SYQ1

FYP Final Report

A Knowledge Graph to Better Understand Medical Text about COVID-19

by

CHAN Tsz Ho, TANG Yutian, WONG Pui Ying and ZHANG Liangwei

SYQ1

Advised by

Prof. Song Yangqiu

Submitted in partial fulfillment

of the requirements for COMP 4981

in the

Department of Computer Science

The Hong Kong University of Science and Technology

2020-2021

Date of submission: April 14, 2021

Abstract

The knowledge graph has served as a tool for scientific research for decades since the emergence of the internet. During the development, numerous knowledge graphs were built to store information from various fields. However, the result from the past does not always satisfy the need for now. As the outbreak of COVID-19, information of the unknown virus is hard to be found in the past bio-knowledge graph. The number of papers that study COVID-19 is keeping increasing, which may cause difficulty in research. Therefore, we built a knowledge graph from a medical text corpus of 29500 papers and provided an interactive web application on it for users to access the information.

Similar to most knowledge graphs, we extract information in an entity-relation way to represent the key of the sentence. By adding specific COVID-19 related entity types, our graph can cover more precise information of the virus. Meanwhile, this project utilizes both manually validated relation extraction and machine learning based bootstrap relation extraction to produce high confidence results. Three models were tested to improve the performance and extend the range of relation detection.

In order to help people to use our project better, we provide web applications with various tools for users. The website demonstrates the extracted result while the searching function and layout would help users with specific needs to get what they want. We hope this project can offer users some assistance in terms of research or information collection to explore more possibilities.

Table of Contents

1. Introduction

1.1. Overview

With the rapid development of modern science and technology, people can now produce and share information very conveniently. However, we now live in an era of information explosion, so people usually cannot quickly summarize data from enormous and complicated datasets on their own using manual procedures. To solve this problem, Google released "Google Knowledge Graph" in 2012 [1]. This is a graph-based knowledge representation describing and connecting real-world entities or concepts. Figure 1 shows a simple example of a knowledge graph (KG) and how it integrates information. Unlike traditional datasets, which require careful design and updating of tables, KGs just connect entities together so that we can easily understand the information by looking at a graph. On a small scale, the cost of querying and understanding information from a KG is about the same as it is with a traditional database. But with large datasets, KGs have an advantage in that they make querying and understanding much easier.

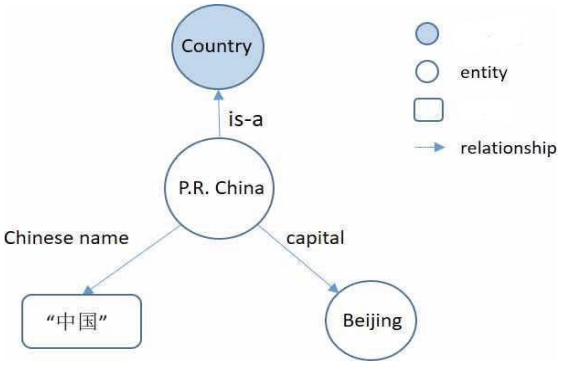

Figure 1– Knowledge graph example.

In 2020, COVID-19 has been a serious issue for people all around the world. With numerous researchers, institutions, and governments focusing their efforts on understanding COVID-19, as well as its spread and containment, it is a challenge to comprehend all novel findings and connect them with those that are already known, especially since the constant rate of the increase of publications is high [2]. Therefore, for summarizing and analyzing the information about COVID-19, KGs can be used to improve the performance of searching for useful information.

We believe that the speed of disease research will continue to accelerate. So in our FYP, we built a KG that contains the most updated entity types related to COVID-19 and a straightforward and user-friendly user interface. Our KG can serve as a useful tool for medical research.

1.2. Objectives

The goal of this project was mainly to construct a knowledge graph of COVID-19 related corpus to help researchers to understand it in an easier way.
During the development of the project, we mainly focused on the following objectives:

1. Integrate and modify the existing algorithms that extract medical text entities and link them, extract potential relations between entities, and use machine learning algorithms to search for further relations.
2. Develop a database that contains the extracted knowledge related to COVID-19, which is integrated into an ontology in the form of a knowledge graph.
3. Develop a web application that contains a basic knowledge graph that visualizes medical text entities and their potential relationships in various types of representations and enable querying through our knowledge graph according to the user's instructions.

To achieve the first objective, we explored existing efficient methods and modified them to perform text mining and knowledge visualization.
To achieve the second objective, we deployed the database on an online cloud platform and found a stable database service provider.
To achieve the third objective, we built a frontend to represent the knowledge graph properly.

The biggest challenge that we met during the development was annotation enrichment. As we decided to obtain annotations out of the existing dataset, we had

to manually define the types of entities in the training set to ensure the accuracy and coverage of the entity extraction algorithm. We overcame this through the manual definition process and three existing popular models.

1.3. Literature Survey

The research of scientific knowledge graphs began to develop rapidly after the emergence of the Internet in the late 1990s. In the past two decades, much work has been done to develop and expand knowledge graph methods and systems, and the work is ongoing. However, most of the existing work focuses on either entity extraction or relation extraction. For our COVID-19 knowledge graph, we are focused on both of them.

Figure 2 – Visualization of the COVID-19 KG in BiKMi [2].

Information extraction serves as a foundation for the construction of knowledge graphs. Since extracting invaluable information from a rapidly increasing number of medical articles has gained popularity among researchers, biomedical text mining models have continuously been improved. Despite some known Natural Language Processing (NLP) advancements like BERT [3], these general models have often had poor performance on specific biomedical-domain corpora. Thus, some pre-trained models which were trained on massive datasets in biomedical domains, such as BioBERT [4] and SciBERT [5], have recently been developed to effectively recognize and capture information from biomedical entities. However, in our project about COVID-19, some specific entities not previously included in common biomedical models should be annotated. In order for our system to provide beneficial assistance on studies of COVID-19-related virus structures, spreading mechanisms

and possible treatment, some fine-tuning techniques have also been applied to improve the performance of biomedical text mining in this new area.

Moreover, it is crucial to extract the factual relations between entities detected. Traditionally, dependency trees have been used to label direct relations between distant words that are syntactically correlated. While these trees have an accuracy of over 95% on some aspects, the performance in medical texts has significantly declined in recent years [6]. In order to obtain a more stable accuracy in relation extraction, graph neural networks (GNNs) [6] were proposed to exploit the most relevant relations between different entities.

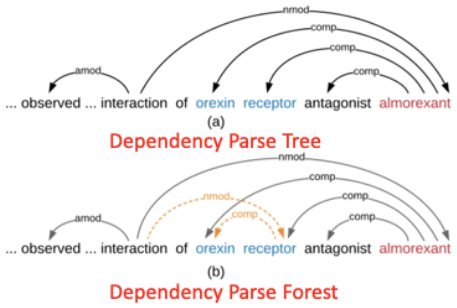

Figure 3 – Dependency parse tree and forest [6]

Most of the current COVID-19-related knowledge graphs recognize and capture medical text from existing entity databases, such as those of biomedical systems, so the entity extraction may not be thorough and accurate. At the same time, the existing COVID-19-related knowledge graphs mainly focus on entities and their relations but rarely consider specific entity types and their descriptions. Therefore, we expanded the range of entities by adding specific entity types and descriptions related to COVID-19 so as to improve the precision of the classification when doing entity extraction. Furthermore, we also tried different functions and layouts to provide a more useful knowledge graph.

2. Methodology

2.1. System Overview

In accordance with our objectives, we built a knowledge graph of COVID-19 medical text which allowed users to search and gain information more efficiently through our web application, as the knowledge graph provided a means for capturing, representing, and formalizing structured information [7].

The building work of the knowledge graph can be shown in two parts: The pipeline that we process the original data and extract confident information; The web application for users to access our knowledge graph conveniently. Those two systems are connected by the database that contains processed information.

The system-chart below shows a comprehensive overview of our Natural Language Processing pipeline.

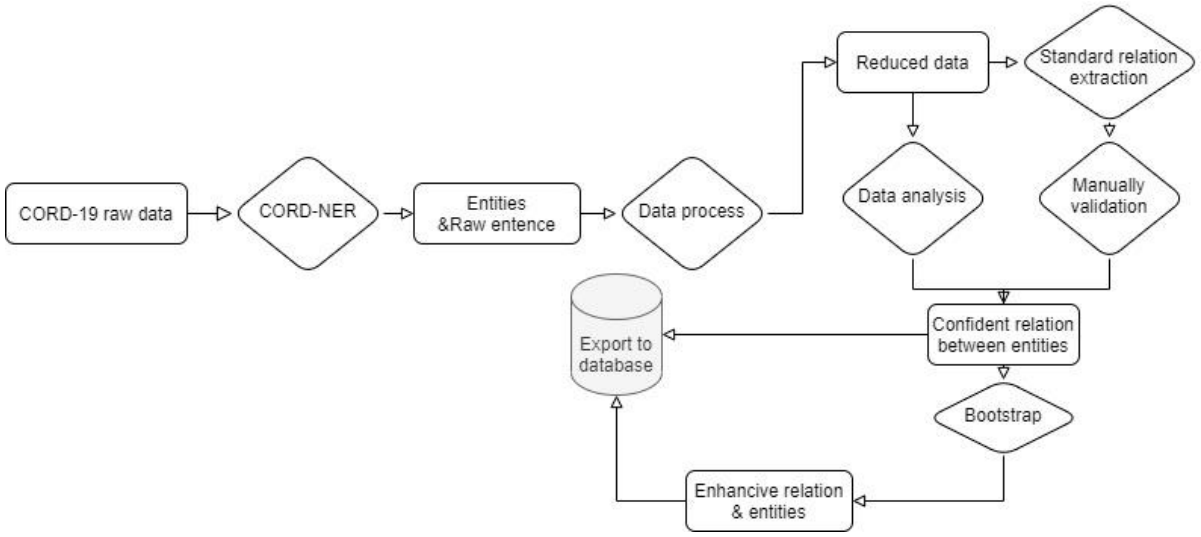

Figure 4 – Pipeline overview

The diagram below demonstrates the design flow for users to access our knowledge graph.

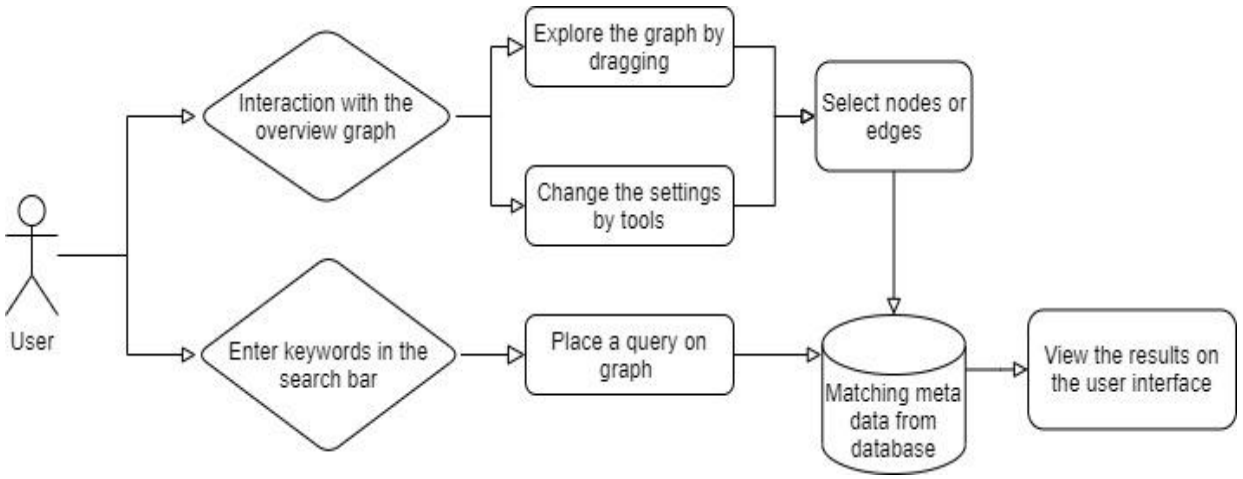

Figure 5 – User-case diagram of the web application

2.2. Data Processing

2.2.1 The Dataset

Our raw COVID-19 dataset [8] contains 29500 COVID-19 and coronavirus-related research papers(e.g. SARS, MERS, etc.) from PubMed's PMC open access corpus and additional COVID-19 research articles by the WHO. We have generated some new insights in support of the fight against this infectious disease based on this raw dataset.

