A Semantic QA Framework for Knowledge Discovery from CORD-19

Master Thesis

by

Manmeet Kumar Chaudhuri

27th October 2020

SRH University Heidelberg

School of Information, Media and Design

Degree course: M.Sc. Big Data and Business Analytics

Supervisors Prof. Dr. Sven Garbade

Prof. Dr. Swati Chandna

Affidavit

Herewith I declare:

- that I have composed the chapters for the Master Thesis for which I am named as the author independently,
- that I did not use any other sources or additives than the ones specified,
- that I did not submit this work at any other examination procedure.

Heidelberg, _____ Signature: _____

Acknowledgment

I would like to express my gratitude to all my Professors for imparting in me the necessary skill sets to undertake the Master Thesis independently. I dedicate the Thesis to my Professors:

- Prof. Dr. Sven Garbade
- Prof. Dr. Swati Chandna
- Prof. Dr. Ajinkya Prabhune
- Prof. Dr. Barbara Sprick
- Prof. Dr. Herbert Schuster
- Anil Keshav

I would like to thank my parents, Mr. Pratap Chandra Chaudhuri and Mrs. Narmada Chaudhuri. Without their love and support, I would not have reached this far in my academic pursuits. I would also like to thank my younger brother and my best friend, Major Dr. Navneet Chaudhuri.

Abstract

Question Answering (QA) is an essential task in Natural Language Processing (NLP). One of the many potential QA systems is automatically suggesting answers to the users based upon a data corpus. In the medical domain, providing researchers with relevant answers to questions from a pool of research documents or publications could be a first-step practical application. It will save precious time by providing support and help to researchers in finding answers to questions that are already answered but lost in the vastness of data. Researchers can thereby be more efficient at solving the task at hand.

This is a difficult and challenging time due to Corona Virus Disease (COVID). The scientific community has been quite busy trying to understand the disease and find a cure for it. To further this cause, the Allen Institute of AI and some leading research groups have released the COVID-19 Open Research Dataset (CORD-19). CORD-19 (Wang et al., 2020 [1]) is a growing resource of scientific papers on COVID-19 and related historical coronavirus research.

As a result, in this Thesis, a state-of-the-art semantic AI framework is developed for knowledge discovery from the CORD-19. The Framework is referred to as the "Semantic QA Framework for Knowledge Discovery." Some user questions related to CORD-19 have been identified, and the Framework is used to retrieve knowledge related to the questions. The Framework helps extract the most scientifically relevant publications regarding queries and analyze and summarize information at the sentence level by combining multiple advanced NLP techniques and deep learning techniques. These relevant publications and sentences were retrieved and ranked using cosine similarity scores. The future scope for improving the Framework is also discussed in detail.

Keywords: QA, NLP, Vectorization, Word Embeddings, Deep Learning, Attention, BERT, Knowledge Graph

Table of Contents

List of Figures

List of Tables

List of Acronyms

AI	Artificial Intelligence
BERT	Bidirectional Encoder Representations from Transformers
CBOW	Continuous Bag-of-Words
CNN	Convolutional Neural Network
CORD-19	COVID-19 Open Research Dataset
COVID	Corona Virus Disease
CRISP-DM	Cross-industry standard process for data mining
DL	Deep Learning
DM	Data Mining
ELMO	Embeddings from Language Model
GPT	Generative Pre-trained Transformers
HAL	Hyperspace Analog to Language
KD	Knowledge Discovery
LSA	Latent Semantic Analysis
LSTM	Long Short Term Memory
NER	Name Entity Recognition
NLP	Natural Language Processing
NN	Neural Net
NNLM	Neural Net Language Model
POS	Part-of-speech
QA	Question Answering
RC	Reading Comprehension
RE	Relation Extraction
RNN	Recurrent Neural Network
SARS	Severe acute respiratory syndrome
SG	Skip-Gram Model
SQuAD	Stanford Question Answering Dataset
TF-IDF	Term frequency-Inverse document frequency

1. Introduction

This chapter covers the research's motivation, formulation of the problem that is being investigated, and the definition of the research questions this work attempts to answer. Finally, the research approach to answer the questions is mentioned, and the contribution of this thesis work is presented in this section.

1.1 Motivation

Question Answering (QA) is a critical task in Natural Language Processing (NLP) from the theoretical as well as the practical point of view. Theoretically, the area of QA in its most refined form could be seen as a machine being able to qualify the Turing Test [2]. The machine will be then Artificial Intelligence (AI) and NLP complete. Many AI and NLP problems can be framed as QA, as presented in [3]. The idea of machines that are able to answer questions just like a human is a compelling motivation for researchers in this area. One of the success stories is IBM's Watson [4], a machine that competed in a quiz show called Jeopardy in 2011 and competed against all-time champions on the show and beat them all to win first place. It was quite a milestone in AI and NLP.

The QA system has many applications in various domains like medical, legal, or literature as it can be employed to support users and experts. For example, IBM's Watson was later also used for support in treating lung cancer and many other applications.

One of the many potential QA applications is automatically suggesting answers to the users based upon a data corpus. In the medical domain, providing researchers with relevant answers to questions from a pool of already answered questions and Frequently Asked Questions could be a practical first-step application. It will save precious time by providing support and help to researchers in finding answers to questions that are already answered. Researchers can thereby be more efficient at solving the task at hand. Another related QA application would be answering questions based upon a paragraph in a way done by humans. The machine would have to deal with logic, semantics, and other vital characteristics of the text.

Deep Learning (DL) is fast evolving as an advanced tool for solving NLP tasks, including QA. A considerable amount of data is used for training DL models and increasing accuracy. For some domains and tasks, unavailability or scarcity of data due to privacy issues or legal issues may make it harder to train a DL model. Sometimes a DL model is restricted in its ability to

generalize well as the model trained on a task like QA may not perform well on a task like Text Summarization. It is primarily caused by the differences in the learning required for different NLP tasks.

The development of Pre-trained language models provided a solution to these issues. Language model pre-training is found to be useful for learning universal language representations that leverage large amounts of unlabeled data. Some of the prominent pre-trained language models are Embeddings from Language Model (ELMO) [5], Generative Pre-trained Transformers (GPT) [6], and Bidirectional Encoder Representations from Transformers (BERT) [7]. These models work on a two-step learning step. The first step is referred to as pre-training. In the second step, these pre-trained language representations are applied on requisite language tasks like machine translation, question answering, text summarization. Any of the two learning strategies can be used in the second step, i.e., feature-based and fine-tuning based approach. These are discussed in detail in chapter 4.

Despite the worthy advancements, direct use of the pre-trained state-of-the-art NLP models to domain-specific tasks may have some shortcomings. BERT is tested mainly on general and domain agnostic datasets like Wikipedia and may not be as useful when applied to a domain-specific textual data.

This project explores the development of an AI-based semantic QA framework that automatically suggests answers to the questions based upon a data corpus. This project studies if these current state-of-the-art word representation pre-trained DL language models like BERT can be used on domain-specific data to be effective for the QA task.

1.2 Problem Statement

DL is a powerful tool for solving complex NLP tasks like the ones involved in QA. These DL models require a large amount of data for generalization and performance. Labeled data is not always available to train deep learning models, thereby affecting the best performance achievement.

Allen Institute of AI and some leading research groups have prepared the COVID-19 Open Research Dataset (CORD-19). COVID stands for Corona Virus Disease. CORD-19 is a free resource of over 100,000 scholarly articles, including around 60,000 papers with full text, about COVID-19, and the coronavirus family of viruses.

A considerable amount of knowledge resides in the CORD-19. There is a growing need for AI-based approaches for Knowledge Discovery (KD) from CORD-19 because of the vastness of existing literature and the rapid acceleration in new coronavirus literature, making it difficult for the research and scientific community to keep up with relevant knowledge. The research community would prefer a method or Framework to retrieve knowledge (or answers) related to a Query that identifies semantically similar paper ids with relevant sentences from the vast CORD-19.

In this project, an AI framework comprising NLP and other DL techniques is developed and applied to the CORD-19 dataset for the QA task. The Framework can support the medical community in keeping up with the literature and discovering knowledge that can be used in the fight against this infectious disease and optimizing individual researchers' time resources.

1.3 Research Questions

NLP methods for QA are applied to the CORD-19 to find answers to the scientific community's questions to tackle this COVID crisis. Considering the problem statement, two main questions that will be answered in the Thesis are as follows:

1. Which DL NLP architecture should be selected for the KD task from CORD-19? KD, in the context of this Thesis, is limited to the QA task.
2. Can an AI Framework be developed incorporating the selected DL NLP architecture to retrieve or discover knowledge from CORD-19? Retrieving the knowledge is limited to identifying relevant publications, sentences from the corpus and creating knowledge graphs using the retrieved knowledge.

Using the AI Framework developed in the Thesis, the following questions specific to the CORD-19 are answered.

- What is known about the origin of virus and management measures at the human-animal interface?
- What is known about transmission, incubation, and environmental stability?
- What has been published about medical care?
- What is known about COVID-19 risk factors?
- What has been published regarding research and development and evaluation efforts for developing vaccines and therapeutics?

These questions are just indicative. The Framework developed in this Thesis can cater to any type of question-related to the CORD-19. The Framework developed in the Thesis provides an answer to the questions by retrieving the most semantically similar research publications, sentences and preparing knowledge graphs.

The relationship between QA and Reading Comprehension (RC) is very close. Some researchers consider RC a kind of specific QA task, and others regard the RC as a kind of method to solve QA tasks. Within the limited scope of this Thesis, both RC and QA are considered to be the same.

1.4 Thesis Contribution

State of the art DL and NLP techniques have been deployed to create an intelligent QA Framework that can answer various questions related to CORD-19. The Framework is referred to as "A Semantic QA Framework for Knowledge Discovery from CROD-19." Some indicative user queries have been identified in section 1.3 that would be answered using the Framework. The answer provided by the Framework is limited to retrieving the most semantically similar research publications and sentences from the dataset based on the query. This Framework will help the research and scientific community retrieve key results or answers or information just by specifying the requisite scientific keywords related to COVID-19 and then performing analysis at the sentence level to efficiently discover knowledge and information from the dataset. The Framework developed in the Thesis is scalable and is agnostic of the size of the dataset.

1.5 Thesis Outline

The Thesis is structured as follows:

- Chapter 1, Introduction: The motivation for this Thesis is first elaborated. Then, the problem statement and the research questions are identified. The chapter also mentions the contribution that this Thesis makes in the field of QA using DL by developing a semantic QA Framework for knowledge retrieval.
- Chapter 2, Theoretical Background: This chapter focuses on the background information related to TM, NLP, DL and Word Embeddings. Different methods used for Word Embeddings are discussed. Pre-trained language models that are used for QA tasks are introduced. Evaluation metrics that are relevant for checking the effectiveness of QA task are also presented

- Chapter 3, Model Selection: This chapter discusses the choice of pre-trained model selection that forms the base of the DL QA framework proposed in this Thesis. Some recent pre-trained DL models are compared on relevant parameters, and a final selection of the model is made.
- Chapter 4, State-of-the-art: BERT: The architecture of state-of-the-art pre-trained model, i.e., BERT chosen as the backbone of the DL QA Framework developed in the Thesis, is explained in the chapter. How BERT is fine-tuned for various NLP tasks, including QA, is also discussed.
- Chapter 5, Proposed Framework: The Framework, "Semantic QA Framework for Knowledge Discovery," developed to address the task of knowledge discovery from CORD-19, is discussed in this section. Key components and various steps involved in the Framework are provided.
- Chapter 6, Experimentation and Result: This chapter talks about the experimental setup and critical parameters required to apply the Framework on questions related to the CORD-19. The findings of the Framework in these questions are presented and discussed.
- Chapter 7, Conclusion and Future Work: The Thesis concludes by summarizing the effectiveness of the approach developed for the QA task on CORD-19. Suitable topics have been identified that could be further pursued to enhance the methodology and approach used here.

2. Theoretical Background

This section contains theoretical concepts related to KD, NLP, DL and other related concepts. Some of the topics that are discussed and addressed in this section are as follows:

- What is KD? How is KD related to TM and NLP?
- What is NLP? How has NLP evolved?
- What is DL? How does DL solve problems related to various NLP tasks?
- What are some of the state-of-the-art DL networks that are useful in NLP today?
- What are word embeddings? How are word embeddings generated? How have deep contextual models transformed NLP and QA?
- Which evaluation matrices are used to measure the quality of QA tasks?

The questions mentioned above are addressed by first understanding the process of KD. Then, the concept of TM and NLP are explained. Finally, the field of DL is explored, and its application in various NLP tasks is discussed. The section is concluded by giving an overview of DL models used for the NLP task related to QA.

2.1 Knowledge Discovery

KD, as the phrase suggests, relates to finding useful and insightful information from data. Fayyad et al. [8] the original author of the process, referred to Knowledge Discovery as a structured process of finding valid, novel, useful, and understandable patterns in the data available in large databases. Data mining (DM) methods are employed to extract knowledge with steps that involve requisite pre-processing and transformation of data. The process of KD and various steps are shown in Figure 2-1.

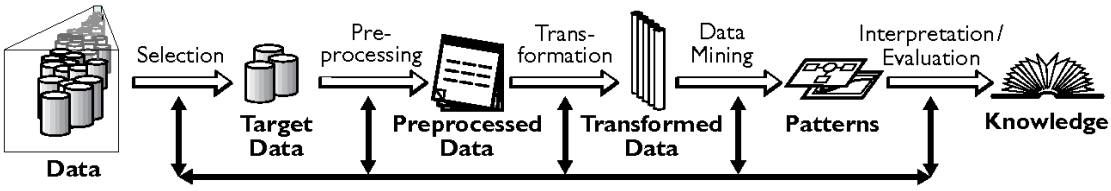


Figure 2-1 Knowledge Discovery Process as per Fayyad et al. [8]

It can be inferred from Figure 2-1 that the overall process is a repetitive process that involves five necessary steps. These steps are mentioned below:

a) **Selection:** The first steps start with understanding the end-user requirement and developing domain knowledge on the task at hand. A dataset is created or selected based upon the requirement on which the knowledge discovery process is applied.
b) **Pre-processing:** The data is cleaned to remove noise and outliers. If there is some missing data, a strategy must be decided to handle such missing data fields. The phrase "garbage in, garbage out" is very much relevant at this step.
c) **Transformation:** Data is transformed by using dimension reduction and transformation methods to bring down the number of variables and better represent the data.
d) **DM:** Suitable DM algorithms are used based upon the KD process's goal, i.e., classification, regression, clustering, sentiment analysis, and language translation.
e) **Interpretation/ Evaluation:** Mined patterns are interpreted, and the knowledge is consolidated in this step.

2.2 Data Mining and Text Mining

DM involves applying algorithms or models on data to extract patterns or solve the task at hand. Figure 2-2 shows various DM techniques and related algorithms.

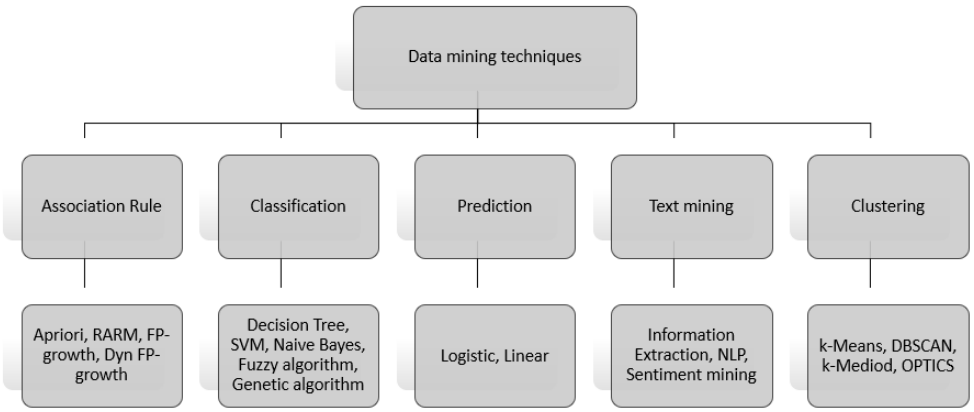


Figure 2-2 Various data mining techniques

The concept of DM is defined in the Cross-industry standard process for data mining (CRISP-DM, 2000) [9]. As per CRISP-DM, DM process comprises six significant phases: Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation, and Deployment.

TM is the DM technique of extracting knowledge from unstructured textual resources like websites, books, emails, reviews, publications. TM is discussed in detail in the next section. Usually, a combination of DM techniques is used to find a solution to a data problem rather than using just one technique.

There are a variety of TM methods that are employed for analyzing textual data, as identified by Talib et al. [10]. Some of the fundamental techniques are Document Classification, Information Retrieval, Web Mining, Document Clustering, Information Extraction, and NLP. The areas of the application under all these techniques overlap and interact with each other. Figure 2-3 shows the various types of techniques, their application areas and the relationship among these areas.

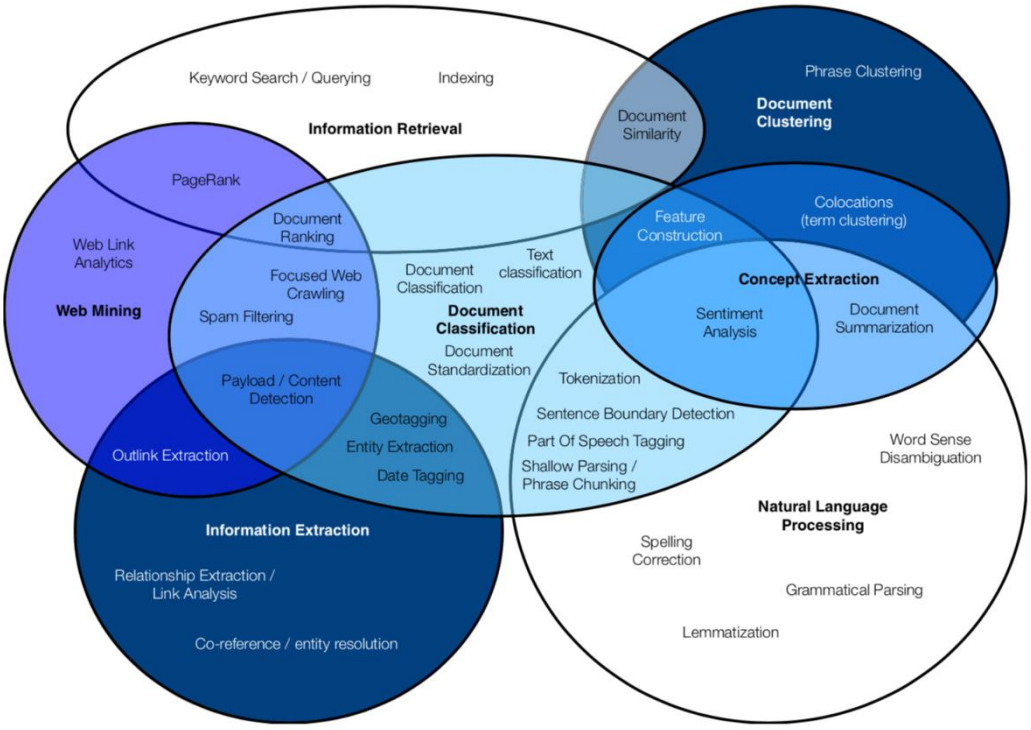


Figure 2-3 Inter-relation among various text mining techniques [10]

2.3 Machine Learning and Deep Learning

Before exploring the area of NLP in detail, it is vital to have a basic understanding of ML and DL. For a DM and Text Mining task, it is essential to choose the correct approach for addressing the task or problem at hand. One of the most recent and advanced methods has been the use of ML methods. The significant difference between humans and machines that

have been around for a long time is that humans tend to learn and improve their way of doing a task with time and experience. Traditionally machines or computers have been just performing their tasks without any mechanism of learning in their approach. It has changed now with the advancement in the field of machine learning. Machines can learn and improve their performance by understanding and analyzing many data with time.

ML approaches can be divided into four groups, i.e., Supervised learning, Unsupervised learning, Semi-supervised learning, Reinforcement learning and DL. Supervised learning incorporates human help to train the pattern-detecting or text mining model, and in Unsupervised learning, machines find knowledge from the data with little or no human intervention. Semi-supervised learning methods fall in between Supervised and Unsupervised learning. Reinforcement learning involves direct interaction with the problem environment to gain knowledge.

DL is a sub-field of ML that determines patterns from the data using a neural network. DL has a variety of architectures such as Neural Network, Convolutional Neural Network (CNN), Deep Belief Networks, Recurrent Neural Network (RNN) used by research engineers across the globe to develop right from simple text recognition to complicated driverless car technologies.

With the recent developments and advancements in computational technologies, DL methods have successfully achieved very high performance in many NLP tasks. Out of all the NLP models, DL models such as sequence-to-sequence, encoder-decoder based neural network algorithms comprising LSTMs and Transformers have made significant advancements in question answering, machine translation, name entity recognition in recent times. These DL models have also outperformed traditional models and approaches.

2.4 Natural Language Processing

NLP is a field of AI where computers perform human-like activities. These activities could be understanding and summarizing a text document, translation in other languages, sentiment mining, text classification. Traditional approaches to understanding human speech using programming languages often underperform. These traditional approaches lack ways to handle nuances in a language, such as dialects, contexts, jargon, and other peculiarities used by the speaker. Many ML-based NLP algorithms evolved over the past decades that handle most of the limitations mentioned above. The advances in DL-based NLP have gained much traction

in recent years, which solves complex tasks that legacy models fail to achieve. A brief breakdown of NLP tasks is provided in Figure 2-4.

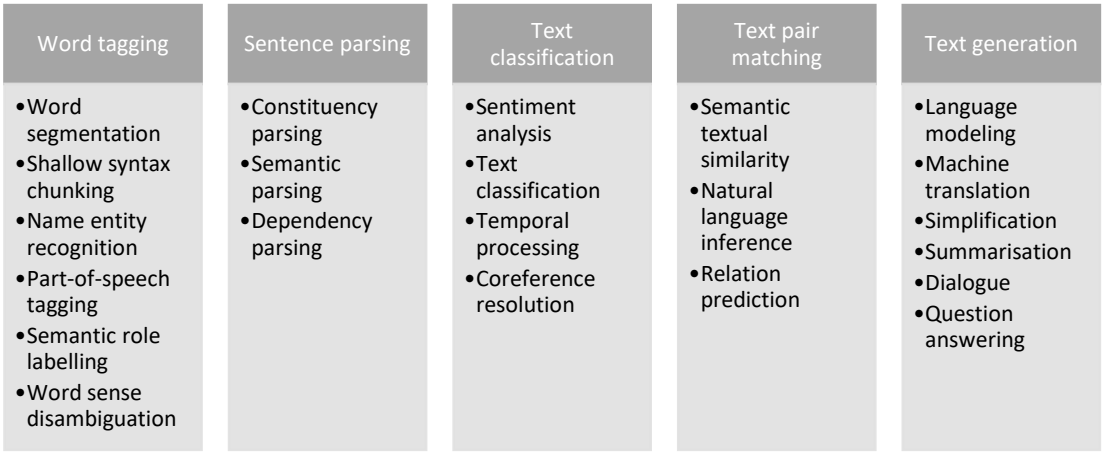


Figure 2-4 Various NLP Tasks

2.5 History of NLP

Research in the domain of NLP has been happening since the 1940s. In this report, for the sake of better understanding, various NLP methods have been divided into two broad categories based upon the timeline [11].

2.5.1 NLP methods before the Deep Learning era

In 1949, Weaver et al. [12] developed the first computer-based application for natural language in the field of Machine translation. Dostert et al., in 1955 [13] through a set of computer experiments known as Georgetown experiments, a collaboration between IBM and Georgetown University, translated more than 60 Russian language sentences into English. Early systems used dictionaries to look up appropriate words for translation and then ordered the words after translation as per word-order and grammar rules.

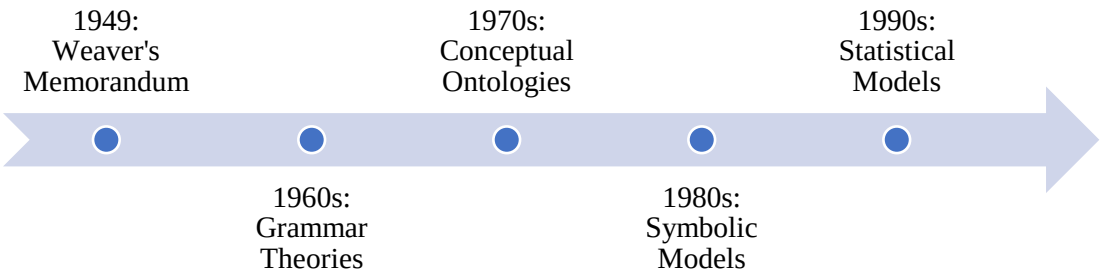


Figure 2-5 Key milestones in NLP during pre-Deep Learning era [11]

In the 1960s, computer researchers developed new theories related to grammar that were computationally tractable for the first time. Many new theories were developed that explained syntactic anomalies and provided semantic representations, such as Fillmore, 1968 [14] regarding case grammar, Collins et al., 1969 [15] related to semantic networks, Woods, 1970 [16] for augmented transition networks Schank, 1972 [17] related to conceptual dependency theory. Many natural language prototypes were also developed during this time, e.g., ELIZA, LUNAR, PARRY.

During the 1970s, new NLP ideas like building conceptual ontologies that structured real-world information into computer-understandable data were explored.

In the 1980s, various researchers like Charniak,1983 [18]; Dyer,1983 [19]; Riesbeck and Martin,1986 [20]; Grosz et al.,1987 [21]; Hirst,1987 [22]; developed symbolic approaches to solving problems in NLP. These symbolic approaches involved the use of complex hard-coded rules and grammars to parse a language. Initially, the text was segmented into meaningless tokens consisting of words and punctuation. Then, representations were created manually by assigning meanings to these tokens. The mutual relationships among these tokens are then established using well-understood knowledge representation schemes and other associated algorithms. These representations were finally used to perform an analysis of linguistic phenomena.

The 1990s saw the revolution in which statistical models came into being for solving NLP tasks. Researched like Bahl et al., 1989 [23]; Brill et al., 1990 [24]; Chitrao and Grishman, 1990 [25]; Brown et al., 1991 [26] developed these statistical methods replacing previous NLP systems that were based on a complex set of handwritten rules. Statistical models could do complex tasks of creating language rules through automated learning.

2.5.2 NLP methods during the Deep Learning era

In 2003, Bengio et al. [27] proposed the first neural network language model that consisted of one hidden layer feed-forward NN. They were one of the first researchers to introduce what is now referred to as the word embedding, a real-valued word feature vector (refer section 2.6 for details). In 2008, Collobert et al. [28] applied multitask learning [29] in NLP using NNs, wherein multiple learning tasks were solved simultaneously. They used a single convolutional neural network architecture that performed several NLP predictions like part-of-speech tagging, named entity tagging, and semantic roles on a given sentence.

