Constructing a Co-occurrence Network Embeddings to Assist Association Extraction Across COVID-19 and Other Coronavirus Diseases

David Oniani[1], Guoqian Jiang, Ph.D., Hongfang Liu, Ph.D.[3], Feichen Shen, Ph.D.[2]

1. **Luther College,**
2. **Kern Center, Mayo Clinic, Rochester MN**
3. **Division of Digital Health Sciences, Mayo Clinic, Rochester MN**

Corresponding author:

Feichen Shen, PhD
Division of Digital Health Sciences
Mayo Clinic
Rochester 55901 MN, USA
E-mail: shen.feichen@mayo.edu

Word count (up to 4000 words): 3978
Structured Abstract (up to 250 words)**:** 248
Tables (up to 4)**:** 4
Figures (up to 6)**:** 3

Abstract

Objective

As COVID-19/Novel coronavirus started its rapid emergence and gradually transformed into an unprecedented pandemic, the need for having a large knowledge base for the disease became crucial. To address this issue, per request of the White House Office of Science and Technology Policy, new COVID-19 machine readable dataset (CORD-19) has been released. Our objective was to use this dataset in order to build a co-occurrence network linking chemicals, diseases, genes, and mutations based on similarity. Additionally, our goal was to mine novel and implicit associations for COVID-19 leveraging network embeddings.

Materials and Methods

We used the Linked Data version of the COVID-19 Open Research Dataset (CORD-19-on-FHIR) as our dataset. We first utilized the SPQRAL query for detecting co-occurrences among chemicals, diseases, genes, and mutations and building a co-occurrence network. The node2vec model with four different edge embeddings operations (L1, L2, Average, and Hadamard) was used for constructing various network embeddings. Six different algorithms (Decision Tree, Linear Regression, Support Vector Machine, Random Forest, Naive Bayes, and Multi-layer Perceptron) were applied for classification purposes. An unsupervised clustering algorithm was also developed incorporating t-SNE and DBSCAN to divide biomedical entity nodes into subgroups.

Results

Random Forest classifier has shown the best performance on link prediction task across all the generated network embeddings. Specifically, for edge embeddings generated using the average operation, Random Forest achieved the optimal ROC of 0.96, average precision of 0.97, and F1 score of 0.90. For unsupervised learning, xxx clusters were formed with silhouette scores of xxx. Use case study on COVID-19.

Conclusion

In this study, we constructed a co-occurrence network embeddings for COVID-19. Results indicated that the generated embeddings is able to extract significant explicit and implicit associations for COVID-19.

Keywords: COVID-19; co-occurrence network embeddings; association extraction

Introduction

Having now affected millions of people worldwide, COVID-19/Novel coronavirus has become a major pandemic of the century. Most of the countries has declared the state of national emergency and took actions effective immediately to slow the spread. Researchers and medical personnel around the world have published and released thousands of papers over a short period of time, covering a vast scientific ground and exploring medical treatments and possible vaccines for the virus []. With all this information, it is important to assemble all the available heterogeneous information and be aware of the explicit or implicit associations amongst subjects related to COVID-19 (e.g., certain genes could be linked to other genes and/or mutations between COVID-19 and other coronavirus diseases). Figuring out which subjects appear together is one of the approaches for identifying these associations and linking them together. Traditionally, text semantic similarity is one of the approaches for detecting links between words or sentences from unstructured data. One limitation is that it is inefficient to apply this approach over a large collection of free-text data, thus lack a global view to detect significant associations across literature from heterogeneous sources and domains. Normalized data stored in semi-structured graph format is more suitable for global link detection, as linked data by nature provides efficient query scheme over triplets to interpolate between "macroscopic" and "microscopic" search.

In the COVID-19 graph-based analysis, some attempts have been made by researchers all over the world. For example, Ahamed and Samad developed a graph-based model using abstracts of 10,683 COVID-19-related scientific articles and applying betweenness-centrality to rank order the importance of keywords related to drugs, diseases, pathogens, hosts of pathogens, and biomolecules. Bellomarini et al. present a report on ongoing work about the application of automated reasoning and knowledge graph technology to address the impact of the COVID-19 outbreak on the network of Italian companies. Tsiotas and Magafas used visibility graphs to study Greek COVID-19 infection-curve as a complex network. Per request of the White House Office of Science and Technology Policy, new COVID-19 machine readable dataset (CORD-19) has been released and several studies have featured the CORD-19 dataset to investigate COVID-19 related topics. For example, Wolinski has used CORD-19 for extract diseases at risk and calculate relevant indicators as well as created VIDAR-19 (VIsualization of Diseases At Risk in CORD-19). Wang, et al. have conducted CORD-19 named entity recognition leveraging the distant supervision strategy. CORD-19-on-FHIR is a Linked Data version of the COVID-19 Open Research Dataset (CORD-19) data. It is represented in FHIR RDF and was produced by data mining the CORD-19 dataset and adding semantic annotations. In addition, Groza [7] has featured CORD-19-on-FHIR in the analysis of how semantically annotated dataset can be applied for detecting and preventing the potential spread of deceptive information regarding COVID-19.

