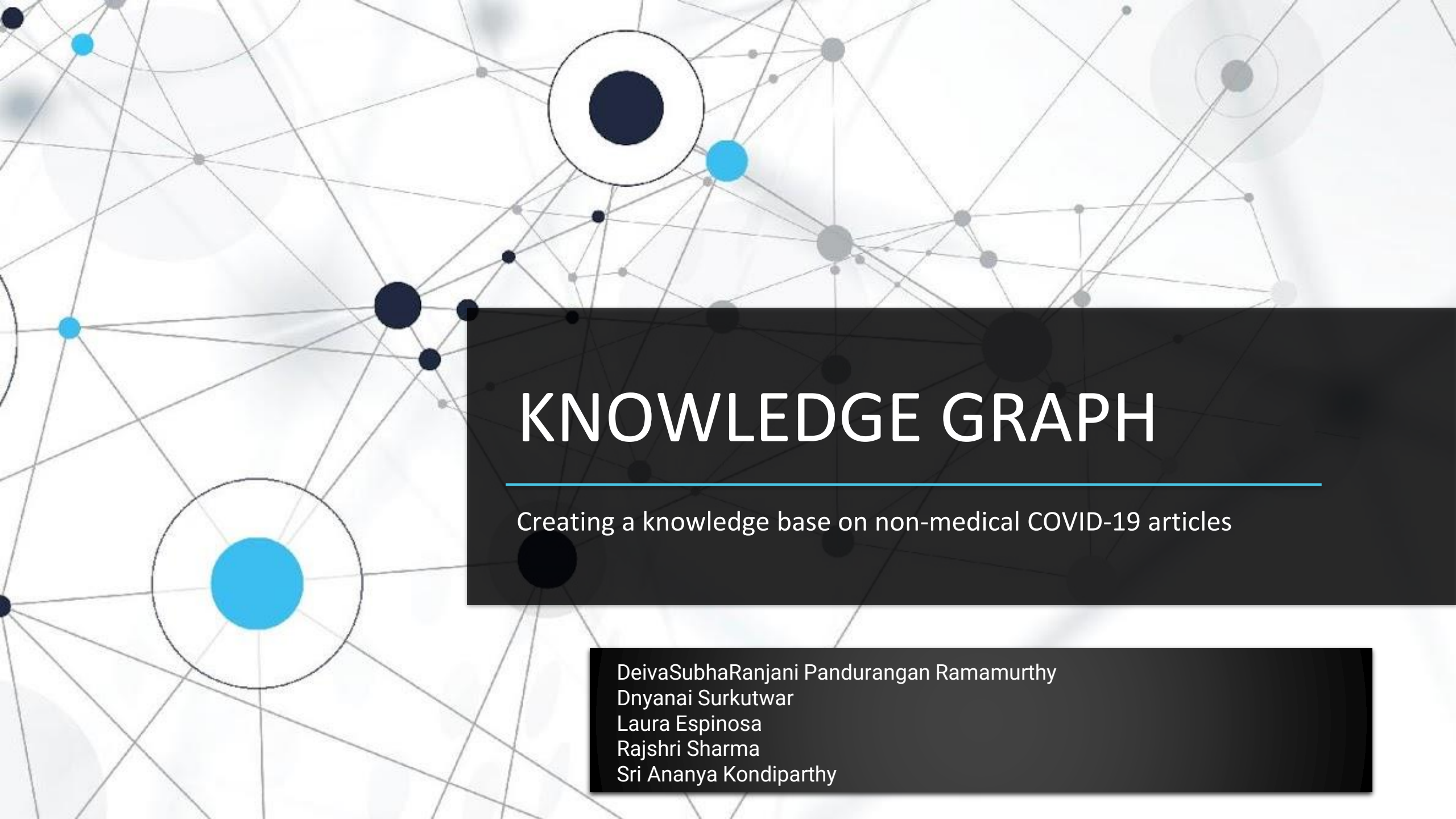


KNOWLEDGE GRAPH

Creating a knowledge base on non-medical COVID-19 articles

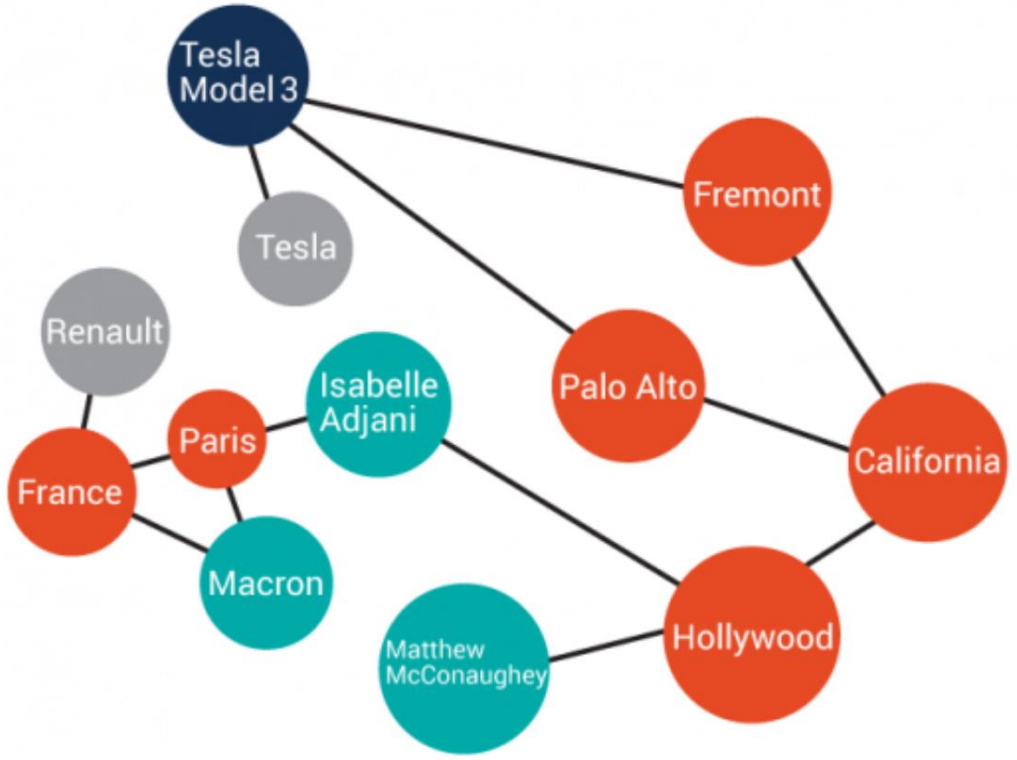
DeivaSubhaRanjani Pandurangan Ramamurthy
Dnyanai Surkutwar
Laura Espinosa
Rajshri Sharma
Sri Ananya Kondiparthy

AGENDA

- What is a Knowledge Graph?
- Motivation
- Learnings and Attempts
- Path I
- Graphical Representation using Neo4j
- Path II
- Graphical Representation using Networkd3
- Future Scope
- Q&A

WHAT IS A KNOWLEDGE GRAPH?

- When we utilize KG for NLP, we are looking at the most common relationships between associated words to infer meaning.
- Other use cases → Netflix utilizes KG to analyze all their data to best recommend content to their end users.
- Key value prop → KG harmonizes data across vast sources, like different data sources or databases.
- Aggregates and creates relationships between different data points to highlight where connections lie.



MOTIVATION

- The Covid-19 crisis has received tremendous response in terms of publications texts. Fewer studies on non-medical impacts on Covid-19 motivated us to extract and formalize this text in a structured and computable form . This can be technically achieved using knowledge graphs in NLP.
- The **Aim** of this project is to create a **prototype of Knowledge Graph** which can be leveraged for information retrieval to show the impact of Business, Technical and Automotive impacts of COVID19.

Dataset

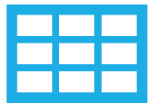
- This data set will let us explore the non-medical impacts of COVID-19 in terms of the social, political, economic and technological dimensions.