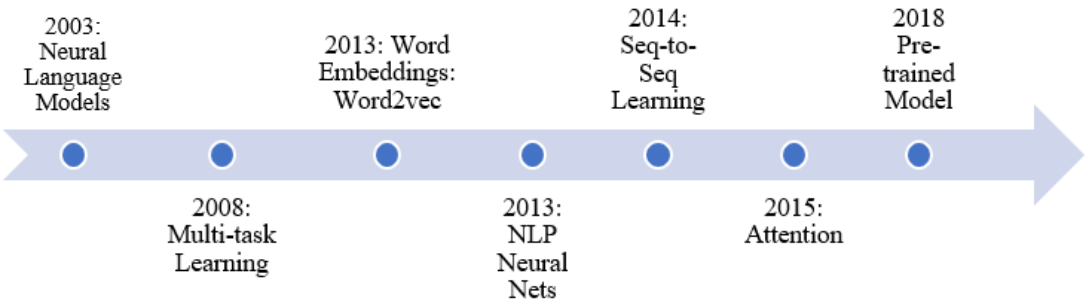


Figure 2-6 Key milestones in NLP during Deep Learning era [11]

In 2013, Mikolov et al. [30] from Google introduced the word embedding models, i.e., Word2Vec. Word2vec is based upon a shallow and efficient neural network. The architecture thus enables training a large number of word or sentence level embeddings on a vast amount of unstructured data. In 2013 various other neural network models were used in NLP. Three other types of neural networks, i.e., RNN [31], CNN, and recursive neural network [32], were used. However, usage of plain and simple RNN models was quickly replaced by Long-Short Term Memory (LSTM) networks (Hochreiter et a., 1997 [33]). LSTMs solved the issue of data persistence in a neural network, and they were not prone to the issue of vanishing and exploding gradient.

Sutskever et al., in 2014 [34], proposed the sequence-to-sequence neural network method. This network comprises an encoder and a decoder. An encoder neural network is fed a sentence letter by letter, and the encoder compresses the sentence into a vector representation. A decoder neural network predicts the output sentence letter by letter using the encoder input and state. Initially, encoders and decoders were based on RNN, but recent architectures are based on LSTM ([35]) and Transformer ([36]). Transformers are more effective in maintaining the data persistence and solving the vanishing gradient problem. Sometimes an amalgamation of LSTM and Transformer ([37]) is used in a sequence-to-sequence neural network. Sequence-to-sequence models are used in NLP tasks like language translation. Google announced in 2016 that it is using a neural sequence-to-sequence model in Google Translate.

Bahdanau and the team [38] introduced the concept of attention in 2015. They used the concept of attention to improve seq-to-seq models for language translation tasks. A new form of attention known as self-attention is more prominent today. Self-attention is the building block of the Transformer architecture. The self-attention mechanism looks at the neighboring words

in a sentence and generates more semantically and contextually vibrant word representations. Attention and self-attention are discussed in detail in Chapter 4.

Pre-trained language models were introduced in 2018 and they transformed the world of NLP. Features in the form of word embeddings or sentence embeddings generated from pre-trained language models can be used as input in other downstream models for NLP specific tasks [5]. Pre-trained language models can also be fine-tuned to a specific language task as done by researchers [7] [39]. Fine-tuned models have shown improved results even when the training data is limited. Pre-trained language models have been able to learn various nuances of language representations using large datasets. Low-level models can be built on top of these large scale pre-trained models to generate impressive results using limited training data.

2.6 Concept of Word Embeddings

A word embedding is a fixed-length, dense, and real-valued vector depiction of a word. Word embeddings have positioned themselves as a fundamental element of many NLP tasks. The building blocks of a natural language are words and sentences. These words and sentences, which are easily comprehended by humans, are not directly understandable by computers. Input text has to be converted into word embeddings so that the text can be understood and processed by machines.

With the onset of DL that solved a number of problems in various fields, NLP researchers started to think of ways to represent textual data numerically so as to fully exploit the potential of DL in solving NLP tasks. Extensive research has been done in representing textual data in machine comprehensible form that captures both the semantic and syntactical meaning of a sentence. There are many ways to represent a word in the form of machine-readable vectors. These methods are discussed in this section.

One method is the use of one-hot encoding. In one-hot encoding, for a given vocabulary of N words, each word is assigned a unique integer index $i \in \{1, ..., N\}$. Each of the words in the vocabulary is then represented in an N-dimensional vector space. The representation comprises of zero in all the positions except for the position corresponding to the word's index where the assigned value is one.

One-hot encoding is a quick and easy way for representing words numerically, but it has two main drawbacks.

One-hot encoding is a quick and easy way for representing words numerically, but it has two main drawbacks.

- With the increasing size of the vocabulary, the vector size of each of the word representation increases. An increase in the size of the word vector leads to an increase in the features and parameters for estimation. This leads to the requirement of more data for training and more computational resources.
- One-hot encoding does not capture the semantics or similarity, or relationship between words in a sentence. The image on the right-hand side of Figure 2-7 shows how the words should ideally be captured in a three-dimensional vector space. The image on the left-hand side of Figure 2-7 shows the actual way in which words are represented in one-hot encoding.

For the reasons identified above, one-hot encoding is hardly used directly as a definitive word vector representation. It is instead used as an input to more refined and complex word embedding models.

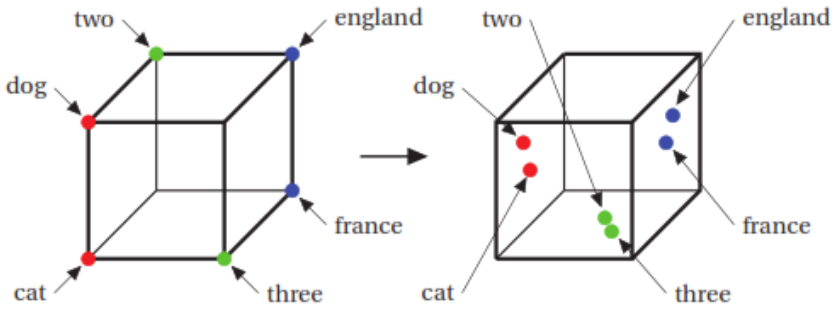


Figure 2-7 One-hot encoding in vector spaces [11]

Various types of word embedding modes are discussed in the next sections.

2.6.1 Frequency-based word embedding models

Count-based or Frequency-based methods are unsupervised methods and use corpus-wide statistics such as frequencies and frequencies to build vector representations of words. These models are based upon the concept of the co-occurrence matrix. The co-occurrence matrix describes how words in a document occur together with respect to each other. It is simply prepared by counting how two or more words occur together in a document or a corpus. These

matrixes are then decomposed by using methods like Principal Component Analysis, Singular Vector Decomposition.

	I	love	Math	Science	tolerate	Chemistry	.
I	0	2	0	0	1	0	2
love	2	0	1	1	0	0	0
Math	0	1	0	0	0	0	1
Science	0	1	0	0	0	0	1
tolerate	1	0	0	0	0	1	0
Chemistry	0	0	0	0	1	0	1
.	2	0	1	1	0	1	0

Figure 2-8 A window-based co-occurrence matrix computed with a window size of 3

Figure 2-8 shows the co-occurrence matrix for a corpus comprising of three sentences: I love Math. I love Science. I tolerate Chemistry.

TF-IDF is another frequency-based word embedding method. Here, the idea is to understand how important a word is in a document that is part of a corpus that comprises several documents. It is a combination of the Term Frequency concept and the Inverse Document Frequency concept. Term Frequency is the number of times a term occurs in a document. Inverse Document Frequency is a logarithmically scaled inverse fraction of the documents in the corpus that contain the word. As a word appears in more documents, the TF-IDF values get close to 0. TF-IDF is always greater than 0 and positive.

Some of the well-known count-based methods are Latent Semantic Analysis (LSA, 1997 [40]), Hyperspace Analog to Language (HAL, 1996 [41]), Correlated Occurrence Analogue to Lexical Semantics (COALS, 2004 [42]) and Hellinger-PCA (2013 [43]). One of the main disadvantages of frequency-based models was that a large amount of memory is required to store the co-occurrence matrix related to a textual corpus.

2.6.2 Prediction-based word embedding models

Prediction-based methods are supervised methods and learn word embeddings by maximizing their ability to predict a word given the context or vice-versa. These are categorized into two types.

Continuous Bag-of-Words Model

In 2013, Mikolov and the team developed the continuous bag-of-words model (CBOW, [30]) for word embeddings. The model uses n number of words before and after the target word to make the prediction.

Given a sequence of words $w_1 \ldots w_n$ and target word $w_t \in V$, where n is the window size (or number of words around the target word) and t is each training step, the task is to minimize the loss function given by [30]

$$L = \frac{1}{T}\sum_{t} \log p(w_t | w_{t-n}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+n}) \tag{2.1}$$

Skip-Gram Model

In 2013, Miklov et al. developed the skip-gram model (SG, [44]). SG model is used to predict the surrounding words given a center word. The objective of a SG model is two-fold. Firstly, the log probabilities of the surrounding n words (on the right and on the left) of the target word w_t are summed together. The summation of the log probabilities is averaged over the number of training steps and then maximized. Maximization of the average of log probabilities as shown below [44]

$$L = \frac{1}{T}\sum_{t=1}^{T} \sum_{-c \le j \le c, j \neq 0} \log p(w_{t+j} | w_t) \tag{2.2}$$

The SG model is the foundation of the second architecture of the Word2Vec method created by Mikolov et al. in 2013 [30] [44]. FastText model developed by Bojanowski et al. in 2017 [45] is also based upon the SG model. The simplified versions of CBOW and SG models are shown in Figure 2-9.

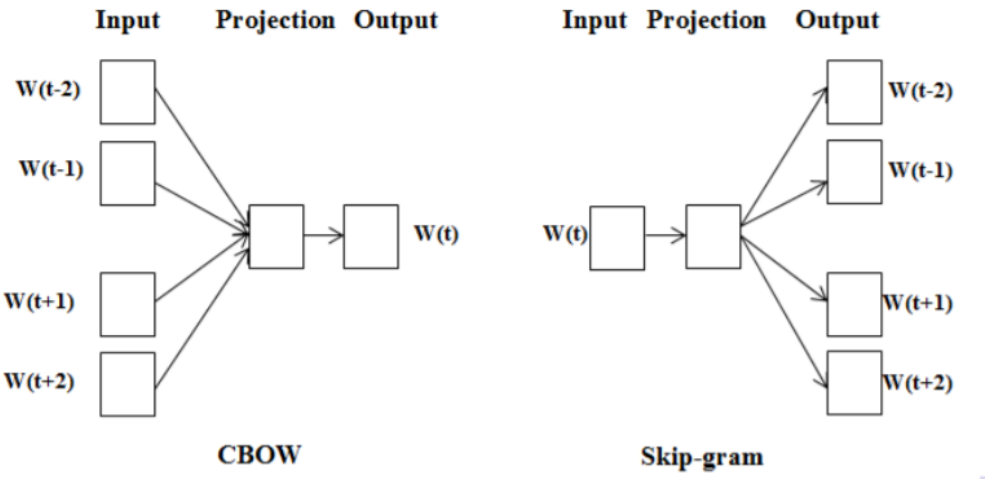


Figure 2-9 Simplified version: Continuous bag-of-words (CBOW) [30] and skip-gram (SG) architectures [30]

2.6.3 Deep contextual models

All word embeddings techniques discussed above missed an important element for fully capturing the semantic and syntactic relationships among words, i.e., the local context. The shortcoming of popular word embedding models like Word2vec is the inability to handle polysemy (polysemous words are words with two or more meanings), as a polysemous word is always mapped to only one unique vector. Hence, such models normally work well with words that represent just one unique concept or meaning (e.g., name of a country, name of fruit). These models could not differentiate between words based on the local context like usage of word bank in the context of "I went to deposit money in the bank" and "The river bank is very green." [11].

Deep contextual models overcame the problem and shortcomings faced by CBOW and SG models by leveraging the use of more complex neural network elements like LSTMs, Transformers, and by using complicated deep learning architectures to train on huge training datasets. The evolution of computational resources also led to the development of Deep contextual models.

Some of the prominent works that focused on learning context-dependent representations are Context2Vec developed by Melamud et al. in 2016 [46], CoVe developed by McCann et al. in 2017 [47], ELMO developed by Peters et al. in 2018 [5] and ULMFiT created by Howard and Ruder in 2018 [48] that all rely on the usage of bidirectional LSTM to encode the context. The introduction of Transformer architecture by Vaswani et al. in 2017 [36] resulted in a variety

of very effective pre-trained language representation models such as BERT developed by Devlin et al. in 2018 [7], XLNet by Yang et al. in 2019 [49] and ERNIE by Zhang et al. in 2019 [50], that proved their forte in solving a wide variety of NLP tasks

2.7 BERT based model: Domain-specific

Over the last couple of months, BERT has been extensively pre-trained on some domain-specific language tasks. Some of these models based on BERT are discussed here.

In 2019, Huang et al. [51] pre-trained a BERT model on clinical documents. The authors compared their new model, ClinicalBERT, with other popular word embedding models, with the help of a clinical word similarity task. They observed that ClinicalBERT shows a higher degree of understanding comparable to human evaluation than BERT on some medical-related concepts. They proved that ClinicalBERT outperforms BERT on certain clinical-related NLP tasks.

In 2019 Beltagy et al. [52] developed SciBERT, a BERT based model that is pre-trained on a dataset comprising of scientific publications. They mainly tried to investigate the effect of fine-tuning a pre-trained BERT model on downstream NLP tasks (fine-tuning based approach) vis-a-vis using task-specific architectures on the top of a frozen pre-trained word embeddings from BERT (feature-based approach). Both the fine-tuning and feature-based approaches were evaluated on three downstream NLP tasks of text classification, sequence labeling, and dependency parsing. The authors concluded that the fine-tuning method gave a better performance on language tasks than the feature-based method. Fine-tuning and feature-based approaches are discussed in section 4.4. The authors also found that fine-tuned SciBERT performed much better than fine-tuned BERT on all chosen NLP tasks.

In 2020 Lee et al. [53] introduced BioBERT, a BERT model that is pre-trained on a huge biomedical dataset. The authors fine-tuned BioBERT on three NLP tasks (named entity recognition, relation extraction and question answering) using the biomedical corpus. It was found that BioBERT outperforms BERT and other state-of-the-art language models on these tasks by a great margin. BioBERT was initialized with the original weights of BERT model.

2.8 Pre-trained language representation and Reading comprehension

RC is an important and challenging task of finding an answer to a question in a paragraph. An ideal RC system should have the ways to tracklist or enumeration, comprehend mathematical

operation, detect and resolve coreference, do logical reasoning, understand the analogy, spatiotemporal relation, causal relation, clause relation, and special sentence structure (Sugawara et al., 2017 [54]). Neural methods have gained significant popularity in the last few years as the production of large data sets allowed supervised methods to be trained and developed. Stanford Question Answering Dataset (SQuAD) is an exhaustive large reading comprehension dataset. This dataset consists of questions posed by crowd workers on Wikipedia articles. The SQuAD 1.0 (Rajpurkar et al., 2016 [55]) includes around 100,000 pairs of question-answer, where every question is answered as a piece of text from the corresponding paragraph. The SQuAD 2.0 (Rajpurkar et al., 2018 [56]) has some 50,000 new question-answer pairs, unanswerable questions written adversarial by crowd workers.

One of the first architectures proposed in BiDirectional Attention Flow (BiDAF) (Seo et al., 2016 [57]) introduced the main idea that the attention flows both ways, i.e., f from the question to the context and from the context to the question. Later on, BERT pre-trained deep bidirectional representations by working on bidirectional contexts. It is also important to mention that all the best performing models in the SQuADv2.0 leader board are all variations (fine-tuned or feature-based variations) of BERT.

2.9 Evaluation Metrics

The evaluation metrics used to evaluate the efficacy of pre-trained language models for QA tasks are discussed in this section. After applying a language model on a QA task, the predicted word offsets are used to retrieve the answer text from the paragraph. They are normalized to remove punctuations, stop words, and extra spaces and are checked for a match between the gold standard and normalized prediction strings. If both the strings match, then the Exact Match (EM) score is 1 for the question, else Exact Match (EM) is 0, an average over all the questions is calculated.

Along with Exact Match, F1 Score for each question is calculated and averaged over all the questions, using the following Equation [58]:

$$F_1 = \left(\frac{2}{recall^{-1} + precision^{-1}}\right) = 2 \cdot \frac{recall \cdot precision}{recall + precision} \tag{2.3}$$

Precision and Recall are computed based on words which are retrieved and are computed as follows [58]:

$$Precision = \left(\frac{len(tokens(prediction) \cap tokens(gold\ standard))}{length(prediction)}\right) \quad (2.4)$$

$$Recall = \left(\frac{len(tokens(prediction) \cap tokens(gold\ standard))}{length(gold\ standard)}\right) \quad (2.5)$$

Where tokens function tokenizes words from the string, and len function calculates the number of words in the string.

3. DL Model Selection

This chapter covers the choice for model selection that forms the backbone of the DL framework proposed for the task of KD from the CORD-19.

3.1 Choice of BERT Model

The field of NLP has been witnessing a growing interest ever since the emergence of the Transformer architecture by Vaswani et al. in 2017 [36], which completely changed the NLP task of machine translation. BERT was one of the first DL language models that were built using Transfomer encoder to perform a variety of NLP tasks. New DL language models are being developed almost every month by tweaking some aspects of BERT architecture to improve results in NLP tasks. Some of the recently developed models based on BERT are GPT by Radford et al. in 2018 [6], Transformer-XL by Dai et al. in 2019 [59], XLM by Lample and Conneau in 2019 [60], GPT-2 by Radford et al. in 2019 [39], ERNIE by Zhang et al. in 2019 [50], XLNet by Yang et al. in 2019 [49], RoBERTa by Liu et al. in 2019 [61], ERNIE 2.0 by Sun et al. in 2019 [62] and CTRL by Keskar et al. in 2019 [63]. BERT has been chosen for the proposed Framework (Chapter 5) because of the reasons that are discussed here.

3.2 BERT: Driver of Google Search Engine

Google Search convincingly dominates the internet search engine market with an approximate market share of 92%, as shown in Figure 3.1, based upon the data from statscounter website. In October 2019, Google made the most significant change in the algorithm of its search engine. The change was termed as “the BERT update.”

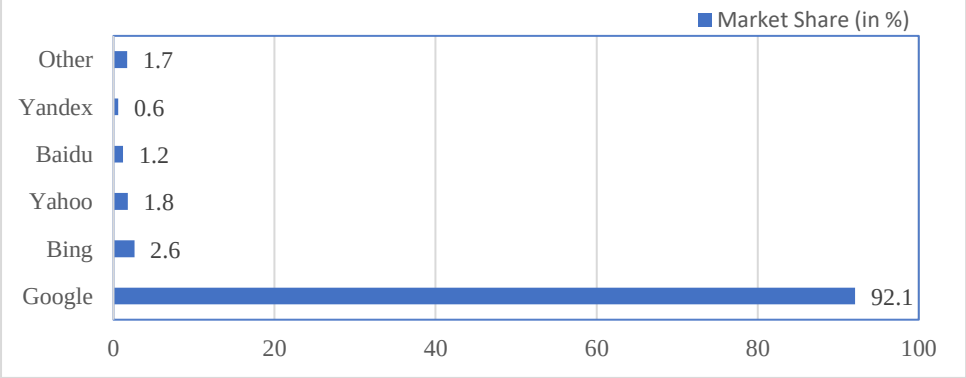


Figure 3-1 Market Share of Search Engines[1]

[1] https://gs.statcounter.com/search-engine-market-share

The result of this change is in front of everyone to see. The fact that the most used and the best ever search engine in the market makes BERT as its backbone is an indication that is strong enough to take a greater interest in BERT and try using it for the Thesis.

3.3 BERT as a Knowledge Base

Recently, many research and publications have been done on the possibility of using the state-of-the-art pre-trained BERT model as a factual knowledge base. Liu et al., in 2019 [64], developed a knowledge base question-answering model by using pre-trained BERT architecture. The model developed by them leverages prior linguistic knowledge that resides in BERT to obtain deep contextual representations. Their experimental results using a pre-trained BERT language model showed state-of-the-art accuracy on QA datasets. In 2019, Petroni et al. [65] established that a pre-trained BERT model already contains relational oracle knowledge comparable to traditional NLP models, thus demonstrating a huge potential of using BERT as a Knowledge base in an unsupervised QA system. In 2019 Poerner et al. [65] supplemented this evidence by proving BERT's good performance on QA datasets is due to BERT's inherent understanding of entity names instead of factual knowledge. Further, in 2019, Peters et al. [66] devised a method to immerse multiple knowledge bases (like WordNet and Wikipedia) into BERT to improve their representations with human-curated and structured knowledge. Their model KnowBERT which was enhanced with further knowledge demonstrated improved performance over BERT in the QA task.

The encouraging results discussed above further strengthened the choice of using BERT in this Thesis. The novel BERT-based Framework discussed in section 5.1 could then be used to solve the QA task for KD discussed in Section 1.1.

3.4 Optimum Model Size of BERT

There is a false misconception that the bigger the DL language model, the better the performance. Recently, NLP researchers have been trying to implement more massive and larger language models in order to increase accuracy. Figure 3-2 shows a comparison between various NLP models and the number of parameters used by them. This trend towards creating bigger models is worrisome. Firstly, big models hinder democratization, and secondly, they restrict scale.

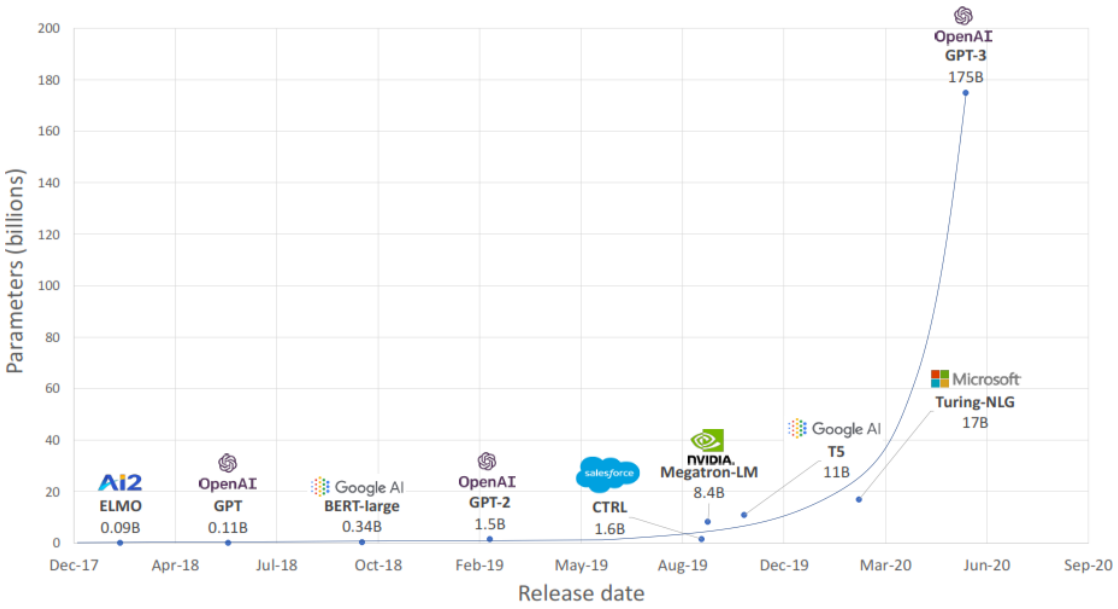


Figure 3-2 Popular Pre-trained Language Models and their parameter count [67]

For this Thesis, as only limited computational resources are available, the candidate models are, i.e., BERT, XLNet, RoBERTa, and ERNIE 2.0 [67].

The extensive availability of documentation on BERT, its open-source nature, and the publication by Liu and team in 2019 [64] that confirmed that BERT is as competitive as some of the other recently released architectures (such as GPT, XLNet) led to the use of BERT-large (architecture discussed in Section 4.2) in this Thesis.

Pre-trained model RoBERTa-large (Liu et al.,2019 [61] are also used as a Knowledge base in the proposed QA framework for the Thesis. RoBERTa is further built on BERT's language masking training task. It modifies key hyperparameters in BERT, including removing BERT's next-sentence pre-training objective and training with much larger mini-batches and learning rates. RoBERTa is also trained on data with an order of magnitude more than BERT and for a more extended amount of time. This training helps RoBERTa in learning representations to generalize even better to downstream tasks compared to BERT.

4. State-of-the-art: BERT

This chapter gives an overview of the state-of-the-art DL model that is the backbone of some recent and most advanced NLP methods. BERT (Devlin et al., 2018 [7]) is the most recent advanced deep learning NLP model developed by Google. BERT model is pre-trained by using deep and bidirectional representations of words from textual data that is unlabeled. It is conditioned jointly on the left and right contexts (as language understanding is bidirectional) in all its layers. Due to the nature of the training undergone by BERT, the pre-trained BERT model can now be fine-tuned with the use of just one additional output layer to cater to a large variety of NLP tasks and provide impressive results that would change the face of the NLP domain (Figure 4-1).

In this chapter, section 4.1 covers the self-attention concept adopted in BERT. Section 4.2 contains a description of the BERT model, architecture, input representations, and hyper-parameters. Section 4.3 explains the pre-training procedure employed by the model. Section 4.4 covers different methods of using the model in downstream NLP tasks.

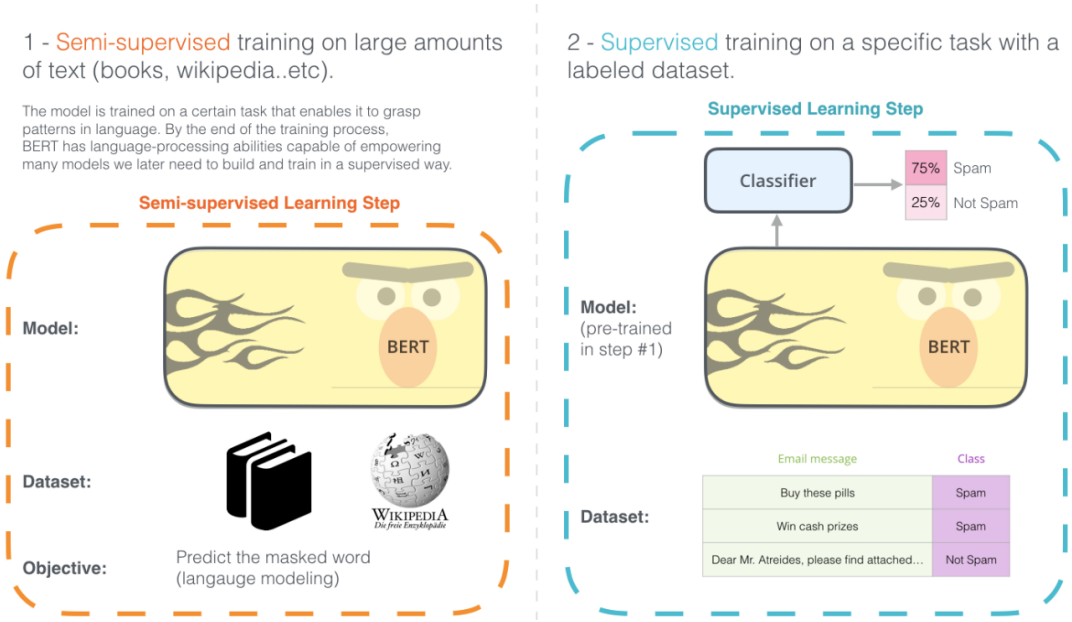


Figure 4-1 Learning steps in BERT (Jay Alammar, 2018 [68])

As shown in the figure above, the learning steps in BERT comprises of two-steps:

a) Pre-training: It encompasses self-supervised training on large amounts of text from books & Wikipedia)

b) Fine-tuning: This step involves supervised training on a specific downstream NLP task with the help of a labeled dataset

4.1 Concept of Self-attention

Attention was first introduced in 2014 by Bahdanau et al. [38]. Self-attention is a special form of attention mechanism. It was first introduced in 2017 by Vaswani et al. in the Transformer arc model ([36]). In simple words, self-attention is an attention mechanism that relates to different positions of a single sentence (or sequence) to generate a contextual representation of each word in that sentence (or sequence), as shown in Figure 4-2. This example below shows how the attention mechanism links “animal” to the word “it” in the sentence “The animal didn’t cross the street because it was too tired.”