Co-occurrence information contained in large amount of CORD-19 datasets preserves huge opportunities to detect novel associations across findings from different research articles. However, such information has been largely unexplored for association extraction. Moreover, the lack of measureable association amongst heterogeneous biomedical entities hampers the capability for a quantitative analysis. Inspired by the success of word embeddings in building distributed semantic representations for each word given a corpus, network embeddings provide a solution to map graph nodes to distributional representations and translate nodes' relationships from graph space to embedding space, which makes association between nodes become measurable. In this study, we filled this gap by constructing network embeddings for the CORD-19 co-occurrence network. Specifically, we first derived a co-occurrence network by querying the CORD-19-on-FHIR and focused on the extraction of biomedical entities fell in four categories: chemical, disease, gene, and mutation. We then applied the node2vec model over the generated network and construct network embeddings. We conducted the evaluation quantitatively and qualitatively. For the quantitative evaluation, we generated different embeddings with four embeddings generation operations using a downstream application on graph link prediction and measured the performance with different machine learning algorithms. For the qualitative evaluation, we visualized clusters generated by the optimal COVID-19 network embeddings and analyze associations of heterogeneous biomedical entities related to COVID-19.

Materials

CORD-19-on-FHIR

The CORD-19-on-FHIR datasets is represented in FHIR Resource Description Framework (RDF) format and was produced by data mining the CORD-19 dataset with additional semantic annotations. The purpose of building this dataset is to represent linkage with other biomedical datasets and enable answering research questions. In this study, we used a subset of CORD-19-on-FHIR datasets annotated by Pubtator and LitCovid, including 3,207 COVID-19 related articles in total. Each article was stored in one specific annotation file. For each file, CORD-19-on-FHIR

provides a way to capture all the annotated biomedical entities for each paragraph respectively. A high level example of data stored in the Terse RDF Triple Language (Turtle) format is shown as:

```
pmc:annotations [
    pmc:id "1" ;
            pmc:infons [ pmc:identifier "MESH:D003371" ; pmc:type "Disease" ] ;
            pmc:locations [ pmc:length "5"^^xsd:int ; pmc:offset "20312"^^xsd:int ] ;
            pmc:text "cough" ],
    pmc:id "2" ;
            pmc:infons [ pmc:identifier "MESH:C000657245" ; pmc:type "Disease" ] ;
            pmc:locations [ pmc:length "19"^^xsd:int ; pmc:offset "14766"^^xsd:int ] ;
            pmc:text "2019-nCoV infection" ],
],
 pmc:annotations [
    pmc:id "5" ;
            pmc:infons [ pmc:identifier "59272" ; pmc:ncbi_homologene "41448" ; pmc:type "Gene" ] ;
            pmc:locations [ pmc:length "31"^^xsd:int ; pmc:offset "1986"^^xsd:int ] ;
            pmc:text "angiotensin-converting enzyme 2" ],
    pmc:id "7" ;
            pmc:infons [ pmc:identifier "MESH:C000657245" ; pmc:type "Disease" ] ;
            pmc:locations [ pmc:length "19"^^xsd:int ; pmc:offset "14766"^^xsd:int ] ;
            pmc:text "2019-nCoV infection" ],
],
```

where “pmc:annotations” was used to differentiate different paragraphs within a same article, “pmc:id” was used to indicate different biomedical entities along with entity type (“pmc:type”), location and offset (“pmc:location”), and the original text from literature (“pmc:text”). It made possible to easily detect co-occurrence of biomedical entities within a single paragraph for building the network across all the literature.

Node2Vec

The node2vec model used a random walk based sampling strategy to balance the graph homophily and structural equivalence. The reason we chose to use node2vec is because its ability to learn node representations with a balance between the breast first search (BFS) and depth first search (DFS), which is essential for learning associations in a graph with both local and global views.

Methods

The workflow of this study is made of three modules, including a CORD-19-on-FHIR based co-occurrence network generation module, a network embeddings construction module, and an unsupervised learning module (Figure 1).