2.2.2 Design

2.2.2.1 Entity Extraction from Medical Text

The most critical elements in our knowledge graph should be those entities that comprehensively represent the topic of COVID-19. In order to train data annotation for entities without much human effort, fine-tuning techniques should be adopted such as Named Entity Recognition (NER) [9]. NER locates and classifies different sorts of entities from unstructured text into predefined categories. It should serve as a fundamental step in information extraction for the construction of a knowledge graph.

2.2.2.2 Data processing

It is of pivotal importance to perform data cleaning and future processing. Python and libraries such as NLTK and Numpy should be used to split the sentences and filter out those with useful information for future steps. In addition, we should detect and analyze some special cases during this process in order to filter out erroneous or abnormal cases to increase the accuracy of information extraction manually.

2.2.3 Implementation

2.2.3.1 Entity Extraction

In order to identify the type of named entity from the input COVID-19 corpus, we have got to know several possible NER models on the original dataset from COVID-19 Open Research Dataset Challenge (CORD-19). Finally, we have decided to adopt the Comprehensive Named Entity Recognition (NER) [9] by Xuan Wang. It revealed better annotation results and accuracy compared to traditional NER models.

Angiotensin-converting enzyme 2 **GENE_OR_GENOME** (ACE2 **GENE_OR_GENOME**) as a SARS-CoV-2 **CORONAVIRUS** receptor: molecular mechanisms and potential therapeutic target. SARS-CoV-2 **CORONAVIRUS** has been sequenced [3 **CARDINAL**] . A phylogenetic **EVOLUTION** analysis [3 **CARDINAL** , 4 **CARDINAL**] found a bat **WILDLIFE** origin for the SARS-CoV-2 **CORONAVIRUS** . There is a diversity of possible intermediate hosts for SARS-CoV-2 **CORONAVIRUS** , including pangolins **WILDLIFE** , but not mice **EUKARYOTE** and rats **EUKARYOTE** [5 **CARDINAL**] . There are many similarities of SARS-CoV-2 **CORONAVIRUS** with the original SARS-CoV **CORONAVIRUS** . Using computer modeling , Xu et al . [6 **CARDINAL**] found that the spike proteins **GENE_OR_GENOME** of SARS-CoV-2 **CORONAVIRUS** and SARS-CoV **CORONAVIRUS** have almost identical 3-D structures in the receptor binding domain that maintains Van der Waals forces **PHYSICAL_SCIENCE** . SARS-CoV spike proteins **GENE_OR_GENOME** has a strong binding affinity to human ACE2 **GENE_OR_GENOME** , based on biochemical interaction studies and crystal structure analysis [7 **CARDINAL**] . SARS-CoV-2 **CORONAVIRUS** and SARS-CoV spike proteins **GENE_OR_GENOME** share identity in amino acid sequences and ……

Figure 6 – NER annotation [9]

Spacy (General NER):

A phylogenetic analysis [3 **CARDINAL** , 4 **CARDINAL**] found a bat origin for the SARS-CoV-2.

SciSpacy (Biomedical NER):

A phylogenetic analysis [3, 4] found a bat origin for the SARS-CoV-2 **SIMPLE_CHEMICAL** .

Ours:

A phylogenetic **EVOLUTION** analysis [3 **CARDINAL** , 4 **CARDINAL**] found a bat **WILDLIFE** origin for the sars_cov_2 **CORONAVIRUS** .

Figure 7 – Annotation results for different NER models [9]

Moreover, the NER model added some unique annotation types that represented information about COVID-19. It combined the common biomedical entity types from pre-trained NER methods from SciSpacy and other COVID-19 newly-related types such as ***Coronavirus***, ***Physical Science***, ***Wildlife***, ***Evolution.*** Finally, we merged those fine-grained entity-types into a NER annotation JSON file for future processing.

CORONAVIRUS	**EVOLUTION**	**WILDLIFE**	**PHYSICAL_SCIENCE**
sars	mutation	bat	positively charged
cov	phylogenetic	wild birds	negatively charged
mers	evolution	wild animals	force field
covid-19	recombination	fruit bats	highly hydrophobic
sars-cov-2	substitutions	pteropus	van der waals interactions
LIVESTOCK	**MATERIAL**	**SUBSTRATE**	**IMMUNE_RESPONSE**
pigs	air	blood	immunization
poultry	plastic	urine	immunity
calves	fluids	sputum	immune cells
chicken	copper	saliva	innate immune
pig	silica	fecal	inflammatory response

Table 1 – COVID-19 new entity types

2.2.3.2 Data processing

Based on our design, we used Python to perform sentence splitting for abstract and body in each article and entity filtering to extract sentences with two entities. In total we obtained 2,477,390 sentences, and filtered out 132,017 duplicated data samples, which accounts for about 5.32%. Finally, we got 2,345,373 sentences from the whole NER dataset. We analyzed some basic information on our data as below.

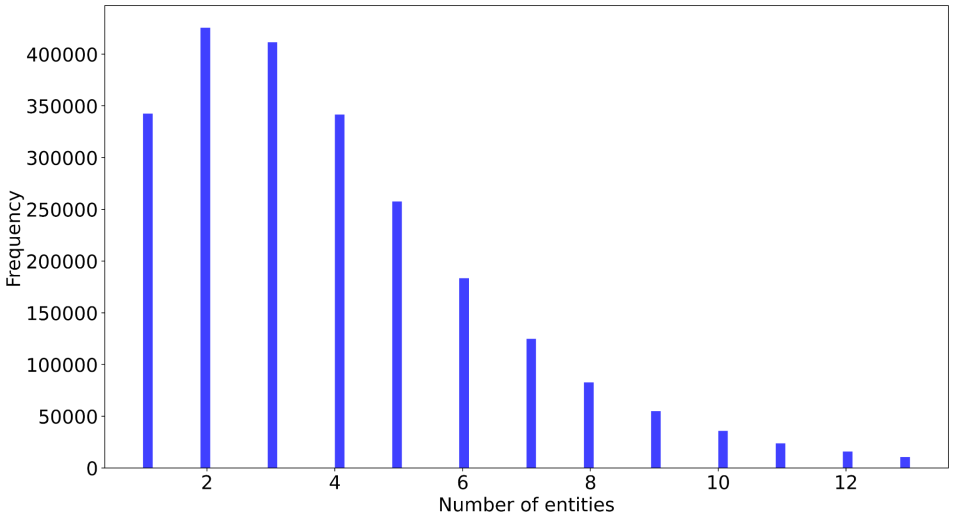

Figure 8 - Distribution of the number of entities in sentences

Although there existed some problems including repetition and loss in this dataset, we still observed that sentences with two entities or three entities accounted for a relatively larger proportion than others. Our project mainly focused on the information and relation extraction on sentences with two entities afterwards.

2.3. Relation Extraction and Analysis

2.3.1 Design

The relation extraction algorithm should be on top of the entity identification to figure out potential relationships between existing entities, while the relations should be dug out from sentences and function as edges in graph construction. As we already had sufficient extracted entities, we decided to perform a pattern extraction method to obtain the possible relations. Due to the insufficient number of good relations from the above method, the Open Information Extraction (OpenIE) in Stanford CoreNLP should be applied to obtain the draft version of relations from their corpus [10], since it more easily produced the relations in sentences. Compared to previous relation extraction methods we proposed in our proposal, the OpenIE should be easier to deploy and requires fewer resources.

By applying the CoreNLP package, the dependency parse should appear in each sentence, which stands for the initial semantic relation between words. With

dependency parse, we performed classical pattern matching on it to adopt common relations between entities.

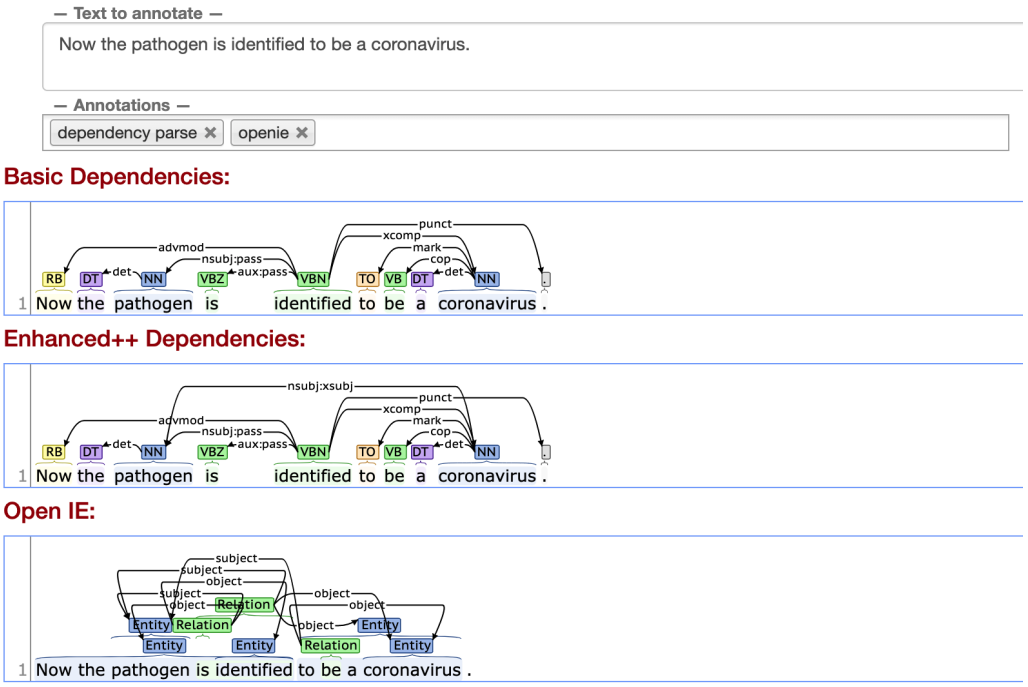

Figure 9 – Sample result of dependency parse and OpenIE

2.3.2 Implementation

2.3.2.1 OpenIE

After we filtered out 420,586 sentences with two entities (17.93% of whole data samples) as entity_2_training dataset, we used the StanfordOpenIE package in Python to extract open-domain relation triples, representing a subject, a relation, and the object of the relation. We used entity labeling to filter out such sentences whose entities were separately subject and object. Thus, we obtained 21,463 sentences with relations found between two entities by OpenIE.

Sentences	Subject	Object	Relation
SARS-CoV-2 is an appropriate name for the new coronavirus.	SARS-CoV-2	coronavirus	is appropriate name for
Host_genetic_factors are important determinants in tuberculosis (TB).	Host_genetic_factors	tuberculosis	are important determinants in
Viral_infection involves a large number of protein-protein interactions (PPIs) between virus and its host.	Viral_infection	PPIs	involves a large number of

Table 2 - Example outputs for OpenIE

However, some exceptions were detected which may lower the accuracy of relation extracted by OpenIE.