200,000 Articles	Dates from Jan 2020 to September 2020	We have columns such as Title, URL, date, domain,author, content, topic area
5 major topics - business, general, finance, tech, science, healthcare, environment, automotive, ai	Major domain - business, general and finance	Choosing articles with business, tech and automotive domain

ATTEMPTS

- **Relation Extraction** is the task of predicting attributes and relations for entities in a sentence. It is a key component for building relation knowledge graphs.
- **Knowledge Graph** uses the *triples* to store knowledge & to describe the facts in the real world.
- Aim - to get **triples - (head entity, relationship, tail entity)**

Why we did not go for NER directly?



Pattern Matching: Noun Verb Noun

Write your own Function!

Noun Chunks

Text Summarization

LEARNINGS – NER

a case of the new virus spreading rapidly in china GPE has been reported in a patient in seattle GPE washington GPE reuters reports. the patient had recently returned from china GPE and is clinically healthy but still being monitored. this is the first ORDINAL us case of the virus which

Using **AWS Comprehend,**

- Too many GPE, ORG, PERSONS.
- Difficult to detect the relationship between the different GPEs.
- Unable to detect important contextual information since missed nouns like *virus* and *patient*.
- Identifying POS tagging is more useful to extract triples.

LEARNINGS – PATTERN MATCHER

```
fing that they believe the threat to the us remains low. the virus is currently known as 2019-ncov. the designation indicate
s that it is a coronavirus the family of viruses that also caused the sars outbreak in 2003. that outbreak killed nearly 800
```

```
# This is a triple and can define a relationship
pattern1 = [{'POS': 'NOUN','OP':'+'},
            {'POS': 'ADP','OP':'?'},
            {'POS': 'NOUN','OP':'+'}]
```

```
#AUX-VERB-ADV-ADP:
  pattern = [{'POS':'AUX','OP':"?"},
             {'POS':'VERB'},
             {'POS':'ADV','OP':"?"},
             {'POS':'ADP','OP':"?"}]
```

Output:

```
Match1 91 93 press briefing
Match1 121 124 family of viruses
```

Using **Spacy Pattern Matcher**,

- Not efficient as we need to create individual patterns to accommodate every scenario. This will not work for all the articles thus overfitting.
- We tried a few patterns, but we ultimately decided to make our own function to detect all Noun-Verb-Noun combinations.

PATH I

Output:

```
Starting position: 0
Verb position: 6
The sentence: [a, case, of, the, new, virus, spreading]
Before - POS_'NOUN': virus
POS_ 'VERB' spreading 6
-----------------------------------
After - POS_'NOUN': china
Starting position: 0
Verb position: 12
The sentence: [a, case, of, the, new, virus, spreading, rapidly, in, china, has, been, reported]
Before - POS_'NOUN': china
POS_ 'VERB' reported 12
```

Noun Verb Noun Dataset:

	ent1_article0	relations_article0	ent2_article0
0	virus	spreading	china
1	china	reported	patient
2	reuters	reports	patient
3	patient	returned	china
4	china	monitored	us