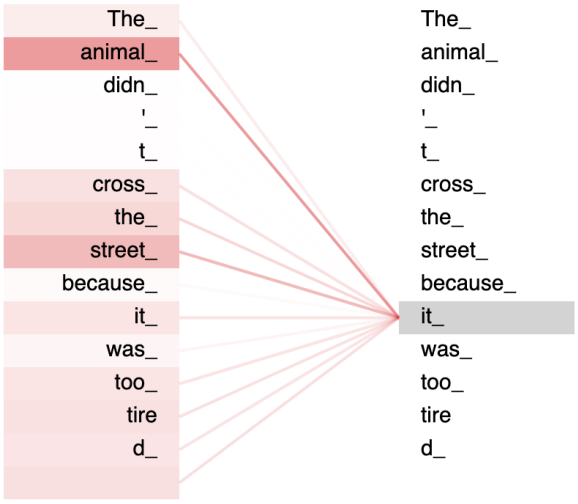


Figure 4-2 Self-attention mechanism ([68])

4.1.1 Multi-head Attention

Transformer based models use the concept of “multi-head” attention. It refers to the use of self-attention mechanism n-number of times instead of just once. Multi-head attention helps BERT in learning complex language representation from different spaces or dimensions. It improves the model’s performance in two ways:

- The model is able to focus on a greater number of different word positions
- The attention layer gets multiple representation spaces

4.2 BERT Model

4.2.1 Architecture

BERT model comprises of a stacked Transformer Encoder as designed by Vaswani et al. in 2017 [36]. The encoder part of the Transformer (refer Figure 4-6) is made up of two sublayers, i.e., a multi-headed self-attention mechanism layer and a fully connected feed-forward neural network layer. Residual connections (He et al., 2016 [69]) are deployed around each of these two sub-layers, followed by a normalization layer. Now, for any input representation x, the output of each of the two sub-layers would be LayerNorm(x + Sublayer (x)), where Sublayer(x) is the function implemented by the sub-layer itself. The feed-forward network uses Gaussian Error Linear Unit activation developed by Hendrycks and Gimpel in 2016 [70].

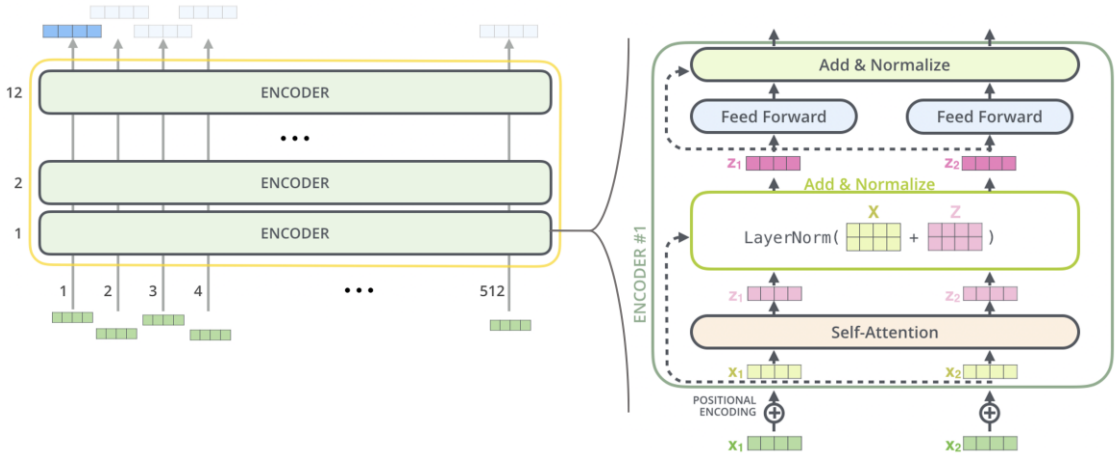


Figure 4-3 Architecture of the Transformer Encoder ([70])

4.2.2 Input

The input to the BERT model comprises of summation of tokens, segment and positional embeddings for representing the cases (downstream NLP tasks) of both single and a pair of sentences in concrete and unambiguous manner.

The input sentence is tokenized with WordPiece embeddings developed by Wu et al. in 2016 [35]. Wordpiece embeddings are created by breaking words into word pieces as per the word unit inventory. Special word boundaries are then added before commencing the training of the model. This ensures that the original sequence of words can be recovered later on from the Word piece sequence without confusion or ambiguity. An example of a word sentence and its corresponding Wordpiece tokens are provided below:

Word tokenizer = Deer is an unpredictable animal

Wordpiece tokenizer = Deer is an un ##predict ##able animal

Apart from Wordpiece tokens, three special types of token embeddings are also used, [CLS], [SEP], and [MASK]. [CLS] token is the first token that is appended at the beginning of every sentence. [SEP] token is a separator and is used to separate two input sentences. [MASK] is used for Mask Language Training objectives.

Segment Embeddings are for NLP tasks where a pair of sentences are needed for training. As for the learned segment embeddings, a sentence A embedding is added to every token on the first sentence and a sentence B embedding to every token on the second sentence. In the case of single-sentence inputs, only A embeddings are used. Positional embeddings are used to capture information related to the relative position of tokens in an input sentence. The learned positional embeddings support a sequence length of up to 512 tokens. The total input length is thus limited to 512 number of tokens. A visual representation of the input embeddings (token, segment, and position) is given in Figure 4-7.

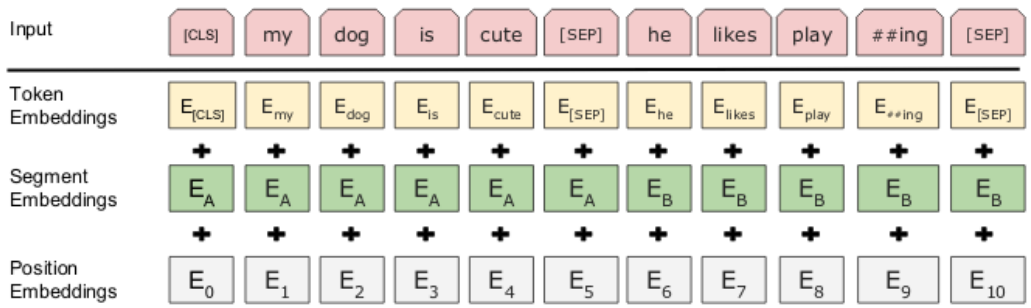


Figure 4-4 BERT input representation ([70])

4.2.3 Parameters

Two BERT models and their important parameters as made available by the authors [7]:

- $BERT_{BASE}$ (BERT-base): with 12 transformer encoder layers, a hidden size of d (dimensionality of input and output of each layer) = 768 and h (number of attentions heads in a self-attention sublayer) = 12 self-attention heads. The total amount of parameters in this model are 110M parameters.
- $BERT_{LARGE}$ (BERT-large): with 24 transformer encoder layers, a hidden size of d = 1024 and h = 16 self-attention heads. The total amount of parameters in this model are 340M parameters.

4.3 Pre-training procedure

BERT has been pre-trained simultaneously on two tasks, i.e., masked language modeling (MLM) and the next sentence prediction (NSP). The training loss is the sum of the likelihood in both the tasks.

MLM is the task of predicting certain words that are randomly masked in the input sequence. It is different from the traditional Language Modeling (LM) task. In LM, the aim is to predict the next word based upon a input sequence of words. The bidirectionality of BERT helps in achieving the MLM objective. For models like BERT, traditional conditional language modeling is not a training objective. The bidirectionality in BERT allows each word to see itself indirectly, and BERT would be trivial in predicting the target word in a multi-layered complicated context. Therefore, BERT is trained using MLM.

NSP is a binary classification task. The model receives a pair of sentences as input. BERT is trained on correctly classifying the second sentence (in the pair of input sentences) as a subsequent sentence. Training on NSP task helps BERT in learning further nuances in the language that it didn't learn while being trained on MLM tasks. NSP training is important for downstream NLP tasks like QA and natural language inference.

4.4 Downstream tasks

Pre-trained language models can be used in various NLP tasks in two ways that are further discussed in this section. These approaches can be either fine-tuning or feature-based. The fine-tuning approach required just minimal task-specific parameters in the model. The model is trained on the downstream NLP tasks by simply fine-tuning all existing pre-trained parameters.

4.4.1 Fine-tuning approach

BERT's two pre-training objectives allow it to be used on a single sequence or sequence pair tasks without making substantial task-specific modifications in the architecture. For each task, task-specific inputs and outputs are to be fed to BERT, and the model parameters need to be fine-tuned for a few epochs. The process of fine-tuning of BERT on different NLP tasks is shown in Figure 4-8 [7].

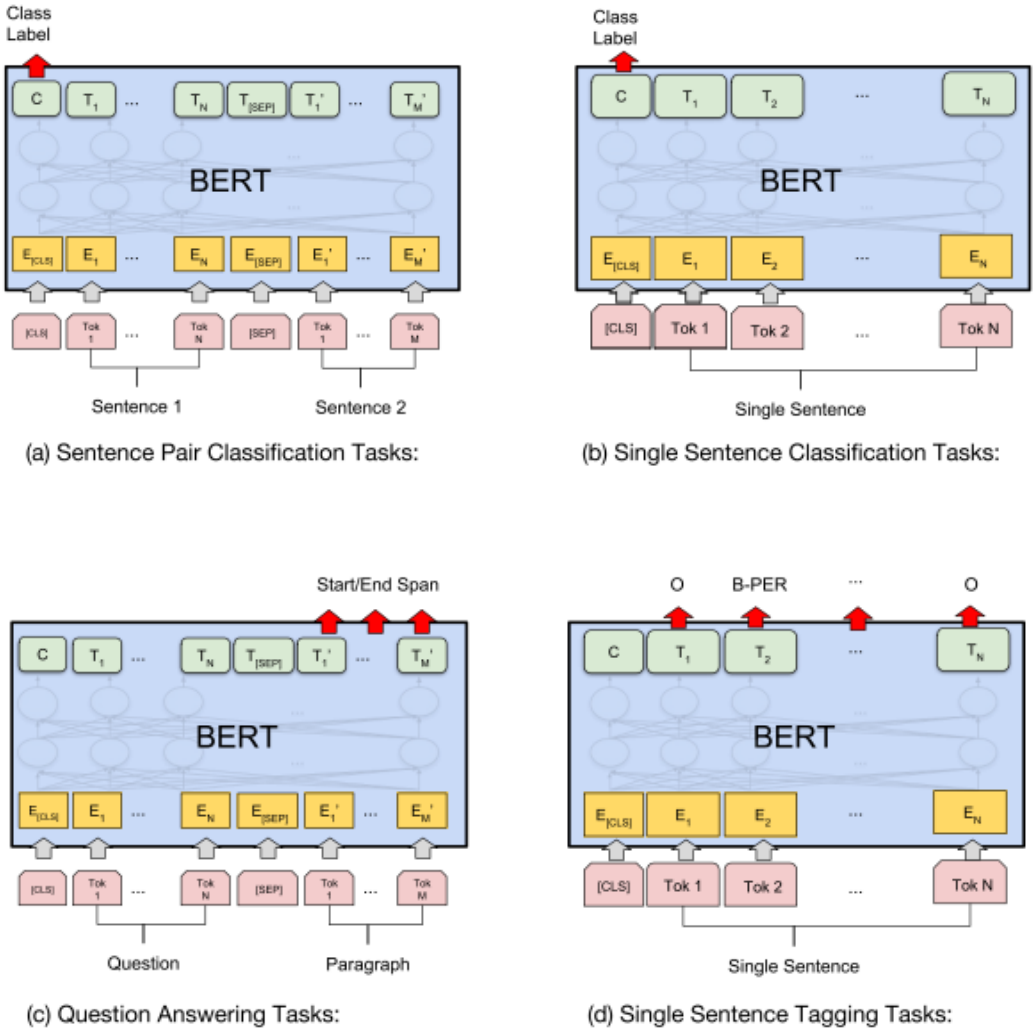


Figure 4-5 Fine-tuning BERT on different tasks (Devlin et al.,2018 [7])

During the input, sentence A and sentence B in pre-training are:

a) hypothesis-premise pairs in textual reasoning or entailment;
b) sentence pairs in translation or paraphrasing;
c) question-passage pairs in QA task;
d) a degenerate text-∅ pair in sequence tagging or text classification.

During the output, the token representations from BERT are fed to an output layer for relevant token-level tasks like question-answering, machine translation. The [CLS] representations are fed to an output layer for text classification tasks like textual entailment, sentiment analysis.

4.4.2 Fine-tuning for QA

For this fine-tuning approach, given a question and a passage containing the answer, the system will have to predict the answer text span in this passage. The input representation is done by merging the question and passage into one sequence. The question and passage are then separated by using the A embeddings for the question and the B embeddings for the passage. In the cases where the input sequence length is larger than 512 tokens, a sliding window is used for feeding the data to the network, and then, the span with the largest context is greedily chosen. Even if in the fine-tuning step all the parameters are updated, there are only two new vectors to be learned for answer span spotting: a start vector $S \in R^d_{model}$ and an end vector $E \in R^d_{model}$, being d_{model} the size of the hidden dimension. The probability of each token to be the start token is computed by the following SoftMax:

$$P_i = \frac{e^{S.T_i}}{\Sigma_i e^{S.T_i}} \tag{3.1}$$

where, $T_i \in R^d_{model}$ denotes the final hidden vector from BERT for the i^{th} input token. The same formula is used for the end token. The maximum scoring span is taken as the prediction.

4.4.3 Feature-based approach

BERT is also used in tasks that require a feature-based approach. Using BERT as a feature-based model is synonymous with using BERT for feature engineering i.e. creating more important features for further end-user application. In the feature-based approach, initially, word representations are extracted from a pre-trained BERT. These extracted features serve as inputs to other task-specific models. Two advantage of using BERT as a feature-based approach over fine-tuning approach are as follows:

- All the NLP tasks cannot be performed by just one architecture (like BERT, GPT, XLNet). There is always a requirement for creating task specific architectures. Feature-based approach helps this cause of creating different architectures for different NLP tasks.
- From the point of view of optimizing computational resources, it is always helpful to create complex and expensive features from the training data just once, and then run experiments on these expensive representations using simple and light models. The feature-based approach is helpful in these kinds of analytics scenarios.

The feature-based approach forms a part of the proposed Framework.

5. Proposed Framework

5.1 Semantic QA Framework for Knowledge Discovery

A state-of-the-art AI framework is developed for the task of knowledge discovery from the CORD-19. The framework is referred to as "Semantic QA Framework for Knowledge Discovery." (referred to as "Framework")

The Framework extracts the most scientifically relevant publications regarding user queries on COVID from CORD-19. It analyzes and summarizes information at the sentence level using multiple advanced NLP methods (CBOW, TF-IDF, sentence embedding using BERT) and deep learning methods.

The Framework is made up of the following key components:

a) NLP driven AI search engine

   A state-of-the-art AI search engine using advanced machine learning and text mining methods like CBOW, TF-IDF and cosine similarity scores is implemented. In the first step, all relevant information user-defined keywords are filtered out and collated. It is followed by evaluation and score (cosine similarity) matching of extracted information. The final output from the search engine is evaluated using word cloud and n-gram implementation.

b) Semantic deep-learning search engine

   The second component involves developing a semantical deep learning-based search engine to extend the framework's ability to extract information at the sentence level using sentence embeddings from BERT. Further insights and information are retrieved from the dataset based on the user queries from the specific scientific area.

c) Visualization using Knowledge graphs

   The third component comprises of deploying a knowledge graph method on the outcome of sentence embeddings to visualize the top-level publications relevant to various user queries. Operations like Sentence Segmentation, Entities Identification and Relations Extraction are carried out to build the knowledge graph. The knowledge graphs are built using limited entities due to computational constraints to show the potential for the final refinement or scale-up in future work.

The Framework is a robust, modular, and analytical implementation with reasonable accuracy based on well-defined keywords and queries from the end-user. The research community can use this Framework for efficient knowledge retrieval and QA task from CORD-19.

5.2 Steps in the Framework

The diagram of the Framework is provided in Figure 5-1. The Framework comprises eight steps that are further discussed in this section.

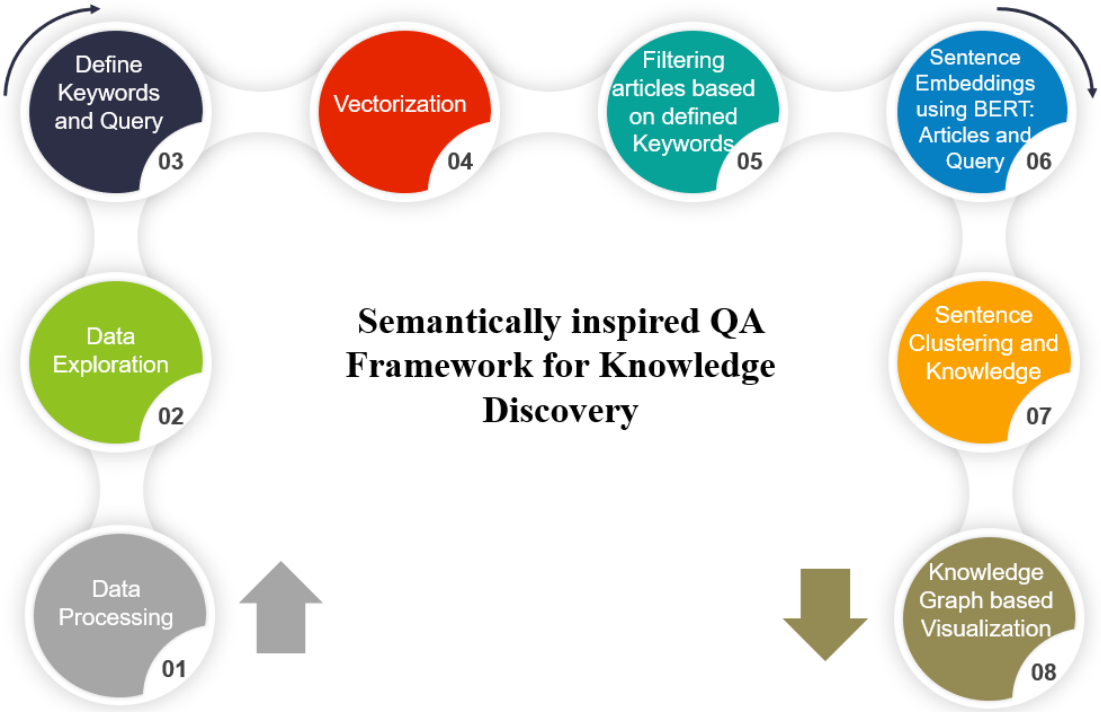


Figure 5-1 Framework Diagram

a) Data Processing

Only English language texts are used for the application of the framework. Data is cleaned and tokenized in this step. The data cleaning process involves the following:

- Replace brackets
- Replace contractions
- Lower the case
- Replace commas
- Lemmatizing
- Stemming

- Remove number words
- Remove punctuations
- Remove stop words

Stemming and Lemmatization mentioned above have the same common aim of reducing the inflectional forms of each word into a common base or root. Stemming cuts off the beginning or the ending of a word, taking into account a list of common prefixes and suffixes, whereas Lemmatization takes into account the morphological composition of the words. After the data has been cleaned, the data (i.e., abstract only) is tokenized.

b) Data Exploration

- Word cloud

  The text corpus that was created after pre-processing is visualized using a word cloud to get insights into the most frequently used words
- Visualize top N uni-grams, bi-grams & tri-grams using vector of word counts

  Vector of word counts based on bag of words model is created, which ignores the sequence of the words, and considers word frequencies only. Using the vectors that are created, vocabulary is built. This allows in visualizing the top 20 unigrams, bi-grams and tri-grams and gain insight from overall articles.

c) Define Keywords and Query

After combining the results of the data exploration and basic domain knowledge (gathered using google search in this thesis), some Keywords are defined, and a refined Query is prepared. Keywords are used to perform searches on the dataset using cosine similarity based on TF-IDF scores. Keyword filtering is a great method for efficient selection and shortlisting of high-quality content from the corpus. The Query is used for creating sentence embedding using BERT and for comparison with the sentence embeddings of shortlisted high-quality papers generated using Keywords. The query does not have to be in the form of a question.

d) Vectorization

TF-IDF scores for the word vector (created for abstract in step b above) are computed. A high weight in TF-IDF is reached when there is high term frequency in the given document and the low document frequency of the term in the whole collection of documents. Thus,

TF-IDF helps to find frequent words in the publications that are not so frequent in the whole dataset. These specific words can be highlighted and interpreted appropriately.

e) Filtering articles based on Keywords

A search engine using cosine similarity between the Keywords and the vocabulary created using abstracts of the papers is implemented. The matching score is based on TF-IDF that was computed for the word vectors in step d above. The goal is to extract more relevant papers and limit the investigation space for sentence embeddings using BERT.

f) Sentence Embeddings using BERT on Articles and Query

Every sentence of the papers is embedded that have been marked as interesting by the TF-IDF search engine. This is done for the abstracts of the papers and the combined text (abstract + body text) of the papers. Subsequently, the Query (from step c above) is also embedded. Sentence embedding is performed by using BERT-large and RoBERTa-large models.

Every sentence in the combined text is compared to the Query in embedding space using the cosine similarity measure. All the sentences in the combined text are ranked based on cosine similarity measure with Query, and top 50 sentences are finally selected. Associated paper id, relevant text from the paper Top 50 sentences and cosine similarity is saved as a result for each of the Queries.

g) Sentence Clustering and Knowledge

It is possible to cluster the embedded sentences (created in step f above) in embedding space and print sentences associated with each found cluster. Thus, a form of knowledge (cluster-wise) is created by the machine.

h) Knowledge Graph-based visualization

The outcome of the sentence embedding is visualized using the concept of a knowledge graph where top-ranked sentences are decomposed into a set of nodes and edges to represent essential relationships between entities. Those entities were identified by parts of speech (POS) tags, and parsing out the dependency tree of the sentence, such as nouns, are treated as nodes and verbs as relationships.

6. Experimentation and Result

This chapter covers the experiments conducted on the CORD-19 and the results obtained during the analysis. Various tools and software used are also identified.

6.1 Environment setup

6.1.1 Programming environment

All the programming in this thesis is done in Colab Notebook. Colab is a free Python development environment that runs in the browser using Google Cloud

6.2.2 Python packages

Important python libraries that were used in implementing the framework are as follows:

a) NLTK

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries.

b) NUMPY

NumPy is an open-source package and project that aims to enable numerical computing with Python. It was developed in 2005 and is built on some early work of the Numerical and Numarray libraries.

c) PANDAS

Pandas is a Python package that provides fast, flexible, and expressive data structures designed to make working with structured (tabular, multidimensional, potentially heterogeneous) and time-series data. It is the fundamental and high-level building block for carrying out real-world data analysis in Python.

d) SCIKIT-LEARN

Scikit-learn is a Python module for machine learning built on top of SciPy and is distributed under the 3-Clause BSD license.

e) SPACY

It is a library for advanced NLP in Python and Cython language. It is built using high research and is designed to be used in real products from day one. It comes loaded with pre-trained statistical models and various word vectors. Spacy currently supports tokenization in more than 60 languages.

f) SENTENCE-TRANSFORMERS

This framework provides an easy method to compute dense vector representations for sentences and paragraphs (also known as sentence embeddings). The models are based on transformer networks like BERT / RoBERTa / XLM-RoBERTa and are tuned specifically meaningful sentence embeddings such that sentences with similar meanings are close in vector space.

g) MATPLOTLIB

Matplotlib is an important library for creating static, animated, and interactive data visualizations in Python.

h) PYTORCH

It is a package that provides mainly two high-level features:

- Computation of Tensors with strong GPU acceleration
- Deep neural networking built on the top of a tape-based autograd system

i) PANDARALLEL

It is an easy to use library to speed up computation (by parallelizing on multi CPUs) with pandas.

j) PICKLE

The pickle library implements binary protocols that are used for serializing and de-serializing a Python object structure. Pickling is used to store python objects lists, dictionaries, class objects, and more.

6.2 Dataset: CORD-19

CORD-19 (Wang et al., 2020 [1]) is a growing resource of scientific papers on COVID-19 and related historical coronavirus research. On 16th March 2020, the Allen Institute for AI, in

collaboration with The White House Office of Science and Technology Policy, the National Library of Medicine, the Chan Zuckerburg Initiative, Microsoft Research, and Kaggle, coordinated by Georgetown University's Centre for Security and Emerging Technology, released the first version of CORD-19. Papers and preprints from several archives are collected and ingested through the Semantic Scholar literature search engine, metadata are harmonized and deduplicated, and paper documents are processed through the pipeline established in Lo et al. in 2020 [71] to extract full text in JSON format. Figure 6-1 shows the broad process of creating CORD-19.

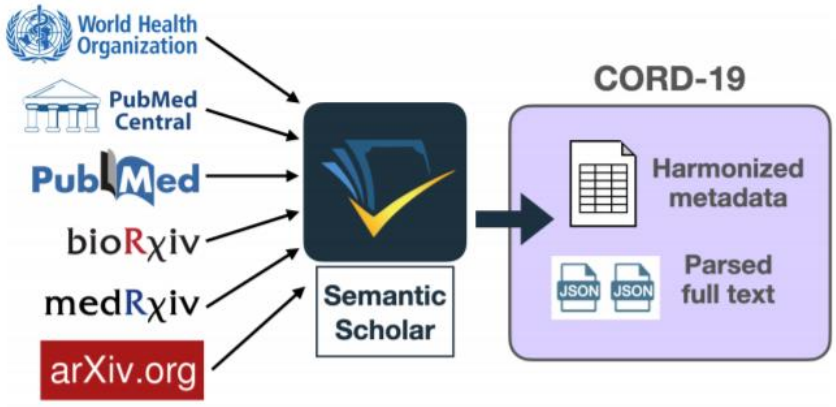


Figure 6-1 Papers and preprints are collected from different sources through Semantic Scholar [1]

6.3 Implementing the Framework on CORD-19

6.3.1 Schematics

The steps in the Framework are schematically described. Figure 6-2 explains the process involved in steps from Data Processing to Filtering papers/articles based on Keywords. Figure 6-3 covers steps involved from creating sentence embeddings using BERT on shortlisted articles and the Query to visualizing Knowledge Graph.

