,
(GO1, GO4): 0.5,
(GO2, GO3): 0.5,
(GO2, GO4): 0
(GO3, GO4): 0.5 }

4. Now, these would act as edges in our GO Subgraph. The similarity scores could have been the weights; however, our graph is unweighted. The similarity score helps us to discard edges if they have 0.0 similarities (no similarity at all).
5. After discarding edges with zero similarity, there emerge two cases to handle.
   - CASE A: The GO subgraph still remains connected.
     Find the most central node(s) based on betweenness centrality and create spanning trees and later join them as done in section [A.8 and A.9].
   - CASE B: The GO subgraph gets split into multiple components due to edge deletions.
     Choose the cluster with the most number of nodes, and as a cluster is connected, we follow the same approach as CASE A.
6. This is how, GO subgraph was created.
   (Note: We had gene Names here, but in [A] we had geneIDs as nodes. So, for the sake of simplicity, geneID was taken as the unique identifier for Genes. Here in [B], while preparing the dictionary we also mapped and replaced geneNames with geneIDs using the dataset in [C.1].

C. DISEASE SUBGRAPH

1. The datasets for gene - disease interactions were collected from BioGrid. Here, Gene ID was extracted as ‘Interactor A’ and Disease ID as ‘Interactor B’.
   (Note: The dataset also had Gene Names corresponding to Gene IDs and Disease Names corresponding to its Disease ID).
   After Processing:
     - Covid - gene - disease dataset: 70210 rows and 2 columns,
     - All gene disease dataset: 1.1 Million rows approx and 2 columns.

   Both of these datasets were combined [D3] to get 1.2 Million rows x 2 columns with a schema that can be seen in Table 2[LOT].
2. Now, for the above table the same methods were used as in [B.2 to B.5], keeping Disease ID as key and list of Gene IDs as values.
3. This is how the Disease Subgraph was prepared. Here, nodes represent diseases and edges came due to the Jaccard Similarity Score between diseases.

D. VARIANT SUBGRAPH

1. The Variants here refers to different SNP variants of Diseases w.r.t. Genes.
2. The dataset was collected from BioGrid and 3 columns were extracted namely snpID, geneID and DiseaseID [D4] as shown in Table 3[LOT].
3. Operations were performed on the above table as done in [B.2 to B.5] with snpID as keys and list of geneIDs and list of diseaseIDs as values.

   Except for the fact that Jaccard similarity was computed as shown below:

   Let J1 be the jaccard similarity score between snpID pairs based on intersection/union of geneIDs and J2 be the same but based on diseaseIDs in place of geneIDs.

   The final Jaccard Similarity score (J) for any pair of variants will be the average of its corresponding J1 and J2.

   $J(rs0x,\ rs0y)\ =\ (J(rs0x,\ rs0y)\ w.r.t.\ geneIDs\ +\ J(rs0x,\ rs0y)\ w.r.t.\ diseaseIDs\ )\ /\ 2$

   $J(rs0x,\ rs0y)\ w.r.t.\ geneIDs\ =\ geneIDs(rs0x) \cap geneIDs(rx0y)\ /\ geneIDs(rs0x) \cup geneIDs(rx0y)$

   $J(rs0x,\ rs0y)\ w.r.t.\ diseaseIDs$

   $=\ diseaseIDs(rs0x) \cap diseaseIDs(rx0y)\ /\ diseaseIDs(rs0x) \cup diseaseIDs(rx0y).$
4. This Undirected Variant subgraph so prepared will have SNP variants of diseases w.r.t. Genes as nodes and Jaccard similarity scores will help to get the edges.

E. SYMPTOMS SUBGRAPH

1. The dataset was collected from Kaggle Machine Learning Laboratory’s Dataset section. It contained Disease Name with 17 other columns labelled from ‘Symptom_1’ to ‘Symptom_17’.
2. The above dataset [D5] when preprocessed can be seen in Table 4[LOT].
3. This table was used to obtain Jaccard Similarity between Symptoms based on Diseases as done in [B.2 to B.5], keeping symptoms as key and list of corresponding diseases as values.
4. Performing the above led to formation of the undirected Symptoms subgraph.