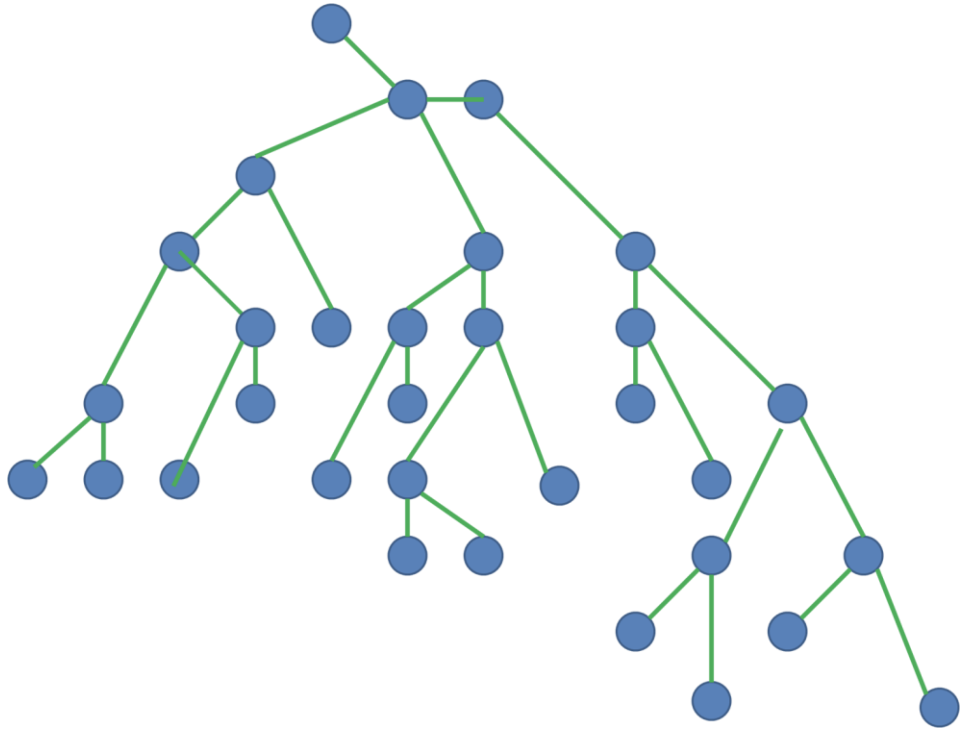
```
#for each in kg.content[0]:
each = kg.content[0]
ent1_0 = []
ent2_0 = []
verbs_0 = []

def getBeforeVerbNoun(start,verb_i,article):
    print(start)
    print(verb_i)

    nn = ['NOUN','PROPN']

    print(list(article)[start:verb_i:1])

    for i in (list(article)[verb_i:start:-1]):

        if i.pos_ in nn:
            print('detected:  ',i)
            return i
```

```
def getAfterVerbNoun(verb_i,article):

    nn = ['NOUN','PROPN']

    for i in (list(article)[verb_i::]):

        if i.pos_ in nn:
            print('detected:  ',i)
            return i
```

```
        start = 0
        end = 0
        for tok in sent:
            if tok.is_sent_start:
                start = tok.i

            if tok.pos_=='VERB':
                verbs_0.append(tok)

                #print(tok.i)
                #print('verb',tok)
                ent_1 = getBeforeVerbNoun(start,tok.i,nlp(each))
                print('verb',tok,tok.i)
                ent1_0.append(ent_1)

                ent_2 = getAfterVerbNoun(tok.i,nlp(each))
                ent2_0.append(ent_2)
```

GRAPHICAL REPRESENTATION- NEO4J

- Neo4j is a native graph database, built from the ground up to leverage not only data but also data relationships.
- Neo4j has a flexible structure defined by stored relationships between data records.
- To generate the dynamic relationship between the nodes we are adding a plug-in to our Neo4j environment called as APOC.
- APOC is an add-on library for Neo4j that provides hundreds of procedures and functions adding a lot of useful functionality.
- It can be installed with a single click in Neo4j Desktop.