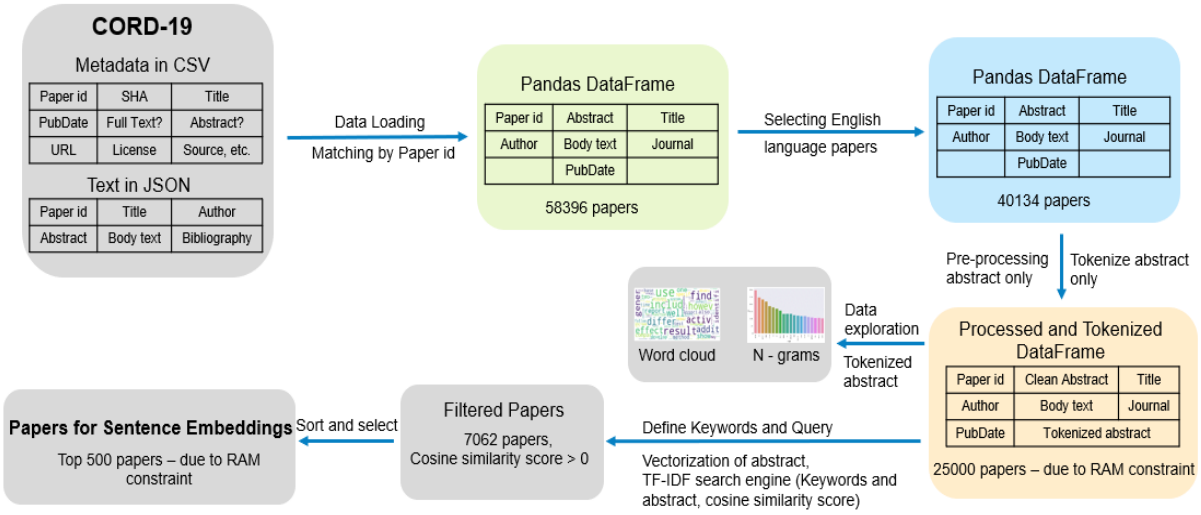


Figure 6-2 Schematics, from Data Processing to Filtering Articles

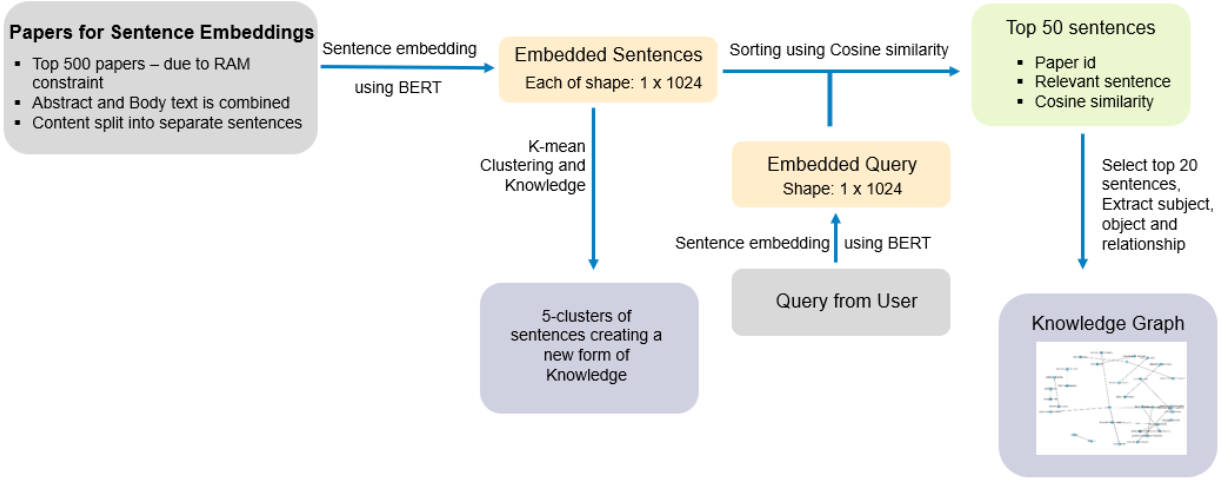


Figure 6-3 Schematics, from Sentence Embeddings to Knowledge Graph

6.3.2 Colab Notebook containing the implementation

The Colab notebook containing the implementation of the Framework is provided in "Appendix A" of this report. All the eight steps are clearly identified in the Colab notebook for detailed logical analysis.

6.3.3 Discussion on implementation

Various steps and intermediate visualizations while implementing the framework in Colab Notebook are as follows:

a) **Data Loading**

The metadata and all .json files are loaded. The meta information and article text are merged into a single pandas dataframe. The metadata is read into a pandas dataframe. The metadata contains the following information. A part of this information is chosen and merged with the actual text. The metadata comprise of the following variables: cord_uid, sha, source_x, title, doi, pmcid, pubmed_id, license, abstract, publish_time, authors, journal, Microsoft Academic Paper ID, WHO #Covidence, has_full_text, full_text_file and url. Eack of json file contains the following information: paper_id, title, author, abstract, body_text, bib_entries, ref_entries.

Data comprises of 58396 papers/publications.

b) **Data Processing**

As the analysis is based on the English language, it is first checked if documents are all in English with the library, langdetect. Each of the abstracts is checked for language. Not all abstracts/documents are in English, e.g., some are in Spanish.

The data is processed to clean the text with the cleaning process identified in Section 5.2a. Word level and sentence level pre-processing is carried out by libraries like contractions, lancasterstemmer from nltk, wordnetlemmatizer from nltk. Both abstract and body-text are tokenized with functionality in nltk package (nltk.word_tokenize).

Data was left with 40134 papers in the English language. However, due to computational limitations, only initial 25000 papers are selected for the next steps and further analysis.

c) **Data Exploration**

Data exploration is performed by using the wordcloud and n-gram frequencies.

Word cloud: The text corpus created after pre-processing is visualized to get insights on the most frequently used words. The word cloud created using the library “wordcloud” is shown in Figure 6-4.

Figure 6-4 Word Cloud created from Abstracts

Visualize top 20 uni-grams, bi-grams & tri-grams: Vectors of word counts based on the bag of words model are created (using countvectorizer method in sklearn library), which ignores the sequence of the words and only considers word frequencies. The fit_transform function is evoked on the vector that was created to learn and build the vocabulary. This allows for the visualization of the top 20 unigrams (Figure 6-5), bi-grams (Figure 6-6) and tri-grams (Figure 6-7), and gain insight from overall articles.

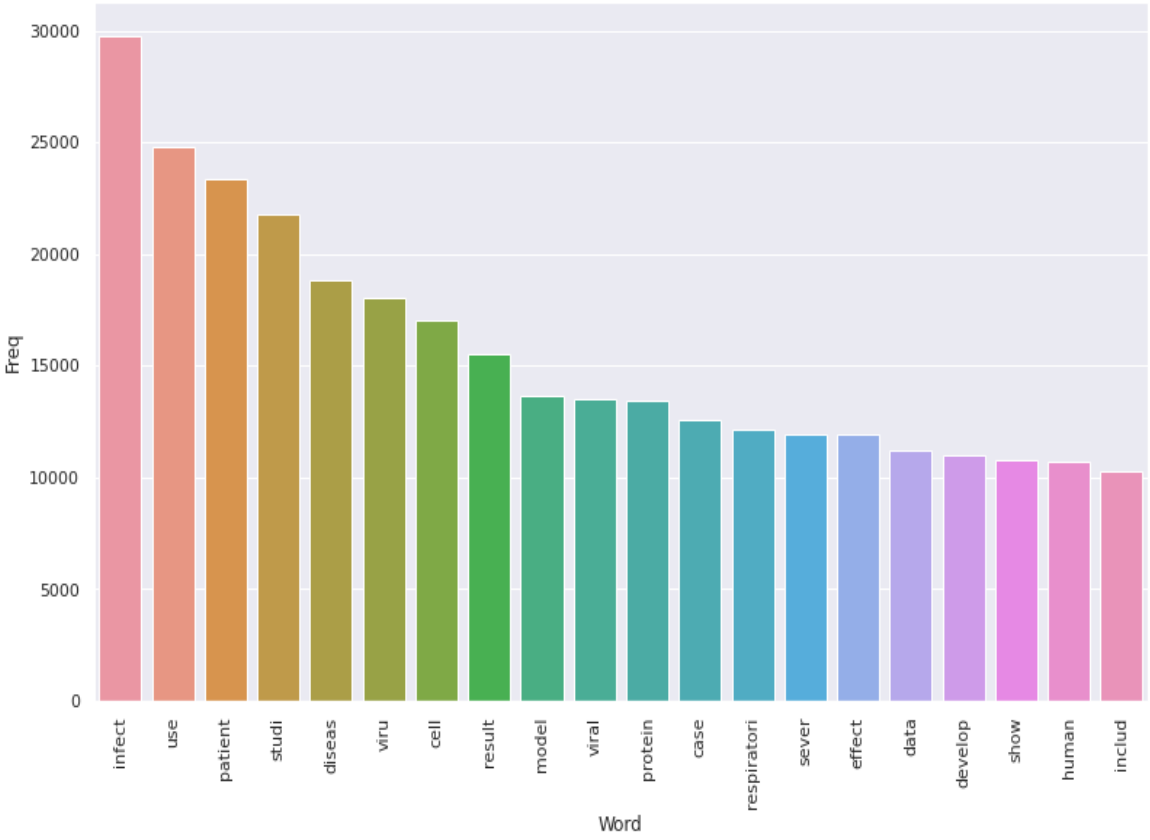


Figure 6-5 Top Unigrams

It can be seen from Figure 6-5 that “infect” is the most used word in the abstract, followed by “use” and “patient.”

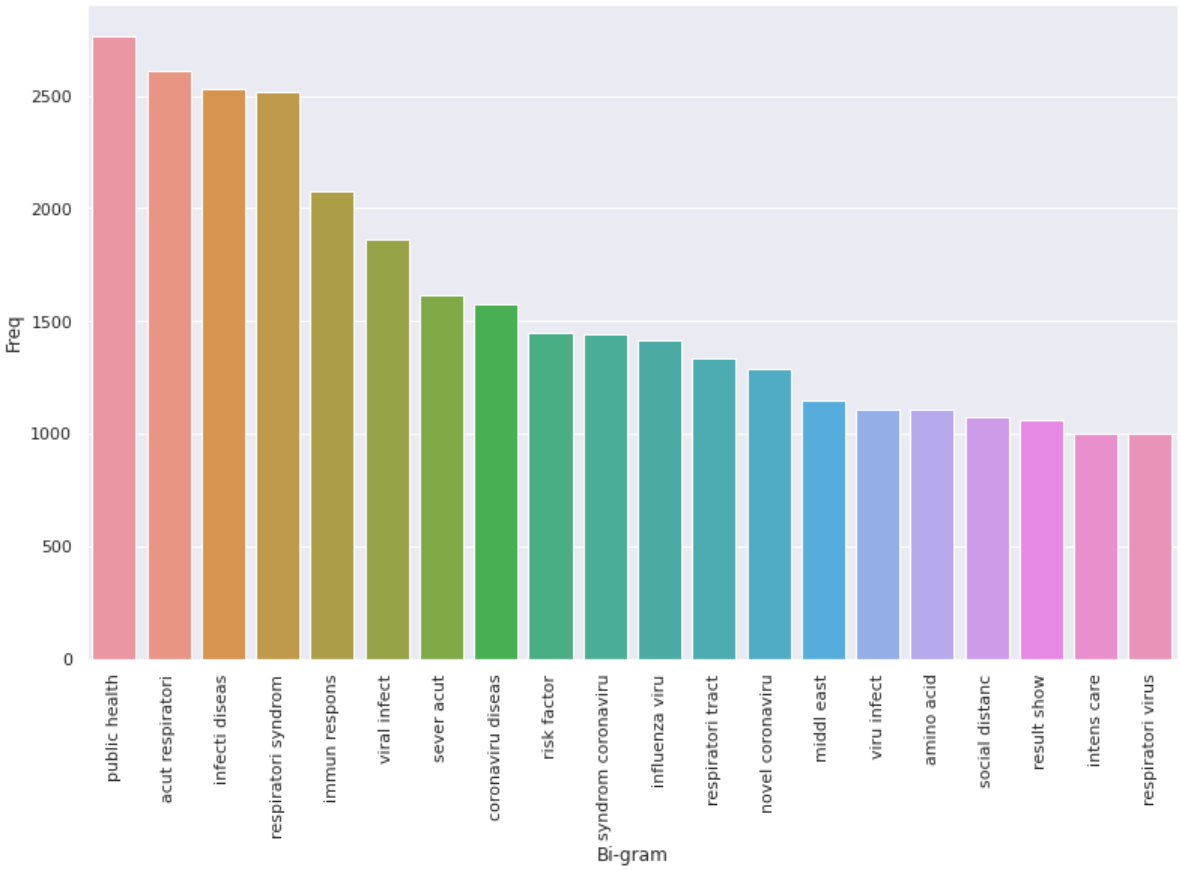


Figure 6-6 Top Bi-grams

It can be seen from Figure 6-6 that “public health” is the most used bi-gram in the abstract, followed by “acut respiratori” and “infecti diseas.”

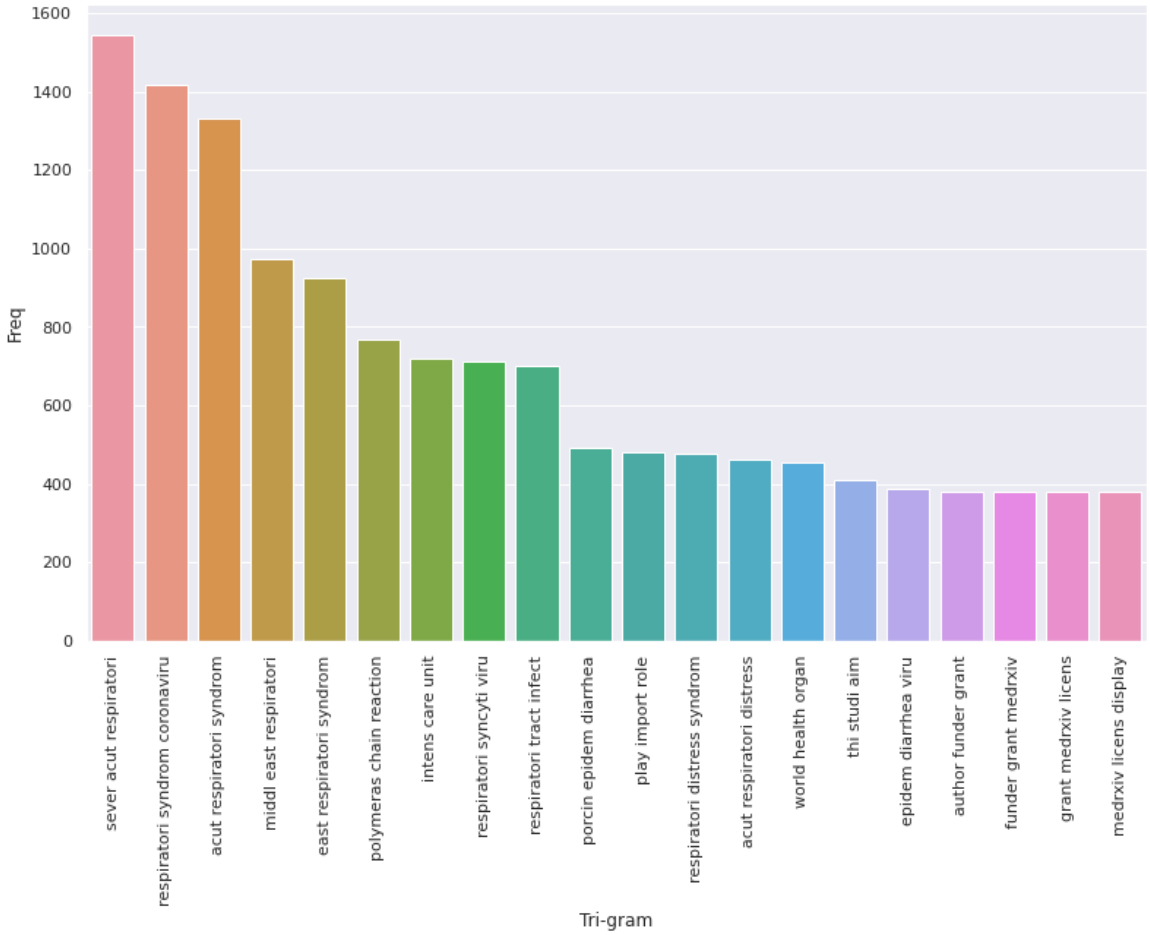


Figure 6-7 Top Tri-grams

From Figure 6-7, it is observed that “sever acut respiratori” is the most used tri-gram in the abstract, followed by “respiratori syndrome coronaviru” and “acut respiratori syndrome.”

d) **Define Keywords and Query**

After combining the data exploration and basic domain knowledge, Keywords can be defined, and the Query can be prepared. The approach for forming Keywords and Query is discussed here.

For example, Keywords and a Query are to be framed for a question related to what is known about the virus origin and management measures at the human-animal interface?

In this question, the focus is on the animal host and any evidence of continued spill-over to humans. Keywords for the initial run were: SARS-CoV-2, Covid-19, HCoV-19, Covid corona, 2019-nCoV, sars, cov2, ncov wuhan, coronavirus, pneumonia. Keywords perform a search on the dataset using cosine similarity based on TF-IDF scores and help in shortlisting of high-quality content from the corpus.

An example of Query for sentence embedding using BERT is " Evidence of SARS-CoV-2 infection in animals and its transmission or spill-over to other hosts, including humans ".

Various Queries (as identified in Section 1.4) that were formed for sentence embeddings are presented in Table 6-1. These Queries are just indicative. The framework is capable of handling any kind of Query related to CORD-19.

Question	**Query**
Question1: What is known about the origin of virus and management measures at the human-animal interface?	Evidence of SARS-CoV-2 infection in animals and its transmission or spill-over to other hosts, including humans
Question 2: What has been published about medical care?	Knowledge of the frequency, manifestations, and course of extrapulmonary manifestations of SARS-CoV-2, including, but not limited to, possible cardiomyopathy and cardiac arrest
Question 3: What is known about the transmission, incubation, and environmental stability of the coronavirus?	Incubation periods for the corona disease in humans, how this varies across age and health status, how long individuals are contagious, even after recovery
Question 4: What is known about COVID-19 risk factors?	Data on potential risk factors: Transmission dynamics of the virus, including the basic reproductive number, incubation period, serial

	interval, modes of transmission and environmental factors
Question 5: What has been published regarding research and development and evaluation efforts for developing vaccines and therapeutics?	Capabilities to discover a therapeutic for the disease, and clinical effectiveness studies to discover therapeutics, including antiviral agents

Table 6-1 Query for QA task on CORD-19

Format of Query

Answers are provided to queries that are in the English language. The format of questions can be divided into two groups: interrogative sentences (e.g., "What has been published about medical care?") and keyword-based queries (e.g., "medical care against COVID-19"). The Framework can provide answers to all the natural language questions irrespective of their format, as they all reflect the need for an answer.

e) **Vectorization**

TF-IDF scores are calculated based on the word vectors created in step c above. Words with top TF-IDF vectors are also analyzed to get some insight into the vocabulary. TfidfVectorizer functionality in sklearn library is used to generate TF-IDF scores.

f) **Filtering articles based on Keywords**

A search engine based upon TF-IDF and cosine similarity with bag-of-word model is implemented. The goal is to extract more relevant information and limit the investigation space within the range of COVID-19 related articles. The keyword used for matching is "SARS-CoV-2 Covid-19 HCoV-19 Covid corona 2019-nCoV sars cov2 ncov wuhan coronavirus pneumonia". This keyword is used to ensure selection screening of all the requisite publications in the CORD-19. The matching score based on TF-IDF and cosine similarity provides a quantitative way to evaluate the search. All articles based on the matching score are sorted. A list of the most relevant (top 500) articles is generated. Sklearn library is primarily used in this step.

g) **Sentence embeddings and BERT**

Every sentence of the papers is embedded (using BERT-large and RoBERTa-large) that are marked as interesting by the TF-IDF search engine in step f above. After

embedding the sentences of the abstracts, combined text and sentence query, every sentence in the abstracts and combined text to the sentence query are compared in embedding space. To make this comparison, a cosine similarity measure is used, and after comparing, for every sentence in the abstracts and combined text, a score is generated. A high score (close to 1) means that the compared sentences are very similar, a low score (close to 0) means that they are very dissimilar, so in this case, a high score means that the compared sentence in the paper's abstract or body text is very comparable to the sentence query.

Subsequently, the sentences are ranked based on their score, and the top 500 sentences for both the abstracts and combined text are chosen. Associated paper id and text of the paper for every sentence in the top 50 are identified and stored the results for human evaluation. Sentence-transformers, itertools and sklearn packages are used in this step for performing analysis.

h) **Sentence Clustering and knowledge**

To better understand the results from the final sentence embedding step and summarize the information obtained, the sentences are first clustered. Sklearn library is used in sentence clustering.

i) **Knowledge Graph-based visualization**

The knowledge graph is drawn to visualize the key logic behind the high score sentences selected by the framework. It helps understand the framework's behavior and helps to dig some hidden knowledge out of the results. Spacy and networkx libraries are used in this preparation of the Knowledge Graph.

6.4 Result

In this section, the ability and utility of the Framework in retrieving knowledge (or providing an answer) in the form of relevant paper id, sentences from the paper and knowledge graph is discussed. The top 3 answers to each of the queries produced by the framework using BERT-large and RoBERTa-large are presented and discussed below.

Question 1: What is known about the origin of virus and management measures at the human-animal interface?	
Query: Evidence of SARS-CoV-2 infection in animals and its transmission or spill-over to other hosts, including humans	
The top 3 answers from the combined abstract and body corpus are as follows:	
BERT	RoBERTa
Paper ID: 14eaf287845975191ed38c75bc22d181806ef167 Important sentence: Thus, SARS-CoV-2 disease can be considered a zoonotic disease (like SARS) that has initially spread from animals to humans. Cosine Similarity metric: 0.781	Paper ID: 32b941d4406406a70dbced2b906b6795b775f5d4 Important sentence: In the last 15 years, two outbreaks of previously unknown highly pathogenic coronaviruses, SARS-CoV and MERS-CoV, have demonstrated that CoVs will continue to spill over into human populations, likely facilitated by interaction between infected animals and humans. Cosine Similarity metric: 0.754
Paper ID: 32b941d4406406a70dbced2b906b6795b775f5d4 Important sentence: In the last 15 years, two outbreaks of previously unknown highly pathogenic coronaviruses, SARS-CoV and MERS-CoV, have demonstrated that CoVs will continue to spill over into human populations, likely facilitated by interaction between infected animals and humans. Cosine Similarity metric: 0.752	Paper ID: bd1973693386dff5704ca4460874257cb1f037dd Important sentence: Finally, since many mammals including domestic animals might be susceptible to SARS-CoV-2, both surveillance and experimental infection should be conducted. Cosine Similarity metric: 0.740
Paper ID: bd1973693386dff5704ca4460874257cb1f037dd Important sentence: Finally, since many mammals, including domestic animals might be susceptible to SARS-CoV-2, both surveillance and experimental infection should be conducted. Cosine Similarity metric: 0.747	Paper ID: 4c222bd3e1c78c48931f4e69164fc6ee7bffb5ff Important sentence: Currently, it is thought that SARS-CoV-2 has been introduced to human by an unidentified intermediary animal and then it has spread from human-to-human. Cosine Similarity metric: 0.736

Table 6-2 Answer to Question, what is known about the origin of virus and management measures at the human-animal interface?

The Framework is able to find out paper ids that are relevant to the question. Resultant paper ids and sentences from both BERT and RoBERTa in Table 6-2 closely match with each other. After studying the top 3 answers provided above for each of the models, it can be concluded that:

- SARS-Cov-2 is a zoonotic disease that spreads from animals to human
- There have two instances in the past wherein coronaviruses, SARS-CoV and MERS-CoV, has spilled over to human
- Domestic animals are also susceptible to SARS-Cov-2
- SARS-Cov-2 spreads from human-to-human

The framework has been quite succesful in finding relevant answers (retrieving relevant paper ids and content) to the question from the corpus that comprises so many publications.

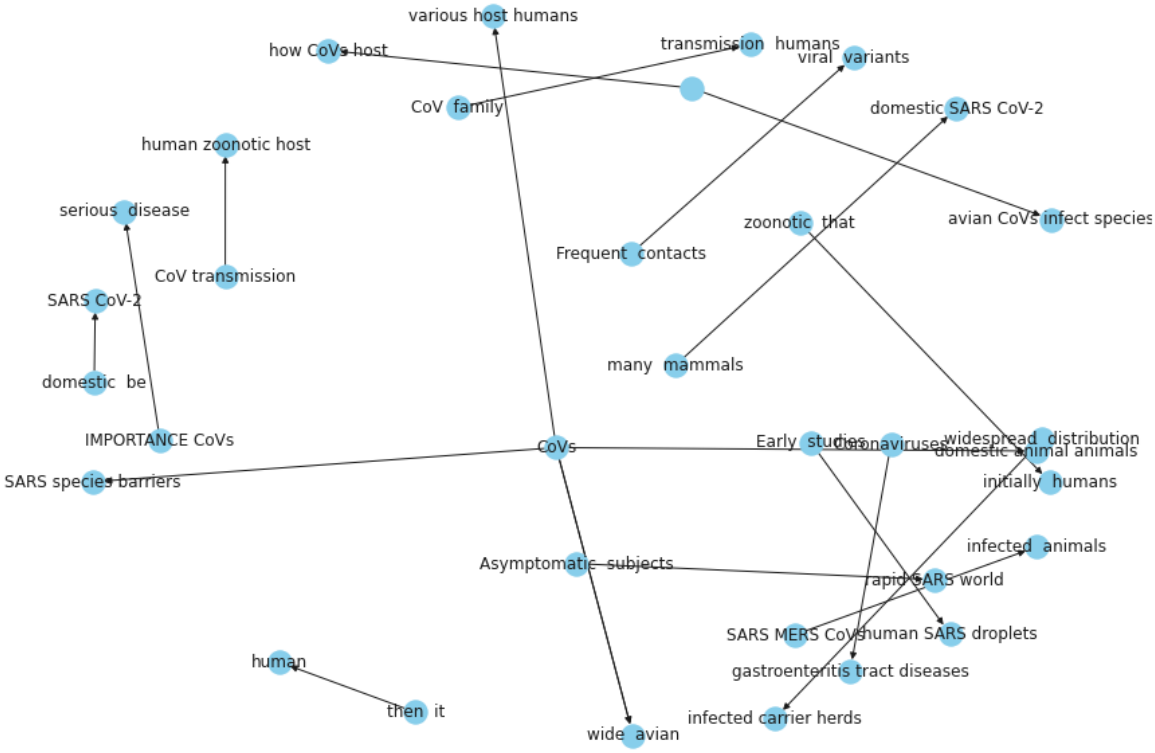


Figure 6-8 Knowledge Graph (Answers to Question 1 using BERT)

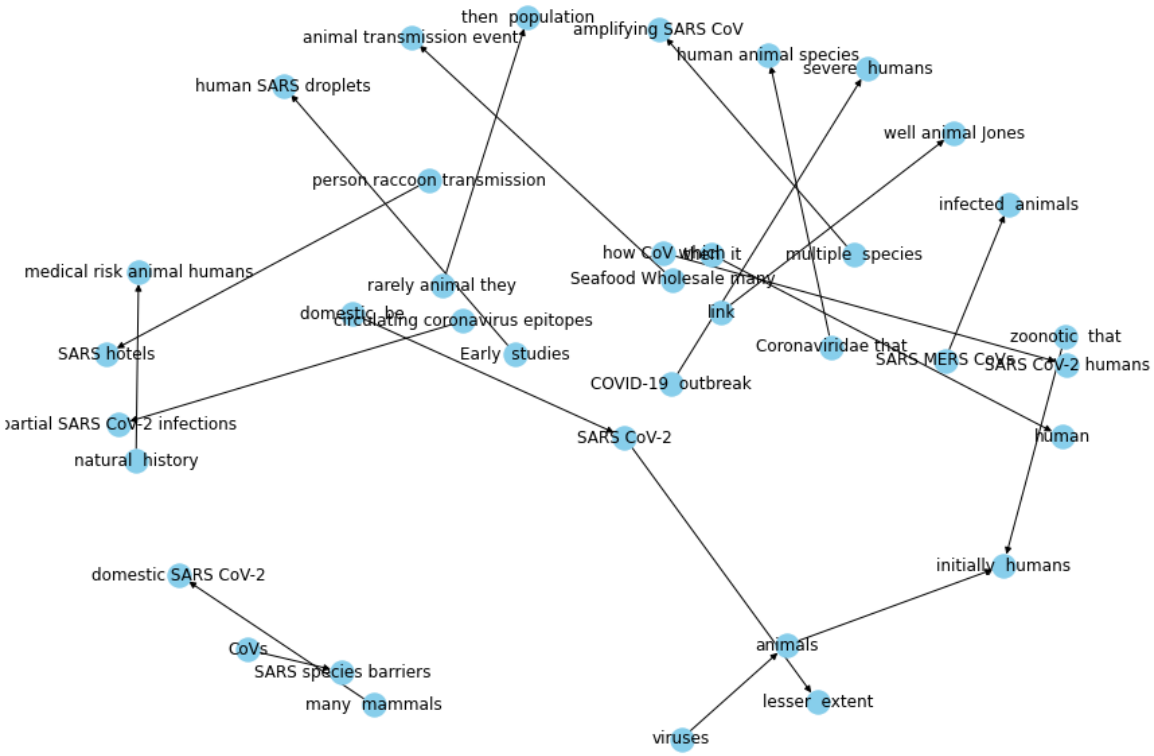


Figure 6-9 Knowledge Graph (Answers to Question 1 using RoBERTa)

Figure 6-8 and Figure 6-9 show the Knowledge graph prepared by using the top 20 answers for Question 1. A clear relationship between humans, animals and COVID-19 can be seen in the graphs. The knowledge graphs are insightful in understanding the relationships between various entities in the top search results. The queries can be further refined by studying and incorporating insights from these graphs.