Co-occurrence network generation

We first designed a SPARQL query statement to extract co-occurrence of biomedical entities mentioned in literature from CORD-19-on-FHIR. Particularly, in order to largely collect coronavirus related diseases and comorbidities, we built a list of keywords for diseases and symptoms to constrain the searching space, which includes *COVID-19*, *SARS*, *pneumonia*, *fever*, *fibrosis*, *diarrhea*, *coronavirus*, *bronchitis*, *Ebola*, *influenza*, and *ZIKA*. We extracted co-occurrences between gene-disease, mutation-disease and chemical-disease pairs using the following SPARQL query by replacing “Biomedical_Entity” with “Gene”, “Mutation”, and “Chemical” respectively:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX fhir: <http://hl7.org/fhir/>
PREFIX pmc: <https://www.ncbi.nlm.nih.gov/pmc/articles#>
SELECT distinct ?pmc_id0 ?text0 ?pmc_id1 ?text1 (count(?text1) as ?count) WHERE {
?pmc pmc:annotations
[ pmc:id ?id0 ; pmc:text ?text0 ; pmc:infons
[ pmc:type ?type0 ; pmc:identifier ?pmc_id0 ] ] .
   FILTER ((?type0 = 'Disease') ) .
   {select * where{
                        ?pmc pmc:annotations
                        [ pmc:id ?id1 ; pmc:text ?text1 ; pmc:infons
                        [ pmc:type ?type1 ; pmc:identifier ?pmc_id1 ] ] .
        FILTER ((?type1=Biomedical_Entity) && (contains (lcase(str(?text1)), "coronavirus") || contains
(lcase(str(?text1)), "sars") || contains (lcase(str(?text1)), "covid-19") || contains (lcase(str(?text1)), "pneumonia") ||
contains (lcase(str(?text1)), "fever") || contains (lcase(str(?text1)), "fibrosis") || contains (lcase(str(?text1)),
"diarrhea") || contains (lcase(str(?text1)), "bronchitis") || contains (lcase(str(?text1)), "ebola") || contains
(lcase(str(?text1)), "influenze") || contains (lcase(str(?text1)), "zika"))).