Sentences	Subject	Object	Relation
The classical falsehood is that vaccines cause autism.	vaccines	autism	cause
To establish molecular_assays, positive_controls are a prerequisite to ascertain specificity and sensitivity.	positive_controls	molecular_assays	establish

Table 3 - Example abnormal outputs needed to be filtered out

The above table revealed typically falsehood for relation extraction. OpenIE ignored the ***The classical falsehood*** on the first sentence and misunderstood the adverbial clause on the second one.

Therefore, we performed manually filtering for sentences and relations outputted by OpenIE in parts of the whole dataset, cleaning those with inaccurate relations.

Finally, through manually filtering and validation, we successfully obtained 1,133 confident OpenIE sentence results and 1,670 related entities, which served as primary content for knowledge graph construction and seed data for embedding models in the following steps.

2.3.2.2 Data Analytic

In order to further search for possible features which may affect relation extraction, we also performed some analytics in Python about both the original data samples (entity_2_training) and the output data samples by OpenIE.

Some analysis on those data samples are shown below:

i. The length of the sentences

	Mean	Standard Deviation	Minimum	Maximum
entity_2_training	118.17	55.89	4	2867
OpenIE	91.09	42.81	10	463

Table 4 - Statistics about the length of sentences in two datasets

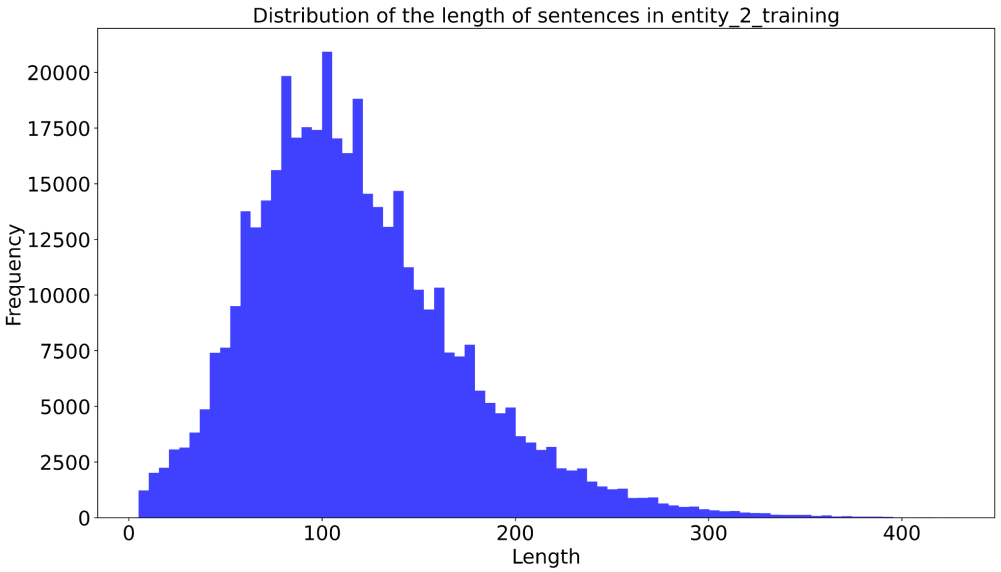

Figure 10 - Distribution of the length of sentences in entity_2_training

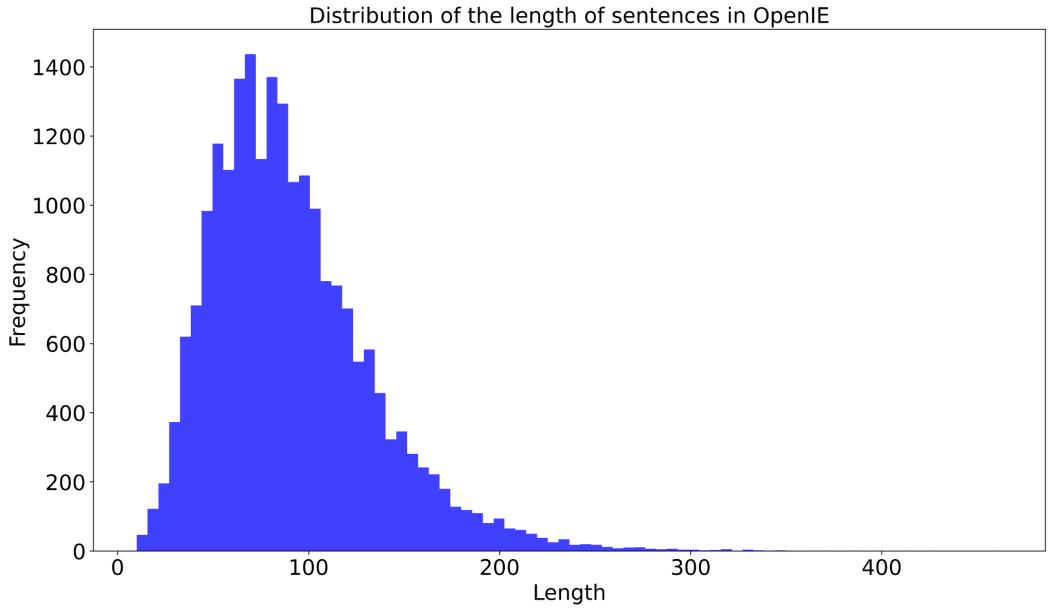

Figure 11 - Distribution of the length of sentences in OpenIE

It was revealed that in both entity_2_training and OpenIE, the number of sentences increased as the length reached around 100, but steadily decreased when the length exceeded 100. The range of the length of sentences was quite broad, which may result in difficulties for word embedding models to extract relations between entities. Especially, due to the special characters and errors in entity labeling, there were abnormal cases for exceptionally short meaningless sentences.

ii. The distance between two entities

Note that the distance refers to the number of characters between two entities.

	Mean	Standard Deviation	Minimum	Maximum
entity_2_training	35.28	34.19	0	822
OpenIE	26.49	19.60	1	197

Table 5 - Statistics about the distance between two entities in two datasets

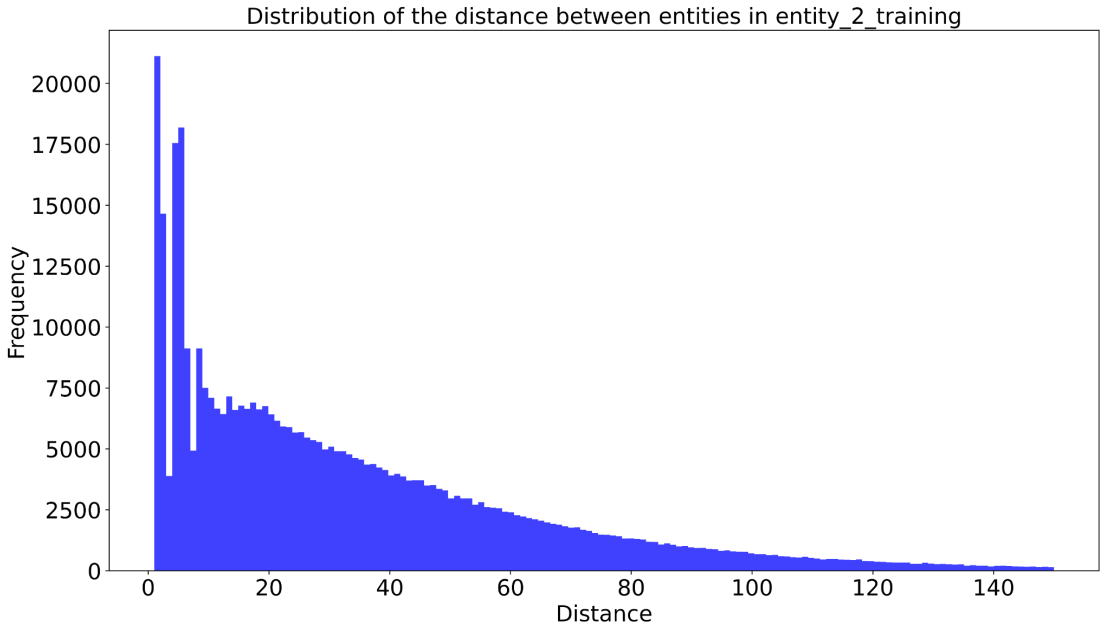

Figure 12 - Distribution of the distance between entities in entity_2_training

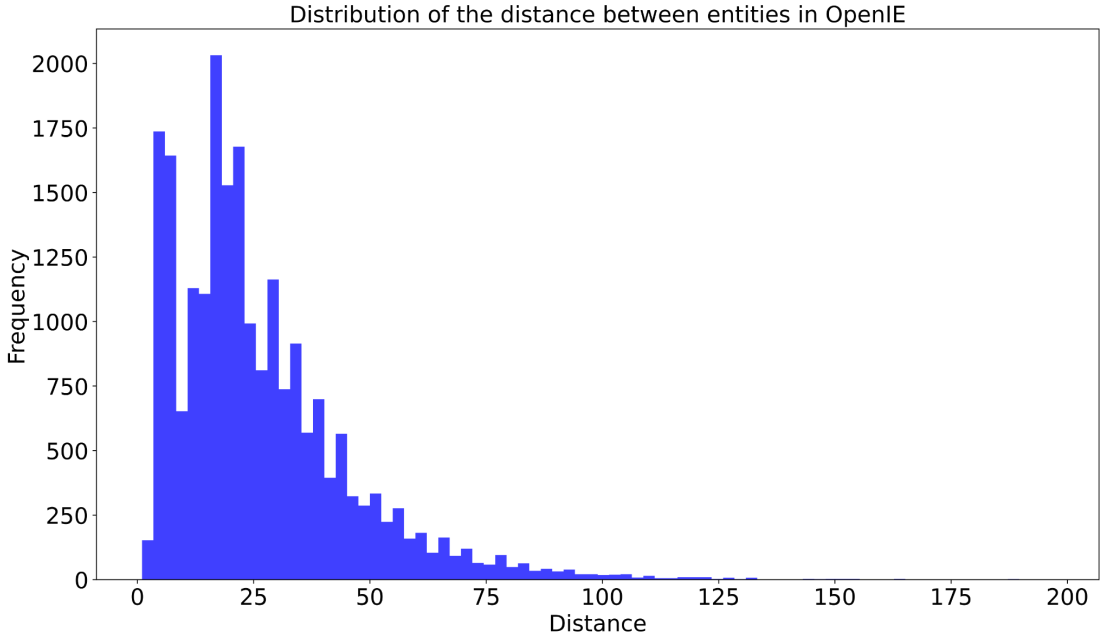

Figure 13 - Distribution of the distance between entities in OpenIE

One of the apparent features was that there were fewer sentences with the distance between entities less than five through the filtering procedure by OpenIE. The majority of sentences with relations had a distance around 25. One possible reason is that the distance less than five usually indicates that two entities had the relation of co-occurrence or even had similar meaning, which was less likely to be detected by OpenIE.