Question 2: What has been published about medical care?	
Query: Knowledge of the frequency, manifestations, and course of extrapulmonary manifestations of SARS-CoV-2, including, but not limited to, possible cardiomyopathy and cardiac arrest	
The top 3 answers from the combined abstract and body corpus are as follows:	
BERT	RoBERTa
Paper ID: 867ebb2be2d466bcb35e5a7319708eaff39c4799 Important sentence: The complications of sedative procedures, of deep sedation and of endotracheal intubation is	Paper ID: 53595a15c4befbdcef60543bccbc76ad9567b640 Important sentence: In addition, peripheral pulmonary opacities were noted which were confirmed

bradycardia and cardiac arrest, 2 case, because of hypoxia and vagal stimulation. Cosine Similarity metric: 0.715	to be secondary to SARS-2-CoV pneumonia. Cosine Similarity metric: 0.751
Paper ID: 85a2db1bb25d3c369b7422b2b5f5a007175bec15 Important sentence: Cardiac complications such as electrocardiography abnormities, diastolic dysfunction, and acute myocardial injury were reported in patients with COVID-19 [124] [125] [126] [127]. Cosine Similarity metric: 0.682	Paper ID: 85a2db1bb25d3c369b7422b2b5f5a007175bec15 Important sentence: Current case reports show that SARS-CoV-2 infection may have cardiovascular symptoms in addition to the typical respiratory symptoms. Cosine Similarity metric: 0.751
Paper ID: 85a2db1bb25d3c369b7422b2b5f5a007175bec15 Important sentence: The activated SNS alters cardiac wall contractility and increases apoptotic pathways in cardiomyocytes, contributing to CVD development. Cosine Similarity metric: 0.659	Paper ID: 734779fad93249a2f6fede6afd10eeff7b37919b Important sentence: The transmission dynamics of the SARS-CoV-2 epidemic will depend on factors including the degree of seasonal variation in transmission strength, the duration of immunity, and the degree of cross-immunity between SARS-CoV-2 and other coronaviruses. Cosine Similarity metric: 0.727

Table 6-3 Answer to Question, what has been published about medical care?

Based on the answers generated in Figure 6-3, it can be summarized that:

- COVID infections may have cardiovascular symptoms (electrocardiography abnormities, diastolic dysfunction, and acute myocardial injury) and respiratory symptoms (pneumonia, peripheral pulmonary opacities)
- Transmission dynamics of COVID-19 is governed by duration of immunity, transmission strength and cross-immunity between SARS-CoV-2 and other coronaviruses

Question 3: What is known about the transmission, incubation, and environmental stability of the coronavirus?
Query: Incubation periods for the corona disease in humans, how this varies across age and health status, how long individuals are contagious, even after recovery
The top 3 answers from the combined abstract and body corpus are as follows:

BERT	RoBERTa
Paper ID: 29d15895b6f3c6472f3bab820d6a99fe efae60df Important sentence: Based on the current data, we do not know whether these patients are only asymptomatic initially after contracting the disease or if they are asymptomatic throughout the course of the disease. Cosine Similarity metric: 0.735	Paper ID: 86314a273c2fbd3015e2646773a98d8e b789a171 Important sentence: Deaths, sick persons who recover, and asymptomatic carriers continue to be found. Cosine Similarity metric: 0.734
Paper ID: 4bc41aaa2b754b3ec526eb40c384a18a b38b8d51 Important sentence: A long incubation period may lead to a high rate of asymptomatic and subclinical infection. Cosine Similarity metric: 0.721	Paper ID: b94ed97de06abc45de3ecc4e688c9ed3 57920162 Important sentence: The source and persistence of the infection in humans remains unknown. Cosine Similarity metric: 0.690
Paper ID: df97f804b68dcf16ffcc3c2c0894528d 6d0c7ff2 Important sentence: Susceptible individuals might acquire the infection at a given rate when they come in contact with an infectious person and enter the exposed disease state before they become infectious and later either recover or die. Cosine Similarity metric: 0.697	Paper ID: b94ed97de06abc45de3ecc4e688c9ed3 57920162 Important sentence: , showing the number of total cases and a timeline of persistence in human populations since onset of the first case. Cosine Similarity metric: 0.675

Table 6-4 Answer to Question, what is known about transmission, incubation, and environmental stability?

From the answers in Table 6-4, it can be inferred that:

- The incubation period is not known based upon the current data as patients can be asymptotic initially after contracting the disease, or they can be asymptotic throughout the disease as well

A variety of insights can also be made from the knowledge graph in Figure 6-10 and 6-11:

- Long incubation period related to asymptomatic infection
- Pulmonary fibrosis related to infection
- The first human case was related to an asymptomatic carrier
- Severe delay in prognosis relates to prolonged rehabilitation

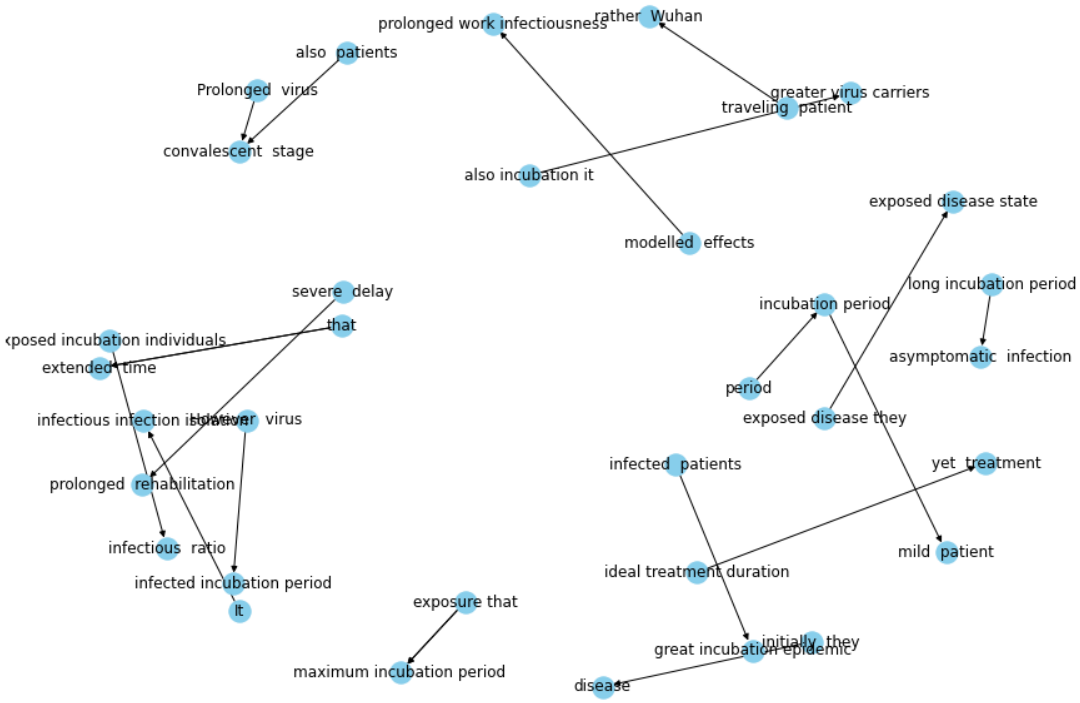


Figure 6-10 Knowledge Graph (Answers to Question 3 using BERT)

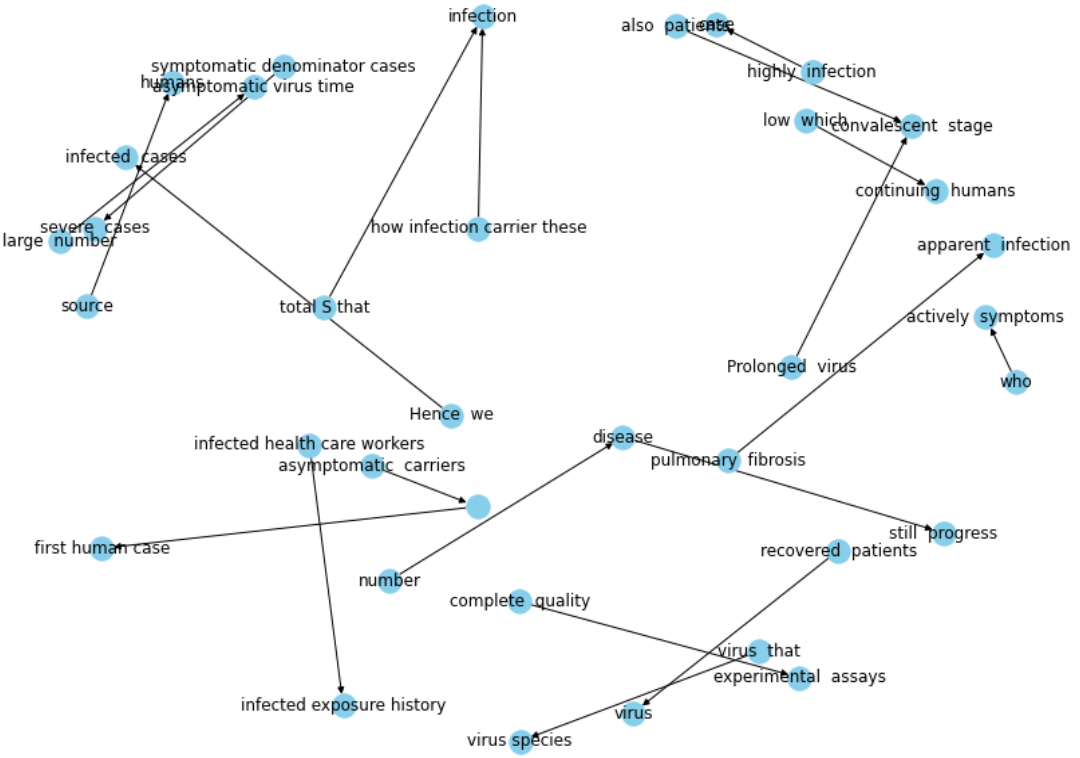


Figure 6-11 Knowledge Graph (Answers to Question 3 using RoBERTa)

Question 4: What is known about COVID-19 risk factors?	
Query: Data on potential risk factors: Transmission dynamics of the virus, including the basic reproductive number, incubation period, serial interval, modes of transmission and environmental factors	
The top 3 answers from the combined abstract and body corpus are as follows:	
BERT	RoBERTa
Paper ID: 7c0e41727a796451e68029e772c970246b71824c Important sentence: Therefore, the particular location of viral shedding events and relevant environmental condition should be taken into account when predicting the potential of interspecies transmission. Cosine Similarity metric: 0.729	Paper ID: 39f33cf580fe6ce842e2c9005570d08115c3ed4b Important sentence: Identification of resistance mutations to antiviral compound candidates in vitro provides an opportunity to assess the concern that resistance may promote viral fitness, leading to enhanced transmission or greater disease severity. Cosine Similarity metric: 0.721
Paper ID: bd1973693386dff5704ca4460874257cb1f037dd Important sentence: In addition, the requirements for successful host switches and the implications of virus evolution on disease severity are also highlighted. Cosine Similarity metric: 0.722	Paper ID: 356381d431c775e013710e6bff7ab89a507eb013 Important sentence: The remarkable frequency of virus infections of multiple species and strains lends itself to an ecological analysis of interactions that may be influential in virus ecology. Cosine Similarity metric: 0.715
Paper ID: 0ddffb5065201f469bdbc98060174a1b16c3f4b5 Important sentence: Reliable data is needed to understand how these infants are infected, and whether transmission of the virus to the fetus occurs pregnancy or sometime during or after delivery. Cosine Similarity metric: 0.714	Paper ID: f99195e087ef19cb9ad37eb6d3c6fc2ca6422323 Important sentence: • Monitor for viral genome mutations that might affect the performance of medical countermeasures, including diagnostic tests. Cosine Similarity metric: 0.709

Table 6-5 Answer to Question, what is known about COVID-19 risk factors?

Based on the answers that were retrieved by the framework, it can be concluded that:

- Transmission of the virus to the fetus during pregnancy cannot be ascertained
- Viral genome mutation could affect the performance of diagnostics test

- Resistance mutation in virus against viral compounds can lead to greater disease severity

Question 5: What has been published regarding research and development and evaluation efforts for developing vaccines and therapeutics?	
Query: Capabilities to discover a therapeutic for the disease, and clinical effectiveness studies to discover therapeutics, including antiviral agents	
Top 3 answers from the combined abstract and body corpus are as follows:	
BERT	RoBERTa
Paper ID: f12a115ffa060877bd7a292df682bac1b99a8193 Important sentence: Various therapies have emerged to treat the various aspects of viral pathogenesis and subsequent immune response, and an understanding of which aspect of disease pathogenesis they target may aide the clinician in knowing when to use them. Cosine Similarity metric: 0.784	Paper ID: 44fd42f34c00ff7d6126a58e38b7129c6e4f940d Important sentence: Therefore, finding new substances with antiviral properties is a required for medical systems. Cosine Similarity metric: 0.833
Paper ID: 6d36596c43f5f3ac4981f36a75765d62e68538f4 Important sentence: Based on the anti-inflammatory and antiviral properties, clinical trials (NCT04323514, NCT04264533, NCT03680274, and NCT04326725) are ongoing for its therapeutic usage. Cosine Similarity metric: 0.782	Paper ID: 050c5232dd292ccc50fe7294d91fb18b72ee6cc2 Important sentence: Recently, researchers have investigated the antiviral potential of various approved drugs as first-line of defense against the newly emerging infectious agents. Cosine Similarity metric: 0.813
Paper ID: cf5622021c8dc0db6cf93aebb6e9240fd9ba65b1 Important sentence: Multiple antiviral regimens are being tried to help patients with severe symptoms of the virus. Cosine Similarity metric: 0.767	Paper ID: 2ad4a32315e1ce061a37ba49106dc886512b5883 Important sentence: This implicates early initiation of antiviral therapies for better outcomes and use of immunosuppressive therapies in the adaptive immune stage. Cosine Similarity metric: 0.811

Table 6-6 Answer to Question, what has been published regarding research and development and evaluation efforts for developing vaccines and therapeutics?

After studying the top 3 answers provided above in Figure 6-6 for each of the models, it can be concluded that:

- Clinical trials are being conducted for therapeutical usage. These clinical trials are, namely, NCT04323514, NCT04264533, NCT03680274, and NCT04326725. The list of clinical trials is just indicative and as retrieved from the dataset
- Various therapies have been identified to treat various aspects of viral pathogenesis.
- Various drugs have been identified as the first line of defense against COVID-19

It can be noted in Figure 6-12 and 6-13 that none of the potential chemical compositions have been captured in the graph. These potential chemical compositions are also not captured in Table 6-6. The reason could be the limited capacity of BERT in understanding domain-specific terminologies.

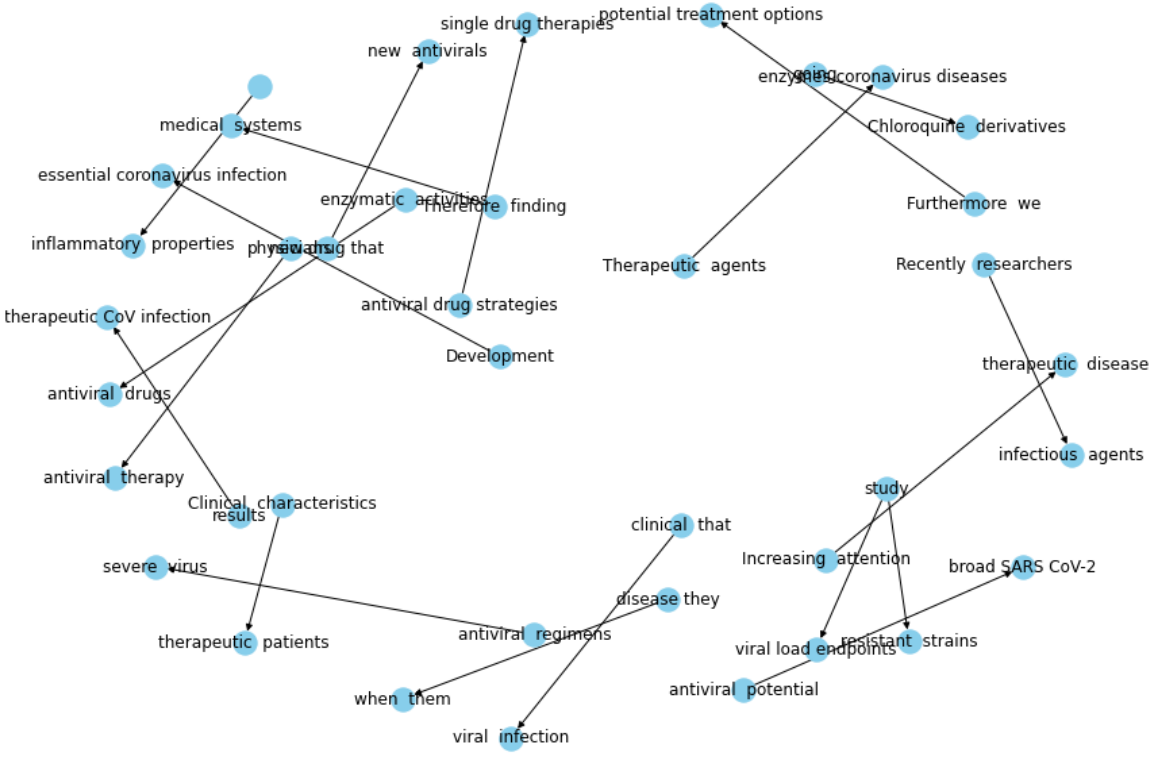


Figure 6-12 Knowledge Graph (Answers to Question 5 using BERT)

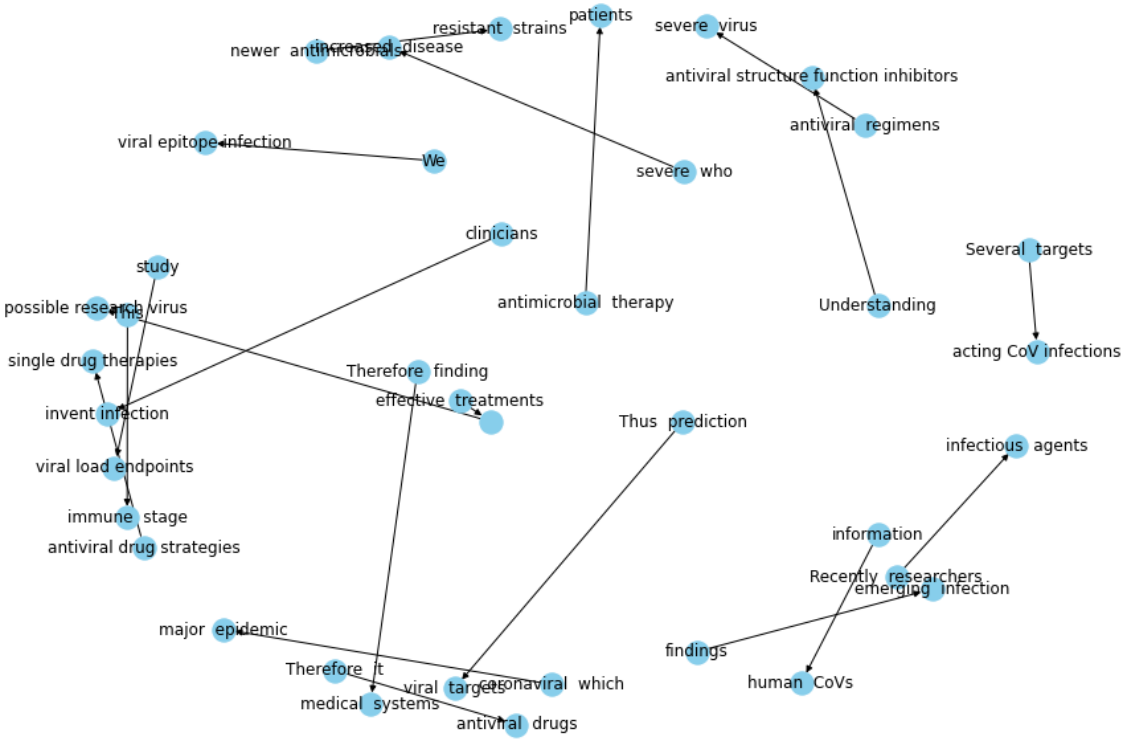


Figure 6-13 Knowledge Graph (Answers to Question 5 using RoBERTa)

6.5 Discussion

Overall, the Framework (answers the second research question in section 1.4) can successfully retrieve knowledge, i.e., answers for a Query from the CORD-19. However, this semantic way of getting the answers is limited to finding the relevant paper id, the relevant sentences and preparing knowledge graphs. As the answers identified are based upon the corpus, any change in the corpus would lead to different answers. Both BERT based and RoBERTa based Framework produced results with varying degree of overlap as expected. RoBERTa is pre-trained on a much larger dataset (10 times) as compared to BERT. Ideally, RoBERTa should produce more semantically similar answers as compared to BERT. However, this does not seem to be the case here as the relevance of results produced by both BERT and RoBERTa based Frameworks seems similar (visual interpretation of results). This could be attributed to the already well-established vocabulary of BERT compared to the vocabulary of RoBERTa for a QA task in the context of CORD-19. The knowledge graph presents various keywords related to the query and relationships among the keywords. The information from the Knowledge Graph can be further used to enhance the quality of the query and finetune the answer using the Framework.

7. Conclusion and Future Work

7.1 Conclusion

With the help of extensive literature review and surveys, it is established that pre-trained language models have been able to capture semantic and syntactic meanings of words: local context. Introduction of the Transformer architecture (Vaswani et al., 2017 [36]) resulted in a series of powerful pre-trained language representation models such as BERT (Devlin et al.,2018 [7]), GPT (Radford et al.,2018 [6]), Transformer-XL (Dai et al.,2019 [59]), XLM (Lample and Conneau, 2019 [60]), GPT-2 (Radford et al.,2019 [39]), ERNIE (Zhang et al.,2019 [50]), XLNet (Yang et al.,2019 [49]), RoBERTa (Liu et al.,2019 [61]), ERNIE 2.0 (Sun et al.,2019 [62]) and CTRL (Keskar et al.,2019 [63]) that proved their effectiveness in a wide variety of language tasks including QA. BERT based models were chosen for developing the QA framework because of three main reasons (Section 2.8 and 3.1). BERT is the backbone of the most dominant Google Search engine, BERT based models have performed very well in supervised QA tasks using datasets like SQuAD, and BERT model size is most efficient as compared to other competing models. This gives a clear answer to the first research question – "Which DL NLP architecture should be selected for KD task from CORD-19?".

As part of this thesis, all phases in the data science/analytics life cycle have been covered. State of the art advanced NLP and machine learning techniques were deployed to develop a semantically intelligent QA framework to study queries related to COVID-19. Requisite queries were formed based on the research questions identified in Section 1.4. The answers to a query comprise relevant paper id, relevant text from the paper and cosine similarity value. The results of the queries were manually studied to establish the performance of the framework. The proposed framework will help the research community retrieve key (top-level) publications and specify the most frequently used COVID-19 scientific keywords and perform semi-automatic analysis at the sentence level to dig out hidden knowledge and novel information. Framework's utilization and effectiveness have been demonstrated by way of utilizing high-level keywords and queries. In the end, a 360^{o} view of the acquired publications is provided through the Knowledge Graph component for ease of use for the research community. It can be concluded that the "Semantic QA Framework for Knowledge Discovery" is not a perfect system for QA on CORD-19. However, it has been quite effective in providing the answers (in the form of the most relevant paper, content and knowledge graph) to the research questions identified in Section 1.4. With various permutation and combination

of keywords, queries and pre-trained BERT based model, requisite knowledge can be mined from the dataset.

7.2 Future Scope

Three areas for further contribution to this thesis has been identified. These areas are:

a) **Extraction of more complex relationships for preparing the Knowledge Graph**

   Currently, noun and verb combinations of the first 20 results are used to prepare the Knowledge Graph. Complex knowledge graph algorithms can be used to extract complicated relationships among words in the entire result. With the help of insights from more advanced and complex knowledge graphs, the queries can be further refined. This will lead to better results while querying.

b) **Experimentation with other pre-trained language models esp. recently developed BioBERT**

   In 2020 Lee et al. [53] introduced BioBERT, a BERT model that is pre-trained on a huge biomedical dataset. The authors fine-tuned BioBERT on three NLP tasks (named entity recognition, relation extraction and question answering) using biomedical corpus. BioBERT outperforms BERT on some language tasks as well. The framework shall remain the same. Only while generating the word embeddings, BioBERT can be used instead of BERT.

c) **Evaluation methodology for QA tasks where no training data is available**

   It is difficult to evaluate the framework's performance for the QA task on the dataset when no training data is available. This situation can be compared to evaluating the performance of various search engines like Google, Bing, Yandex on some defined metrics or parameters. Had there been a standard formula, Google would not have been so dominant in the market with 92% market share. The search engine's performance can only be measured by the perceived relevance of search results for the users. Similarly, the framework used in the thesis can only be evaluated by actual users of the CORD-19, i.e., researchers and scientists. The finalization of the framework would be an iterative process with constant feedback from the end-users and then making requisite modification in the framework. These changes can include trying various pre-trained language models, vectorization methods, iterative refining of queries.