COMMANDS TO GENERATE GRAPH IN NEO4J

- **To load the data**

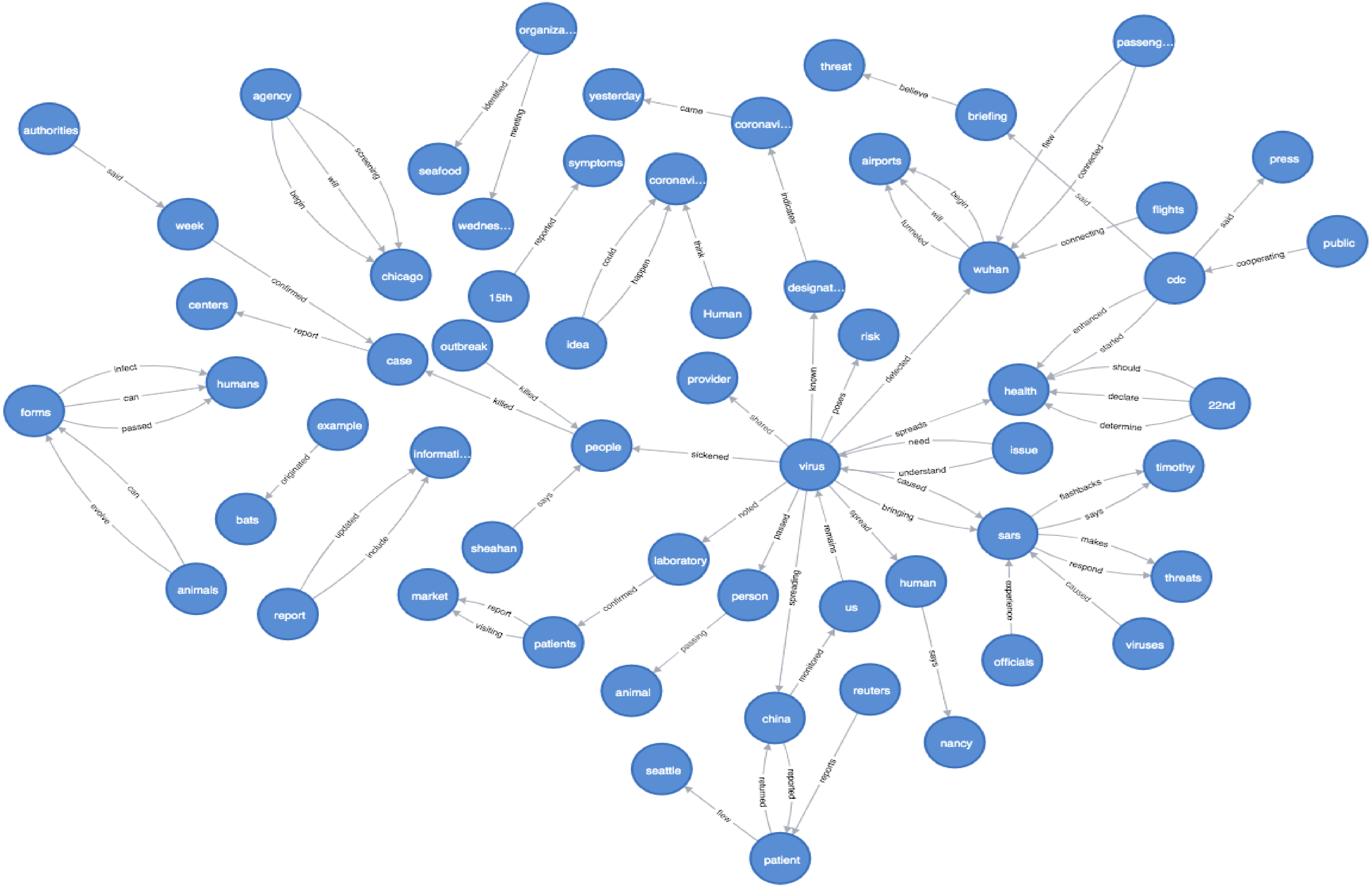
```
LOAD CSV FROM 'file:///article0_triples_simple.csv' AS row
WITH row[0] AS ID, row[1] AS Subject, row[2] AS Verb, row[3] AS Predicate
RETURN ID, Subject, Verb, Predicate
```

- **To create the nodes and dynamic relationship between them**

```
LOAD CSV WITH HEADERS FROM 'file:///article0_triples_simple.csv' as row
MERGE (s:MyGraph {Name: row.Subject})
MERGE (p:MyGraph {Name: row.Predicate})
with s,p,row
CALL apoc.create.relationship(s,row.Verb,{},p) yield rel
return rel
```

- **To display the graph**

```
MATCH (s:MyGraph) RETURN s
```


organiza...
agency
authorities
seafood
yesterday
coronavi...
threat
passeng...
briefing
symptoms
coronavi...
airports
press
flights
wednes...
week
chicago
15th
designat...
wuhan
cdc
public
centers
Human
idea
outbreak
case
risk
provider
health
22nd
humans
forms
example
people
virus
issue
timothy
informati...
bats
sheahan
laboratory
sars
threats
animals
report
market
person
us
human
patients
officials
viruses
animal
reuters
china
nancy
seattle
patient
identified
meeting
said
screening
will
begin
came
believe
flew
connected
indicates
think
could
happen
reported
confirmed
report
infect
can
passed
evolve
originated
updated
include
killed
says
sickened
known
poses
detected
shared
spreads
need
understand
caused
bringing
noted
spread
remains
spreading
flashbacks
makes
respond
experience
connecting
cooperating
enhanced
started
should
declare
determine
funneled
visiting
passing
monitored
returned
reports