iii. Distribution of entities in sentences

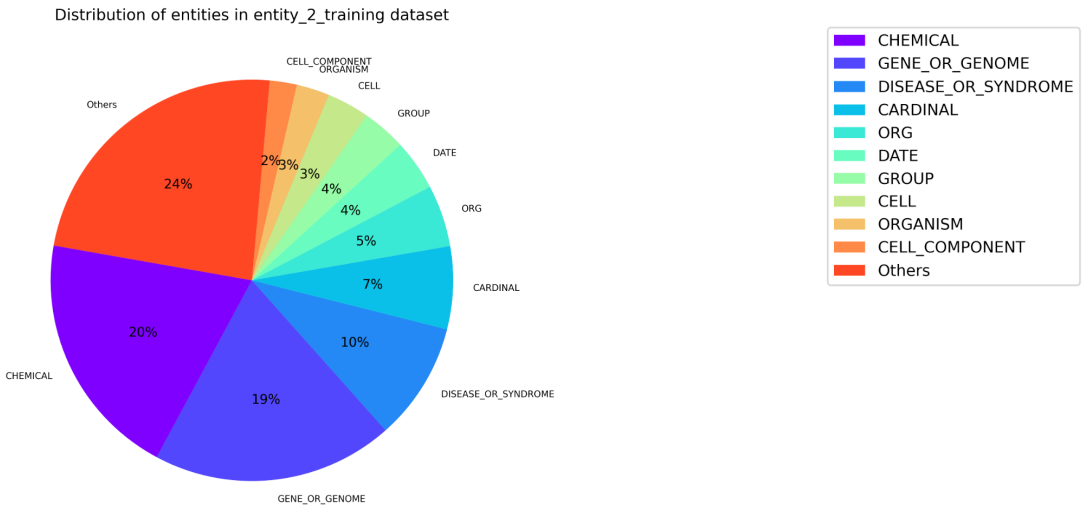

Figure 14 - Distribution of entities in entity _2_training

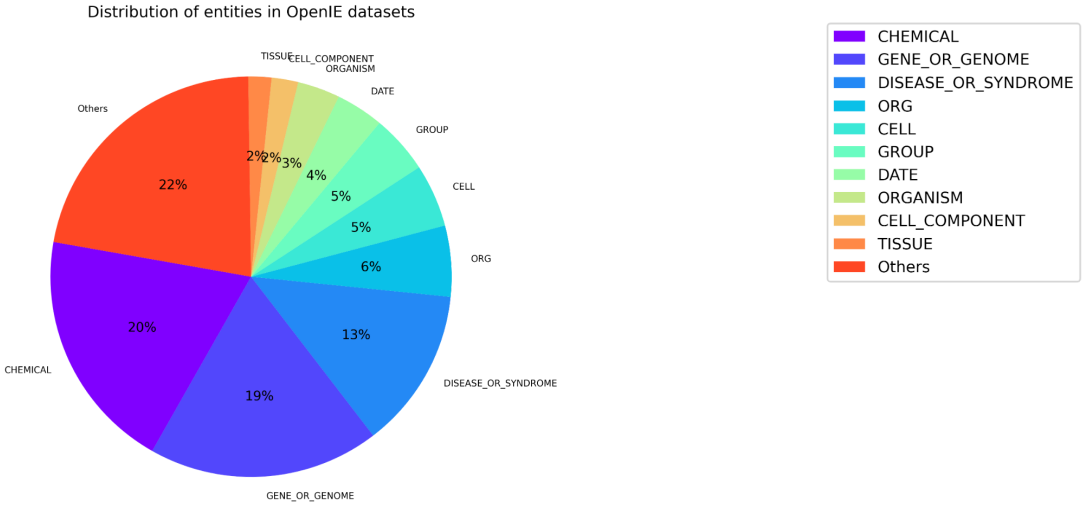

Figure 15 - Distribution of entities in OpenIE

There was no significant difference in the distribution of entities between the entity_2_traning and OpenIE data samples. Around half of entities belonged to ***Chemical***, ***Gene or Genome*** and ***Disease or Syndrome***.

2.4. Bootstrap for extension

2.4.1 Design

After standard relation extraction and manual validation, we obtained some highly confident relations as a result. However, for a knowledge graph that comes from a corpus about COVID-19 text, the number of relations should be much more to show the overall value. In this case, we adopted the idea from Professor Song to perform Bootstrap on the pattern in sentences, which utilizes the power of machine learning to find more potential relations from our processed data. The basic idea of Bootstrap is a repeating process that uses some high confidence data as seed to train a neural network model or computation function and applies the model or function to the overall dataset to gain more confident results.

We applied a CNN model (SNOWBALL) [11] and a Semi-Supervised Bootstrap with embedding (BRED) [12] respectively on our confident result, which worked as a seed to produce more.

2.4.2 Implementation

2.4.2.1 SNOWBALL

The structure of the CNN encoder of snowball mainly followed the CNN structure of Nguyen and Grishman (2015) [11]. This encoder used the word embedding matrix as input [11]. The CNN simply included a convolutional layer, a max pooling layer and a fully connected layer with dropout and softmax output [11].

The snowball experiment setting required a dataset with precise labels, i.e. precise manually annotations. Moreover, the dataset should be large enough to train the CNN encoder. From the paper, the only fitted dataset they found is FewRel. This dataset included about 100 different relations and 70,000 entities from Wikipedia [13].

After reappearing the Snowball CNN encoder training experiment with the FewRel dataset, we applied it on our dataset with our own manually labelling. However, as the property of our validated data differed from that of FewRel, there was little effect

in increasing the number of relations.

2.4.2.2 BRED

Different from SNOWBALL that apply CNN to the embedding matrix, this model simply performed statistical training on the similarity function of word embedding matrix [12], which held a good performance when the sentence format did not have huge variance.

As the model took embedding as input, we firstly passed our original dataset into the Gensim package to obtain the same format embedding of confident data and overall dataset. By applying the subset that holds most sentences under the same entity-pair to train, we got some expansion on our relation data, which would be more as iteration.

2.4.3 Result

As the attempt of bootstrap to produce high confidence data did not provide a satisfactory outcome, we decided to go back and adopt the extraction result from StanfordOpenIE package to be the resulting dataset in our knowledge graph. And the manually validated subset of it will be exported separately in our database. The accuracy of those results will be tested later.

2.5. Database

2.5.1 Design the Database Structure

After we obtained the extracted entities and relations from the corpus, we designed a database to store the essential information of it, which should be accessed by the web application to formulate the knowledge graph and support the searching function. In particular, instead of only reserving the nodes and edges for graph formation, the database should also store the metadata of the original sentence and document to provide a thorough understanding of the result in the knowledge graph.

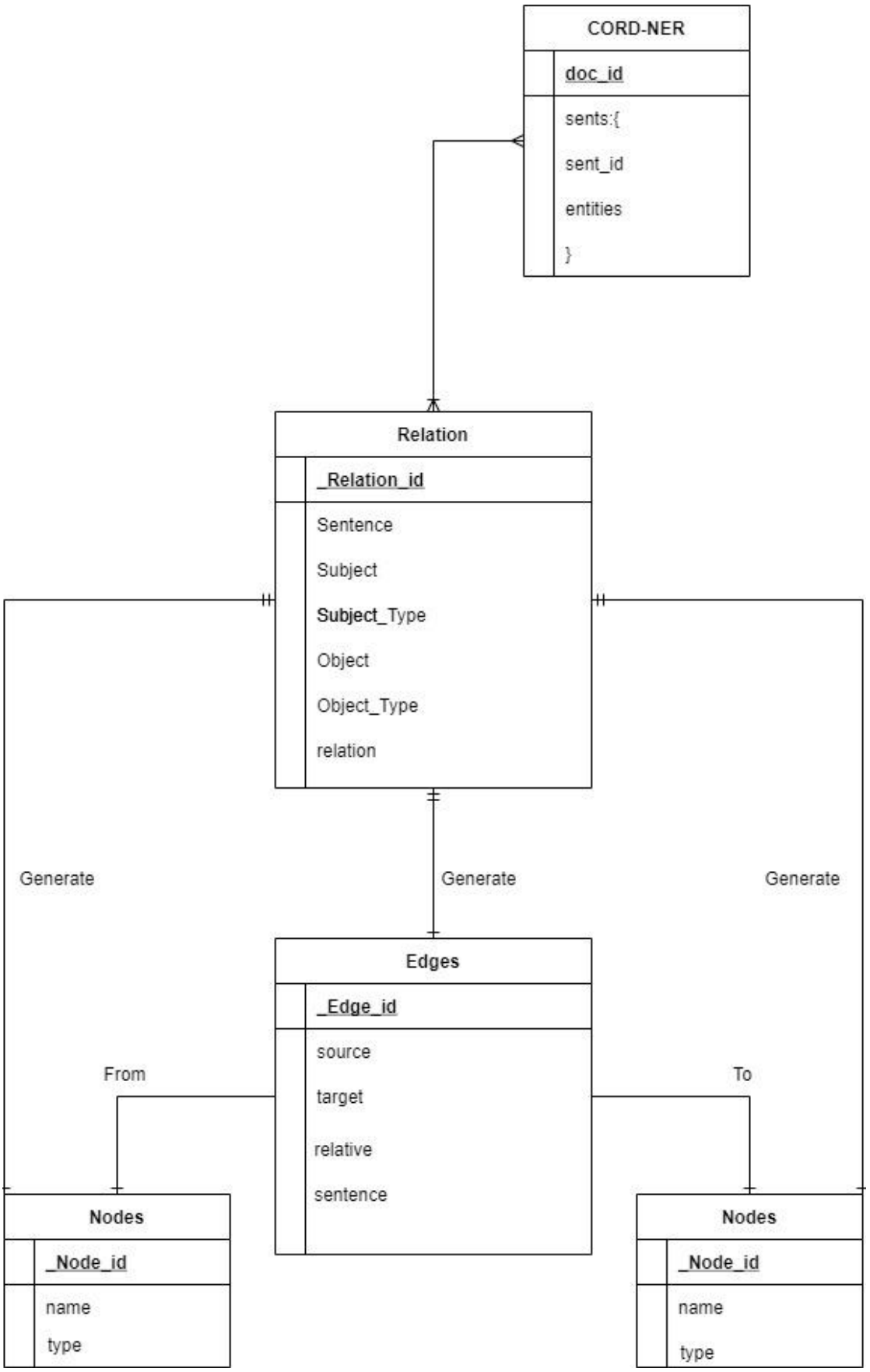

Figure 16 – Database Structure

2.5.2 Build the Database

Based on our designed structure, we chose MongoDB to deploy our databases as it did not require other cloud platforms to finish online tasks and provides great support to javascript. Apart from those reasons, the easy-use Graphic User interface also contributed to the selection. After completing the configuration of the database, we created an API for the web application to establish a connection, and the API and connection were now under testing with the draft of the knowledge graph. This database was used to store the edges and nodes in the graph and provide metadata for searching results on web applications.