     }
   }
}Group by ?pmc_id0 ?text0 ?pmc_id1 ?text1 Order by DESC(?count)
```

The outputs of the query were composed of a list of pairwise biomedical entities with co-occurrence frequency. We then built a network based on this list by adding a link between any two biomedical entities if they have co-occurred at least once. As shown in Figure 1, the co-occurrence network was represented by source-target pairs, which were then used as input data for training node representations.

Network embeddings representation learning

We applied the node2vec model in this module. Node2vec implements a 2nd order random walk over the graph topological structure, denoting that three types of node are involved in a specific walk, namely source entity, intermediate entity, and target entity. Given any source entity as $\boldsymbol{E_s}$, target entity as $\boldsymbol{E_t}$, intermediate entity that exists on the path between $\boldsymbol{E_s}$ and $\boldsymbol{E_t}$ as $\boldsymbol{E_i}$, normalization constant as $\boldsymbol{Z}$, the distribution of entity $\boldsymbol{E_t}$ with a fixed length of random walk can be represented as:

$$\boldsymbol{P(E_t|E_i)} = \begin{cases} \frac{\boldsymbol{\pi(E_i,E_t)}}{\boldsymbol{Z}} & \boldsymbol{if\ (E_i,E_t)\ is\ an\ edge} \\ \boldsymbol{0} & \boldsymbol{otherwise} \end{cases} \quad (1)$$

where $\boldsymbol{\pi(E_i,E_t)}$ is a transition probability between entities $\boldsymbol{E_i}$ and $\boldsymbol{E_t}$. Given the weight over edge $\boldsymbol{(E_i,E_t)}$ as $\boldsymbol{w(E_i,E_t)} = \boldsymbol{1}$, $\boldsymbol{\pi(E_i,E_t)}$ could be calculated as:

$$\pi(E_i,\boldsymbol{E_t}) = \alpha(E_s,\boldsymbol{E_t}) \cdot w(E_i,\boldsymbol{E_t}) \quad (2)$$

$\boldsymbol{\alpha(E_s,E_t)}$ is a searching bias term developed in node2vec. Specifically, node2vec introduced two hyperparameters $\boldsymbol{p}$ and $\boldsymbol{q}$ to balance between the BFS and DFS searching strategy for both local and global optimization. Given the shortest distance between $\boldsymbol{E_s}$ and $\boldsymbol{E_t}$ as $\boldsymbol{sd(E_s,E_t)}$, $\boldsymbol{\alpha}$ for entities $\boldsymbol{E_s}$ and $\boldsymbol{E_t}$ is computed based on $\boldsymbol{p}$ and $\boldsymbol{q}$:

$$\boldsymbol{\alpha(E_s,E_t)} = \begin{cases} \boldsymbol{\frac{1}{p}} & \boldsymbol{if\ sd(E_s,E_t) = 0} \\ \boldsymbol{1} & \boldsymbol{if\ sd(E_s,E_t) = 1} \\ \boldsymbol{\frac{1}{q}} & \boldsymbol{if\ sd(E_s,E_t) = 2} \end{cases}$$

After learning the sampled network data using random walk, we then leveraged the Skip-gram model to train entity representations on the sampled data. For each entity node $\boldsymbol{E_s} \in \boldsymbol{E}$ and all its sampled neighbors $\mathbf{N}(\boldsymbol{E_s})$, the loss function for entity representation learning could be described as:

$$\max_f \sum_{E_s \in E} \log P(\mathrm{N}(E_s)|f(E_s)) \quad (4)$$

In the end, we normalized the prediction distribution by using a nonlinearity function (e.g., softmax) and optimize this loss function using the Stochastic gradient descent algorithm.

Unsupervised clustering of network embeddings