Bibliography

[1] L. Wang, K. Lo, Y. Chandrasekhar and R. Reas, "CORD-19: The COVID-19 Open Research Dataset," *arXiv*, vol. 2004.10706v4, 2020.

[2] A. Turing, "Computing machinery and intelligence," *Mind*, vol. 59, no. 236, pp. 433-460, 1950.

[3] A. Kumar, O. Irsoy, P. Ondruska, M. Iyyer, J. Bradburry and I. Gulrajani, "Ask me anything: Dynamic memory networks for natural language processing," *JMLR Workshop and Conference Proceedings*, vol. 48, pp. 1378-1387, 2016.

[4] D. A. Ferrucci, E. W. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. Kalyanpur, A. Lally, J. W. Murdock, E. Nyberg, J. M. Prager, N. Schlafer and C. A. Welty, "Building watson: An overview of the deepqa project," *AI Magazine*, vol. 31, no. 3, pp. 59-79, 2010.

[5] M. E. Petere, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee and L. Zettlemoyer, "Deep contextualized word representations," in *16th Annual Conference of the North American Chapter of the Association for Computational Linguistics*, New Orleans, 2018.

[6] A. Radford, K. Narasimhan, T. Salimans and I. Sutskever, "Improving language understanding by generative pre-training," 11 06 2018. [Online]. Available: https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/language-unsupervised/language_ understanding_paper. pdf. [Accessed 15 10 2020].

[7] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv*, vol. abs/1810.04805, 2018.

[8] G. Piatetsky-Shapiro, U. Fayyad and P. Smyth, "From Data Mining to Knowledge Discovery: An Overview," *Advances in Knowledge Discovery and Data Mining, MIT Press*, 1996.

[9] R. Wirth and J. Hipp, "Crisp-dm: towards a standard process modell for data mining," in *Wirth2000CrispdmTA*, 2000.

[10] R. Talib, S. Ayesha and M. Kashif Hanif, "Text Mining: Techniques, Applications and Issues," *(IJACSA) International Journal of Advanced Computer Science and Applications,* vol. 7, no. 11, pp. 414-418, 2016.

[11] A. Louis, "NetBERT: A Pre-trained Language Representation Model for Computer Networking," 25 06 2020. [Online]. Available: https://matheo.uliege.be/handle/2268.2/9060. [Accessed 01 10 2020].

[12] W. Weaver and S. Claude, "Theory of information," in *The Mathematical Theory of Communication*, Urbana, University of Illinois Press, 1949, pp. 36-41.

[13] L. E. Dostert, "The georgetown-ibm experiment," in *Machine translation of languages,* New York, John Wiley& Sons, 1955, pp. 124-135.

[14] C. Fillmore, "The case for case," *Bach and Harms (Ed.): Universals in Linguistic Theory,* pp. 1-25, 1968.

[15] A. M. Collins and R. M. Quillian, "Retrieval time from semantic memory," *Journal of Verbal Learning and Verbal Behavior,* vol. 8, no. 2, p. 240 – 247, 1969.

[16] W. A. Woods, "Transition network grammars for natural language analysis," *Communications of the ACM,* vol. 13, no. 10, p. 591–606, 1970.

[17] R. C. Schank, "Conceptual dependency: A theory of natural language understanding," *Cognitive psychology,* vol. 3, no. 4, pp. 552-631, 1972.

[18] E. Charniak, "Passing markers: A theory of contextual influence in language comprehension," *Cognitive Science,* vol. 7, no. 3, p. 171–190, 1983.

[19] M. G. Dyer, "The role of affect in narratives," *Cognitive Science,* vol. 7, no. 3, p. 211–242, 1983.

[20] C. K. Riesbeck and M. C, "Direct memory access parsing," in *Experience, memory and Reasoning,* 1986, p. 209–226.

[21] B. J. Grosz, D. E. Appelt, P. A. Martin and F. C. Pereira, "Team: an experiment in the design of transportable natural-language interfaces," *Artificial Intelligence,* vol. 3, no. 2, p. 173–243, 1987.

[22] H. Graeme , "Semantic interpretation and ambiguity," *Artificial Intelligence,* vol. 34, no. 2, p. 131 – 177, 1987.

[23] L. R. Bahl, P. F. Brown, P. V. de Souza and R. L. Mercer, "A tree based statistical language model for natural language speech recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* vol. 37, no. 7, p. 1001–1008, 1989.

[24] E. Brill, D. Magerman, M. Marcus and B. Santorini, "Deducing linguistic structure from the statistics of large corpora," in *Proceedings of the 5th Jerusalem Conference on nformation Technology,* Jerusalem, 1990.

[25] M. V. Chitrao and R. Grishman, "Statistical parsing of messages," in *Speech and Natural Language: Proceedings of a Workshop,* Pennsylvania, June 24-27, 1990.

[26] P. Brown, J. Cocke, S. Della Pietra and V. Della Pietra, "A statistical approach to machine translation," *Computational linguistics,* vol. 16, no. 2, p. 79–85, 1991.

[27] Y. Bengio, R. Ducharme, V. Pascal and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research,* vol. 3, no. 2, pp. 1137-1155, 2003.

[28] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep," *Proceedings of the 25th international conference on Machine Learning,* pp. 160-167, 2008.

[29] S. Ruder, "An overview of multi-task learning in deep neural networks," *arXiv,* vol. abs/1706.05098, 2017.

[30] T. Mikolov, K. Chen, G. Corrado and J. Dean, "Efficient estimation of word representations in vector space," *arXiv,* vol. 1301.3781, 2013.

[31] J. L. Elman, "Finding structure in time," *Cognitive Science,* vol. 14, no. 2, pp. 179-211, 1990.

[32] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631-1642, 2013.

[33] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735-1780, 1997.

[34] I. Sutskever, O. Vinyals and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104-3112.

[35] Y. Wu, M. Schuster, Z. Chen, M. Norouzi, W. Macherey, Q. V. Le, M. Krikun, Y. Cao, Q. Gao and K. Macherey, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv*, vol. 1609.08144, 2016.

[36] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, pp. 5998-6008, 2017.

[37] M. X. Chen, O. Firat, A. Bapna, M. Johnson, W. Macherey, G. Foster, L. Jones, N. Parmer, M. Schuster and Z. Chen, "The best of both worlds: Combining recent advances in neural machine translation," *arXiv*, vol. 1804.09849, 2018.

[38] D. Bahdanau, K. Cho and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv*, vol. 1409.0473, 2014.

[39] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei and I. Sutskever, *Language models are unsupervised multitask learners*, OpenAI Blog, 2019.

[40] T. K. Landauer and S. T. Dumais, "A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge," *Psychological Review*, vol. 104, no. 2, p. 211, 1997.

[41] K. Lund and C. Burgess, "Producing high-dimensional semantic spaces from lexical co-occurence," *Behavior research methods, instruments, & computers*, vol. 28, no. 2, pp. 203-208, 1996.

[42] D. Rohde, L. Gonnerman and D. Plaut, "An improved method for deriving word meaning from lexical co-occurance," *Cognitive Psychology,* vol. 7, pp. 573-605, 2004.

[43] R. Lebert and R. Collobert, *Word emdeddings through hellinger pca,* arXiv preprint arXiv:1312.5542, 2013.

[44] T. Mikolov, G. Corrado, J. Dean and I. Sutskever, "Distributed representations of words and phrases and their compositionality," *Advances in neural information processing systems,* pp. 3111-3119, 2013.

[45] P. Bojanowski, E. Grave, A. Joulin and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics,* vol. 5, pp. 135-146, 2017.

[46] O. Melamud, J. Goldberger and I. Dagan, "contect2vec: Learning generic context embeding with bidirectional lstm," *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning,* pp. 51-61, 2016.

[47] B. McCann, J. Bradburry, C. Xiong and R. Socher, "Learned in translation: Contextualized word vectors," *Advances in Neural Information Processing Systems,* pp. 6294-6305, 2017.

[48] H. Jeremy and S. R. , "Universal language model fine-tuning for text classification," *arXiv,* vol. 1801.06146, 2018.

[49] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," *Advances in neural information processing systems,* pp. 5754-5764, 2019.

[50] Z. Zhengyan , X. Han, L. Zhiyuan , J. Xin, S. Maosong and L. Qun, "Enhanced language representation with informative entities," *arXiv,* vol. 1905.07129, 2019.

[51] K. Huang, J. Altosaar and R. Ranganath, "Modeling clinical notes and predicting hospital readmission," *arXiv,* vol. 1904.05342, 2019.

[52] I. Beltagy, A. Cohan and K. Lo, "Scibert: Pretrained contextualized embeddings for scientific text," *arXiv*, vol. 1903.10676, 2019.

[53] J. Lee, W. Yoon, K. Sungdong, K. Donhyeong and K. Sunkyu, "Biobert: a pre-trained biomedical language representation model for biomedical text minning," *Bioinformatics*, no. 436, pp. 1234-1240, 2020.

[54] S. Sugawara, Y. Kido, H. Yokono and A. Aizawa, "Evaluation metrics for machine reading comprehension: Prerequisite skills and readability," *Proceedings of ACL*, pp. 806-817, 2017.

[55] P. Rajpurkar, J. Zhang, K. Lopyrev and P. Liang, "Squad: 100,000+ questions for machine comprehension of text.," *arXiv*, vol. 1606.05250, 2016.

[56] P. Rajpurkar, R. Jia and P. Liang, "Know What You Don't Know: Unanswerable Questions for SQuAD," *arXiv*, vol. 1806.03822, 2018.

[57] M. Seo, A. Kembhavi, A. Farhadi and H. Hajishirzi, "Bidirectional Attention Flow for Machine Comprehension," *arXiv*, vol. 1611.01603, 2016.

[58] A. Kent, M. Berry, F. U. Jr. and J. Perry, "Machine literature searching VIII. Operational criteria for designing information retrieval systems," in *American Documentation. 6 (2): 93. doi:10.1002/asi.5090060209*, 1955.

[59] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. Lee and R. Salakhutdinov, "Transformer-xl: Attentive language models beyond a fixed-length context," *arXiv*, vol. 1901.02860, 2019.

[60] G. Lample and A. Conneau, "Cross-lingual language model pretraining," *arXiv*, vol. 1901.07291, 2019.

[61] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv*, vol. 1907.11692, 2019.

[62] Y. Sun, S. Wang, Y. Li, S. Feng, H. Tian, H. Wu and H. Wang, "Ernie 2.0: A continual pre-training framework for language understanding," *arXiv,* vol. 1907.12412, 2019.

[63] N. Kesker, B. McCann, L. Varshney, X. Caiming and R. Socher, "Ctrl: A conditional transformer language model for controllable generation," *arXiv,* vol. 1909.05858, 2019.

[64] A. Liu, Z. Huang, X. Wang and C. Yuan, "Bb-kbqa: Bert-based knowledge base question answering," *China National Conference on Chinese Computational Linguistics*, pp. 81-92, 2019.

[65] N. Poerner, U. Waltinger and H. Schutze, "Factual knowledge vs. name-based reasoning in unsupervised qa," *arXiv,* vol. 1911.03681, 2019.

[66] M. Peters, M. Neumann, R. Schwartz, V. Jochi, S. Singh, N. Smith, R. L and L. IV, "Knowledge enhanced contextual word representations," *arXiv,* vol. 1909.04164, 2019.

[67] huggingface, "Pretrained models," Huggingface, 2020. [Online]. Available: https://huggingface.co/transformers/pretrained_models.html. [Accessed 12 October 2020].

[68] J. Alammar, "The Illustrated Transformer," 27 June 2018. [Online]. Available: http://jalammar.github.io/illustrated-transformer/. [Accessed 10 October 2020].

[69] K. He, G. Gkioxari, P. Dollar and R. Girshick, *Deep residual learning for image recognition,* In CVPR, 2016.

[70] D. Hendrycks and K. Gimpel, "A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks," *arXiv,* vol. 1610.02136, 2016.

[71] K. Lo, L. L. Wang, M. Neumann, R. Kinny and D. S. Weld, *S2ORC: The Semantic Scholar Open Research Corpus,* Proceedings of ACL, 2020.

[72] Z. S. Harris, "Distributional structure," *Word,* vol. 10, no. 2, pp. 146-162, 1954.

[73] J. R. Firth, "A synopsis of linguistic theory," *Studies in linguistic analysis,* 1957.

[74] Y. Bengio, R. Ducharme, P. Vincent and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research,* vol. 3, no. 2, p. 1137–1155, 2003.

[75] T. Mikolov, M. Karafiát, J. Černocky and S. Khudanpur, "Recurrent neural network based language model.," *Eleventh annual conference of the international speech communication association,* 2010.

Appendix A – Colab Notebook

The Colab Notebook containing the implementation of the Framework in python language is provided here.

Semantic QA Framework for Knowledge Discovery from CORD-19

October 24, 2020

1 Semantica QA Framework for Knowledge Discovery from CORD-19

Installing and importing packages

[ ]:
```
# install packages
!pip install nltk
!pip install owlready2
!pip install pronto
!pip install ipynb-py-convert
!pip install langdetect
!pip install contractions
!pip install inflect
!pip install num2words
!pip install tables
!pip install h5py
!pip install sentence-transformers
!pip install pandas
!pip install tqdm
!pip install seaborn
!pip install numpy
!pip install scipy
!pip install matplotlib
!pip install numpy
!pip install bottleneck
!pip install pandarallel
!pip install wordcloud
!pip install spacy
!pip install https://github.com/explosion/spacy-models/releases/download/
↪en_core_web_sm-2.2.0/en_core_web_sm-2.2.0.tar.gz
```

[2]:
```
from collections import defaultdict
import glob
import itertools
import json
import pickle
import os
```

```
import re
import bs4
import contractions
import inflect
from langdetect import detect
import matplotlib.pyplot as plt
import networkx as nx
import nltk
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')
from nltk import tokenize
from nltk.corpus import wordnet as wn
from nltk.corpus import stopwords
from nltk.stem import LancasterStemmer
from nltk.stem import PorterStemmer
from nltk.stem import WordNetLemmatizer
import numpy as np
import pandas as pd
from pandarallel import pandarallel
from PIL import Image
import requests
import seaborn as sns
from sentence_transformers import SentenceTransformer
from sklearn.cluster import KMeans
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
import spacy
from spacy import displacy
nlp = spacy.load('en_core_web_sm')
from spacy.matcher import Matcher
from spacy.tokens import Span
from tqdm import tqdm
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator

# Initialize pandarallel
pandarallel.initialize(use_memory_fs=False,nb_workers=2)
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Unzipping corpora/wordnet.zip.
INFO: Pandarallel will run on 2 workers.
INFO: Pandarallel will use standard multiprocessing data transfer (pipe) to
transfer data between the main process and workers.
```

[3]:
```
# mounting google drive
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

[4]:
```
# pandas options
pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)
pd.set_option('display.expand_frame_repr', False)
pd.options.mode.chained_assignment = None

tqdm.pandas()
```

1.1 Data Loading

[5]:
```
# Help functions and class
# help function to generate file path
def filepath(*args):
    if len(args) < 1:
        return None
    elif len(args) == 1:
        return args[0]
    else:
        return f'{args[0]}/{filepath(*args[1:])}'

# Add time bar to loop
def addtimebar(L, threshold=1000):
    if len(L) > threshold:
        return tqdm(L)
    else:
        return L

# File Reader Class
class FileReader:
    def __init__(self, file_path):
        with open(file_path) as file:
            content = json.load(file)
            self.paper_id = content['paper_id']
            self.abstract = []
            self.body_text = []
            # Abstract
            try:
                for entry in content['abstract']:
                    self.abstract.append(entry['text'])
            except KeyError:
```

```
                pass

            # Body text
            try:
                for entry in content['body_text']:
                    self.body_text.append(entry['text'])
            except KeyError:
                pass
            self.abstract = '\n'.join(self.abstract)
            self.body_text = '\n'.join(self.body_text)
    def __repr__(self):
        return f'{self.paper_id}: {self.abstract[:200]}... {self.body_text[:
 ↪200]}...'

# Helper function adds break after every words when character length reach to␣
 ↪certain amount. This is for the interactive plot so that hover tool fits the␣
 ↪screen.
def get_breaks(content, length):
    data = ""
    words = content.split(' ')
    total_chars = 0

    # add break every length characters
    for i in range(len(words)):
        total_chars += len(words[i])
        if total_chars > length:
            data = data + "<br>" + words[i]
            total_chars = 0
        else:
            data = data + " " + words[i]
    return data

## composition function
## example: compose(f1,f2,f3)(x, y) = f3(f2(f1(x, y)))
def compose(*funcs):
    *funcs, penultimate, last = funcs
    if funcs:
        penultimate = compose(*funcs, penultimate)
    return lambda *args: penultimate(last(*args))
```

[6]:
```
# file path
path = "/content/drive/My Drive/Thesis"
meta = "metadata.csv"
# path for all json files
all_jsons = glob.glob(filepath(path, '**', '*.json'), recursive=True)
```

[7]:
```
# dataframe for meta data
meta_df = pd.read_csv(filepath(path, meta),
                      dtype={'pubmed_id': str,
                             'Microsoft Academic Paper ID': str,
                             'doi': str,
                             'journal':str
                             },
                      low_memory=False)
```

[8]:
```
# Load the text data into DataFrame
dict_ = {'paper_id': [], 'abstract': [], 'body_text': [], 'authors': [],␣
 ↪'title': [], 'publish_time':[], 'journal': [], 'abstract_summary': []}
for entry in addtimebar(all_jsons):
    content = FileReader(entry)

    # get metadata information
    meta_data = meta_df.loc[meta_df['sha'] == content.paper_id]
    # no metadata, skip this paper
    if len(meta_data) == 0:
        continue

    dict_['paper_id'].append(content.paper_id)
    dict_['abstract'].append(content.abstract)
    dict_['body_text'].append(content.body_text)

    # also create a column for the summary of abstract to be used in a plot
    if len(content.abstract) == 0:
        # no abstract provided
        dict_['abstract_summary'].append("Not provided.")
    elif len(content.abstract.split(' ')) > 100:
        # abstract provided is too long for plot, take first 300 words append␣
 ↪with ...
        info = content.abstract.split(' ')[:100]
        summary = get_breaks(' '.join(info), 40)
        dict_['abstract_summary'].append(summary + "...")
    else:
        # abstract is short enough
        summary = get_breaks(content.abstract, 40)
        dict_['abstract_summary'].append(summary)

    # get metadata information
    meta_data = meta_df.loc[meta_df['sha'] == content.paper_id]

    try:
        # if more than one author
        authors = meta_data['authors'].values[0].split(';')
        if len(authors) > 2:
```

```
            # more than 2 authors, may be problem when plotting, so take first␣
 ↪2 append with ...
            dict_['authors'].append(". ".join(authors[:2]) + "...")
        else:
            # authors will fit in plot
            dict_['authors'].append(". ".join(authors))
    except Exception as e:
        # if only one author - or Null valie
        dict_['authors'].append(meta_data['authors'].values[0])

    # add the title information, add breaks when needed
    try:
        title = get_breaks(meta_data['title'].values[0], 40)
        dict_['title'].append(title)
    # if title was not provided
    except Exception as e:
        dict_['title'].append(meta_data['title'].values[0])

    # add publish time
    try:
        publish_time = get_breaks(meta_data['publish_time'].values[0], 40)
        dict_['publish_time'].append(publish_time)
    # if publish time was not provided
    except Exception as e:
        dict_['publish_time'].append(meta_data['publish_time'].values[0])

    # add the journal information
    dict_['journal'].append(meta_data['journal'].values[0])

df_covid = pd.DataFrame(dict_, columns=['paper_id', 'abstract', 'body_text',␣
 ↪'authors', 'title',  'journal', 'publish_time', 'abstract_summary'])
df_covid.head()
```

[9]:
```
# save data
df_covid.to_pickle('/content/drive/My Drive/Thesis/df_all.pkl')
# # load saved data
# with open('/content/drive/My Drive/Thesis/df_all.pkl', 'rb') as fp:
#     df_covid = pickle.load(fp)
```

[10]:
```
print(len(df_covid)) # number of lines in df_covid
# look at the top 3 rows in the dataframe
df_covid.head(3)
```

```
58396
```

[10]:

	paper_id	abstract	body_text	authors	title

```
journal publish_time                                    abstract_summary
0  2ad337f148249bbc7fa34527302927081cd36d31  To aid the ongoing battle against
hospital-acq...  The emergence of the severe acute respiratory ...        Damji,
S..  Barlow, G. D...   An audit of the use of isolation facilities i...
Journal of Hospital Infection   2005-07-31   To aid the ongoing battle
against<br>hospital...
1  9e9aaaacb26e95ceec9d7465ce65972e1fd84a7a  The solar radiation can heat the
building oute...  The environmental and public health issues are...
Mu, Di.  Gao, Naiping...   CFD investigation on the effects of wind and<...
Building and Environment   2018-06-30   The solar radiation can heat the
building out...
2  55d762875ac9c5b43832336f3a88d0e70ff8ab9c  Background: Our recent meta-
analysis indicated...  In controlled trials, vitamin C has improved e...
Hemilä, Harri.  Chalker, Elizabeth   Vitamin C may reduce the duration of
mechanic...               J Intensive Care   2020-02-07   Background: Our recent
meta-analysis<br>indic...
```

1.2 Data Processing

1.2.1 Select English articles

[11]:
```
# function to check if text of certain column in dataframe is written in␣
 ↪certain language
def is_lang(row, item, lang, dropNA=True):
    if (row[item] != None and row[item] != '' and row[item] != 'None' and␣
 ↪isinstance(row[item], str)):
        try:
            return detect(row[item]) == lang
        except Exception as e:
            #print("Non-readable entity will be droped from data.frame")
            return False
    else:
        return not dropNA

# select article written in certain language
def select_article_lang_multi(df, basedon='abstract', lang='en'):
    return df[df.parallel_apply(lambda text: is_lang(text, basedon, lang),␣
 ↪axis=1)]

df_covid_eng = select_article_lang_multi(df_covid)
print('Number of English Articles: {}/{}'.format(len(df_covid_eng),␣
 ↪len(df_covid)))
df_covid_eng.head(n=2)
```

```
Number of English Articles: 40134/58396
```

[11]:
```
                                    paper_id
abstract                                                body_text
authors                                                    title
journal publish_time                                         abstract_summary
0  2ad337f148249bbc7fa34527302927081cd36d31  To aid the ongoing battle against
hospital-acq...  The emergence of the severe acute respiratory ...  Damji, S..
Barlow, G. D...   An audit of the use of isolation facilities i...  Journal of
Hospital Infection   2005-07-31   To aid the ongoing battle
against<br>hospital...
1  9e9aaaacb26e95ceec9d7465ce65972e1fd84a7a  The solar radiation can heat the
building oute...  The environmental and public health issues are...      Mu, Di.
Gao, Naiping...   CFD investigation on the effects of wind and<...
Building and Environment   2018-06-30   The solar radiation can heat the
building out...
```

[12]:
```
# save intermidiate data
df_covid_eng.to_pickle('../data/df_all_eng.pkl')
# load saved data
# with open('/content/drive/My Drive/Thesis/df_all_eng.pkl', 'rb') as fp:
#    df_covid_eng = pickle.load(fp)
```

[13]:
```
print(len(df_covid_eng)) # number of lines in df_covid_eng
# look at the top 3 rows in the dataframe
df_covid_eng.head(3)
```

```
40134
```

[13]:
```
                                    paper_id
abstract                                                body_text
authors                                                    title
journal publish_time                                         abstract_summary
0  2ad337f148249bbc7fa34527302927081cd36d31  To aid the ongoing battle against
hospital-acq...  The emergence of the severe acute respiratory ...        Damji,
S..  Barlow, G. D...   An audit of the use of isolation facilities i...
Journal of Hospital Infection   2005-07-31   To aid the ongoing battle
against<br>hospital...
1  9e9aaaacb26e95ceec9d7465ce65972e1fd84a7a  The solar radiation can heat the
building oute...  The environmental and public health issues are...
Mu, Di.  Gao, Naiping...   CFD investigation on the effects of wind and<...
Building and Environment   2018-06-30   The solar radiation can heat the
building out...
2  55d762875ac9c5b43832336f3a88d0e70ff8ab9c  Background: Our recent meta-
analysis indicated...  In controlled trials, vitamin C has improved e...
Hemilä, Harri.  Chalker, Elizabeth   Vitamin C may reduce the duration of
mechanic...               J Intensive Care   2020-02-07   Background: Our recent
meta-analysis<br>indic...
```

1.2.2 Text cleanning and tokenization

[14]:
```
# Pre-processing functions
## text level processors
def replace_brackets_with_whitespace(text):
    text = text.replace('(', '')
    text = text.replace(')', '')
    text = text.replace('[', '')
    text = text.replace(']', '')
    return text

def replace_contractions(text):
    return contractions.fix(text)

# remove special characters
def strip_characters(text):
    t = re.sub('\(|\)|:|,|;|\.||||\?|%|>|<', '', text)
    t = re.sub('/', ' ', t)
    t = t.replace("'",'')
    return t

## word level processors:
def to_lowercase(word):
    return word.lower()

def do_stemming(stemmer):
    return lambda word: stemmer.stem(word)

def do_lemmatizing(lemmatizer):
    return lambda word: lemmatizer.lemmatize(word, pos='v')

# help function to test if word is stopword
def is_stopword(word):
    return word in stopwords.words('english')

# function to process word
def process_word_by(word_cleanner, uniqueYN):
    def cond(word):
        return (len(word) > 1 and
                not is_stopword(word) and
                not word.isnumeric() and
                word.isalnum() and
                word != len(word) * word[0])

    def clean_byword(text):
        return list(take_unique(uniqueYN)((word_cleanner(word) for word in text␣
 ↪if cond(word))))
```

```
    return clean_byword

# function to decide making a set (unique words) from text or not
def take_unique(YN):
    return set if YN else lambda x:x

# function to pre_processing the text
## compose text and word processors by combine every individual processor␣
 ↪together
text_processor = compose(replace_brackets_with_whitespace,␣
 ↪replace_contractions, strip_characters)
word_processor = compose(to_lowercase, do_lemmatizing(WordNetLemmatizer()),␣
 ↪do_stemming(PorterStemmer())) # it is crucial to do stemming after␣
 ↪lemmatization

## pre_processing function taking a dataframe and text and word processor␣
 ↪functions as input and clean the text and tokenize the specified column
def pre_processing(df, text_tools, word_tools):
    def inner(col, uniqueYN=False):
        return df[col].parallel_apply(text_tools).parallel_apply(nltk.
 ↪word_tokenize).parallel_apply(process_word_by(word_tools,uniqueYN=uniqueYN))
    return inner
```

[15]:
```
# sort by publish time
tokenized_df = df_covid_eng.sort_values(by='publish_time', ascending=False)
tokenized_df.head(n=3)
```

[15]:
```
                                     paper_id
abstract                                                body_text
authors                                                    title
journal publish_time                                         abstract_summary
30894  7d3d591550199dfc1b1e818a2c4b93a3c91db83a  A B S T R A C T Analysis of
bodily fluids usin...  Although concepts of biomedical applications o...
Byrne, Hugh J..  Bonnier, Franck...   Quantitative analysis of human blood
serum<br...                      Clinical Spectroscopy   2020-12-31   A B S T R
A C T Analysis of bodily fluids usi...
26049  a452fffd88c36ba02e3cbde8d2aa992b15290bbb  Colorimetric biosensors can be
used to detect ...  An infectious virus particle is made up of nuc...  Zhao,
Victoria Xin Ting.  Wong, Ten It...   Colorimetric biosensors for point-of-
care<br>...  Materials Science for Energy Technologies   2020-12-31
Colorimetric biosensors can be used to detect...
39262  8697a9c71461dbf2fcbdb7e6c8459a3c5ffe080b  Based on its forensic capacity
and experience ...  The International Committee of the Red Cross (...
Finegan, Oran.  Fonseca, Stephen...   International Committee of the Red
Cross<br>(...    Forensic Science International: Synergy   2020-12-31   Based on
its forensic capacity and experience...
```

[16]:
```
# created processor function with chosen text and work processors and apply it␣
 ↪to all articles to clean and tokenize all abstracts
processor = pre_processing(tokenized_df, text_processor, word_processor)
tokenized_df['abstract_token'] = processor('abstract')

# reset index (this is necessary for cosine similarity search)
tokenized_df = tokenized_df.reset_index(drop=True)

# Processor function is a generic procedure to clean and tokenize any column␣
 ↪with user specified column name, such as 'abstract' or 'body_text'
# Because processing body_text takes too long, only process abstract
# tokenized_df['body_text_token'] = processor('body_text')
```

[17]:
```
# store the dataframe
tokenized_df.to_pickle('/content/drive/My Drive/Thesis/df_all_eng_tokenized.
 ↪pkl')
# # open the dataframe
# with open('/content/drive/My Drive/Thesis/df_all_eng_tokenized.pkl', 'rb') as␣
 ↪fp:
#     tokenized_df = pickle.load(fp)

# have a look at the head of the cleanned and tokenized abstract column
tokenized_df.head()['abstract_token']
```

[17]:
```
0    [analysi, bodili, fluid, use, vibrat, spectros...
1    [colorimetr, biosensor, use, detect, particula...
2    [base, forens, capac, experi, gain, worldwid, ...
3    [homolog, human, viral, protein, establish, fa...
4    [hydrogen, technolog, fuel, cell, offer, alter...
Name: abstract_token, dtype: object
```

[18]:
```
len(tokenized_df)
```

[18]:
```
40134
```

[19]:
```
# Due to RAM contraint, select the initial 25000 publications for further␣
 ↪analysis
tokenized_df = tokenized_df[:25000]
```

[20]:
```
tokenized_df.head()
```

[20]:
```
                                  paper_id
abstract                                                body_text
authors                                                    title
journal publish_time                                     abstract_summary
abstract_token
0  7d3d591550199dfc1b1e818a2c4b93a3c91db83a  A B S T R A C T Analysis of bodily
fluids usin...  Although concepts of biomedical applications o...         Byrne,
Hugh J..  Bonnier, Franck...   Quantitative analysis of human blood serum<br...
Clinical Spectroscopy   2020-12-31   A B S T R A C T Analysis of bodily fluids
```

```
usi...  [analysi, bodili, fluid, use, vibrat, spectros...
1  a452fffd88c36ba02e3cbde8d2aa992b15290bbb  Colorimetric biosensors can be used
to detect ...  An infectious virus particle is made up of nuc...  Zhao,
Victoria Xin Ting.  Wong, Ten It...  Colorimetric biosensors for point-of-
care<br>...  Materials Science for Energy Technologies  2020-12-31
Colorimetric biosensors can be used to detect...  [colorimetr, biosensor, use,
detect, particula...
2  8697a9c71461dbf2fcbdb7e6c8459a3c5ffe080b  Based on its forensic capacity and
experience ...  The International Committee of the Red Cross (...
Finegan, Oran.  Fonseca, Stephen...  International Committee of the Red
Cross<br>(...  Forensic Science International: Synergy  2020-12-31  Based on
its forensic capacity and experience...  [base, forens, capac, experi, gain,
worldwid, ...
3  7220ce6df74c050673c37d340c06f7b94b6c7ced  Homology between human and viral
proteins is a...  Autopsies of Chinese citizens who have died fr...
Lyons-Weiler, James  Pathogenic priming likely contributes to<br>s...
Journal of Translational Autoimmunity  2020-12-31  Homology between human and
viral proteins is ...  [homolog, human, viral, protein, establish, fa...
4  131244022c7f60fa4d34c09a5c1b62d4bfc77687  Hydrogen technologies and fuel
cells offer an ...  Numerous countries across the world are now ur...  Thomas,
John Meurig.  Edwards, Peter P...  Decarbonising energy: The
developing<br>inter...  Journal of Energy Chemistry  2020-12-31
Hydrogen technologies and fuel cells offer an...  [hydrogen, technolog, fuel,
cell, offer, alter...
```

1.3 Data Exploration

1.3.1 Word cloud

[21]:
```
# create corpus
tokenized_df['abstract_corpus'] = tokenized_df['abstract_token'].apply(lambda␣
 ↪tokens: ' '.join(tokens))
corpus = tokenized_df['abstract_corpus'].tolist()

#Word cloud
from os import path
from PIL import Image
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
import matplotlib.pyplot as plt
%matplotlib inline
wordcloud = WordCloud(
                          background_color='white',
                          stopwords=stopwords.words('english'),
                          max_words=100,
                          max_font_size=50,
                          random_state=42
                         ).generate(' '.join(corpus))
```

```
fig = plt.figure(1)
fig.set_size_inches(15,12)
plt.imshow(wordcloud)
plt.axis('off')
plt.show()
```

1.3.2 Visualizing top 20 uni-grams, bi-grams & tri-grams

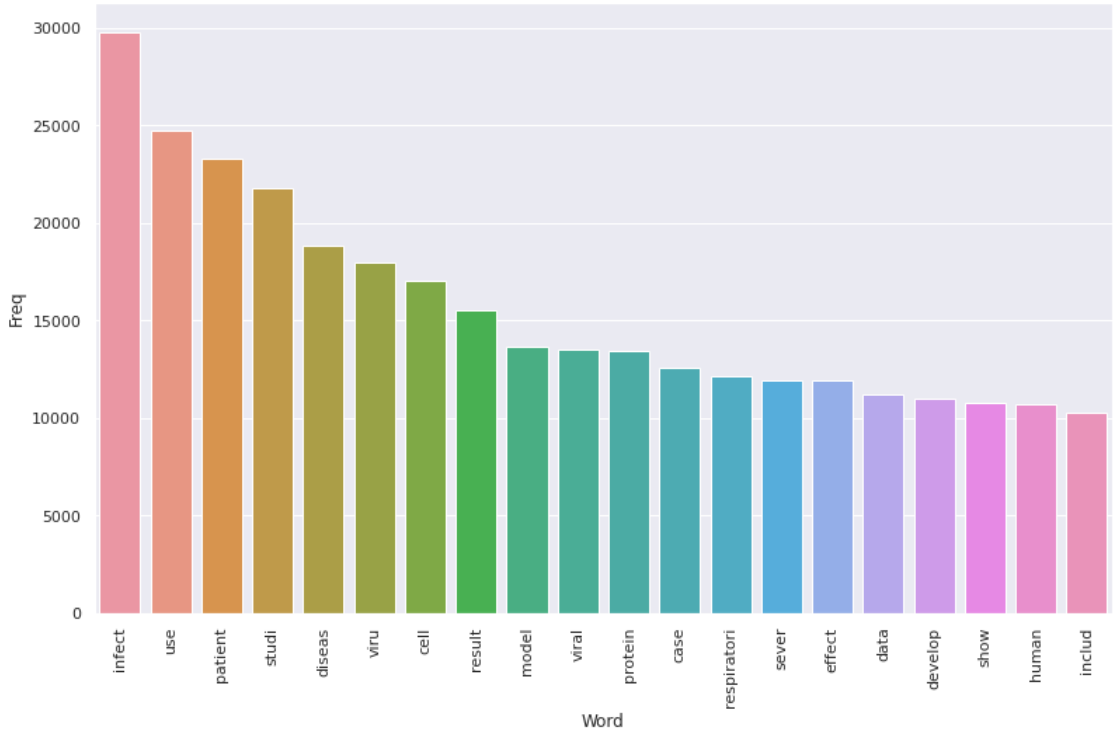
[22]:
```
# Function to get top n-grams
def get_top_nK_words(corpus, K=1, n=None):
    vec1 = CountVectorizer(max_df=0.7,stop_words=stopwords.words('english'),␣
 ↪ngram_range=(K,K),
            max_features=2000).fit(corpus)
    bag_of_words = vec1.transform(corpus)
    sum_words = bag_of_words.sum(axis=0)
    words_freq = [(word, sum_words[0, idx]) for word, idx in
                  vec1.vocabulary_.items()]
    words_freq =sorted(words_freq, key = lambda x: x[1],
                reverse=True)
    return words_freq[:n]
```

Top Uni-grams

[23]:
```
#Convert most freq words to dataframe for plotting bar plot
top_words = get_top_nK_words(corpus, K=1, n=20)
top_df = pd.DataFrame(top_words)
top_df.columns=["Word", "Freq"]
#Barplot of most freq words
```

```
sns.set(rc={'figure.figsize':(13,8)})
g = sns.barplot(x="Word", y="Freq", data=top_df)
g.set_xticklabels(g.get_xticklabels(), rotation=90)
```

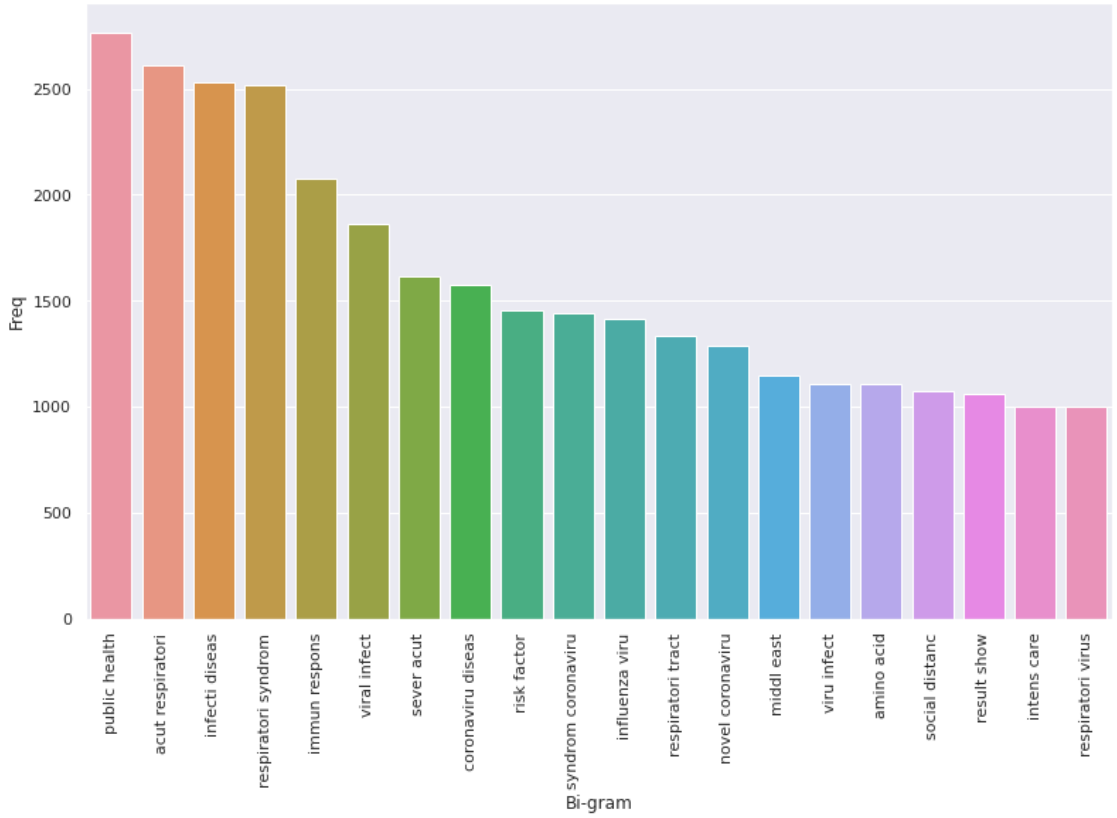
[23]:
```
[Text(0, 0, 'infect'),
 Text(0, 0, 'use'),
 Text(0, 0, 'patient'),
 Text(0, 0, 'studi'),
 Text(0, 0, 'diseas'),
 Text(0, 0, 'viru'),
 Text(0, 0, 'cell'),
 Text(0, 0, 'result'),
 Text(0, 0, 'model'),
 Text(0, 0, 'viral'),
 Text(0, 0, 'protein'),
 Text(0, 0, 'case'),
 Text(0, 0, 'respiratori'),
 Text(0, 0, 'sever'),
 Text(0, 0, 'effect'),
 Text(0, 0, 'data'),
 Text(0, 0, 'develop'),
 Text(0, 0, 'show'),
 Text(0, 0, 'human'),
 Text(0, 0, 'includ')]
```



Top Bi-grams

[24]:
```
# Top bi-grams
top2_words = get_top_nK_words(corpus, K=2, n=23)
top2_df = pd.DataFrame(top2_words)
top2_df.columns=["Bi-gram", "Freq"]
top2_df = top2_df.drop([top2_df.index[5], top2_df.index[6], top2_df.index[16]]).
 ↪reset_index(drop=True)
print(top2_df)
#Barplot of most freq Bi-grams
import seaborn as sns
sns.set(rc={'figure.figsize':(13,8)})
h=sns.barplot(x="Bi-gram", y="Freq", data=top2_df)
h.set_xticklabels(h.get_xticklabels(), rotation=90)
fig = h.get_figure()
```

```
               Bi-gram  Freq
0         public health  2767
1      acut respiratori  2611
2        infecti diseas  2531
3   respiratori syndrom  2518
4         immun respons  2078
5          viral infect  1865
6            sever acut  1615
7     coronaviru diseas  1576
8           risk factor  1452
9    syndrom coronaviru  1439
10       influenza viru  1416
11    respiratori tract  1334
12     novel coronaviru  1285
13           middl east  1149
14          viru infect  1109
15           amino acid  1107
16       social distanc  1073
17          result show  1061
18          intens care   999
19    respiratori virus   998
```

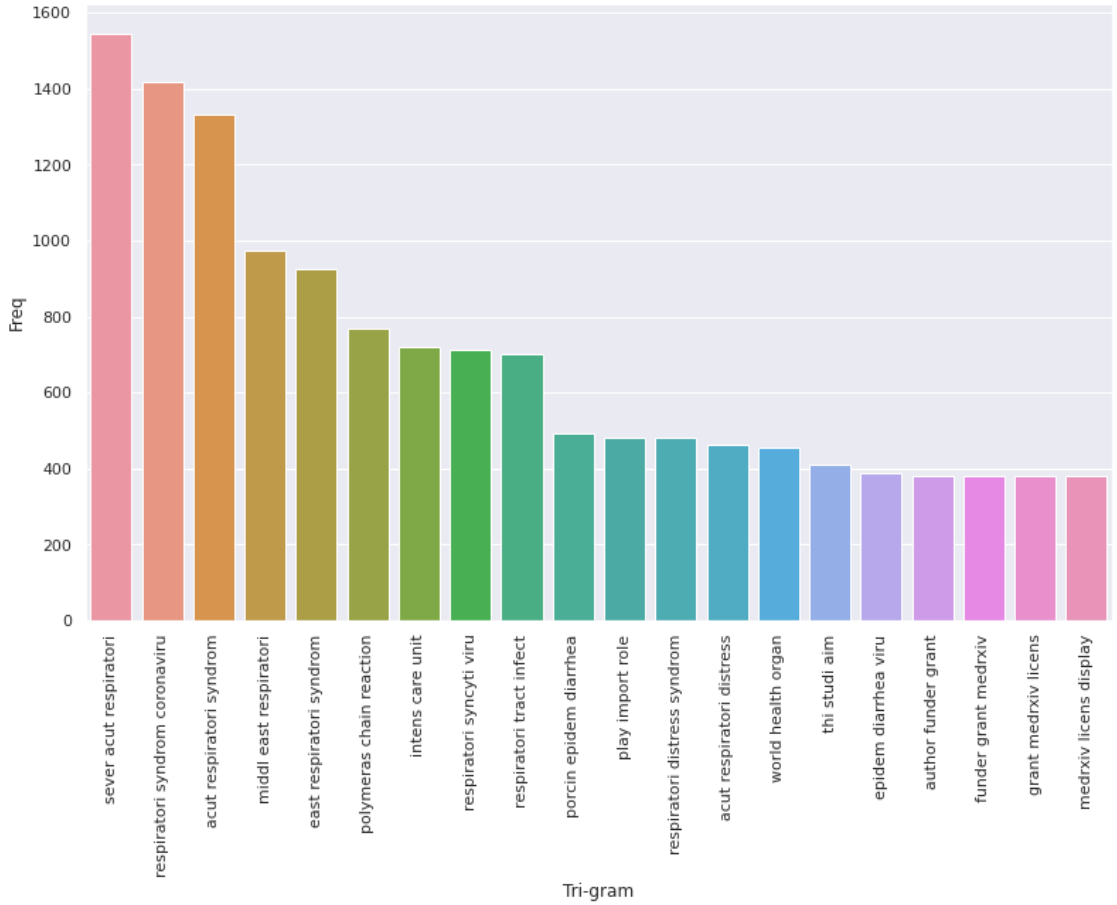


Top Tri-grams

[25]:
```
top3_words = get_top_nK_words(corpus, K=3, n=26)
top3_df = pd.DataFrame(top3_words)
top3_df.columns=["Tri-gram", "Freq"]
top3_df=top3_df.drop([top3_df.index[9], top3_df.index[10], top3_df.index[11],␣
 ↪top3_df.index[18], top3_df.index[20], top3_df.index[21]]).
 ↪reset_index(drop=True)
print(top3_df)
#Barplot of most freq Tri-grams
import seaborn as sns
sns.set(rc={'figure.figsize':(13,8)})
j=sns.barplot(x="Tri-gram", y="Freq", data=top3_df)
j.set_xticklabels(j.get_xticklabels(), rotation=90)
fig = j.get_figure()
```

```
                       Tri-gram  Freq
0          sever acut respiratori  1546
1   respiratori syndrom coronaviru  1418
2         acut respiratori syndrom  1332
3           middl east respiratori   974
```

4	east respiratori syndrom	925
5	polymeras chain reaction	769
6	intens care unit	719
7	respiratori syncyti viru	714
8	respiratori tract infect	701
9	porcin epidem diarrhea	491
10	play import role	480
11	respiratori distress syndrom	480
12	acut respiratori distress	464
13	world health organ	454
14	thi studi aim	410
15	epidem diarrhea viru	387
16	author funder grant	379
17	funder grant medrxiv	379
18	grant medrxiv licens	379
19	medrxiv licens display	379



1.4 Define Keywords and Query

Keywords:"ARS-CoV-2, Covid-19, HCoV-19, Covid corona, 2019-nCoV, sars, cov2, ncov wuhan, coronavirus, pneumonia"

Query for sentence embedding: "Knowledge of the frequency, manifestations, and course of extrapulmonary manifestations of SARS-CoV-2, including, but not limited to, possible cardiomyopathy and cardiac arrest.".

1.5 Vectorization

TF-IDF

Implement a search engine based on TF-IDF and cosine similarity with bag of word model.

1.5.1 Compute TF-IDF scores for word vectors

Compute TF-IDF scores based on word vectors

[ ]:
```
# compute TF-IDF scores for word vectors
def tfidf_(df):
    myvectorizer = TfidfVectorizer()
    vectors = myvectorizer.fit_transform(df['abstract_token'].
 ↪parallel_apply(lambda x:' '.join(x))).toarray()
    feature_names = myvectorizer.get_feature_names()
    veclist = vectors.tolist()
    out_tfidf = pd.DataFrame(veclist, columns=feature_names)
    return out_tfidf

tfidf_(tokenized_df[:20]).head()
```

1.5.2 Finding words with highest TF-IDF scores in the vectors

[27]:
```
# extract key-words with tfidf score
tfidf_scores_df = tfidf_(tokenized_df[:20])
N = 15 # Number of min/max values
u = np.argpartition(tfidf_scores_df, axis=1, kth=N).values
v = tfidf_scores_df.columns.values[u].reshape(u.shape)
maxdf = pd.DataFrame(v[:,-N:]).rename(columns=lambda x: f'Max{x+1}')
maxdf.head()
```

[27]:
```
      Max1      Max2     Max3     Max4     Max5   Max6  Max7          Max8
Max9    Max10    Max11   Max12   Max13     Max14 Max15
0  administ    adjust    adequ  address    addit   acut   act  aceinhibitor
accumul  account   accord    year  access  abstract    à3
1    admiss  administ   adjust    adequ  address  addit  acut           act
aceinhibitor  accumul  account  accord  access  abstract    à3
2  administ    adjust    adequ  address    addit   acut   act  aceinhibitor
worldwid  accumul  account  accord  access  abstract    à3
3    admiss  administ   adjust    adequ  address  addit  acut           act
aceinhibitor  accumul  account  accord  access  abstract    à3
```

```
4    adjust     adequ  address     wind  without  addit  acut          act
aceinhibitor  accumul  account  accord  access  abstract    à3
```

1.6 Filtering articles based on Keywords

1.6.1 Implementing a search engine with cosine similarity based on the TF-IDF scores

[28]:
```
# convert query token to vector
def gen_vector_T(tokens):
    Q = np.zeros((len(vocabulary)))
    x= tfidf.transform(tokens)
    #print(tokens[0].split(','))
    for token in tokens[0].split(','):
        #print(token)
        try:
            ind = vocabulary.index(token)
            Q[ind]  = x[0, tfidf.vocabulary_[token]]
        except:
            pass
    return Q

# calculate cosine similarity
def cosine_sim(a, b):
    cos_sim = np.dot(a, b)/(np.linalg.norm(a)*np.linalg.norm(b))
    return cos_sim

# Function to get transformed tfidf model
def tfidf_tran(mydf):
    vectorizer = TfidfVectorizer()
    vectors = vectorizer.fit_transform(mydf['abstract_token'].
 ↪parallel_apply(lambda x:' '.join(x)))
    return vectors

# Define wordLemmatizer
# WordNetLemmatizer requires Pos tags to understand if the word is noun or verb␣
 ↪or adjective etc. By default it is set to Noun
def wordLemmatizer(data):
    tag_map = defaultdict(lambda : wn.NOUN)
    tag_map['J'] = wn.ADJ
    tag_map['V'] = wn.VERB
    tag_map['R'] = wn.ADV
    file_clean_k =pd.DataFrame()
    for index,entry in enumerate(data):

        # Declaring Empty List to store the words that follow the rules for␣
 ↪this step
        Final_words = []
```

```
        # Initializing WordNetLemmatizer()
        word_Lemmatized = WordNetLemmatizer()
        # pos_tag function below will provide the 'tag' i.e if the word is␣
↪Noun(N) or Verb(V) or something else.
        for word, tag in nltk.pos_tag(entry):
            # Below condition is to check for Stop words and consider only␣
↪alphabets
            if len(word)>1 and word not in stopwords.words('english') and word.
↪isalpha():
                word_Final = word_Lemmatized.lemmatize(word,tag_map[tag[0]])
                Final_words.append(word_Final)
            # The final processed set of words for each iteration will be␣
↪stored in 'text_final'
                file_clean_k.loc[index,'Keyword_final'] = str(Final_words).
↪lower()
                file_clean_k=file_clean_k.replace(to_replace ="\[.", value =␣
↪'', regex = True)
                file_clean_k=file_clean_k.replace(to_replace ="'", value = '',␣
↪regex = True)
                file_clean_k=file_clean_k.replace(to_replace =" ", value = '',␣
↪regex = True)
                file_clean_k=file_clean_k.replace(to_replace ='\]', value = '',␣
↪regex = True)
    return file_clean_k
```

[29]:
```
def cosine_similarity_T(k, query, text_token_df):
    preprocessed_query  = re.sub("\W+", " ", query).strip()
    tokens = nltk.word_tokenize(text_processor(str(preprocessed_query).lower()))
    tokens = [word_processor(token) for token in tokens if
              len(token) > 1 and
              not is_stopword(token) and
              not token.isnumeric() and
              token.isalnum() and
              token != len(token) * token[0]]
    q_df = pd.DataFrame(columns=['q_clean'])
    q_df.loc[0,'q_clean'] =tokens
    q_df['q_clean'] = wordLemmatizer(q_df.q_clean)
    d_cosines = []
    #print(q_df['q_clean'])
    query_vector = gen_vector_T(q_df['q_clean'])
    #print(query_vector)
    #print(q_df['q_clean'])
    #print(sum(query_vector))
    for d in tfidf_tran.A:
        d_cosines.append(cosine_sim(query_vector, d))
    #print(d_cosines)
    out = np.array(d_cosines).argsort()[-k:][::-1]
```

```
    d_cosines.sort()
    #print(out)
    a = pd.DataFrame()
    firsttime=True
    for i,index in enumerate(out):
        try:
            a.loc[i, 'Paper ID'] = text_token_df['paper_id'][index]
            a.loc[i,'Title'] = text_token_df['title'][index]
            a.loc[i, 'Summary'] = text_token_df['abstract_summary'][index]
        except KeyError as e:
            if firsttime:
                print("Fewer matches are found than requested {}".format(k))
                firsttime=not firsttime
                pass
    for j,simScore in enumerate(d_cosines[-k:][::-1]):
        a.loc[j,'Score'] = simScore
    return a
```

[30]:
```
import nltk
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
```

[30]:
```
True
```

1.6.2 Applying TF-IDF search engine with Keywords

[31]:
```
## Create Vocabulary
vocabulary = set()
for tokens in tokenized_df.abstract_token:
    vocabulary.update(tokens)
vocabulary = list(vocabulary)

# Intializating the tfIdf model
tfidf = TfidfVectorizer(vocabulary=vocabulary)

# Transform the TfIdf model
tfidf_tran=tfidf.fit_transform(tokenized_df['abstract_token'].
 ↪parallel_apply(lambda x:' '.join(x)))

# search engine using cosine similarity + TF-IDF
TFIDF_output = cosine_similarity_T(20000,'SARS-CoV-2 Covid-19 HCoV-19 Covid␣
 ↪corona 2019-nCoV sars cov2 ncov wuhan coronavirus pneumonia',tokenized_df)
TFIDF_output_significant = TFIDF_output[TFIDF_output['Score'] > 0]
TFIDF_output_significant.head()
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:17: RuntimeWarning:
invalid value encountered in double_scalars
```

[31]:
```
                                  Paper ID
Title                                                 Summary     Score
0  b628598eb595b481c3be76b2bf433007b9184a23   Nitric oxide inhalation as an
interventional<...                                                COVID-  0.290575
1  769d953d28605c51f20b2fd049ffabc4d73583a7   On the assessment of more reliable
COVID-19<b...                                  COVID-19 (SARS-CoV-2  0.266964
2  30b5c7b8faf95265f67ae59f4686eaf9b2772893   Estimated effectiveness of
traveller<br>scree...   14 Traveller screening is being used to limit...
0.266812
3  924855757497658770d841c959cec97eebff5c41   Trends and Prediction in Daily New
Cases and<...   Results: There were 636,282 new cases and<br>...  0.264267
4  a6af28a7f0a0d8d1e4d5b06f5ae38412298bbcb4   Direct Measurement of Rates of
Asymptomatic<b...   The pandemic potential of the novel<br>corona...  0.258771
```

[32]:
```
# store the dataframe to google drive
TFIDF_output_significant.to_pickle('/content/drive/My Drive/Thesis/
 ↪TFIDF_output_significant_all.pkl')

# The amount of the most significant search results
len(TFIDF_output_significant)
```

[32]:
```
7062
```

1.7 Sentence embedding using BERT on Articles and Query

The data stored by the TF-IDF search engine is loaded, only get the papers that it found and for the sentences of those papers obtain and store the sentence embeddings.

[33]:
```
# TFIDF_output_significant.to_pickle('/content/drive/My Drive/Thesis/
 ↪TFIDF_output_significant_all.pkl')
```

[34]:
```
get_top = 500
top_to_print = 10

#with open('/content/drive/My Drive/Thesis/TFIDF_output_significant_all.pkl',␣
 ↪'rb') as fp:
#    TFIDF_output_significant = pickle.load(fp)
# with open('/content/drive/My Drive/Thesis/df_all_eng.pkl', 'rb') as fp:
#     df_covid_eng = pickle.load(fp)

df_covid_eng.drop_duplicates(subset=['paper_id'], inplace=True)
TFIDF_output_significant.drop_duplicates(subset=['Paper ID'], inplace=True)

papers_to_embed = df_covid_eng.loc[df_covid_eng['paper_id'].isin(
    TFIDF_output_significant['Paper ID'])].copy()
```

```
sort_papers = TFIDF_output_significant.loc[
    TFIDF_output_significant['Paper ID'].isin(
        papers_to_embed['paper_id'])].sort_values(
    by='Score', ascending=False)['Paper ID'].to_list()
papers_to_embed = papers_to_embed.set_index('paper_id').loc[sort_papers].
 ↪reset_index()
```

[35]:
```
tqdm.pandas(desc='Combining abstracts and body text')
papers_to_embed['combined_text'] = papers_to_embed.progress_apply(
        lambda x: x['abstract'] + ' ' + x['body_text'], axis=1)

tqdm.pandas(desc='Splitting abstracts into sentences')
papers_to_embed['abstract_sentence'] = papers_to_embed[
    'abstract'].progress_apply(tokenize.sent_tokenize)

tqdm.pandas(desc='Splitting papers into sentences')
papers_to_embed['combined_text_sentence'] = papers_to_embed[
    'combined_text'].progress_apply(tokenize.sent_tokenize)
```

```
Combining abstracts and body text: 100%|| 7062/7062 [00:00<00:00,
34593.87it/s]
Splitting abstracts into sentences: 100%|| 7062/7062 [00:02<00:00,
2546.70it/s]
Splitting papers into sentences: 100%|| 7062/7062 [00:36<00:00,
193.01it/s]
```

[36]:
```
# Select BERT
embedder = SentenceTransformer('bert-large-nli-stsb-mean-tokens')

# # Select RoBERTa
# # embedder = SentenceTransformer('roberta-large-nli-stsb-mean-tokens')
```

```
100%|| 1.24G/1.24G [00:14<00:00, 85.7MB/s]
```

[37]:
```
sent_to_embed_abstr = list(itertools.chain(
    *papers_to_embed['abstract_sentence']))
sent_to_embed_comb = list(itertools.chain(
    *papers_to_embed['combined_text_sentence'].iloc[:get_top]))

abstract_embed = np.array(embedder.encode(
    sent_to_embed_abstr, batch_size=64, show_progress_bar=True))
comb_text_embed = np.array(embedder.encode(
    sent_to_embed_comb, batch_size=64, show_progress_bar=True))
```

```
HBox(children=(FloatProgress(value=0.0, description='Batches', max=1236.0,
style=ProgressStyle(description_wid
```

```
HBox(children=(FloatProgress(value=0.0, description='Batches', max=1210.0,
style=ProgressStyle(description_wid
```

[38]:
```
# save intermidiate data in case needed
np.save('/content/drive/My Drive/Thesis/finalreport/abstr_data_encodings',␣
 ↪abstract_embed)
np.save('/content/drive/My Drive/Thesis/finalreport/comb_text_data_encodings',␣
 ↪comb_text_embed)
```

Here the sentence embeddings are loaded and the query is embedded.

[39]:
```
# # load intermidiate data in case needed
# abstract_embed = np.load('/content/drive/My Drive/Thesis/finalreport/
 ↪abstr_data_encodings.npy')
# comb_text_embed = np.load('/content/drive/My Drive/Thesis/finalreport/
 ↪comb_text_data_encodings.npy')
```

[40]:
```
questions = [
    ('Knowledge of the frequency, manifestations, and course of extrapulmonary␣
 ↪manifestations of SARS-CoV-2, including, but not limited to, possible␣
 ↪cardiomyopathy and cardiac arrest.')
]
```

[41]:
```
questions_embed = np.array(embedder.encode(
    questions, batch_size=64, show_progress_bar=True))
```

```
HBox(children=(FloatProgress(value=0.0, description='Batches', max=1.0,
style=ProgressStyle(description_width=
```

Comparing every sentence in the abstracts and combined text with the sentence query in embedding space using the cosine similarity measure. The sentences are ranked based on their cosine similarity measure. Top 50 sentences for both the abstracts and combined text are kept.

[42]:
```
similarity_abstr = cosine_similarity(abstract_embed, questions_embed).squeeze()
similarity_comb = cosine_similarity(
    comb_text_embed, questions_embed).squeeze()

sort_args_abstr = np.argsort(similarity_abstr)[::-1]
sim_sort_abstr = similarity_abstr[sort_args_abstr]
sort_args_comb = np.argsort(similarity_comb)[::-1]
sim_sort_comb = similarity_comb[sort_args_comb]

paper_id_abst = np.array(list(itertools.chain(
    *papers_to_embed.progress_apply(
        lambda x: [x['paper_id']] * len(x['abstract_sentence']),
```

```
        axis=1).tolist())))
paper_id_comb = np.array(list(itertools.chain(
    *papers_to_embed.iloc[:get_top].progress_apply(
        lambda x: [x['paper_id']] * len(x['combined_text_sentence']),
        axis=1).tolist())))
```

```
Splitting papers into sentences: 100%|| 7062/7062 [00:00<00:00,
52769.73it/s]
Splitting papers into sentences: 100%|| 500/500 [00:00<00:00,
56964.61it/s]
```

[43]:
```
interest_paper_id_abstr = paper_id_abst[sort_args_abstr]
interest_sentences_abstr = np.array(
    sent_to_embed_abstr)[sort_args_abstr]
interest_abstracts = papers_to_embed.set_index('paper_id').loc[
    interest_paper_id_abstr]['abstract'].tolist()

interest_paper_id_comb = paper_id_comb[sort_args_comb]
interest_sentences_comb = np.array(
    sent_to_embed_comb)[sort_args_comb]
interest_comb_text = papers_to_embed.set_index('paper_id').loc[
    interest_paper_id_comb]['combined_text'].tolist()

with open('/content/drive/My Drive/Thesis/finalreport/
 ↪interesting_papers_based_on_abstract.txt', 'w') as f:
    for paper, sent, abst, metric in zip(
        interest_paper_id_abstr, interest_sentences_abstr, interest_abstracts,␣
 ↪sim_sort_abstr):
        _ = f.write('Paper ID: ' + paper + '\n')
        _ = f.write('Important sentence: ' + sent + '\n')
        # _ = f.write('Associated abstract: ' + abst + '\n')
        _ = f.write('Cosine Similarity metric: ' + '{0:.3f}'.format(metric) +␣
 ↪'\n')
        _ = f.write('\n')

with open('/content/drive/My Drive/Thesis/finalreport/
 ↪interesting_papers_based_on_comb_text.txt', 'w') as f:
    for paper, sent, comb_text, metric in zip(
        interest_paper_id_comb, interest_sentences_comb, interest_comb_text,␣
 ↪sim_sort_comb):
        _ = f.write('Paper ID: ' + paper + '\n')
        _ = f.write('Important sentence: ' + sent + '\n')
        _ = f.write('Cosine Similarity metric: ' + '{0:.3f}'.format(metric) +␣
 ↪'\n')
        # _ = f.write('Associated body text: ' + comb_text + '\n')
```

```
        _ = f.write('\n')
```

Results are printed

[44]:
```
print('Results based on abstract:')
print('"""')
with open('/content/drive/My Drive/Thesis/finalreport/
 ↪interesting_papers_based_on_abstract.txt', 'r') as f:
    print('\n'.join(f.read().splitlines()[:4*top_to_print]))
print('"""')
print('')
print('Results based on abstract and body text:')
print('"""')
with open('/content/drive/My Drive/Thesis/finalreport/
 ↪interesting_papers_based_on_comb_text.txt', 'r') as f:
    print('\n'.join(f.read().splitlines()[:4*top_to_print]))
print('"""')
```

```
Results based on abstract:
"""
Paper ID: 9da2ea921dcea8312b8c1491c1dd004bd42a53a8
Important sentence: Potential contributors to acute cardiac injury in the
setting of COVID-19 include 1) acute changes in myocardial demand and supply due
to tachycardia, hypotension, and hypoxemia resulting in type 2 myocardial
infarction; 2) acute coronary syndrome due to acute atherothrombosis in a
virally-induced thrombotic and inflammatory milieu; 3) microvascular dysfunction
due to diffuse microthrombi or vascular injury; 4) stress-related cardiomyopathy
(Takotsubo syndrome); 5) non-ischemic myocardial injury due to a
hyperinflammatory cytokine storm; or 6) direct viral cardiomyocyte toxicity and
myocarditis.
Cosine Similarity metric: 0.710

Paper ID: 57dc2f7140d1f539058c3b5961e2c80901341397
Important sentence: Cardiac manifestations related to COVID-19 include demand
ischemia, fulminant myocarditis, myocardial infarction and arrhythmias.
Cosine Similarity metric: 0.688

Paper ID: 9da2ea921dcea8312b8c1491c1dd004bd42a53a8
Important sentence: Acute cardiac injury, manifest by increased blood levels of
cardiac troponin, electrocardiographic abnormalities, or myocardial dysfunction,
occurs in up to ~60% of hospitalized patients with severe COVID-19.
Cosine Similarity metric: 0.679

Paper ID: cd94213e0bab4d53ef4d968a25fff5d7c42a5d91
Important sentence: Summary SARS-CoV-2 infection related acute cardiac injury
cannot be ignored, and its underlying mechanisms remain speculated.
Cosine Similarity metric: 0.670
```

```
Paper ID: e07cbaa2259e508e40acbfd78f1f0496e94d4109
Important sentence: Furthermore, patients with pre-existing cardiac disease are
likely to experience a more severe disease course with COVID-19.
Cosine Similarity metric: 0.660

Paper ID: e56de2a1f43c2756876348e5482549729ee93f1b
Important sentence: Early clinical evidence suggests that severe cases of
coronavirus disease 2019 , caused by the severe acute respiratory syndrome
coronavirus 2 (SARS-CoV-2), are frequently characterized by hyperinflammation,
imbalance of renin-angiotensin-aldosterone system, and a particular form of
vasculopathy, thrombotic microangiopathy, and intravascular coagulopathy.
Cosine Similarity metric: 0.658

Paper ID: b4fe7209f1fe1cd59c4680cc3811c3fec9144911
Important sentence: Severe acute respiratory syndrome coronavirus 2 is
associated with a prothrombotic state in infected patients.
Cosine Similarity metric: 0.655

Paper ID: c99b9b3f8734b2c25ea3a1eace4607adb4bb60d2
Important sentence: It has been postulated that CIC may be an uncontrolled
immunothrombotic response to COVID-19, and there is growing evidence of venous
and arterial thromboembolic events in these critically ill patients.
Cosine Similarity metric: 0.655

Paper ID: cd94213e0bab4d53ef4d968a25fff5d7c42a5d91
Important sentence: Although SARSCoV-2 causes primarily respiratory problems,
concurrent cardiac injury cannot be ignored since it may be an independent
predictor for adverse outcomes.
Cosine Similarity metric: 0.652

Paper ID: 1769e07fe8ccfc53345a96818c2f39ae22fa6fdc
Important sentence: Conclusions Both ischemic and hemorrhagic stroke can
complicate the course of COVI-19 infection.
Cosine Similarity metric: 0.648

"""

Results based on abstract and body text:
"""
Paper ID: 867ebb2be2d466bcb35e5a7319708eaff39c4799
Important sentence: The complications of sedative procedures, of deep sedation
and of endotracheal intubation is bradycardia and cardiac arrest, 2 case,
because of hypoxia and vagal stimulation.
Cosine Similarity metric: 0.715

Paper ID: 85a2db1bb25d3c369b7422b2b5f5a007175bec15
Important sentence: Cardiac complications such as electrocardiography
abnormities, diastolic dysfunction, and acute myocardial injury were reported in
```

patients with COVID-19 [124] [125] [126] [127] .
Cosine Similarity metric: 0.682

Paper ID: 85a2db1bb25d3c369b7422b2b5f5a007175bec15
Important sentence: The activated SNS alters cardiac wall contractility and increases apoptotic pathways in cardiomyocytes, contributing to CVD development .
Cosine Similarity metric: 0.659

Paper ID: 929545880647bc518d1b5ab4112d452dcf9b36d9
Important sentence: This may be similar to what is observed in many patients with acute respiratory illnesses; or it may indicate myocardial injury because of the virus as ACE-2 receptors are widely expressed on cardiomyocytes.
Cosine Similarity metric: 0.658

Paper ID: f12a115ffa060877bd7a292df682bac1b99a8193
Important sentence: A prior study demonstrated that cardiac injury and cardiovascular events in the form of elevated troponin and left ventricular systolic dysfunction are common post-CAR-T; in a cohort of 137 patients with post-CAR-T cytokine release syndrome (CRS), 21% had elevated troponin and 12% developed cardiovascular events including cardiac arrest, decompensated heart failure, and arrhythmias [65] .
Cosine Similarity metric: 0.652

Paper ID: 85a2db1bb25d3c369b7422b2b5f5a007175bec15
Important sentence: Current case reports show that SARS-CoV-2 infection may have cardiovascular symptoms in addition to the typical respiratory symptoms.
Cosine Similarity metric: 0.643

Paper ID: 934f45d3ce46b7e090b1a3d76a96e2acb705cc91
Important sentence: Our results demonstrate two consistent and dramatic phenotypes associated with MHV-induced macropinocytosis: ruffling with vesicle internalization and filopodia associated with cell fusion and recruitment into syncytia.
Cosine Similarity metric: 0.642

Paper ID: 9af82ab15c96ec049b8766fa2d5db9d3d27c3461
Important sentence: First, the observed ILI surge may represent more than just SARS-CoV-2 infected patients.
Cosine Similarity metric: 0.640

Paper ID: ba139493dfc91d8e847d162613747f40950add89
Important sentence: Furthermore, some coronaviruses have been demonstrated able to spread via a synapse-connected route to the medullary cardiorespiratory center from the mechanoreceptors and chemoreceptors in the lung and lower respiratory airways.
Cosine Similarity metric: 0.636

```
Paper ID: f12a115ffa060877bd7a292df682bac1b99a8193
Important sentence: This systemic release of cytokines, characterized by
increased IL-2, IL-6, IL-10, GCSF, IFN-, MCP-1, MIP-1-, and TNF-, likely
contributes to cardiac injury in a situation analogous to cardiotoxicity in the
setting of chimeric antigen receptor (CAR)-T cell therapy.
Cosine Similarity metric: 0.628

"""
```

1.8 Sentence Clustering and Knowledge

To better understand the results from the final sentence embedding step, the sentences are clustered in 5 groups.

1.8.1 Sentence embedding clustering

Cluster the embedded sentences in embedding space. Print sentences associated with each found cluster.

[45]:
```
rows_to_sample = np.random.randint(len(comb_text_embed), size=1000)

sentences_subset = np.array(sent_to_embed_comb)[rows_to_sample].tolist()
embeddings_subset = comb_text_embed[rows_to_sample]

# Perform kmean clustering
num_clusters = 5
clustering_model = KMeans(n_clusters=num_clusters)
_ = clustering_model.fit(embeddings_subset)
cluster_assignment = clustering_model.labels_

clustered_sentences = [[] for i in range(num_clusters)]
for sentence_id, cluster_id in enumerate(cluster_assignment):
    clustered_sentences[cluster_id].append(sentences_subset[sentence_id])

for i, cluster in enumerate(clustered_sentences):
    print("Cluster ", i+1)
    print(cluster[:10])
    print("")
```

```
Cluster  1
['Whether distantly related proteins in HCoV-OC43 and HCoV-HKU1 might have
similar activity remains to be determined.', 'Percentage of patients receiving
ribavirin and corticosteroids did not differ between patients with and without
ARF (92% and 91.2%, respectively).', 'A single report described isolation and
characterization of bovine-like CoVs from giraffes during the outbreak of
diarrhoea in an Ohio wild animal park in 2003 (Hasoksuz et al., 2007) .',
'However, introduction of two point mutations (H183R and Y241H) in the S protein
```

of HCoV-OC43 led to a higher degree of XBP1 cleavage, followed by a strong activation of caspase-3 and nuclear fragmentation [93] .', 'The nuclear transportation of NSP2 starts at the earliest during the infection, and the nuclear localization is mediated through the pentapeptide PRRRV (aa 647-651) (Rikkomen, 1996) .', 'The transmission of infection starts when a pathogen enters a host body and starts infecting its protein which in turn affects its directly or indirectly connected neighborhood proteins.', 'Genetically engineered unrelated viral genome with deficient packaging elements for encoding targeted gene Spike and nucleocapsid proteins [100, 104] Spike and nucleocapsid proteins [87, 88] Safety; stronger and specific cellular and humoral immune responses [77] .', 'Selected clones were taken forward for expression analysis.', 'This study emphasizes the importance of bat species selection for studying CoVs in bats.', 'Similar to HR1 from \uf061-HCoVs, sequence alignment across HCoVs revealed that a 14-amino acid insertion also exists in the HR2 region of \uf061-HCoVs (Fig.']

Cluster 2
['Wuhan is the major air and train transportation hub of central China (figure 1).', '3A).', 'two intertwined ACE2R-ACE2R; [ACE2R] 4).', '31 .', '2B) .', '.', 'and K.Z.', 'Pakistan has trade and travel with Iran and China.', 'In contrast, a single prominent cleavage product was observed in the present study.', '(Pascal et al.']

Cluster 3
['In humans, CoVs cause mainly respiratory tract infections.', 'Bats can carry and transmit CoVs into local bat populations via migration even though little is known about the migratory patterns of these animals.', 'Community-acquired pneumonia (CAP) is a common illness that affects millions of people each year, and it is one of the most common infectious diseases that can lead to morbidity and mortality worldwide.', 'and X.C., more than 15 years of experience) reached a consensus and were blinded to the clinical and laboratory findings.', 'The percentage of heart injury in COVID-19 patients who died, approximately 28% to 89% [12, 13, 132] , was higher than that in those who survived.', 'This virus is significantly different from previously reported coronaviruses for many reasons, such as the short anchor of the S protein, the specific number and location of small ORFs, and the presence of only one copy of PLP pro .', 'Because of the ability of fly and migrate and the tendency to live in the large sizes of social groups, bats are predisposed to maintain and transport viruses to other mammals for the possible outbreaks of epidemics (Bennett, 2006) .', 'However, seasonal pattern variations and viral outbreaks can considerably alter the prevalence of certain viruses between surveillance studies, especially for RSV and influenza virus [12] .', 'In addition, we emphasize the role of extended thromboprophylaxis in discharged patients.', 'These data highlighted extensive cross-reactivity of HCoV patient sera with SARS-CoV-2 S, and specific recognition of the S1 subunit only by COVID-19 patient sera.']

Cluster 4
['However, as for a novel disease, common people have more panic and anxiety on

it than other diseases.', 'Our incidence rates were calculated only for LRTIs, because URTIs are underreported in our hospital patient administrative system.', '10, 11, [104] [105] [106] [107] [108] [109] Many of these trials used the WHO definition of pneumonia, which is solely based on clinical symptoms and signs, is highly sensitive, but also has a low specificity.', 'The entry receptor (DPP4) for MERS-CoV is also highly expressed in the kidney, causing renal dysfunctions by either hypoxic damage or direct infection of the epithelia [42] .', '8 For example, household air pollution is an important risk factor for acute lower respiratory infections in children (with a population attributable fraction of 52%) and accounts for 39ů1 million disability-adjusted life years lost and 455 000 deaths in 2014.', 'It will be essential to strike the correct balance between the rights of damaged individuals and the proper safeguarding of health-care professionals in a frame that considers the responsibilities of the whole health system.', 'As expected, a slightly larger value of the RMSD is observed for the protein, as a consequence of its larger flexibility compared to the rigid G4 structures (Figure 2 d, c) .', 'The nadir absolute neutrophil and lymphocyte counts dropped to 3.95 Œ 10 9 /L (0.8 to 9.9) and 0.3 Œ 10 9 /L (0.1 to 0.7), respectively.', 'Culture methods have been the gold standard for diagnosis but are too insensitive producing a result after several days or even several weeks and are therefore not relevant for the management of acute illness.', '5B) , indicating that this construct has a dominant negative effect on viral replication.']

Cluster 5
['Risk management and assessment are performed in an open and transparent environment based on communication and dialog.', 'Details of these mathematical models are provided in S1 File.', 'If the coordinates listed for a particular bat were on the line between grid cells, it was arbitrarily included in the grid cell with higher latitude/longitude.', 'Nsp10 performs several functions for SARS-CoV.', 'Then active notification was proceeded and infection control interventions were activated.', 'Methods: We prospectively investigated the existence of MDROs, including MRSA, in residents of six LTCFs and their environments from January to December 2016.', 'https://doi.org/10.1101/2020.03.23.20034058 doi: medRxiv preprint reporting system, these cases were reported with a time lag.', 'Continuing education programs, especially if mandatory, also play a significant role.', 'https://doi.org/10.1101/2020.02.11.20021493 doi: medRxiv preprint the author/funder, who has granted medRxiv a license to display the preprint in perpetuity.', 'To compare the predictive value of the resulting model with a model based on the NEWS2 total score alone.']

1.9 Knowledge Graph based Visualization

[46]:
```
file_to_read = '/content/drive/My Drive/Thesis/finalreport/
 ↪interesting_papers_based_on_comb_text.txt'
content = None
with open(file_to_read) as f:
    content = f.readlines()
content = [x.strip() for x in content]
content = [string for string in content if string != ""]
top_results = content[0:100] # Select the first n elements.
selected_top_sentences = []
for elem in top_results:
    if elem.startswith('Important sentence:'):
        selected_top_sentences.append(elem.replace('Important sentence:','').
 ↪strip())
# Select the first n sentences.
selected_top_sentences = selected_top_sentences[0:20]

# Some settings for the plot.
pd.set_option('display.max_colwidth', 200)

# The main idea is to go through a sentence and extract the subject and the␣
 ↪object
# as and when they are encountered.
def get_entities(sent):
    ## chunk 1
    ent1 = ""
    ent2 = ""
    prv_tok_dep = ""  # dependency tag of previous token in the sentence
    prv_tok_text = ""  # previous token in the sentence
    prefix = ""
    modifier = ""
    for tok in nlp(sent):
        ## chunk 2
        # if token is a punctuation mark then move on to the next token
        if tok.dep_ != "punct":
            # check: token is a compound word or not
            if tok.dep_ == "compound":
                prefix = tok.text
                # if the previous word was also a 'compound' then add the␣
 ↪current word to it
                if prv_tok_dep == "compound":
                    prefix = prv_tok_text + " " + tok.text
            # check: token is a modifier or not
            if tok.dep_.endswith("mod") == True:
                modifier = tok.text
```

```
                # if the previous word was also a 'compound' then add the␣
 ↪current word to it
                if prv_tok_dep == "compound":
                    modifier = prv_tok_text + " " + tok.text
            ## chunk 3
            if tok.dep_.find("subj") == True:
                ent1 = modifier + " " + prefix + " " + tok.text
                prefix = ""
                modifier = ""
                prv_tok_dep = ""
                prv_tok_text = ""
            ## chunk 4
            if tok.dep_.find("obj") == True:
                ent2 = modifier + " " + prefix + " " + tok.text
            ## chunk 5
            # update variables
            prv_tok_dep = tok.dep_
            prv_tok_text = tok.text
    return [ent1.strip(), ent2.strip()]

# Relation/Predicate Extraction.
def get_relation(sent):
  doc = nlp(sent)
  # Matcher class object
  matcher = Matcher(nlp.vocab)
  #define the pattern
  pattern = [{'DEP':'ROOT'},
             {'DEP':'prep','OP':"?"},
             {'DEP':'agent','OP':"?"},
             {'POS':'ADJ','OP':"?"}]
  matcher.add("matching_1", None, pattern)
  matches = matcher(doc)
  k = len(matches) - 1
  span = doc[matches[k][1]:matches[k][2]]
  return(span.text)

entity_pairs = []

for i in tqdm(selected_top_sentences):
  entity_pairs.append(get_entities(i))

# extract relationship
relations = [get_relation(i) for i in tqdm(selected_top_sentences)]
print(relations)

# extract subject
source = [i[0] for i in entity_pairs]
```

```
print(source)

# extract object
target = [i[1] for i in entity_pairs]
print(target)

kg_df = pd.DataFrame({'source':source, 'target':target, 'edge':relations})

# Use the library networkx to build graph.
# Create a directed graph from the dataframe first.
sentence_graph = nx.from_pandas_edgelist(kg_df,
                                         "source",
                                         "target",
                                         edge_attr=True,
                                         create_using=nx.MultiDiGraph())
plt.figure(figsize=(12,9)) # Change this to make plotting area for knowledge␣
 ↪graph bigger or smaller.
#pos = nx.spring_layout(sentence_graph)
pos = nx.spring_layout(sentence_graph,
                       k = 1.3,
                       iterations = 100,
                       fixed = None,
                       center = (0,0),
                       scale = 4)
nx.draw(sentence_graph,
        with_labels=True,
        node_color='skyblue',
        edge_cmap=plt.cm.Blues,
        pos=pos)
plt.show()
```

```
100%|| 20/20 [00:00<00:00, 71.23it/s]
100%|| 20/20 [00:00<00:00, 104.53it/s]
```

```
['is bradycardia', '127', 'contractility', 'be similar', 'elevated', 'show',
'demonstrate', 'represent more', 'demonstrated able', 'contributes to cardiac',
'increased in', 'caused by cerebral', 'were', 'develop over', 'be analogous',
'be peripheral', 'confined to', 'underwent', 'reported', 'appears']
['complications', 'Cardiac  complications', '', 'myocardial  receptors', '21
cytokine release %', 'SARS CoV-2 infection', 'results', 'observed ILI surge',
'Furthermore  coronaviruses', 'systemic  release', 'Cardiac troponin Ilevels',
'1  Cardiac', 'ICU care admission', 'communications', 'Cardiac  damage',
'peripheral SARS which', 'respiratory  they', 'severe laboratory parameters',
'recent  studies', 'SARS CoV-2']
['hypoxia endotracheal stimulation', 'myocardial electrocardiography COVID-19',
'apoptotic CVD development', 'widely  cardiomyocytes', 'cardiac  arrest',
'respiratory  symptoms', 'induced cell syncytia', 'infected  patients',
'cardiorespiratory  lung', 'chimeric antigen receptor', 'milder SARS CoV-2
```

disease', '1 case', 'more Group', 'venous collaterals', 'CAR T.', 'due brain barrier inflammation', 'neurological Central diseases', 'deceased brain Table', 'cardiovascular JAMA detail', '128']