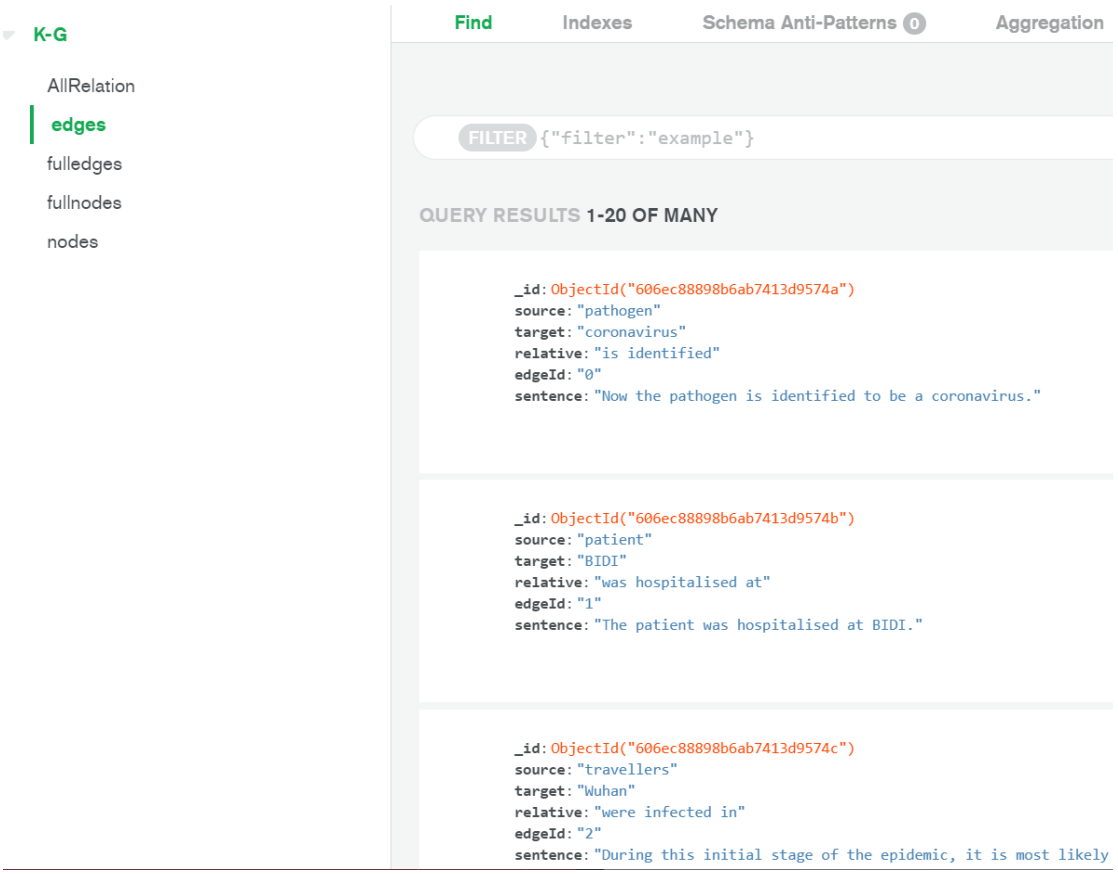

Figure 17 – The deployed MongoDB

2.6. Web Application

2.6.1 Design the Web Application

A user-friendly user interface should be used to visualize our knowledge graph, which allows users to perform simple functions and provide related information in a proposed format.

We provide different layouts of the graph: one is the concentric circle layout, and another is the cluster-based layout. Users can switch the picture by their preference.

The minimap at the bottom left corner should provide an overview of our graph to the user. Moreover, it should show which part of the knowledge graph the user's currently watching.

We used different colors to represent different types of entities, which can provide users an intuitive view of the connection between entity types. For example, in the knowledge graph, blue nodes indicate that these entities are of chemical type.

There is an infobox to provide users with extra detailed information. If users click on an edge, the infobox will pop up with the relative data of the connected nodes. The infobox should contain the name and relation of two entities and the original sentence where we extract this relationship.

In addition, a toolbar was set in the upper right corner that has combined a series of functions for users to interact with our graph: Search, Zoom In, Zoom Out, and Fisheye functions. We designed a simple search function to allow users to make queries about our knowledge graph. After the user searches for a word, the window of our knowledge graph should automatically move to the corresponding node and highlight it for the user. The user can use the zoom in and zoom out buttons to adjust the ratio of the knowledge graph in the view window. If users want to view a particular area closer, they can use the fisheye function to partially enlarge the chart.

Here is a video link showing all the basic functions: https://youtu.be/PXo2FZuV7BA

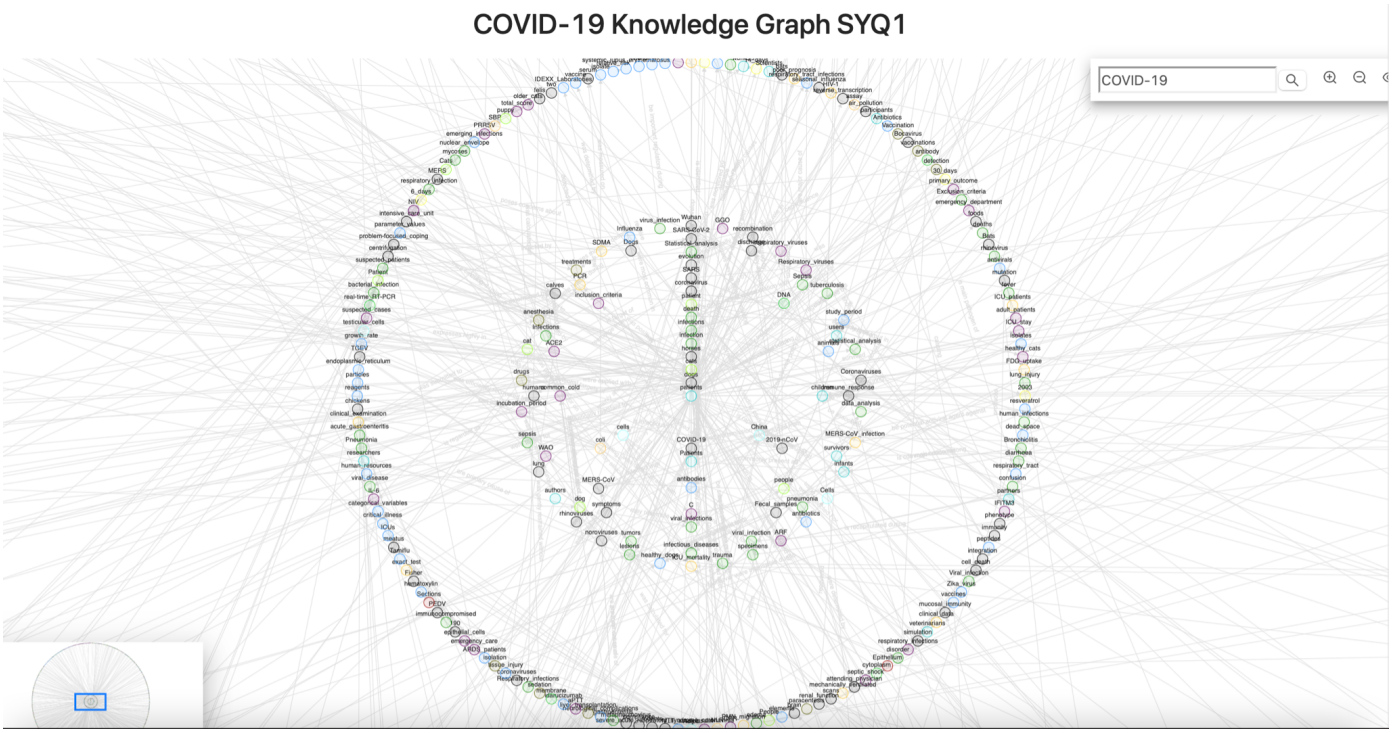

Figure 18 – Concentric circle layout of the knowledge graph

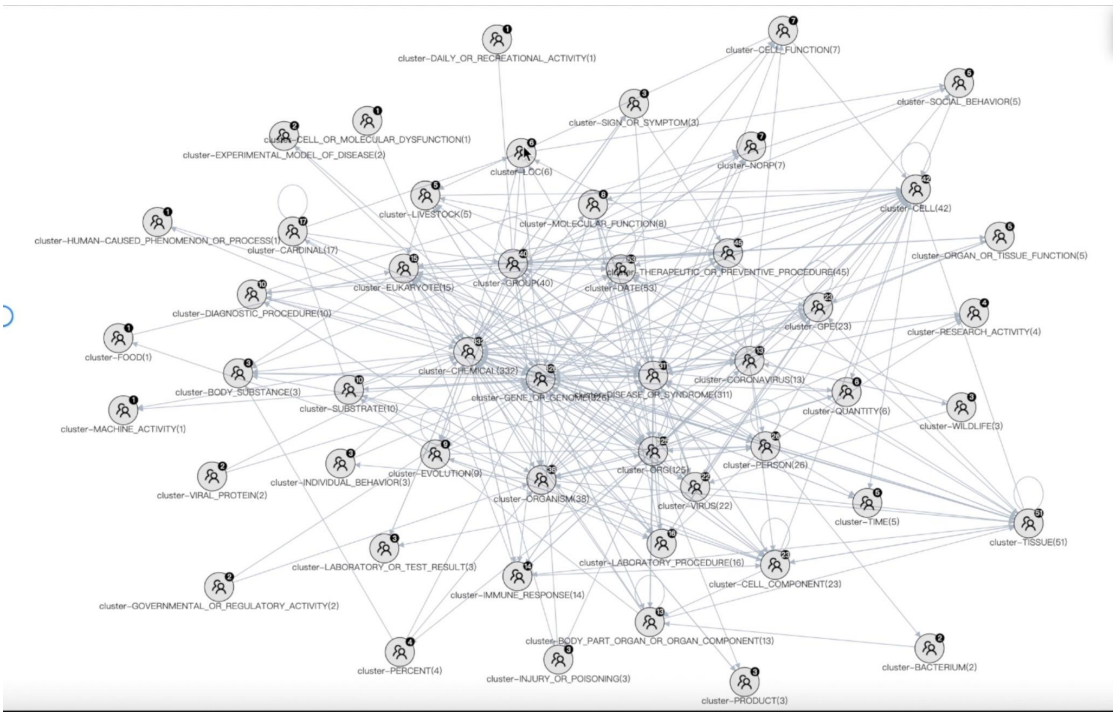

Figure 19 – Force layout of the knowledge graph

2.6.2 Develop the Web Application

2.6.2.1 Frontend Development

Based on our design, we used reactJS and Javascript coding to develop the user interface. With the G6-based React package Graphin, we can easily create a user-friendly user interface that contains different layouts and interactive functions of our knowledge graph.

We used Graphin's built-in concentric circle layout and G6's built-in cluster basic layout. For different layouts, we provided different data structures to the react component. We modified the setting in order to provide a better vision in our user interface. Graphin would calculate and produce the graph automatically. The cluster base layout used the ref from Graphin to access the G6 pre-set layout.

We used nodes' colors to represent the entity type, we designed and applied a color dictionary to assign nodes to different colors by their entity type. In order to show the minimap of our knowledge graph, we directly used the built-in component in Graphin. To implement the information box that displays detailed information on the edge, we rewrote the content in the tooltip component to show the information in our designed text format.

For the toolbar that combines search, zoom in, zoom out, and fisheye functions, we rewrote the usage of the component to make it fit the toolbar framework. For the search function, the search content changed with the user's input. Each time it changed, the search API would find the corresponding nodes in the whole knowledge graph. Zoom in, zoom out and fisheye function were the built-in components in Graphin. We can embed those functions into the toolbar simply. However, we adjusted the way that the fisheye opened and closed by using the button in the toolbar to switch the status.

2.6.2.2 Backend Development

We used NodeJS to develop our API that returns the required data in the backend.

For concentric circle layout, it required an array that contained "nodes" and "edges".

The "nodes" was an array that included the JSON format of every node's id and type. The "edges" was an array that included the JSON format of every edge's source, target, relation, and original sentence. It's the direct format of the data we stored in the database, such that we aggregated all the data into the required data format simply.

Figure 20 – The return data format of concentric circle layout

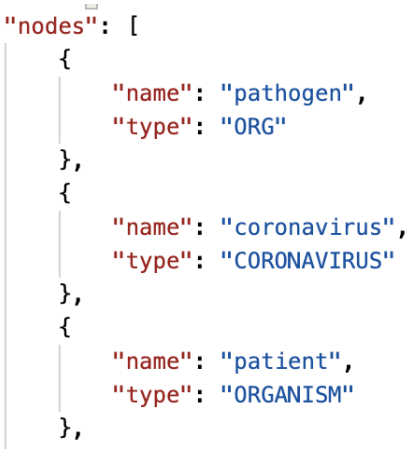

Figure 21 – The return data format of nodes

Figure 22 – The return data format of edges

For cluster-basic layout, besides the "nodes" and "edges" required above, it also required the "clusters" and "clusterEdges". The "clusters" was an array that included the JSON format of every cluster's id, the number of nodes contained, and an array of nodes under the current cluster. The "clusterEdges" was an array that included the JSON format of every edge among clusters, which requires source, target, and the number of edges between those two clusters. We had to regroup and calculate the data from the dataset to achieve this data structure.


```
"statusCode": 200,
"message": "Data fetched Successfully",
"result": {
    "nodes": [⋯
    ],
    "edges": [⋯
    ],
    "clusters": [⋯
    ],
    "clusterEdges": [⋯
    ]
}
```


Figure 23 – The return data format of clusters-base layout

For "clusters", we first grouped the nodes according to their types and then calculated the total number of nodes under each group. We used entity type as the cluster name and id.


```
"clusters": [
    {
        "id": "ORG",
        "sumTot": 125,
        "node": [
            {
                "name": "pathogen",
                "clusterId": "ORG"
            },
            {
                "name": "public_health_viewpoint",
                "clusterId": "ORG"
            },
            {
                "name": "acute_phase",
                "clusterId": "ORG"
            },
```


Figure 24 – The return data format of clusters

For "clusterEdges", since we were using the node's name as the source and target of each edge, and the "edges" collection did not include the type of it, such that we

needed to get the source and target type first. We can join the "nodes" and "edges" collection together to get the result. Afterward, for each combination, it counted the number of source type and target type.


```
"clusterEdges": [
    {
        "source": "ORG",
        "target": "CORONAVIRUS",
        "sumTot": 5
    }
```


Figure 25 – The return data format of clustersEdges

3. Testing

Throughout the implementation, we performed tests on most of the functions and systems when they were done. After we finished building the whole system, we conducted integration testing.

3.1. Test the knowledge graph

To test the knowledge graph, we checked the following:

- Efficiency
- Operability
- Export rate

To test the accuracy, we utilize an evaluation model proposed in [14], and we also perform tests manually on a small scale.

For the manual tests, the overall approach is to use some certain readable literature to do a small-scale test to examine the accuracy by ourselves.

After the overall building, we examine the ratio between number of exportable data and the number of raw data for reference.

3.1.1 Test the dataset

In particular, we examine the result dataset to see whether there is a bias distribution between it and the original one. Then the some subsets are manually examined to see the score and performance of overall resulting data.

3.1.2 Test the extraction result

Instead of testing the accuracy of model performance, we chose to compute the accuracy of the final result to represent the precision of the knowledge graph.

As this is a result from Natural Language Processing, it is hard to find a fitted tool or standard to simply state the confidence score of it. We decided to randomly produce several sentence subsets from the result and omit the extracted relation to make it a raw sentence with entities specified. These subset would be given away to volunteers

from different backgrounds to perform manually pattern labeling, which will be compared with our result to produce an accuracy score.

3.2. Test the database

The testing of the database was mainly conducted by retrieving large amounts of data at one time and performing related operations(join, map, query..) to show the reliability of the database. Besides, we also checked the completeness of the data within the database.

3.3. Test the user interface

The user interface is directly interacting with the user, so we mainly care about the usability of it.

Here is our testing Checklist [15]:

1. Style conformity.
2. Adaptability. Check how all UI elements are displayed on screens of different sizes and in portrait and landscape orientations.
3. Compliance with standards.
4. The functionality used.
5. Check fields and standard items.
6. Information elements. Check how error messages, notifications, and other elements related to this category look and are positioned.

3.3.1 User study and volunteer test

To test the user interface, we invite some volunteer testers from both technical and non-technical backgrounds to try out our web application. After that, some feedback and questions were collected for us to improve the user experience of the interface.

4. Evaluation

As our projects consist of multiple parts that work separately, we choose to evaluate those parts one by one to see whether they fulfil our objectives, and then perform overall evaluation as a whole system.

4.1 Summary

Objectives	Failed	Partially Succeeded	Succeeded
Integrate and modify the existing algorithms that extract medical text entities and link them, extract potential relations between entities, and use machine learning algorithms to search for further relations.		√	
Develop a database that contains the extracted knowledge related to COVID-19, which is integrated into an ontology in the form of a knowledge graph.			√
Develop a web application that contains a basic knowledge graph that visualizes medical text entities and their potential relationships in various types of representations and enable querying through our knowledge graph according to the user's instructions.			√

Table 6 - Summary of the evaluation on objectives

Among the three objectives, we encountered much more difficulties in the first one. We failed to search for further potential relations by bootstrapping with a satisfying accuracy.

4.2 Evaluate the model performance

The models we use in our project are in bootstrap session to enrich our relation data, which determines the size of the knowledge graph.
For both models (SNOWBALL and BRED) the performance is not satisfactory to add confident data. Especially in the case of SNOWBALL, due to the characteristic of CNN model, the insufficient number of input validated data resulting in high error in prediction and bringing no new confident relation to our dataset.

As for the BRED, we predict 6000+ sentences under the same entity-pair with 10 seeds, which bring a result with the highest confidence score at 0.52 while most results are at 0.30 level. The low successful rate is to be discussed to find the possible reason.

```
instance: Touro College  Israel  score:0.33333333333333326
sentence: Her bachelor 's degree apparently came from <LOC>New York City</LOC> 's <ORG>Touro
College</ORG> branch in <LOC>Israel</LOC> , an institution of lesser standing , at a time when such
foreign branches were not under government supervision , <ORG>Army Radio</ORG> reported .
pattern_bef: City 's
pattern_bet: branch in
pattern_aft: , an
passive voice: False

instance: Citibank       Seoul    score:0.26463670285457497
sentence: After questioning a dozen people overnight , prosecutors said Thursday they had narrowed
the allegations to a report that local hotel owner <PER>Lee Chang-soo</PER> was holding dlrs 132
million in false-name accounts in a <ORG>Citibank</ORG> branch in southern <LOC>Seoul</LOC> .
pattern_bef: in a
pattern_bet: branch in southern
pattern_aft: .
passive voice: False

instance: Ipswich        England  score:0.25142535646182573
sentence: Though many of the stores set for closure had been virtually cleared , there were bargains
to be had -- 38-year-old <PER>Teresa Stewart</PER> managed to buy a box filled with clothes for two
pounds ( 2.9 dollars , 2.1 euros ) after queueing for nearly half an hour outside the <ORG>Ipswich</
ORG> branch in eastern <LOC>England</LOC> .
pattern_bef: outside the
pattern_bet: branch in eastern
pattern_aft: .
passive voice: False

instance: Citibank       Athens   score:0.24601727063039758
sentence: A bomb exploded outside a <ORG>Citibank</ORG> branch north of <LOC>Athens</LOC> early on
Monday , causing significant damage but no apparent injuries , <MSC>Greek</MSC> police said .
pattern_bef: outside a
pattern_bet: branch north of
pattern_aft: early on
passive voice: False

instance: Livent Toronto  score:0.17138302112395376
```

Figure 26 – The confident score of bootstrap result

4.3 Evaluate the extraction result

We produce 4 subsets of result with size 50 to perform volunteer test, the performance are stated below:

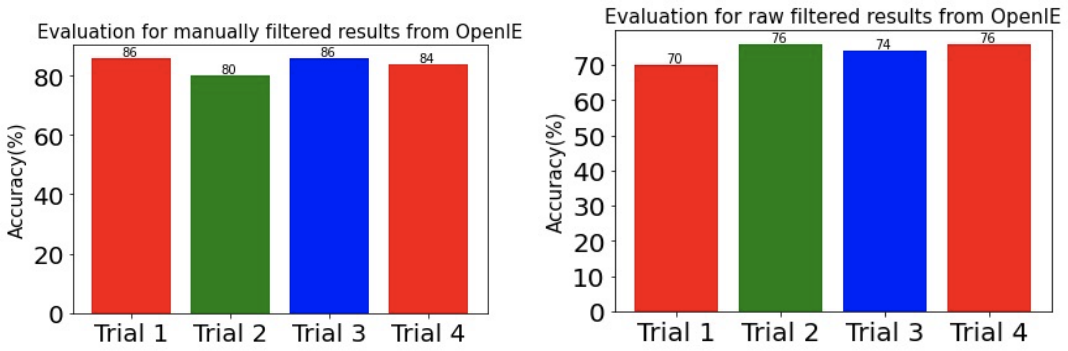

Figure 27 – The Accuracy of extraction result

	Manually filtered result from OpendIE	Raw result from OpenIE
Overall accuracy	84%	74%

Table 7 – The Overall Accuracy of extraction result

As the full set from OpenIE is used to be the resulting information in our knowledge graph due to the unsatisfactory performance bootstrap, the accuracy 74% is the average accuracy of our relation extraction outcome while the manually validated data can achieve a higher score around 84%. Though the raw data from OpenIE can not achieve the level of manually validated one, it is still a good result for our original dataset.

4.4 Evaluate database

The database acts as a connector between web frontend and the Natural Language Processing pipeline, so the ability of maintaining data safely and transporting the data steadily is essential.

The overall database performance is beyond our expectation as it is able to afford a large amount of requests at a time and allow our web application to access it frequently while fetching numerous data without losing.

4.5 Evaluate the User Interface

To do the user evaluation, we asked 5-10 volunteer testers to use our web application and tell us about their experiences. These testers test our web applications with

different browsers(mainly Safari and Google Chrome). And we recorded the bugs raised by the testers and their suggestions. The following is the summary of the tests results:

- In total: The knowledge graph is professional but too academic. The nodes and edges are too many, so it is hard to find specific nodes or edges.
- UI: The layout and the style conformity is good. Basic tools are all set. But some tools are lagging, especially the highlight and fisheye.
- Search: Users cannot search relations or paths among nodes. And the case of the search keyword and the entity must be the same, otherwise the search results will not be displayed.
- User guide: Users did not know what different colors of nodes mean. They have no clue about searching. Some users need a tutorial to use the web application.
- Suggestions: Add infobox for nodes, which can appear after clicking a node, and explain the meaning of nodes and edges. Enlarge node and relation when highlighted. Add more hints in search, tutorials.

To summarize, most testers have a high degree of acceptance to our web APP, though there are still some bugs. For the UI, the showed nodes and edges are too many, but can fully present the entire knowledge graph. The style of the graph and the style conformity of the graph and toolbar is good. User hints and tutorials are needed. Functions cannot work together and have some lags. The search function is case sensitive and not comprehensive enough.

5. Discussion

In this section, we will outline a brief summary of the steps we have taken to achieve the objective. Especially, we will illustrate some of the challenges we faced and solutions we came up with to resolve the problems.

5.1 Data Preprocessing

Initially we chose the CORD-NER dataset [9]. Although it had utilized relatively efficient NER methods, there were inevitable errors for the annotation. Special characters such as brackets were mistakenly annotated as entities, which contributed to the errors for the future steps. In order to resolve this problem, we first observed many abnormal data samples and cases, and then removed special brackets and symbols for reference annotations in articles in the preprocessing steps. Moreover, we manually filtered some parts of samples by ourselves. Though performed many processing steps, this greatly decreased the accuracy of information rendered on our final prototype.

5.2 Relation Extraction by OpenIE

OpenIE was utilized to extract the relations in sentences efficiently. However, we encountered some major problems. In addition to the one mentioned in section **2.3.2.1**, we also observed that there were multiple pairs of tuples including subject, object and relation for some sentences. For instance, for one example sentence:
The final_size of an outbreak is greatly affected by transmission_events early during the outbreak process.
The results by OpenIE for ***final_size*** and ***transmission_events*** include '***is affected early by', 'is greatly affected early by', 'is affected by'*** and ***'is greatly affected by'***.
It was necessary for us to pick out the most suitable one. For such a large dataset, it was hardly possible for our group to manually filter all of them. Therefore, we first randomly chose 5,000 sentences with multiple results. During the process of manually filtering, it was revealed that most of the suitable relations are longest with

the most detailed information. Finally we adopted the relation with the largest length, despite some errors for some exceptional cases.

5.3 Bootstrap for Further Extraction

As mentioned in Evaluation, we found it difficult to get satisfying results on those bootstrap models. Based on the analytics in section **2.3.2.2**, we found some possible factors which contributed to the failure. First, the distribution of the length of sentences was quite large, compared to the sample data on the original models. Secondly, the distribution of distances of entities in sentences was also quite sparse, especially in those bootstrap models, where all the sentences and the positions of the entities were of similar type. Thirdly, as we adopted the CORD-NER dataset to obtain an entity recognition with more specific entity type related to COVID-19, the error propagation from NER result to our machine learning driven model could be a problem.

Finally, although the total number of sentences seemed quite adequate (around 420,586), there were inadequate numbers for each pair of entities.

Entity pairs	Number	Ratio
('CHEMICAL', 'CHEMICAL')	21845	5.19%
('GENE_OR_GENOME', 'GENE_OR_GENOME')	20284	4.82%
('GENE_OR_GENOME', 'CHEMICAL')	16225	3.86%
('CHEMICAL', 'GENE_OR_GENOME')	15524	3.69%
('CHEMICAL', 'DISEASE_OR_SYNDROME')	8309	1.98%
('GENE_OR_GENOME','DISEASE_OR_SYNDROME')	7934	1.89%

Table 8 - Top number of data samples for entity pairs

Compared to the original model and application [11], it was suggested the data size

for each entity pair should be around one million. The lack of data resulted in the low accuracy of the model.

5.4 User Interface Improvement by Feedback

Through the collected opinions, we will further improve our user interface. We mainly divided the feedback into three categories: First, the system problem. Second, suggestions on interface design. Third, demand for certain interactive functions with our knowledge graph.

Our web page takes time when fetching data. Function such as fisheye has a delay problem. Those mentioned above are mainly due to the relatively large amount of data, and the calculation of the function takes time. For example, fisheye needs to calculate the center node and the insignificant node during the movement of the fisheye, therefore it can enlarge the center node and bounce off other nodes. In addition, some of the functions cannot be used at the same time, such as the fisheye and drag-and-drop functions. This is a conflict problem among the components in the package we used. The solution is to improve the functions and algorithms inside the package, and we will not focus on that.

During the testing, some testers felt that we could improve our user interface design to give users a better experience. We will try to emphasize the important information in the future development.

Testers also mention that they want to have more interactive functions with our knowledge graph. We will try to provide more interactive functions in the future development.

6. Conclusions

6.1 Project Summary

In this project, we constructed a knowledge graph of COVID-19 medical text and represented it by a web application.

For the construction of our knowledge graph, we adopted the existing named entity recognition result with the updated COVID-19 type recognition as our dataset. After comparing the performance of different standard relation extraction models, we decided to use OpenIE as our draft model and then filter the meaningful result manually to build the primary result dataset. Finally bootstrap strategy was tried to enrich our graph relation dataset. As for the web application of our knowledge graph, we implement the different layout of the visualization and provide a series of functions for users to interact with our knowledge graph. Users can simply view the entire knowledge graph and gain the information they need.

From our experience, previewing the dataset is very important. Since the medical text is formal and academic, the relation extraction algorithm cannot locate the correct relation position. Furthermore, neither the original corpus nor the NER dataset we used do not hold one standard format like the training dataset we got in touch with in class and contains many disturb outputs. These are also reasons that the accuracy of our tried model is low. Furthermore, if the user interface wants to contain personalized design, it’s better to use a react package that contains more basic components.

6.2 Future Plan

6.2.1 Better User Interface

We will enlarge the node and show all the relevant information boxes on edges when the node is selected. We aim to reduce the total number of clicks for the user to achieve the information they are interested in.

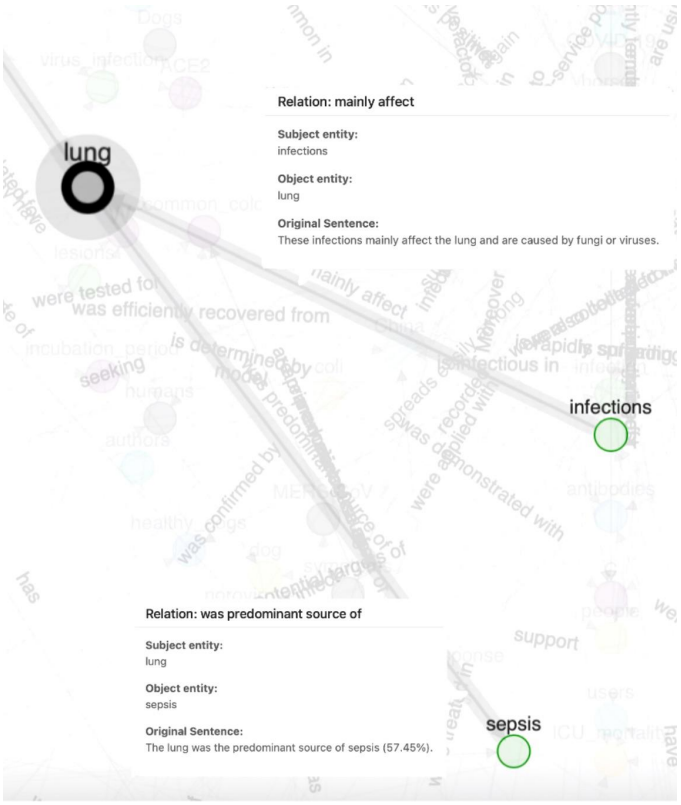

Figure 28 – Expected User Interface when the user click on node "lung"

6.2.2 More interactive tools for user

We will consider applying the breadth-first search (BFS) or depth-first search (DFS) algorithm to allow the user to find the shortest path between two nodes. Furthermore, we will highlight the whole path in our knowledge graph to show the result.

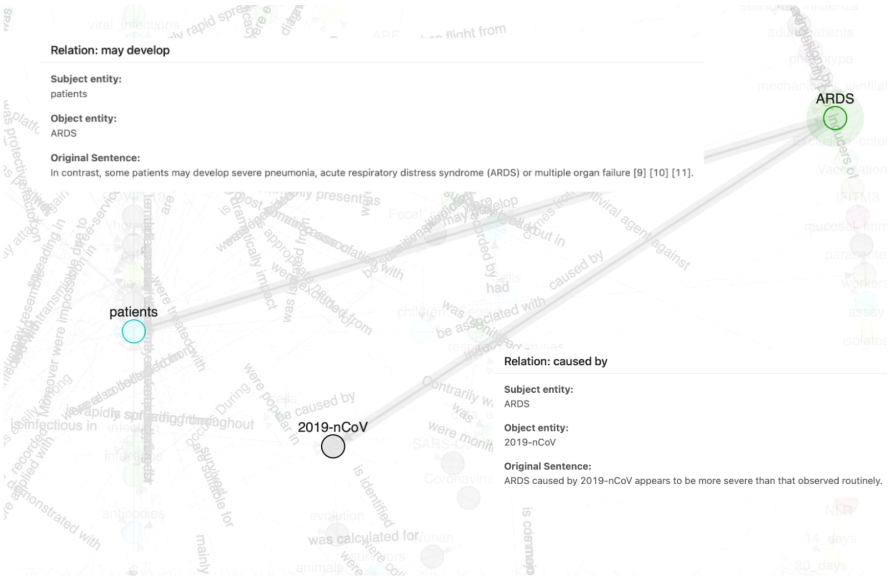

Figure 29 – Expected User Interface when the user select "patients" as source and "2019-nCov" as target

6.2.3 Enrichment and Auto Updating

It is better if the knowledge graph's information is up to date. We will try to modify and utilize a crawl function for accessing the updated dataset in the COVID-19 medical text website. And then enable our implementation on updating the knowledge graph with the latest COVID-19 open dataset.

7. Project Planning

7.1. Distribution of Work

Task	CHAN	TANG	WONG	ZHANG
Do the literature survey	○	○	○	●
Design the goals	●	○	●	○
Learning related to knowledge graph	●	●	●	●
Discuss the technologies to be used	●	●	●	●
Design entity extraction algorithms	○	○	●	●
Design relation extraction algorithms	●	●	○	○
Design the knowledge graph structure	●	●	●	●
Design the user interface	○	○	●	○
Design the database	●	○	●	○
Design the bootstrap algorithm	●	●	○	○
Data preprocessing	●	●	●	●
Implement Entity and relation extraction	●	●	●	●
Build the database	●	○	●	○
Build the knowledge graph	●	●	●	●
Build the user interface	●	○	●	○
Test the knowledge graph	○	●	○	○
Test the user interface	○	○	●	○

Test the search function	○	○	○	●
Perform integration testing	○	○	●	●
Debugging	●	●	○	○
Write the proposal	●	●	●	●
Write the monthly reports	●	●	●	●
Write the progress report	●	●	●	●
Write the final report	●	●	●	●
Prepare for the presentation	●	●	○	○
Design the FYP poster	○	○	●	●

● Leader ○ Assistant

7.2. GANTT Chart

Task	July	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr
Do the literature survey										
Design the goals										
Learning related to knowledge graph										
Discuss the technologies to be used										
Design entity extraction algorithms										
Design relation extraction algorithms										
Design the knowledge graph structure										
Design the user interface										
Design the database										

Find existing bootstrap approaches										
Data preprocessing										
Entity and relation extraction										
Build the database										
Build the knowledge graph										
Build the user interface										
Test the knowledge graph										
Test the user interface										
Test the search function										
Perform integration testing										
Debugging										
Write the proposal										
Write the monthly reports										
Write the progress report										
Write the final report										
Prepare for the presentation										
Design the FYP poster										

8. Required Hardware & Software

8.1. Hardware

Development PC:	MacBook with macOS
Minimum Display Resolution:	1920 * 1080 with 16-bit color
Server PC:	PC with 512GB hard drive

8.2. Software

MongoDB Atlas	Database
Python, Javascript, NodeJS	Programming languages
GitHub	Co-work platform

9. References

[1] A. Singhal. (May 2012). Introducing the knowledge graph: things, not strings. [Online]. Available: https://blog.google/products/search/introducing-knowledge-graph-things-not/

[2] Daniel D.F. (April 2020).COVID-19 Knowledge Graph: a computable, multi-modal, cause-and-effect knowledge model of COVID-19 pathophysiology. [Online]. Available: https://www.biorxiv.org/content/10.1101/2020.04.14.040667

[3] J. Devlin, M. W. Chang, K. Lee, K. Toutanova. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. [Online]. Available: https://arxiv.org/pdf/1810.04805.pdf

[4] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, H. S. Chan and J. Kang. (2019).BioBERT: a pre-trained biomedical language representation model for biomedical text mining. [Online]. Available: https://arxiv.org/pdf/1901.08746.pdf

[5] I. Beltagy, K. Lo and A. Cohan. (2019). SciBERT: A Pretrained Language Model for Scientific Text. [Online]. Available: https://arxiv.org/pdf/1903.10676.pdf

[6] L.F. Song, Y. Zhang, D. Gildea, M. Yu, Z.G. Wang and J.S. Su. (2019). Leveraging Dependency Forest for Neural Medical Relation Extraction. [Online]. Available: https://arxiv.org/pdf/1911.04123.pdf

[7] W. Nelson, M. Zitnik, B. Wang, J. Leskovec, A. Goldenberg and R. Sharan. (2019). To embed or not: network embedding as a paradigm in computational biology. *Frontiers in genetics*, *10*, 381.

[8] L. L. Wang, K. Lo, Y. Chandrasekhar and R. Reas. (2020). CORD-19: The Covid-19 Open Research Dataset. [Online]. Available: https://arxiv.org/pdf/2004.10706.pdf

[9] X. Wang, X. C. Song, Y. J. Guan, B. Z. Li and J. W. Ha. (2020). Comprehensive Named Entity Recognition on CORD-19 with Distant or Weak Supervision. [Online]. Available: https://arxiv.org/pdf/2003.12218.pdf

[10] C. D. Manning. et al. (2014) The Stanford CoreNLP Natural Language Processing Toolkit. [Online]. Available: https://www.aclweb.org/anthology/P14-5010.pdf

[11] Nguyen, T. H., and Grishman, R. 2015. Relation extraction: Perspective from convolutional neural networks. In *Proceedings of the Workshop on Vector Space Modeling for NLP*, 39–48.

[12] David. Batista, Bruno. Martin. (2015). Semi-Supervised Bootstrap of Relationship Extractors with Distributional Semantics. [Online]. Available:

https://www.aclweb.org/anthology/D15-1056.pdf

[13] T.Y. Gao, X. Han, R.B. Xie and Z.Y.Liu. (2019). Neural Snowball for Few-Shot Relation Learning. [Online]. Available: https://arxiv.org/pdf/1908.11007.pdf

[14] J. Gao, X. Li, Y.E. Xu, B.Sisman, X.L. Dong, J. Yang. (July 2019). Efficient Knowledge Graph Accuracy Evaluation. [Online]. Available: https://users.cs.duke.edu/~jygao/KG_eval_vldb_full.pdf

[15] R.Telvak. "A Lite Checklist for UI (User Interface) Testing" Lvivity. https://www.seobility.net/en/seocheck/.

10. Appendix A: Meeting Minutes

10.1. Minutes of the 1st Project Meeting

Date: July 17th, 2020

Time: 22:00

Place: Via Zoom meeting, online.

Present: Professor Song Yangqiu,
and CHAN Tsz Ho, Tang Yutian, Wong Pui Ying, Zhang Liangwei

Absent: None

Recorder: Wong Pui Ying

1. Learning about Knowledge Graph(K-G)

   1.1 Learned the concept and development of Knowledge Graph.

   1.2 Introduced the recent approach of Knowledge Graph.

2. Discussion items

   2.1 Discussed about the functions and objectives of our project towards Knowledge Graph.

   2.2 Discussed the potential improvement against the existing Knowledge Graph.

   2.3 Professor Song demonstrated a couple of algorithms that can construct and lie out the Knowledge Graph.

3. Goals for the coming weeks

   3.1 Have more solid and fundamental comprehension to K-G building and K-G representing according to the information provided by Professor Song.

   3.2 Try out algorithms that are commonly used in the field of K-G.

   3.3 Form ideas and objectives to achieve in the new system/software.

4. Meeting adjournment

   The meeting was adjourned at 23:15.

10.2. Minutes of the 2nd Project Meeting

Date: Aug 27th, 2020

Time: 15:00

Place: Via Zoom meeting, online.

Present: CHAN Tsz Ho, Tang Yutian, Wong Pui Ying and Zhang Liangwei

Absent: None

Recorder: CHAN Tsz Ho

1. Approval of minutes

   The minutes of the last meeting were approved without amendment.

2. Report on progress

   2.1 All group members have studied materials provided by Professor Song.

   2.2 All group members have learnt the concept about construction and representation of Knowledge Graph(K-G) with some novel implementation were mentioned and recorded.

   2.3 All group members have thought about our own objective and idea to accomplish in this project.

3. Discussion items

   3.1 Discuss the choice of algorithms to modify and adopt in this project and whether to integrate new algorithms.

   3.2 Decide to adopt additional features to the existing approach of K-G, including mind map and search engine.

   3.3 Discuss the aiming population of this project.

   3.4 Discuss the possible questions to ask Professor Song.

4. Goals for the coming week

   4.1 All group members will read the papers provided by Professor Song's email.

   4.2 All group members will need to try the flow of novel algorithms provided by Professor Song.

4.3 Draft the proposal

5. Meeting adjournment

The meeting was adjourned at 17:00.

10.3. Minutes of the 3rd Project Meeting

Date: Sep 13th, 2020

Time: 15:00

Place: Via Zoom meeting, online.

Present: CHAN Tsz Ho, Tang Yutian, Wong Pui Ying and Zhang Liangwei

Absent: None

Recorder: Tang Yutian

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 All group members have studied related materials and merged into the proposal.

2.2 All group members have tried out some algorithms used in the field of K-G separately.

3. Discussion items

3.1 Share the tried algorithm to each other about how it works and what role it plays in K-G.

3.2 Discuss about the feasibility of the event extraction and event-relation extraction.

3.3 Discuss the choice of search method applied on K-G.

4. Goals for the coming week

4.1 Check the latest update about COVID-19 and Knowledge Graph.

4.2 Finish the proposal

5. Meeting adjournment

The meeting was adjourned at 16:00.

10.4. Minutes of the 4th Project Meeting

Date: Sep 30th, 2020

Time: 21:30

Place: Via Zoom meeting, online.

Present: Professor Song Yangqiu,

and CHAN Tsz Ho, Tang Yutian, Wong Pui Ying, Zhang Liangwei

Absent: None

Recorder: Zhang Liangwei

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 The proposal is submitted and every group member reports about the work plan for the coming semester.

2.2 Demonstrate the result of getting entities from the corpus.

3. Discussion items

3.1 Professor Song points out the comment for the submitted proposal and provides advice for future work.

3.2 Professor Song demonstrates some possible ways to find the potential relation within a sentence.

3.3 Discuss the pattern-determined data mining method.

3.4 Discuss the possible ways of searching function

4. Goals for the coming week

4.1 Try out some frequency mining methods.

4.2 Perform frequency ranking on mining relations.

5. Meeting adjournment

The meeting was adjourned at 22:00.

10.5. Minutes of the 5th Project Meeting

Date: Nov 3rd, 2020

Time: 20:30

Place: Via Zoom meeting, online.

Present: CHAN Tsz Ho, Tang Yutian, Wong Pui Ying and Zhang Liangwei

Absent: None

Recorder: Wong Pui Ying

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 Every group member finished the assigned frequency mining try out.

2.2 Report the result of relation frequency mining.

3. Discussion items

3.1 Discuss the storage method used in our project.

3.2 Discuss the co-work platform to be used.

3.3 Discuss the possibility to perform N-ary relation extraction.

4. Goals for the coming week

4.1 Continue to perform frequency mining methods.

4.2 Set up the co-work platform and online database.

5. Meeting adjournment

The meeting was adjourned at 21:30.

10.6. Minutes of the 6th Project Meeting

Date: Nov 25th, 2020

Time: 21:30

Place: Via Zoom meeting, online.

Present: Professor Song Yangqiu

and CHAN Tsz Ho, Tang Yutian, Wong Pui Ying, Zhang Liangwei

Absent: None

Recorder: CHAN Tsz Ho

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 Every group member reports the result of frequency mining and the challenges.

3. Discussion items

3.1 Professor Song provides possible solutions for our met challenges.

3.2 Professor Song demonstrates the expected outcome of frequency mining.

3.3 Discuss the potential ways to reduce the ambiguity.

4. Goals for the coming week

4.1 Finish the draft of the pipeline and draft of the outcome.

4.2 Try out the machine learning algorithm for mining more relation from pos-tag.

5. Meeting adjournment

The meeting was adjourned at 22:10.

10.7. Minutes of the 7th Project Meeting

Date: Jan 7th, 2021

Time: 22:00

Place: Via Zoom meeting, online.

Present: Professor Song Yangqiu

and CHAN Tsz Ho, Tang Yutian, Wong Pui Ying, Zhang Liangwei

Absent: None

Recorder: Zhang Liangwei

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 Every group member reports the result of extending frequency mining and the challenges.

3. Discussion items

3.1 Professor Song points out comments on our present work and provides possible solutions for our met challenges.

3.2 Professor Song provides possibilities for us to present the outcome of the project.

3.3 Discuss the quality of the dataset and the ways to cope.

3.4 Discuss the details about machine learning algorithms for bootstrapping.

4. Goals for the coming week

4.1 Start draft the progress report.

4.2 Modify the machine learning algorithm for mining more relation from pos-tag.

4.3 Find possible ways to increase the quality of our present outcome.

4.4 Start to construct the frontend of the knowledge graph.

5. Meeting adjournment

The meeting was adjourned at 22:40.

10.8. Minutes of the 8th Project Meeting

Date: Feb 10th, 2021

Time: 20:00

Place: Via Zoom meeting, online.

Present: CHAN Tsz Ho, Tang Yutian, Wong Pui Ying and Zhang Liangwei

Absent: None

Recorder: CHAN Tsz Ho

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 Finish of progress report.

2.2 Basic framework of the frontend web application

3. Discussion items

3.1 Change the priority of development to web application.

3.2 The further possible methods for bootstrapping the existing relation data.

3.3 Data analysis for future model selection.

4. Goals for the coming holiday

4.1 Finish assigned web-application function.

4.2 Perform bootstrap and examine the outcome.

4.3 Construct analysis report on current data.

5. Meeting adjournment

The meeting was adjourned at 21:40.

10.9. Minutes of the 9th Project Meeting

Date: Feb 28th, 2021

Time: 15:00

Place: Via Zoom meeting, online.

Present: CHAN Tsz Ho, Tang Yutian, Wong Pui Ying and Zhang Liangwei

Absent: None

Recorder: Tang Yutian

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 Demonstration the assigned functions in web-application.

2.2 Report the result of tried bootstrap

3. Discussion items

3.1 Possible reasons for neutral performance on bootstrap.

3.2 The usability of web-application and improvement.

3.3 The analysis results on current data.

4. Goals for the coming week

4.1 Enhance the web-application.

4.2 Begin the study on bootstrap.

4.3 Refine the previous information extraction pipeline.

5. Meeting adjournment

The meeting was adjourned at 17:10.

10.10. Minutes of the 10th Project Meeting

Date: April 1st, 2021

Time: 21:00

Place: Via Zoom meeting, online.

Present: CHAN Tsz Ho, Tang Yutian, Wong Pui Ying and Zhang Liangwei

Absent: None

Recorder: Wong Pui Ying

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 Demonstration of the newly-add functions on web-application.

2.2 The final result of data.

3. Discussion items

3.1 Possible ways to present the final result to the professor.

3.2 Test case of testing the web-application and user study.

4. Goals for the coming holiday

4.1 Finish the draft version of knowledge graph web-application.

4.2 Perform tests towards our project.

4.3 Invite friends to do the user study.

5. Meeting adjournment

The meeting was adjourned at 21:40.

10.11. Minutes of the 11th Project Meeting

Date: April 8th, 2021

Time: 22:00

Place: Via Zoom meeting, online.

Present: Professor Song Yangqiu

and CHAN Tsz Ho, Tang Yutian, Wong Pui Ying, Zhang Liangwei

Absent: None

Recorder: Zhang Liangwei

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 Finalize the web-application..

2.2 Result of testing.

3. Discussion items

3.1 Professor Song points out comments on our pipeline and provides possible solutions for us to show it clearly in the report.

3.2 Professor Song provides possibilities for us to present the final outcome of the project.

3.3 Discuss the details about possible future work.

4. Goals for the coming holiday

4.1 Start drafting the final report.

4.2 Produce the demonstration video for final report.

5. Meeting adjournment

The meeting was adjourned at 22:40.