To render the relatively high-dimensional embedding representations of network embeddings into a lower-dimensional space, we utilized the t-distributed stochastic neighbor embedding (t-SNE) algorithm to render the embeddings for all entity nodes into a 2D space. t-SNE does not perform clustering in and of itself, but instead renders each node embedding into a (x, y) coordinate. As such, additional post-processing is needed to re-group these points into discrete clusters. The Density-based spatial clustering of applications with noise (DBSCAN) algorithm was therefore used over output generated by the t-SNE to further partition different entity groups into distinct clusters. Given a parameter ε that denotes how close points should be to each other and another parameter k that indicates the minimum number of points, the DBSCAN clustered similar entity nodes together based on density according to the pre-defined two parameters.

Experiments

From CORD-19-on-FHIR dataset, we extracted 49,696 co-occurred biomedical entities for 3,626 coronavirus related diseases, 5,741 genes, 524 mutations, and 6,878 chemicals. Thus the derived co-occurrence network contains 16,769 nodes and 49,696 edges in total.

For quantitative evaluation, we generated the optimal network embeddings by performing a downstream link prediction task. Link prediction is a procedure where the goal is to predict the relationship between any two nodes and use the performance of a prediction to evaluate the quality of the generated network embeddings. Edge embeddings were used in this task in order to investigate the relationships between nodes leveraging distributional representations provided by entity embeddings. For any given nodes $\boldsymbol{E_s}$, $\boldsymbol{E_t}$ and their corresponding entity representations $\boldsymbol{f(E_s)}$ and $\boldsymbol{f(E_t)}$, edge embeddings were calculated using four operations, namely Hadamard, Average, L1 and L2 as shown in Equations 5-8 respectively:

$$\boldsymbol{Hadamard(E_s, E_t) = f(E_s) * f(E_t)} \quad (5)$$

$$\boldsymbol{Average(E_s, E_t) = \frac{f(E_s)+f(E_t)}{2}} \quad (6)$$

$$\boldsymbol{L1(E_s, E_t) = |f(E_s) - f(E_t)|} \quad (7)$$

$$\boldsymbol{L2(E_s, E_t) = |f(E_s) - f(E_t)|^2} \quad (8)$$

To evaluate the performance on different edge embeddings on link prediction task, we used six conventional classification algorithms to achieve the goal, including Decision Tree (DT), Linear Regression (LR), Support Vector Machine (SVM), Random Forest (RF), Naive Bayes (NB), and Multi-layer Perceptron (MLP). Specifically, The Boolean function $\boldsymbol{L(E_s, E_t)}$ was used to determine the existence of edge(s) between nodes $\boldsymbol{E_s}$ and $\boldsymbol{E_t}$, where $\boldsymbol{L(E_s, E_t) = 1}$ indicates positive links and $\boldsymbol{L(E_s, E_t) = 0}$ represents negative links. We fit features of edge embeddings with labels provided by $\boldsymbol{L(E_s, E_t)}$ to train the model. For positive examples, for each of the four networks, 60%, 10%, and 30% of all their edges were used for training, validation, and testing purposes respectively. For negative examples, an equal number of node pairs were randomly sampled (with the same ratio among training, validation, and testing sets as 60%, 10%, and 30% respectively).