F. HUMAN PHENOTYPE (HPO) SUBGRAPH

1. The dataset was collected from BioGrid, and preprocessed to get the HPO ID and Gene ID as columns [D6] as its schema can be seen in Table 5[LOT].
2. This table was used to obtain Jaccard Similarity between HPO based on genes as done in [B.2 to B.5], keeping hpoIDs as key and list of corresponding geneIDs as values.
3. Performing the above led to formation of the undirected HPO subgraph.

As all the subgraphs as spanning trees, thus, #edges = #nodes - 1. The number of nodes in each of the subgraphs can be seen in Table 6[LOT].

4.2. Joining The Subgraphs

Let's see how two subgraphs are merged together.

- GENE SUBGRAPH + GO SUBGRAPH
  Both the subgraphs are undirected and highly sparse (close to a tree).
  Let n1 and n2 be the Nodes of CG and GO sub-graphs respectively.

```
for each pair(n1, n2), if (n1, n2) appears as related in [D2],
     then put an edge in CG + GO subgraph to make them
     connected.
```

Similar is the case for **joining all other subgraphs** and they were joined in the following fashion

1. CG + GO + Disease subgraph.
   Here Disease nodes were connected to CG nodes based on edges from [D3].
2. CG + GO + Disease + Variant subgraph
   Here variant nodes were connected with the nodes of CG as well as Disease subgraph based on edges from [D4].
3. CG + GO + Disease + Variant + Symptoms + HPO subgraph.
   Here Symptom subgraph nodes were connected to Disease subgraph nodes based on edges from [D5] and HPO with CG subgraph nodes based on edges from [D6].

This is how the KG was formed, having #nodes = 70392 and #edges = 419848.

4.3. Generating Triplets from KG

In the main knowledge graph we have CG, GO, Disease, Variant, Symptoms and HPO subgraphs integrated together.

The nodes in each of these subgraphs can be distinguished in the following manner:

a. Nodes in CG subgraph - an integer type. [Token = CG]
b. Nodes in GO subgraph - starts with 'GO'. [Token = GO]
c. Nodes in Disease subgraph - starts with 'C'. [Token = DIS]
d. Nodes in Variant subgraphs - starts with 'rs'. [Token = VAR]
e. Nodes in HPO subgraphs - starts with 'HP'. [Token = HPO]
f. Nodes in Symptom subgraphs - are generally string values, i.e., every node except the nodes that follows one of the above 5 conditions [a to e]. [Token = SYM]

For all edges (u, v), such that u, v ∈ [KG], the triplets can be represented as:

$$\begin{aligned}&< head, relation, tail >\\&= \; < u, typeOf(u) \; + \; \text{‘\_’} \; + \; typeOf(v), v > \; and \; < v, typeOf(v) \; + \; \text{‘\_’} \; + \; typeOf(u), u >\end{aligned}$$

The typeOf(x) function gives which subgraph node ‘x’ belongs to, by using the observations made in [a to f].

E.g.: *If u = 2033 and v = C201, the triples will be, <2033, CG_DIS, C201> and <C201, DIS_CG, 2033>, as 2033 is of integer type, hence it is a node of the CG subgraph and as C201 starts with ‘C’, it is a node of the disease subgraph.*

In this way, we generated all the 839,696 triples using the edges in the KG.

4.4. Embedding KG Triplets and Feeding it to LSTM

The KG triplets formed above were taken into a Pandas Data Frame having columns named Head, Relation, Tail and Label as shown in Table 7[LOT]. Here, Head and Tail are nodes of the KG, Relation represents the relation between them and label is always 1 (saying such relation exists between head and tail).

Assuming each row in the DF as a separate sentence, word2vec embedding was applied here. The size of the embedding was 839696 rows and each row having 8 words (padding inclusive) and dimension for representing each word was chosen to be 16. A single embedding can be seen in Figure 1[LOF].

This embedding was then fed to an LSTM Model with 50 hidden neurons with RELU activation function, ADAM optimizer and MSE loss function.
The results can be seen in Figure 2[LOF].