PATH II

```
noun_chunks_01234 = []


for i in [0,1,2,3,4]:
    for chunk in nlp(kg.cleaned[i]).noun_chunks: #or in [patterns]
        noun_chunks_01234.append(chunk)
```

Disadvantage of using **Noun Verb Noun,**

(i) Loss of context,

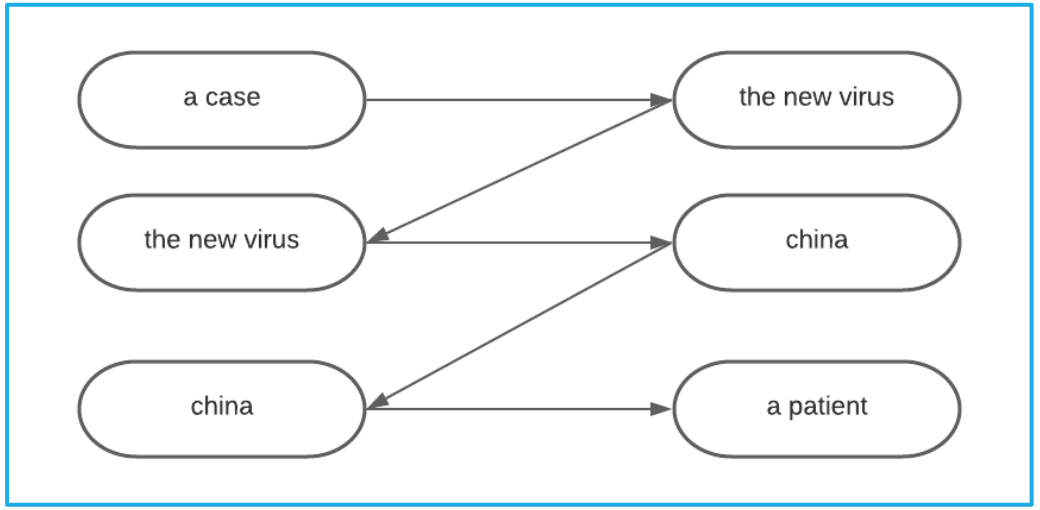
(ii) We were not able to accurately get triples (only ~10-20% triples were correct)

Noun Chunks,

- Noun chunks are “base noun phrases” – flat phrases that have a noun as their head.
- We used Spacy to extract the Noun chunks from first 5 articles.
- Custom stop words to keep Prepositions/Conjunctions(POS_ : ADP) like ‘of’ and ‘in’ along with verbs to define a relationship between 2 nouns.

HOW WE CREATED A TRIPLE?

- How we separated the two entities?
- Manual check - relations (VERB,ADP) between these entities to get the triples.
- Accuracy is based on how many triples give the correct relation between two noun phrases.
- For a DataFrame consisting of 137 number triples in ‘of’ and ‘in’ relation, we got correct triple accuracy of ~45%.




```
print(edge_1[:5])
print(relations_[:5])
print(edge_2[:5])
```

```
[a case, the new virus, china, a patient, seattle washington reuters]
[of, spreading, in, reported, in]
[the new virus, china, a patient, seattle washington reuters, the patient]
```

	N1	relations	N2
0	a case	of	the new virus
1	the new virus	spreading	china
2	china	in	a patient
3	a patient	reported	seattle washington reuters
4	seattle washington reuters	in	the patient

“A case ‘of’ the new virus spreading rapidly ‘in’ china has been reported ‘in’ a patient ‘in’ seattle washington reuters reports.
The patient had recently returned from china..

GRAPHICAL REPRESENTATION – NETWORKD3

To create networkd3 graph,

- We created 2 files:

(i) Links – Consisting of our triples dataframe, we created a key for each entity column.

(ii) Nodes – Consisting of only unique entities with their keys.

We used R script to implement the graph.

FUTURE SCOPE

Our dataset has around 200K articles,

- Due to its high dimensionality, we also implemented Text Summarization which we think will be our next step.
- Moreover, the pattern matcher gave us some complex relationships which we can add to our existing dataframe.
- We also worked on automating the process of detecting correct triples based on the idea of our Noun Verb Noun algorithm which we plan to extend to our Noun Chunks dataframe.

Q&A