For each classifier, we plotted the receiver operating characteristic (ROC) curve and computed the area under the ROC curve (AUROC) to report link prediction performance**.** Moreover, as shown in Equations 9-12, we used average precision, precision, recall, and F1 score to quantify the link prediction performance amongst four edge embeddings.

$$\boldsymbol{Precision} = \frac{|\{\boldsymbol{True\ Relations}\} \cap \{\boldsymbol{Predicted\ Relations}\}|}{|\{\boldsymbol{Predicted\ Relations}\}|} \quad (9)$$

$$\boldsymbol{Recall} = \frac{|\{\boldsymbol{True\ Relations}\} \cap \{\boldsymbol{Predicted\ Relations}\}|}{|\{\boldsymbol{True\ Relations}\}|} \quad (10)$$

$$\boldsymbol{F1\ score} = \frac{\boldsymbol{2 * precision * recall}}{\boldsymbol{precision + recall}} \quad (11)$$

$$\boldsymbol{AP} = \sum_n (\boldsymbol{Recall_n - Recall_{n-1}})\boldsymbol{Precision_n} \quad (12)$$

For qualitative evaluation, we first visualized the network embeddings clustering output and used the silhouette score to evaluate the performance of clustering. Silhouette score is used to calculate the average distance to entities in the same cluster with the average distance to entities in other clusters, and a high silhouette score indicate well clustered. Specifically, given any entity node $\boldsymbol{e}$ in cluster $\boldsymbol{C_e}$, the internal mean distance is defined as:

$$m(e) = \frac{1}{|C_e|-1}\sum_{t \in C_e, e \neq t} d(e,t) \quad (13)$$

where $d(e,t)$ is the distance between node e and t in C_e. Similarly, external mean distance is described as:

$$n(e) = \min_{k \neq e} \frac{1}{C_k}\sum_{t \in C_e} d(e,t) \quad (14)$$

Overall, the silhouette score is calculated incorporating both internal and external mean distances:

$$s(e) = \begin{cases} 1 - \frac{m(e)}{n(e)}, if\ m(e) < n(e) \\ 0, \qquad if\ m(e) = n(e) \\ \frac{n(e)}{m(e)} - 1, if\ m(e) > n(e) \end{cases} \quad (15)$$

For some selected coronavirus diseases, we also located the cluster they belonged to and checked the most similar entities within the cluster using cosine similarity. Let E_s denote any given coronavirus disease and E_t denote any biomedical entity inferred by network embeddings, and $f(E_s)$ and $f(E_t)$ represent the embeddings for E_s and E_t respectively, cosine similarity were calculated as shown in Equation 16.

$$cosine\_similarity\ (E_s, E_t) = \frac{f(E_s) \cdot f(E_t)}{\|f(E_s)\| \|f(E_t)\|} \quad (16)$$

Results

Different embeddings were generated by using neighbor size of 10, number of walks of 10, window size of 10, and dimensionality of 128. The optimal p and q were also tuned as 0.5 and 0.25 for each training process respectively. As shown in Tables X, we presented the evaluation results of four different edge embedding operations along with six different classification algorithms.

Quantitative evaluation

We found that, in general, the embeddings trained by the Average operation achieved the best performance across all the evaluation metrics. The optimal average precision, ROC score, precision and recall was reached when the RF was used, and the optimal F1 score was achieved by using NB. L1 and L2 had roughly the same performance, both peaking at ROC = 0.95 and AP = 0.96 with RF classifier. Among all four approaches, Hadamard yielded the worst performance, peaking at ROC = 0.89 and AP = 0.92 with RF classifier. Across all six classification algorithms, the worst performance, on the other hand, was shown by DT and MLP classifiers.