5. Conclusion and Future Work

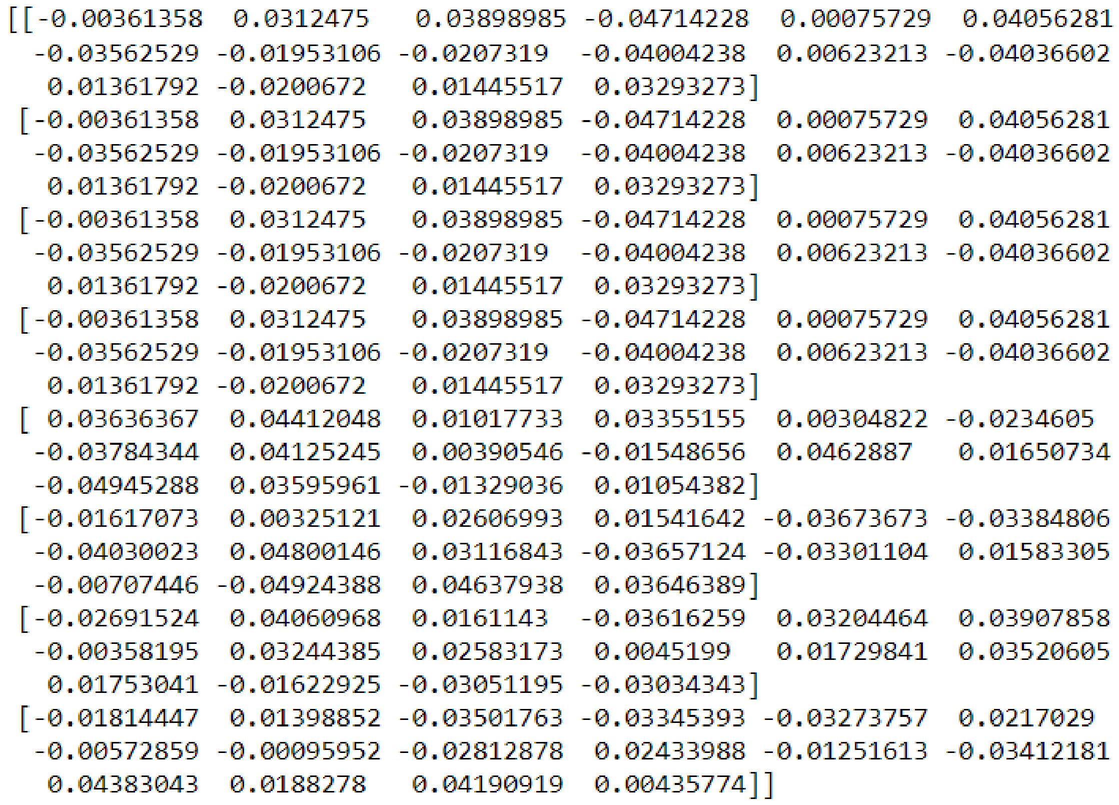
As we took the edges from the KG and built the triplets, then used its embedding to feed to a learning model, we only have positive instances both for training, validation and testing data. Hence our model's learning/testing is biased. By just training on a basic LSTM model with single layer 50 neurons having RELU activation function, ADAM optimizer and Mean Squared Error (MSE) Loss Function, the accuracy from the second epoch reached 100%. Hence the model is over fitted due to bias in input.

So, the first work would be to **pollute the input set with negative instances**. Also, further we will be modifying the KG such that the edges have weights associated to it to help better predict comorbidity.

Also, we will be trying different combinations of different kinds of embedding with various Learning models to make a comparative study.

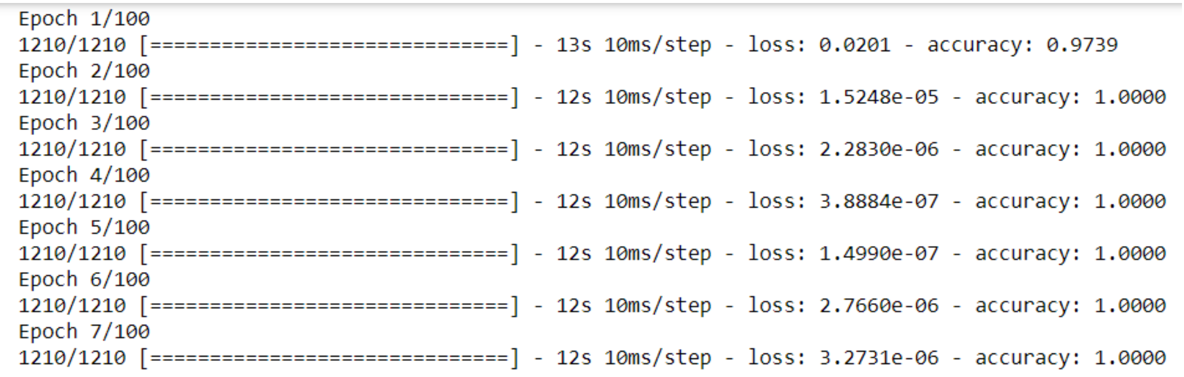
LIST OF FIGURES

Figure 1: An 8 x 16 word2vec embedding.



```
[[-0.00361358  0.0312475   0.03898985 -0.04714228  0.00075729  0.04056281
  -0.03562529 -0.01953106 -0.0207319  -0.04004238  0.00623213 -0.04036602
   0.01361792 -0.0200672   0.01445517  0.03293273]
 [-0.00361358  0.0312475   0.03898985 -0.04714228  0.00075729  0.04056281
  -0.03562529 -0.01953106 -0.0207319  -0.04004238  0.00623213 -0.04036602
   0.01361792 -0.0200672   0.01445517  0.03293273]
 [-0.00361358  0.0312475   0.03898985 -0.04714228  0.00075729  0.04056281
  -0.03562529 -0.01953106 -0.0207319  -0.04004238  0.00623213 -0.04036602
   0.01361792 -0.0200672   0.01445517  0.03293273]
 [-0.00361358  0.0312475   0.03898985 -0.04714228  0.00075729  0.04056281
  -0.03562529 -0.01953106 -0.0207319  -0.04004238  0.00623213 -0.04036602
   0.01361792 -0.0200672   0.01445517  0.03293273]
 [ 0.03636367  0.04412048  0.01017733  0.03355155  0.00304822 -0.0234605
  -0.03784344  0.04125245  0.00390546 -0.01548656  0.0462887   0.01650734
  -0.04945288  0.03595961 -0.01329036  0.01054382]
 [-0.01617073  0.00325121  0.02606993  0.01541642 -0.03673673 -0.03384806
  -0.04030023  0.04800146  0.03116843 -0.03657124 -0.03301104  0.01583305
  -0.00707446 -0.04924388  0.04637938  0.03646389]
 [-0.02691524  0.04060968  0.0161143  -0.03616259  0.03204464  0.03907858
  -0.00358195  0.03244385  0.02583173  0.0045199   0.01729841  0.03520605
   0.01753041 -0.01622925 -0.03051195 -0.03034343]
 [-0.01814447  0.01398852 -0.03501763 -0.03345393 -0.03273757  0.0217029
  -0.00572859 -0.00095952 -0.02812878  0.02433988 -0.01251613 -0.03412181
   0.04383043  0.0188278   0.04190919  0.00435774]]
```

Figure 2: LSTM Model execution details.



```
Epoch 1/100
1210/1210 [==============================] - 13s 10ms/step - loss: 0.0201 - accuracy: 0.9739
Epoch 2/100
1210/1210 [==============================] - 12s 10ms/step - loss: 1.5248e-05 - accuracy: 1.0000
Epoch 3/100
1210/1210 [==============================] - 12s 10ms/step - loss: 2.2830e-06 - accuracy: 1.0000
Epoch 4/100
1210/1210 [==============================] - 12s 10ms/step - loss: 3.8884e-07 - accuracy: 1.0000
Epoch 5/100
1210/1210 [==============================] - 12s 10ms/step - loss: 1.4990e-07 - accuracy: 1.0000
Epoch 6/100
1210/1210 [==============================] - 12s 10ms/step - loss: 2.7660e-06 - accuracy: 1.0000
Epoch 7/100
1210/1210 [==============================] - 12s 10ms/step - loss: 3.2731e-06 - accuracy: 1.0000
```

Figure 3: Our Built KG

Figure 4: Extracting only human-Genes out of all genes, for only Gene Subgraph Dataset.

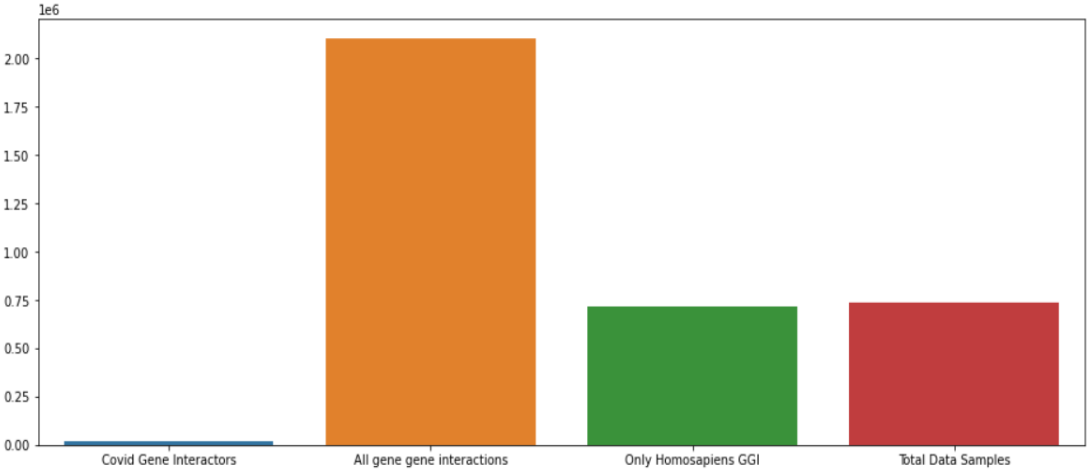


Figure 5: Gene Subgraph

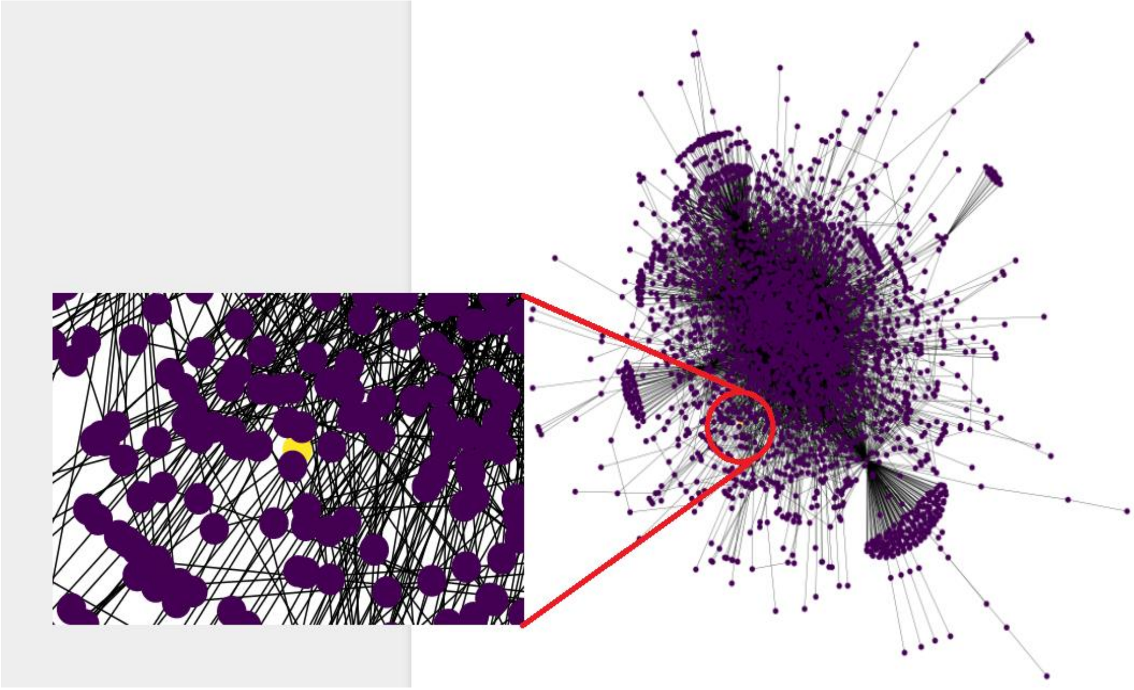


Figure 6: Gene Ontology Subgraph

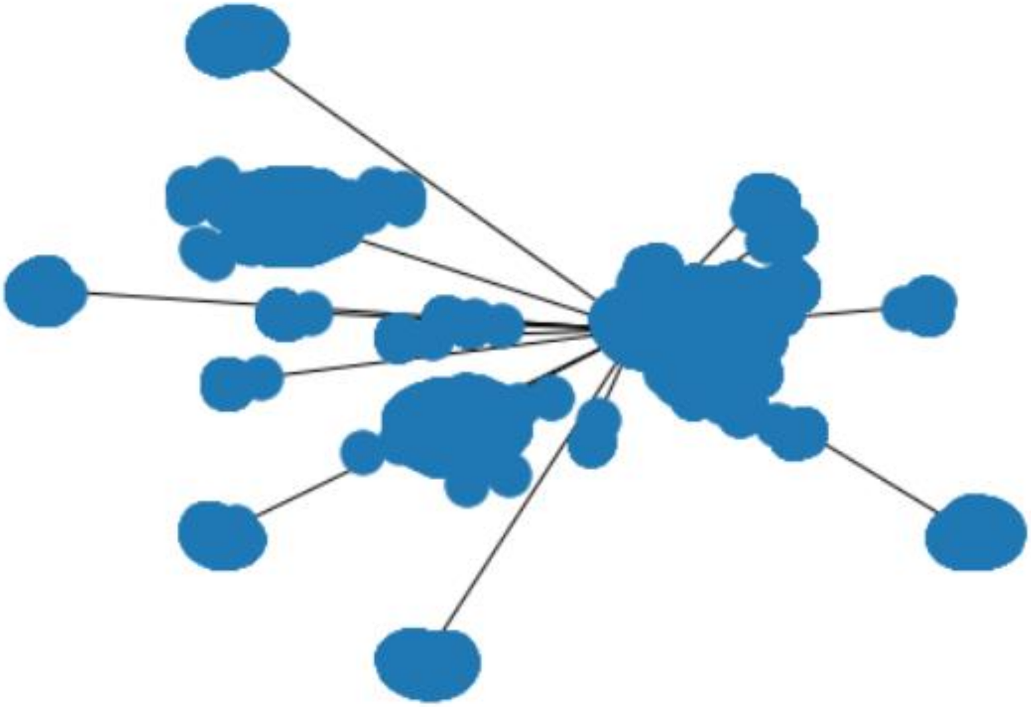


Figure 7: Disease Subgraph

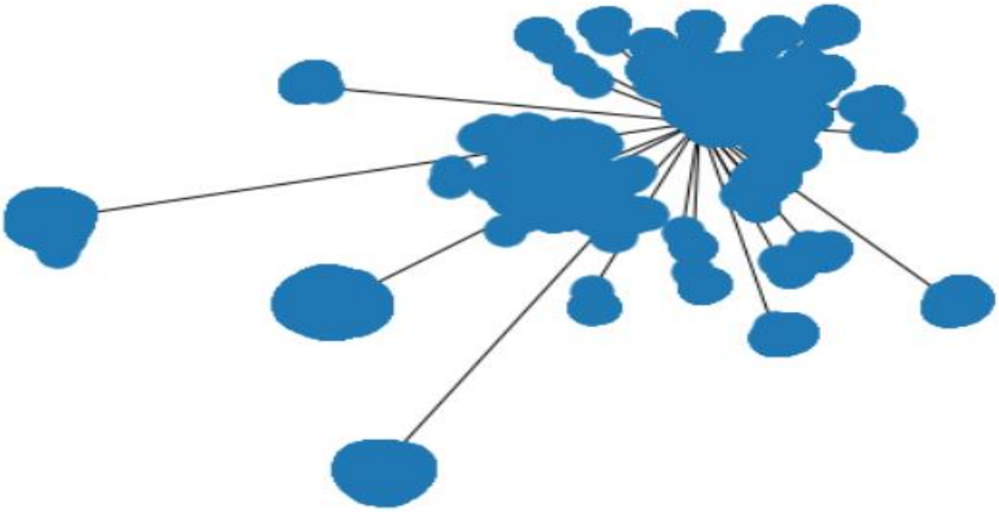
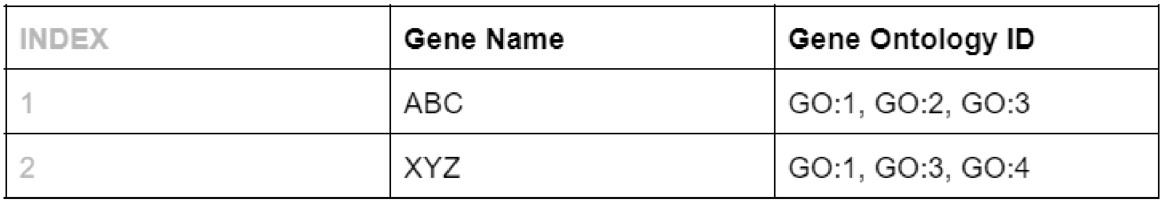


Figure 8: Variant Subgraph

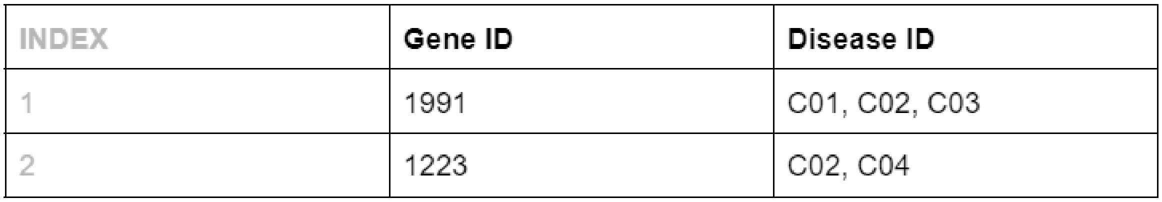
LIST OF TABLES

Table 1: Basic Structure of “Gene with corresponding GOs” Preprocessed Dataset.



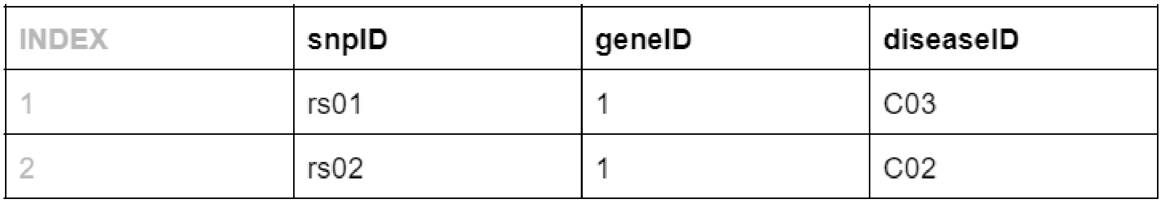
INDEX	Gene Name	Gene Ontology ID
1	ABC	GO:1, GO:2, GO:3
2	XYZ	GO:1, GO:3, GO:4

Table 2: Basic Structure of “Gene with involved Diseases” Preprocessed Dataset.



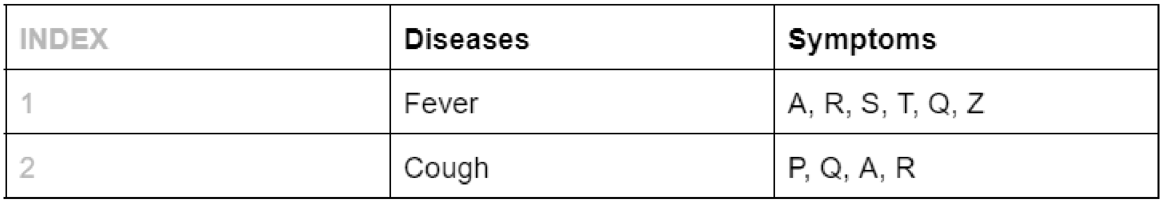
INDEX	Gene ID	Disease ID
1	1991	C01, C02, C03
2	1223	C02, C04

Table 3: Basic Structure of “Variant of Genes and Diseases” Preprocessed Dataset.



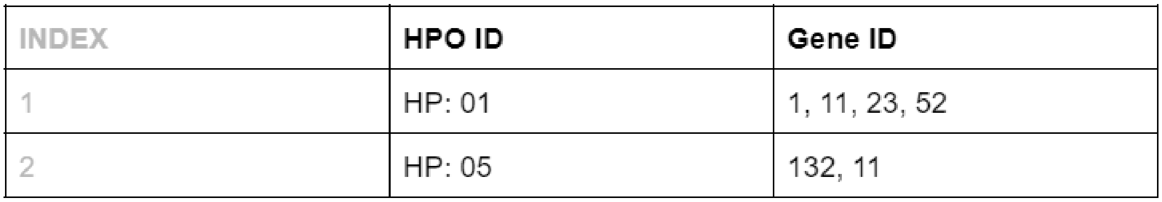
INDEX	snpID	geneID	diseaseID
1	rs01	1	C03
2	rs02	1	C02

Table 4: Basic Structure of “Diseases with their Symptoms” Preprocessed Dataset.



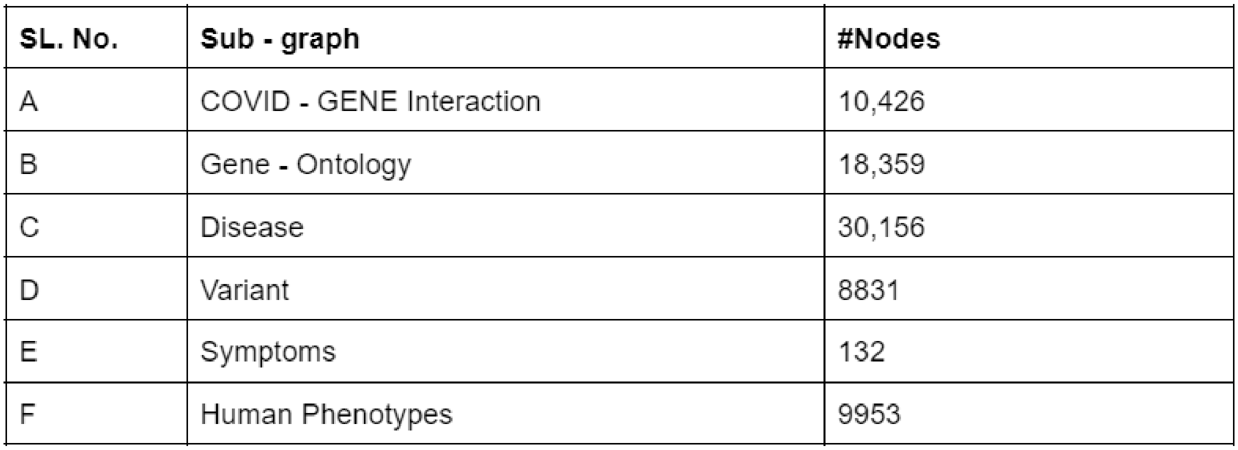
INDEX	Diseases	Symptoms
1	Fever	A, R, S, T, Q, Z
2	Cough	P, Q, A, R

Table 5: Basic Structure of “HPO with their Genes” Preprocessed Dataset.



INDEX	HPO ID	Gene ID
1	HP: 01	1, 11, 23, 52
2	HP: 05	132, 11

Table 6: Number of Nodes in each subgraphs.



SL. No.	Sub - graph	#Nodes
A	COVID - GENE Interaction	10,426
B	Gene - Ontology	18,359
C	Disease	30,156
D	Variant	8831
E	Symptoms	132
F	Human Phenotypes	9953

Table 7: KG Triplets into a Pandas Data Frame.

Head	Relation	Tail	Label
C1332979	Dis_Gene	6336	1
Fever	Sym_Gene	2132	1

ABBREVIATIONS

- GO – Gene Ontology.
- HPO – Human Phenotype Ontology.
- DF – Pandas DataFrame.
- DL – Deep Learning.
- ML – Machine Learning.
- LOT – Refer to LIST OF TABLES section.
- LOF - Refer to LIST OF FIGUES section.

References

1. Saikat Biswas, et al, “Relation Prediction of Co-morbid Diseases Using Knowledge Graph Completion”. – IEEE 2019.

2. Rui wang, et al, “Analysis of SARS-CoV-2 mutations in the United States suggests presence of four substrains and novel variants” – NATURE 2021.

3. Zhen Wang, et al, “Knowledge Graph Embedding by Translating on Hyperplanes” – AAAI Conference 2014.

4. Liu Qiao, et al, “Knowledge Graph Construction Techniques” - Journal of Computer Research and Development 2016.

5. Mohommad Ali Moni, et al, “COMOR: a software for disease comorbidity risk assessment” - Journal of Clinical Bioinformatics 2014.

6. Adekunle Sanyaolu, et al, “Comorbidity and its Impact on Patients with COVID-19” – NATURE 2020.