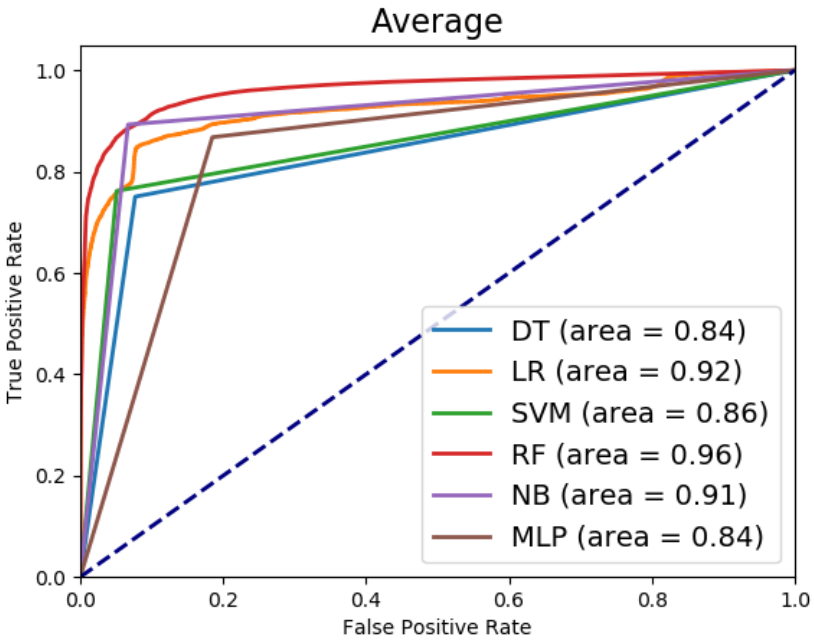
Regarding different classification algorithms, for the L1 embeddings embedding operation, RF and LR had similar performance, with ROC_RF = 0.96, AP_RF = 0.97 and ROC_LR = 0.96, AP_LR = 0.95. SVM, NB, and MLP were also not much different from each other. DT had the worst performance with ROC = 0.80 and AP = 0.75. Similarly, for L2, RF and LR had roughly the same performance. SVM, NB, and MLP were also equally performant. DT had the worst performance with ROC = 0.80 and AP = 0.75. For the Average operation, RF has outperformed all the other classification methods with ROC = 0.96 and AP = 0.97. LR and NB had similar performance. DT, SVM, and MLP were similar, yet all of them were behind RF, LR, and NB. For Hadamard, RF has shown the best performance with ROC = 0.89 and AP = 0.92. The rest of the methods have shown roughly the same performance except for MLP which has the worst performance across all classification algorithms as well as across every embedding generation approach, with ROC = 0.60 and 0.56.

We finalized the network embeddings generated by the Average operation as the optimal one and a ROC curve for the performance across six classification algorithms is shown in Figure 2.

Table X

Operations	Algorithms	Average	ROC score	Precision	Recall	F1 score

		Precision				
Hadamard	DT	0.79	0.82	0.84	0.82	0.81
	LR	0.89	0.83	0.86	0.82	0.81
	SVM	0.80	0.81	0.85	0.81	0.81
	RF	0.92	0.89	0.87	0.86	0.86
	NB	0.82	0.84	0.86	0.84	0.84
	MLP	0.56	0.60	0.63	0.60	0.57
Average	DT	0.81	0.84	0.85	0.84	0.84
	LR	0.94	0.92	0.87	0.85	0.85
	SVM	0.83	0.86	0.87	0.86	0.85
	RF	**0.97**	**0.96**	**0.91**	**0.91**	0.90
	NB	0.88	0.91	**0.91**	**0.91**	**0.91**
	MLP	0.78	0.84	0.84	0.84	0.84
L1	DT	0.75	0.80	0.80	0.80	0.80
	LR	0.95	0.94	0.89	0.89	0.89
	SVM	0.87	0.89	0.90	0.89	0.89
	RF	0.96	0.95	0.89	0.88	0.88
	NB	0.85	0.88	0.89	0.88	0.88
	MLP	0.87	0.89	0.89	0.89	0.89
L2	DT	0.75	0.80	0.80	0.80	0.80
	LR	0.94	0.93	0.89	0.88	0.88
	SVM	0.87	0.88	0.90	0.88	0.88
	RF	0.96	0.95	0.89	0.88	0.88
	NB	0.85	0.87	0.88	0.87	0.87
	MLP	0.85	0.87	0.88	0.87	0.87



Qualitative evaluation

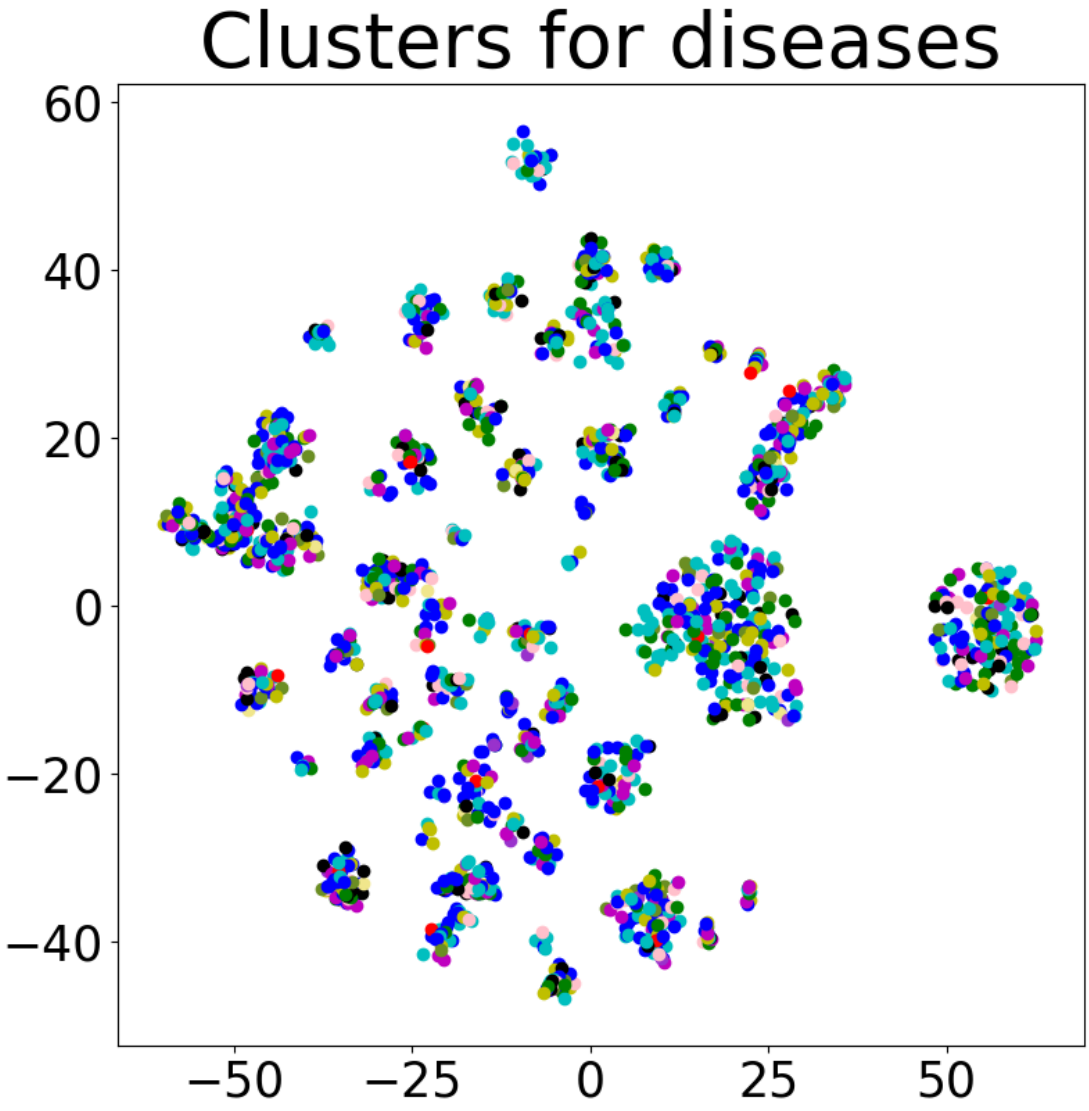
We clustered the network embeddings by selecting the optimal hyperparameters $\boldsymbol{\varepsilon = 1.5}$ and $\boldsymbol{k = 37}$ for the DBSCAN algorithm. 64 clusters were generated with a Silhouette score of 0.128. The visualization

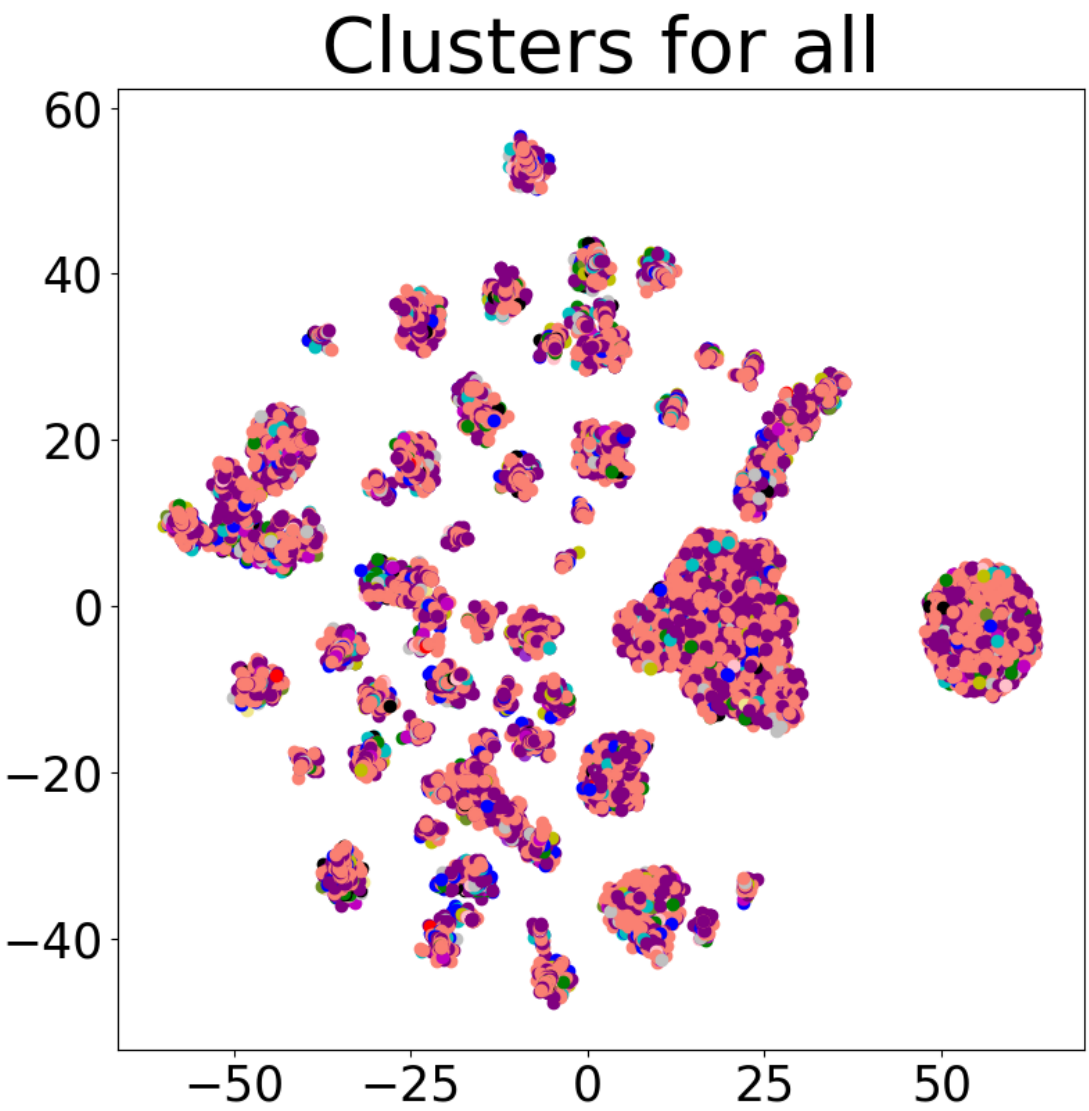
We used the network embeddings generated through the optima Average operation to conduct a further qualitative evaluation. We first selected five coronavirus diseases and list top 5 closest entities using cosine similarity.

Coronavirus diseases	**Top 10 closest entities**	**Cosine similarity score**
COVID-19 (Cluster 6)	Vp35 (Gene)	0.97769576
	Hd11 (Gene)	0.97743833
	Coronavirus infection process (Disease)	0.9699671
	Fibroblast growth factor (fgf)-2 (Gene)	0.96551824
	Acute respiratory infection illness (Disease)	0.9596049
	Pig's (Gene)	0.9575937
	Tgfalpha (Gene)	0.9571036
	Sfpq (Gene)	0.9560543

	Tumour necrosis factor (Gene)	0.95491123
	Praziquantel (Chemical)	0.9537396
Pulmonary coronavirus infection (Cluster 1)	Ptp (Gene)	0.9753737
	Sars-cov-infected hae culturesa (Disease)	0.96988034
	"5'-tgg gat tca aca" (Chemical)	0.9672154
	Tracheanasal respiratory epithelial cells nd llamas (lama glama) in (Disease)	0.96580225
	Suppressor of cytokine signaling 3 (Gene)	0.96197987
	Kat (Gene)	0.9604422
	Cd32 (Gene)	0.957286
	Maternal sars infection (Disease)	0.9552708
	Respiratory syndrome coronavirus (mers-cov) infections (Disease)	0.9546926
	S27 (Gene)	0.95460874
Sars-cov infection damages lung (Cluster 2)	il-1-alpha (Gene)	0.9599887
	sucralfate prn (Chemical)	0.95891565
	acute respiratory syndrome-cov infection (Disease)	0.9555107
	il-5- and il-13-producing ilc-iis (Gene)	0.9486904
	hap1 (Gene)	0.9341973
	fsk (Chemical)	0.9336599
	low fever (Disease)	0.93280876
	hiv and ebola virus infection (Disease)	0.9326824
	ykl-40 (Gene)	0.928821
	etf (Gene)	0.9280162
Coronavirus upper respiratory	viruses actinobacillus	0.9889679

infection (Cluster 23)	pleuropneumoniae (Disease)	
	plasmin (Gene)	0.9718952
	jam-1 (Gene)	0.96537995
	tnf receptor-associated factor 6 (Gene)	0.96479356
	gpc3 (Gene)	0.9613348
	renin (Gene)	0.9582391
	zo-1 (Gene)	0.9563359
	cathepsin g (Gene)	0.9556179
	rs5743313 (Mutation)	0.9547445
	alpha1 antitrypsin (Gene)	0.95441306
Coronavirus-infected pneumonia (Cluster 10)	respiratory syncytial viral infection (Disease)	0.9923498
	pegylated interferon-alpha (Chemical)	0.98907185
	ifitm6 (Gene)	0.9872188
	feline b (Chemical)	0.98577255
	e119v (Mutation)	0.985413
	epac2 (Gene)	0.9850465
	gfpt2 (Gene)	0.9848544
	hepatitis coronavirus infection (Disease)	0.98425007
	ouabain (Chemical)	0.9797489
	ly6g (Gene)	0.97861314


Clusters for diseases
60
40
20
0
−20
−40
−50
−25
0
25
50



Discussion and future work

Keep PMID in the future to track evidence.

Add more disease into sparql query as more research outcome indicate co-related diseases

No Filtering co-occurrences automatically using crossvalidation. In the future, add frequency over edge to indicate weight

Conclusion

The study has explored co-occurrence-based graph generation through several approaches. We have used the FHIR version of CORD-19 – CORD-19-on-FHIR and built the co-occurrence graph utilizing the sparql querying language. node2vec was used for generating embeddings. Four different approaches for generating embeddings were compared, namely L1, L2, Average, and Hadamard. And 6 different classification algorithms were compared. Average appeared to be the best approach for generating embeddings and in all cases, Random Forest classifier has shown the best performance. t-SNE algorithm was employed for data dimensionality reduction and an interactive

visualization of the network was also created, linking and clustering chemicals, diseases, genes, and mutations based on similarity.

Acknowledgement

This work has been supported by the National Institute of Health (NIH) grants U01TR0062-1.

Competing Interests: None

*COVID-19, NCOVID-19, COVID-19 pneumonia, COVID-19 infected, SARS, SARS coronavirus, SARS-COV-*infection, *SARS-COV, SARS-COV-2-pheumonia, pneumonia, fever, fibrosis, diarrhea, coronavirus, bronchitis, Ebola infection, influenza, initiation of ZIKA virus infection*.

Reference: