Question Answering on Convid-19 information using Knowledge Graphs

Theodora Bakaloudi

SID: 3308200003

SCHOOL OF SCIENCE & TECHNOLOGY

A thesis submitted for the degree of

Master of Science (MSc) in Data Science

JANUARY 2023

THESSALONIKI – GREECE

Question Answering on Convid-19 information using Knowledge Graphs

Theodora Bakaloudi

SID: 3308200003

Supervisor: Dr. Christos Berberidis

SCHOOL OF SCIENCE & TECHNOLOGY

A thesis submitted for the degree of

Master of Science (MSc) in Data Science

JANUARY 2023

THESSALONIKI – GREECE

Abstract

This dissertation was written as a part of the MSc in Data Science at the International Hellenic University.

In this study we analyze the different approaches for creating a knowledge graph by using documents as an input as well as how we can leverage the knowledge graphs with Natural Language processing in order to improve question answering related to Covid-19. This dissertation consists of two parts, the theoretical one and the practical. In the first part, where the literature review is presented, the whole process of building a knowledge graph is described in detail. Terms like Named Entity Recognition, entity prediction, entity linking, relation prediction, open information extraction are presented and defined. The main purpose, apart from understanding how a knowledge graph is constructed, was to understand how the knowledge graphs can contribute to question answering systems, what is their relation with search engines and their association with Covid-19 data. In the second part an effort was conducted to build a knowledge graph by using part of the CORD-19 dataset as well as to use haystack, an open-source framework for building search systems, in our case based on Covid-19 data from CORD-19 dataset as well. In both parts, is clearly shown how Knowledge graphs can help improve question answering both in general and specifically in our case for Covid-19 information.

I would like to express my deepest gratitude to my thesis supervisor Dr Christos Berberidis and Ioannis Konstantinidis for their valuable help and assistance, their availability and the numerous explanations throughout the process.

Theodora Bakaloudi

1/3/2023

Contents

1 Introduction

Recent times are interrelated with rapid technological advancements, the continuous evolution of tools and processes, and the huge importance of semantic metadata. Humans are unable to fully absorb, understand, and take complicated decisions on the volume of data created today by both humans and machines. Artificial intelligence, which is also the future of all complex decision-making, is the foundation of all computer learning.

Knowledge graphs that practically constitute a graph representation of a semantic network, a network of real-world entities and their interrelations, have a strong connection with the above-mentioned terms and can be considered the future of Artificial Intelligence. Knowledge graphs can be created manually, from semi-structured sources, or from totally unstructured sources of data. The second option tends to be the most popular and easy among the 3. Wikipedia, DBpedia and Yago can be considered sources of semi-structured data that can be used to build a semantic network of entities.

Knowledge graphs have 3 components: the nodes, the edges, and the labels. The nodes are objects or concepts, the edges are the links between the nodes (or we can say the nodes' relations) and the labels are attributes that characterize the relationship of the nodes. The two nodes that are linked together along with the edge that connects them, form what we call a "triple". This triple is translated as "subject-predicate-object", so practically the first entity/node (the subject) is connected with another entity/node (object) via a link (predicate).

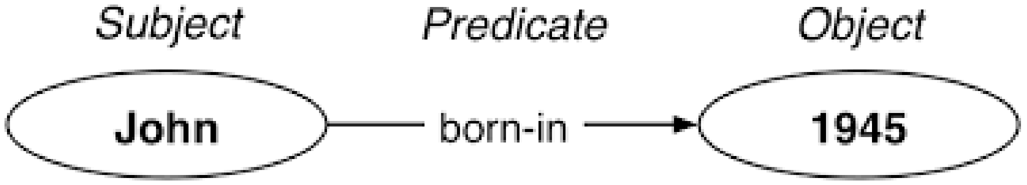


Figure 1 Triple example "Subject – Predicate – Object"

The World Wide Web Consortium (W3C) developed and standardized a Framework that is a way to make statements about entities. It is called RDF and triple models are defined by this method of interconnected web data representation.

Very many computer science subfields intersect with knowledge graphs such as Machine learning – Artificial Intelligence (ML-AI), Natural language processing (NLP), Information retrieval, and data management systems, contributing to technological advancement.

Natural language processing (NLP) techniques can be used to extract information from documents and build a knowledge graph from text.

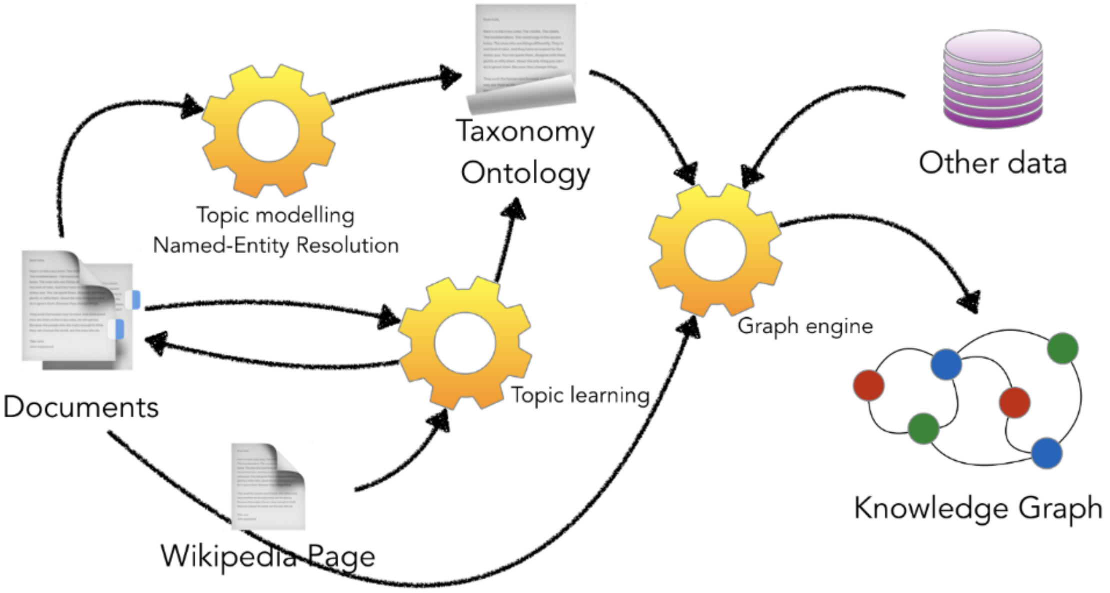


Figure 1 Create knowledge graph from text

After the cleaning and the preprocessing of data, the next step is called "Named entity recognition" and it is the stage where the entities are classified into a set of predefined categories, like names, places, or dates. Entity prediction, entity linking, and relation prediction are the following steps in order to finally achieve information extraction. Moving forward, a set of candidate triples is chosen and after taking into consideration the corresponding domain ontology and the language model, we achieve the creation of the knowledge graph from documents, as shown in Figure 3 below. We go into all these in detail later in this thesis, in chapter 2.6.

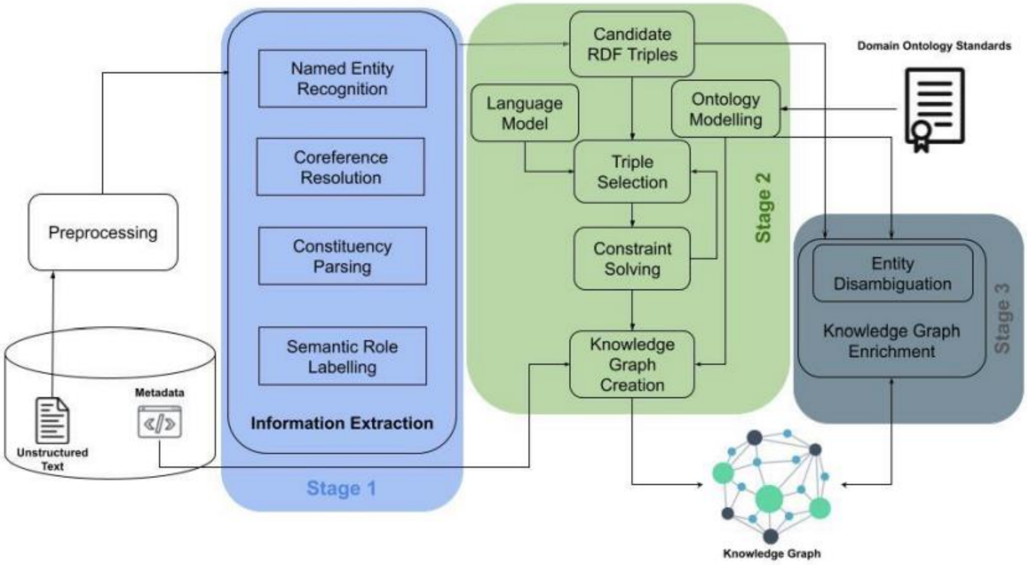


Figure 3 Information Extraction & Domain Ontology

There is no doubt that this technology can benefit the ongoing and emerging pandemic "COVID-19" from various aspects. This very deathly and contagious disease named Coronavirus first appeared in Wuhan, China. It quickly spread over the entire world, becoming a global pandemic caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2). A huge community of researchers not only from the medical field but from the tech field as well is trying to tackle this major issue of humanity with every resource that they can. Part of these joint efforts has also been the knowledge graphs.

Even for professionals, looking through the enormous quantity of scholarly material on the coronavirus topic that already exists, including new research that has surfaced in the wake of the crisis, is a challenging effort. For professionals who need answers to issues that call for bringing together various pieces of information across papers, a simple keyword search over such corpora is not enough. The cutting-edge AI's child known as Knowledge Graph (KG), could be applied to meet such sophisticated information requirements.

A question like "In what types of cells in SARS-CoV-2 receptor ACE2 primarily expressed?" could be expressed in the form of RDF triples as "cells SARS-CoV2 ACE2 express." in order to be interpreted by a search engine appropriately. It is widely known that the answer for the above question is presented in more than one paper available online and that these papers have been processed and indexed by search engines already. Given that, what a search engine can do best to satisfy the user who formed the question, is to present all the relevant pages that contain the answer, so the user can navigate and get the chance to meet what they need. [1]

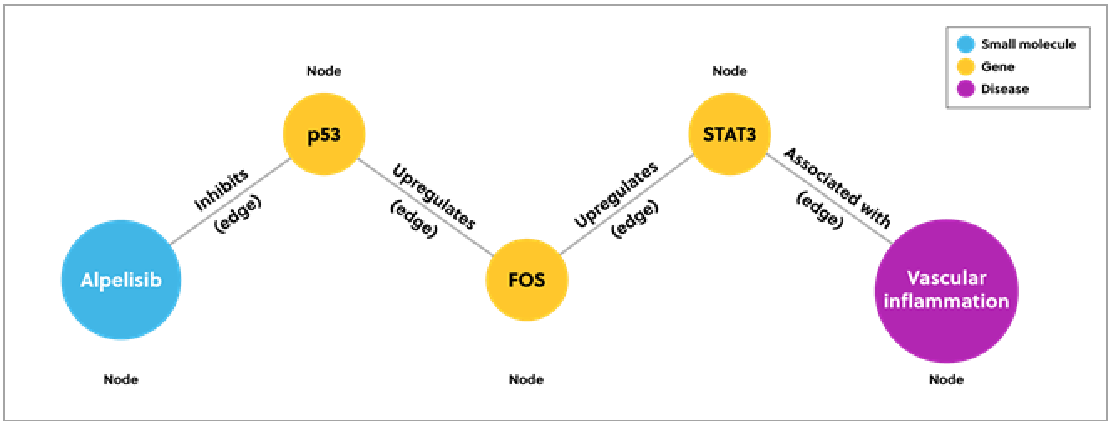


Figure 4 KG Nodes

2 Literature Review

This chapter analyzes and provides an overview of the scientific research made on knowledge graphs and their main relation and applications on various emerging fields. More specifically the aim is to get a better understanding and try to focus on two important questions that concern the scientific area:

1. What are the approaches for creating a knowledge graph from documents?

2. How knowledge graphs with NLP can improve question answering for Covid-19 information?

2.1 Knowledge Graphs

With the rapid technological advancements and the growth of the semantic web the terms knowledge graph and knowledge model are now frequently heard and seen in multiple and interdisciplinary areas of science, research, business and more. More specifically, when Google introduced its Knowledge Graph project in 2012, the key term achieved a lot of popularity while a wide variety of academic papers were published linking the term Knowledge Graph with semantic searches. [2]

However, a Knowledge Graph is more than a semantic search or a graph-based representation of data. There are a lot of definitions given to Knowledge Graphs during the past years, mainly describing them as a large network of real-world entities -or a semantic network- that is represented by a graph, along with their relationships. [3] The information that is stored in a graph database and visualized as a graph is used to integrate knowledge and is considered valuable knowledge in the information technology field. Knowledge graphs have a significant impact on the amelioration of computing methodologies, information extraction, artificial intelligence, NLP, Machine Learning, Knowledge Representation, Data Management systems, Question Answering systems, Search Engines, and many more fields of research of great interest. [4]

What constitutes a knowledge graph are nodes and edges. A node can be an item, a user, or an entity and an edge is the relationship that two nodes may have. [5] The information stored can come from multiple sources where the datasets can contain diverse heterogeneous data in a structured, semi-structured, or unstructured format. Natural Language Processing helps the presented data to become more comprehensive and gives a different meaning to the relationship that two nodes (objects) have. This way, knowledge graphs play a key role in search engines and at the same time they facilitate question answering as they are based on a knowledge database.

Among others, knowledge graphs have facilitated knowledge discovery and discovery in the medical area in general as well. Gathering and managing medical data from heterogeneous sources such as PubMed and others is now an easy task with the use of knowledge graphs. Researchers can save a great amount of time and increase productivity by using them as a tool to obtain insights from related documented work in an automated way by finding multiple relationships between entities, and potential trends and even combining internal data from clinical trials with public open data. [6]

Pharmaceuticals have also benefit by this technology. Ontotext, a company that started as an R&D lab, was chosen by a global pharma company to develop and create a solution system that manages large volumes of medical data and provides accurate and content related answers to a huge number of questions that is needed to be answered in a limited period of time [7]. Ontotext managed to reduce the time spent in analyzing the diverse data from company's archived questions, to give meaning to the data and apparently uncover knowledge that already existed. A big challenge to them, was that the company's stored documents were in a PDF format that is difficult to take advantage of in terms of searching. Additionally, the lack of indexing in documents made the knowledge discovery process almost impossible. The team developed a solution that: [7]

1) Automatically retrieves and categorizes Q&A pairs based on their content from existing archived documents in Pharma's repository.
2) Questions are semantically indexed so they can be used as a reference for new questions to be linked with the existing ones.
3) Knowledge graph is used to present relationships between documents' entities.
4) Uses GraphDB, a semantic graph database, and RDF triples to handle massive amounts of queries in real-time and to match semantically related words.

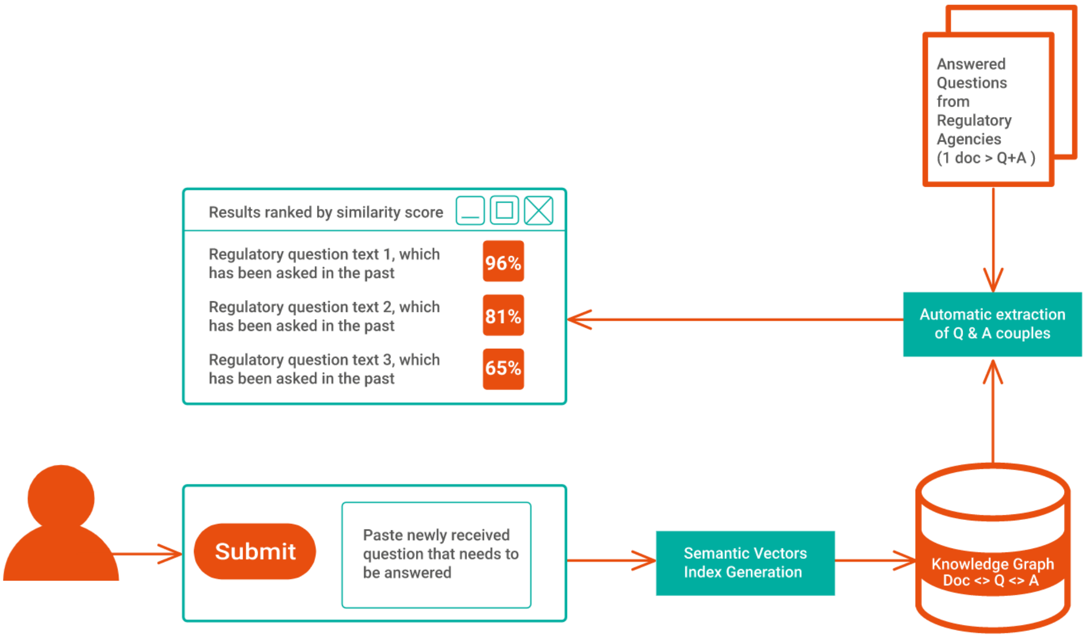

Answered Questions from Regulatory Agencies (1 doc > Q+A)
Results ranked by similarity score
Regulatory question text 1, which has been asked in the past
96%
Regulatory question text 2, which has been asked in the past
81%
Regulatory question text 3, which has been asked in the past
65%
Automatic extraction of Q & A couples
Submit
Paste newly received question that needs to be answered
Semantic Vectors Index Generation
Knowledge Graph Doc <> Q <> A

Figure 2 Ontotext's smart semantic similarity search solution.

2.2 Knowledge Graphs for Question Answering

A question answering system is a system designed to search and provide the most accurate answer to a question set by the user. The answer type and length can vary depending on both the question and the developed system.

Database querying, information retrieval, and knowledge graphs are the fields where a question-answering system can be developed [8]. There is no need any more for the user to fully understand the KG schema, it is just required to pose a natural language question (NLQ) in order to retrieve the requested information. Although, there are challenges as the posed question is in an unstructured form while the given answer is in a structured one. Entity linking, relation linking, and answer retrieval are some of the "child" challenges one may encounter trying to address the gap between the structured and unstructured data, which is the initial challenge, as Mohammad Yani and Adila Alfa Krisnadhi mention at their survey. [9]

To address this issue most approaches, use existing lexicons and templates that help with the different representation of the data. However, they do not focus on the overall understanding of the natural question posed, slowing down the whole process. Most of the time semantic parsing is used in order to understand the question. The natural language question is translated into SPARQL queries, a computational representation. Fabiano Ferreira Luz and Marcelo Finger use vector representations and neural networks and follow an RDF schema in their approach. [10]

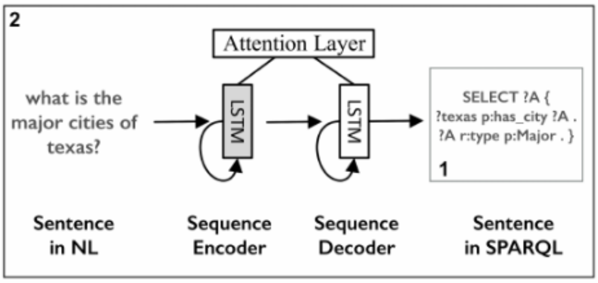

2
Attention Layer
what is the major cities of texas?
LSTM
LSTM
SELECT ?A { ?texas p:has_city ?A . ?A r:type p:Major . }
1
Sentence in NL
Sequence Encoder
Sequence Decoder
Sentence in SPARQL

Figure 3 General overview of the neural network approach

Although, the abovementioned translation can result into polysemy and synonymy that lead to ambiguity. A polysemic term has more than one meaning, which means that an entity that is part of the question can be linked to an entity with the same string but with a different definition. Regarding synonymy, entities analogous to entity mentions do not have the same representations, and this results to inability of mapping the first with the last. Sangjin Shin, Xiongnan Jin, Jooik Jung and Kyong-Ho Lee propose the PCQA, Predicate Constraints based Question Answering. [11] The authors are trying to find the intention of a question using predicate constraints following an RDF schema. Three lexicons were constructed, one for the predicate constraints and the other two containing entity and relation variations. By using relation matching on predicates and due to predicate constraints, that help omit inappropriate candidates, they managed to reduce the complexity of producing a graph. [11]

Saurabh Srivastava, Mayur Patidar, Sudip Chowdhury, Puneet Agarwal, Indrajit Bhattacharya and Gautam Shroff extended the KG related research field by developing the CGA-NMT model which is a multi-task BERT based Neural Machine Translation model oriented to the corporate world having as a scope to address and answer questions related to an enterprise system. [12] The initial query passes through 4 modules before the requested answer is given. Entity detection and path prediction using seq2seq, question type prediction, and answer entity type prediction using classification are the aforementioned modules used. It is shown that in contrast with other approaches, CGA-NMT model scores better results in terms of accuracy due to its ability in handling duplicate KG entities as well as complex and undiscovered topologies. [12]

Another approach that outperforms the state-of-the-art methods is those of Xiao Huang, Jingyuan Zhang, Dingcheng Li and Ping Li. [13] The authors developed the Knowledge Embedding based Question Answering (KEQA) framework which aims to translate in an automatic way the users' unstructured questions (NLQ) into structured answers, bridging the gap between the data volume and complex structures. What makes the KEQA model differ from the rest is that instead of using the head entity and predicate directly it trains both a predicate and a head entity learning model that return back each one's representation in a form of a low dimensional vector that is part of the predicate embedding space and the entity embedding space accordingly. By analyzing the order and the importance of words the output vector is the closest possible to the predicate representation. [13]

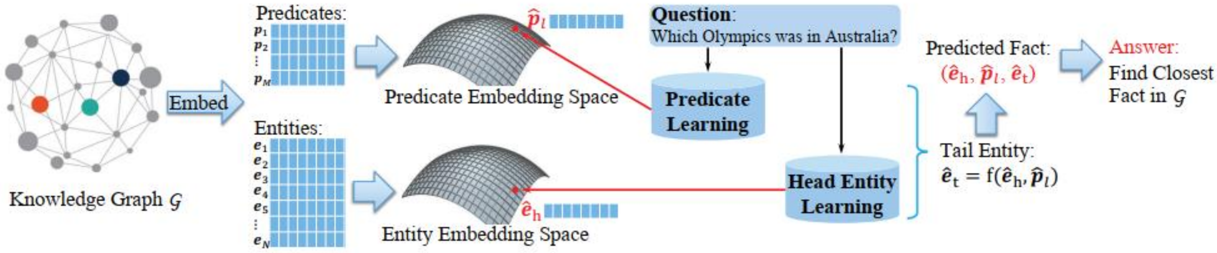


Figure 4 Question answering over knowledge graph (QA-KG)

2.3 Knowledge graphs for search engines

Knowledge Graphs have a crucial impact in search engines' betterment. Complex queries acquire knowledge and semantic meaning. While search engines use the keywords of a given query and quickly identify items and give answers that correspond to those particular keywords to the users, they lack of accuracy, simplicity and explainability when the query is complex which discourages the user, making them lose their trust. [14]

Authors Xuejiao Zhao, Huanhuan Chen, Zhenchang Xing, Chunyan Miao in their recent work, present DeveloperBot, a search engine assistant that is based on a knowledge graph and has the ability to perform like a human brain, responding to complex questions (queries given by a user) in a comprehensive and simple way. [14]

Recent work of Xingping Wu et al., [15] reveals the development of an intelligent search engine (ISE) that is based on knowledge graph and can understand the users' needs and inclinations, providing them with direct and accurate answers. [15]

When a user inputs a random query into a search engine, it is interpreted as a sequence of words with no actual meaning. What a structured knowledge graph has to offer in search engines is to give essence to those queries, making the searching process more targeted and semantic and in the end generating knowledge for the user by retrieving the most accurate answers/results. The keyword queries are transformed into queries with semantic meaning or as Google had firstly said when promoting the Google Knowledge Graph, "Things, not Strings". [16]

Google Search, probably the most popular search engine globally, is relied to knowledge graph to perform the searching, in order to provide to the users, the most accurate results in the shortest possible time. The users now have access to unlimited sources of information and knowledge for free. Google Search's knowledge graph contains around 500 billion facts for 5 billion entities. This facilitates the searching experience and ameliorates user satisfaction. [17]

Besides Google, amazon's chatbot Alexa is powered by a knowledge graph to interact with the user and provide the user question the most accurate answer. Airbnb and eBay are using KG technologies to return to the user the most appropriate results based on the input criteria they have added, making this whole process seem straightforward and very easy. Of course, a lot of companies have embraced knowledge graph technologies such as LinkedIn, Pinterest, IBM, Stardog, and many others.

Stardog identifies 3 steps to use a knowledge graph to build a better search experience.

Step 1: Ask a better question.

Stardog checks to make sure it is asking the appropriate question before looking for an answer. This is an important step because humans frequently do not ask the proper inquiry. In order to better understand the data sources consolidated in the knowledge graph, Stardog rewrites the query depending on the context of related terms. [16]

Stardog's foundation is a semantic graph data structure, which forms a network by connecting relevant pieces of information. Stardog can read real-world meaning and efficiently rewrite queries for improved interpretation thanks to this expressive syntax. Reading the context of the search terms ensures that the search application appropriately reflects user intent. [17]

Step 2: Search across sources.

In this step, the query is reviewed in two stages after being updated to appropriately reflect the user's context. The full-text search is run first. Stop words, case inconsistencies, and fuzzy matches are taken into account, and the pertinent full-text index results are ranked and returned. To make sure that the search covers the full range of pertinent information, Stardog then enables you to compare those results to all other knowledge graph data, including structured, semi-structured, and other unstructured data. To speed up the extraction and labeling of data from unstructured sources like research, rules, and other documents, Stardog collaborates with a large number of specialized NLP partners. [18]

Virtualization is required for scalable search algorithms to function. Virtualization eliminates the need for an intricate and time-consuming ETL system to combine data from dozens or even hundreds of systems and outside suppliers because it directly accesses the source data. [19] [20] Working with copies of data instead of source data lowers data quality and increases the chance of human mistake (data is not current or comprehensive). Instead, virtualization makes sure consumers always have access to the most recent information. [21] [22]

Step 3: Refine results.

Stardog uses business logic at this step to refine results. At the moment of a query, Stardog applies business logic to the underlying data in order to give outcomes that are contextually optimized. [23] This business logic is centralized in a data model. The low-code knowledge graph centrally stores business logic as opposed to building it on top of a database that has this data. All these are taken care of Stardog, that wants to make sure that there is no need for the user to scan through the results and this resumes Stardog’s approach to a Semantic Search application with the use of Knowledge Graphs. [24]

2.4 Knowledge Graphs for Covid-19

In December 2019 a contagious disease named Coronavirus (COVID-19) appeared in Wuhan, China. The disease soon enough was spread all over the world turning into a pandemic. It is caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) and its symptoms include cough, fever, loss of smell and taste, myalgia and others. [18] The fact that the virus can be very easily and quickly transmitted via the respiratory system as well as the severity of the disease that can cause hospitalization and death made it the most important subject of research not only for the pharmaceutical industry, that are focusing on developing a vaccine safe and effective and the rest biomedical community but for a variety of professionals that can contribute to the battle against it as well.

Living under the ongoing COVID-19 pandemic an emerging need of linking separate pieces of academic literature with information about this evolving crisis is born in the scientific area. Researchers focusing on this previously unknown phenomenon try to integrate biomedical data in various research applications having a general scope to broaden the field of knowledge and contribute to society. [19]

Knowledge graphs that analyze and connect scientific data, play a significant role in the combat against COVID-19, as they contribute to the scientific area as well as to the academic research by providing meaningful insights for the crisis. Artificial intelligence and natural language processing serve to help gain knowledge by giving answers to complex questions where billions of unstructured data need to be processed and analyzed. The aforementioned KGs consist of the entities, the main objects of interest that mostly are academic papers and journals, the attributes that represent the attributes of the entities (such as the author or the title of the paper) and the relations between the entities or the attributes of the entities. [20]

Mayank Kejriwal lists quality, scale, development of frontend tools and visualizations, and social challenges as potential challenges that the creation of a COVID-19 KG may encounter. [21] It is very likely not relevant or reliable and untrustworthy data may be used to develop and train algorithms and models which will later be used for COVID-19-related knowledge graphs which will end up giving equally untrustworthy outputs, having a huge negative impact on various fields (research, science, information). Also, COVID-19 datasets could not be characterized as ‘Big Data’ as the volume of collected data isn’t that big yet. [22] There has been more research on the construction of a knowledge graphs than on its visualization, which is something that would definitely strengthen the user experience. People having a more conservative mindset may be afraid of potential loss of privacy because of automation which is a crucial challenge for the trust to AI and the technological advancements. [16]

Scientists, researchers, and academics show great interest in CORD-19, an open COVID-19 research dataset that counts more than 50,000 scientific articles not only for the current ongoing pandemic but also for older coronaviruses. [23]

The medical society needs new tools and technologies to strengthen their work, extend their field of knowledge and gain meaningful insights about whether a treatment or an experiment is accurate and relevant or not. In addition, there is a tremendous need for developing new and efficient drugs that would receive FDA approval as a COVID-19 treatment. Many researchers are focusing on that purpose and use Knowledge Graphs for retrieving biomedical knowledge.

Jacob Al-Saleem, Roger Granet, et al. have developed the CAS Biomedical Knowledge Graph that identifies and analyzes small molecules that are of great clinical interest for targeting biological processes involved in COVID-19. [24] The fact that knowledge graphs can combine data from multiple heterogeneous sources and present them in a user-friendly environment contributes to the exploration and management of different fields of interest in an easy and timely manner. Having analyzed human diseases, proteins, virus, and other COVID-19 data the CAS Biomedical KG detects which small molecules are better candidates to be used for the development of COVID-19 drugs. [24] The model was designed to prioritize those molecules that cause the least side effects and target proteins that are only found at COVID-19 cases. The algorithm's validity and success could be proven by the fact that 11 out of 50 small molecules that are most likely to become drug repurposing candidates, are used in clinical trials related to the pandemic. [24]

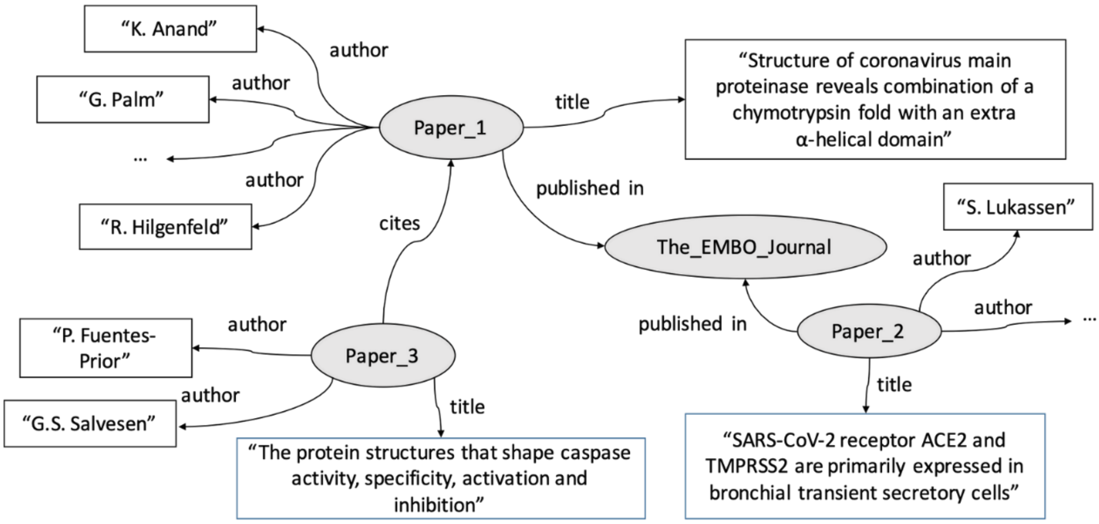

"K. Anand"
author
"G. Palm"
author
...
author
"R. Hilgenfeld"
Paper_1
title
"Structure of coronavirus main proteinase reveals combination of a chymotrypsin fold with an extra α-helical domain"
published in
cites
The_EMBO_Journal
"S. Lukassen"
author
author
...
published in
Paper_2
title
"SARS-CoV-2 receptor ACE2 and TMPRSS2 are primarily expressed in bronchial transient secretory cells"
"P. Fuentes-Prior"
author
Paper_3
author
"G.S. Salvesen"
title
"The protein structures that shape caspase activity, specificity, activation and inhibition"

Figure 5 A knowledge graph (KG) fragment

The construction of this knowledge graph relies in three different types of objects, which are the entities, the attributes, and their relations. We can observe the entities inside the oval nodes; in this case the entities are papers and journals. The entities are the primary object of interest in the knowledge graph. Following, we have the attributes represented by rectangles that can be considered as the attributes of the entities. Lastly, we have the relations between the entities and the attributes, depicted as arrows that connect the abovementioned two. The arrows have both a label that characterizes them and a direction from the entity to the attribute or from one entity to another. [16]

There is structure, rules and constraints behind the knowledge graph's construction that leads to powerful interconnections between its elements. E.g., a paper has an author while a journal has an editor, a fact that seems rational to a human brain, however it cannot be taken as granted in KG's construction process. [25]

2.5 Knowledge Graphs and Ontologies for medical domain

The rise and the advancement of Artificial Intelligence nowadays dictates a better representation of knowledge gained among various fields of interest and research which leads to the inspection of the better use of ontologies. The term ontology refers to the representation of a set of attributes sharing the same characteristics and relationships or could be the representation of the relationships between those attributes themselves. Ontologies are mainly used to represent knowledge about an abstract, conceptual term of interest that we seek to analyze. [26]

In order to surpass the various challenges a researcher may address trying to facilitate the "searching" methods of information management systems that are composed of immense amount of data and information, the use of ontology-based approaches is mandatory. A semantic representation of data can be combined with the corresponding domain knowledge. With the comprehension of the different ontologies used, search engines can define and provide more accurate links and results that are related to a given query by a user. Thus, we can understand that data searching strategies can be built thanks to ontologies. [27]

As one may observe, knowledge graphs and ontologies share many similarities, as they both serve for representing knowledge about a specific domain of interest and its entities and the relationship between those entities. The combination of those two can both facilitate the representation of complex relationships among entities in a way that a user can easily understand and empower the share of knowledge by extending the information stored in a KG to other KGs with the use of mappings. [28]

Ontologies related to the pharmaceutical and medical area play a key role to the knowledge management by all the related stakeholders. The need for developing a system that gives essence to the data and classify them in order to model a domain of knowledge and contribute to the share of knowledge is beyond question among the biomedical community. Therefore, the biggest challenge for the health care community in the construction of a knowledge graph using medical ontologies would be the special expertise on the context and the time consume on labeling biomedical domain specific entities. [28]

Jianbo Yuan, Zhiwei Jin, Han Guo, et al., describe their approach to the construction of such a knowledge graph using medical ontologies. [29] Their framework relies on minimum supervision and uses unstructured scientific biomedical literature as input. The authors recognize the inefficiency of NLP techniques without the amplification of external domain-specific resources. Focusing on entity recognition and entity and relation embedding, the authors based on convolutional neural network and multi-instance learning have achieved a 95% score in terms of accuracy and efficiency. The entity recognition, the entity and relation embedding, the relation generation on clusters according to relation embedding and the relation refinement were all done automatically. The human intervention was needed to assign the semantical relations to the various clusters. [29]

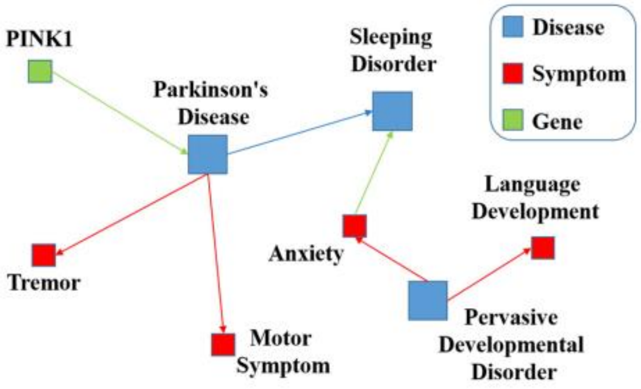


Figure 9 An example of the constructed biomedical knowledge graph

2.6 Knowledge Graphs creation from text

It is beyond question that building a knowledge graph manually can be demanding in terms of going through all the available documents to identify the entities along with their relationships. This is where we can leverage and deploy machines to do the work for us, as they are more capable of handling this activity easily, rapidly and accurately. However, even for machines there are challenges, since they are not able to "understand" the human language. Natural language processing (NLP) enters the scene in this situation. [30]

Making our machine comprehend the human language (natural language) is crucial if we want to create a knowledge graph from the text. NLP methods including sentence segmentation, parts of speech tagging, and entity recognition can be used to do this.

As we have seen several times in this paper, knowledge graphs are the structured representation of an unstructured text or document. It is a set of triples extracted from a text that hold entities and the relationships between them. To reach the structured format/ expression of data extracted from the text, the below steps must be followed. [30]

1) Sentence Segmentation:

The preliminary step to start building a Knowledge Graph from text, is to split this text into sentences. These sentences then, will be filtered so we can keep only those with 1 subject and 1 object storing them in a list. For example, the below text will be split into 4 sentences, which will be filtered afterwards so we can only keep the ones that meet the abovementioned criteria.

"Indian tennis player Sumit Nagal moved up six places from 135 to a career-best 129 in the latest men's singles ranking. The 22-year-old recently won the ATP Challenger tournament. He made his Grand Slam debut against Federer in the 2019 US Open. Nagal won the first set."

Sentence 1: Indian tennis player Sumit Nagal moved up six places from 135 to a career-best 129 in the latest men's singles ranking

Sentence 2: The 22-year-old recently won the ATP Challenger tournament

Sentence 3: He made his Grand Slam debut against Federer in the 2019 US Open

Sentence 4: Nagal won the first set

We will only keep Sentence 2 and Sentence 4 since they are the ones that contain only 1 subject and 1 object.

Sentence	**Subject**	**Object**
2: The 22-year-old recently won the ATP Challenger tournament	22-year-old	ATP Challenger tournament
4: Nagal won the first set	Nagal	first set

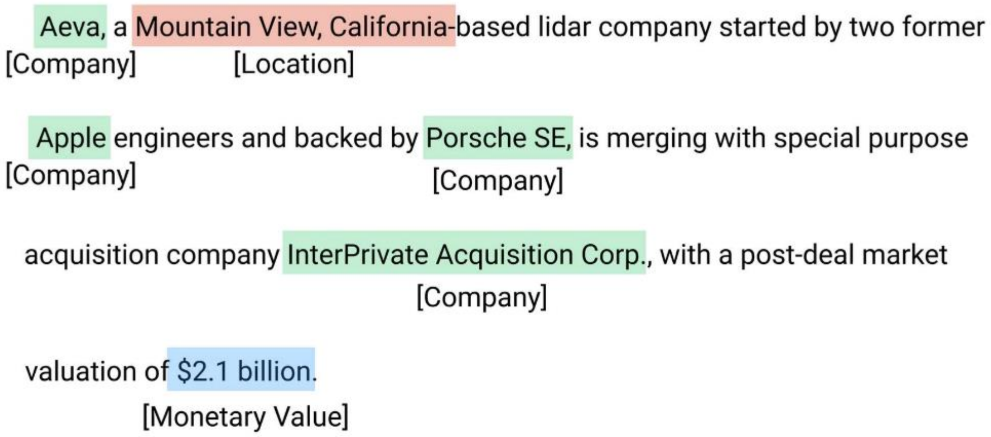
2) Named Entity Recognition:

The term Named Entity Recognition (NER) given by Behrang Mohit [31] addresses the process of firstly detecting proper nouns of a certain topic into a text and secondly classifying them into a set of predefined categories, like names, places, dates, and others. NER plays a key role on the betterment of many language processing tasks, such as Question Answering, Information Extraction, Search Engines and Machine Translation among others. Alireza Mansouri, Lilly Suriani Affendey and Ali Mamat at Named Entity Recognition Approaches, [32] list Rule-based NER, Machine Learning-based NER and Hybrid NER, as different NER approaches for mining text. [33] The first one relies on a set of human-made rules regarding the grammar, the syntax and the orthography and the use of dictionaries. The second one is more probabilistic approach as it occupies classification statistical models and machine learning algorithms, both supervised and unsupervised. The last one uses the combination of the abovementioned and develops new methods using the assets from each method. [34]

3) Entity Extraction:

Entity extraction is a text analysis method that employs Natural Language Processing (NLP) to extract particular information from unstructured text and categorize it**.** [30]

These are named entities, which are the words or phrases that serve as noun representations. Proper names as well as numerical expressions of time or quantity, such as phone numbers, monetary amounts, or dates, are included in this. [35]



Aeva, a Mountain View, California-based lidar company started by two former Apple engineers and backed by Porsche SE, is merging with special purpose acquisition company InterPrivate Acquisition Corp., with a post-deal market valuation of $2.1 billion.

Figure 6 Example of entities

This technique can help to mine massive amounts of unstructured text data for valuable information and structured knowledge. It would take numerous hours of manual labor to sort through hundreds of documents, but with automated entity extraction, one may quickly obtain the information they require. [35]

If we use the “Sentence 2” of the example mentioned in “Sentence Segmentation” above, and deploy a simple Python code using the spaCy library, we will extract the below dependency tags:

The … det

22-year … amod

– … punct

old … nsubj

recently … advmod

won … ROOT

ATP … compound

Challenger … compound

tournament … dobj

. … punct

We can observe that instead of having “the 22-year-old” as the subject of the sentence, we just have the word “old”. The same happens with the object, we wanted the “ATP Challenger tournament” to be considered as the object of the sentence, but instead we only got the word “tournament”.

To address these issues and get the desired subject and object we must apply some rules. In the first case of the subject, we would have to take into consideration the modifier “22-year” and the punctuation mark “-” along with the subject “old”. In the second case, similarly, we would have to take into consideration the compound words “ATP” and “Challengers” along with the object “tournament” to take the desired result. [30]

What was done above is called “dependency parsing” and it is used for entity extraction.

The automatic extraction of information can be done in a variety of ways. Let's examine each one in turn:

i. Rule-based Approach: First we must construct a set of grammatical rules about the syntax and then to apply them to our text so we can achieve information extraction.

ii. Supervised: Given a particular and concrete sentence S, that contains two entities E1 and E2, we deploy a supervised machine learning model to examine whether the 2 entities E1 and E2 have some kind of relation (R). This indicates that using this approach the task of relation extraction now becomes the task of relation detection (we will analyze this term later in this section). The only drawback of the method is that it needs a large amount of labeled data to train a model.

iii. Semi-supervised: In the absence of sufficient labeled data, we can create high-precision patterns from a group of seed examples (triples) that can be applied to extract more relations from the text. [30]

4) Entity Linking:

Entity linking is considered the process where entities mentioned in documents are automatically correlated with entities in a knowledge base. Of course, there are limitations and challenges due to entity ambiguity. Entities with the same name are used for totally different cases having another meaning. This makes the entity linking task more difficult to interpret. [36]

5) Relation Extraction:

Except of the entity extraction the next very important step to build a knowledge graph is the relation extraction. We have mentioned many times already that a knowledge graph consists of nodes and edges. With entity extraction we can form the nodes, but we still miss the edges. The task of defining the relationship that 2 nodes of a KG have, is called relation extraction, and this will be represented by the edges that link the nodes in a KG. [30]

Let’s use again the example with the below sentences:

Sentence
2: The 22-year-old recently won the ATP Challenger tournament
4: Nagal won the first set

As we saw earlier for the first sentence the “22-year-old” is the subject, while the “ATP Challenger tournament” is the object. Same way, by using the spaCy library for the second sentence, we can easily extract the subject “Nagal” and the object “first set”.

Nagal … nsubj

won … ROOT

the … det

first … amod

set … dobj

. … punct

One can easily observe that the 2 sentences have the same relation, the verb “won”. To extract this relation, we simply have to search to our code for the “ROOT” of the sentence, which in both cases is the verb “won”.

By using entity and relation extraction we can now build our knowledge graph like this:

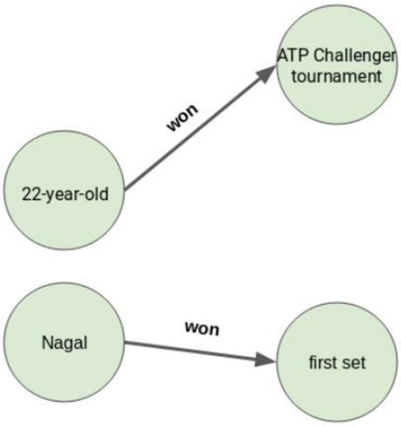


Figure 7 Example of entities relation

The nodes are formed by our entities, the subjects, and the objects of the two sentences and the edges of the graph are formed by the relation of the subjects with the objects. [30]

6) Entity Prediction:

A further step to entity extraction and relation extraction is the entity prediction. Given the fact that a knowledge graph is basically a set of entities and relationships between them presented as RDF triples, let's suppose (h, r, t,) where *h* is the head entity, *t* the tail entity and *r* the relationship these 2 have, the task of entity prediction is about predicting the entity "*t*" if "*h"* and "*r"* are known. [37]

7) Relation prediction:

This task usually called as Knowledge Graph Completion (KGC) serves for predicting the missing relationships among entities of a knowledge graph. This is a very crucial task since it is a common incident that knowledge graphs could be incomplete or noisy. As hundreds of new entities and relationships are being added every minute in real-worlds KGs, traditional methods fail on predicting the probability of relationships among entities, as they are trained to make predictions based on existing and well-linked entities and current KG's topology. The entities have their own separate embedding that learns from the closest connected neighbors. Newest approaches like open-world KGC models aim to find missing. [38]

A relationship-dependent framework is created by fusing entity and word embeddings. A group of researchers Baoxu Shi and Tim Weninger, developed the ConMask model, following this framework. They used mapping functions that return word vectors for each relationship and functions that calculate the weights of words, assigning higher and lower weights to words depending on whether they appear in the relationship or not, outperforming this way other KGC models. [38]

8) Open information extraction:

Information extraction -IE- systems take as input natural language text in unstructured format and return a structured representation of it in the format of RDF triples (2 entities and their relation). This task requires a lot of manual effort and user participation, from the subtask of naming the target relations to the subtask of finding new extraction rules and patterns. Open IE, in opposed to traditional methods, tends to reduce that manual effort. It analyzes a given text and extracts all the relations found between text's entities. There are 3 challenges for open IE systems, which are the automation, the corpus heterogeneity, and the efficiency. [39]

3 Materials and Methods

This chapter analyzes the methodologies used on this thesis for creating a knowledge graph from text, the dataset used, as well as the haystack framework to construct a question-answering system.

3.1 Introduction

As the ongoing pandemic of Covid-19 is one of the most emerging items that concern the society and the medical scientific area, a large volume of data is available, making it difficult for researchers to go through all the published papers each time a query comes up. This method simplifies a lot all this process. Whenever the user provides a query, the model goes through the repository of all the papers, finds and returns back to the user the related ones along with the paragraphs where the relevant answers to those queries can be found.

This model uses the combination of Knowledge graphs with MeSH ontology and Bert embeddings. The graph enabled search helps the user search based on very specific medical terminology. It can also be extended to give inference on how different documents are related to each other (based on author, body, title, etc.)

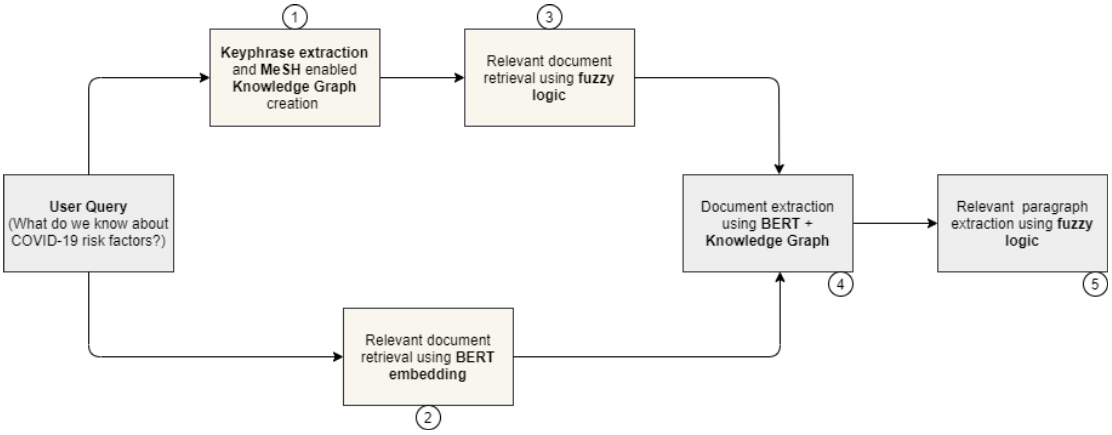

1
Keyphrase extraction and MeSH enabled Knowledge Graph creation
3
Relevant document retrieval using fuzzy logic
User Query (What do we know about COVID-19 risk factors?)
Document extraction using BERT + Knowledge Graph
4
Relevant paragraph extraction using fuzzy logic
5
Relevant document retrieval using BERT embedding
2

Figure 8 High Level Flow- Diagram

The National Library of Medicine created the regulated, hierarchically ordered Medical Subject Headings (MeSH) thesaurus. It is used to index, classify, and search material pertaining to biomedicine and healthcare.

Bi-directional Encoder Representations from Transformers, or BERT, is what it stands for. A group at Google AI Language released it in 2018. The BERT model is created by piling several encoders from the transformer design on top of one another, as suggested by its name. By concurrently conditioning on both left and right context in all layers, the BERT architecture is intended to pretrain deep bidirectional representations from the unlabeled text.

Because of this, the pre-trained BERT model may be improved with just one more output layer, leading to state-of-the-art models for a variety of tasks, including question answering and language inference, without requiring significant task-specific architecture changes.

BERT uses three different embeddings to create its input embeddings. The process of combining the embeddings to create the final input token is depicted in the diagram below.

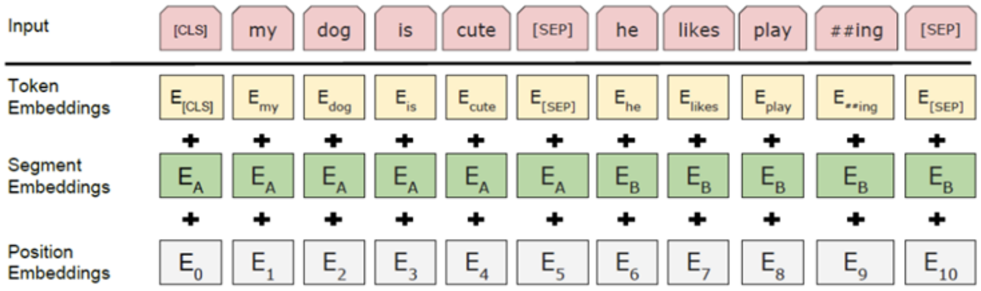

Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} E_{**ing} $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}

Figure 9 BERT Input representation

3.2 Dataset

The dataset that was used for this thesis is CORD -19, the Covid-19 Open Research Dataset. The Allen Institute for AI released the first version of CORD-19 on March 16 of 2020 and it was consisted of approximately 28K structured full text papers related to Covid-19, SARS and MERS. CORD-19 was updating each week, while the latest version was released on June 2 of 2022, having a significant growth that reaches over 1M papers. It is an open-source scientific dataset, designed to smooth the data and text mining process and to contribute to the most emerging area of scientific research of the past 2 years, Covid-19 pandemic. A subset of 10.000 documents of CORD-19 was used due to computational limitations. [23]

3.3 Methodologies

This chapter analyzes the methodologies used both to the experimentation of creating a knowledge graph from JSON text documents and for the creation of a question answering system using Haystack framework.

3.3.1 Entities and Relations extraction

The preliminary step was to extract entities from JSON documents with the use of Spacy library as well as, DBPedia and SciSpacy models. To achieve this task, the acquisition and the cleaning of the raw data were essential in order to proceed with the design of the domain ontology, having as final purpose to focus on COVID-19 related entities.

First, JSON objects were created for the paper id, the title, the authors, the abstract, the references and the body of each document. We wanted to take into consideration only papers in English language that had both abstract and title. To achieve this, 2 filters were applied. The first one was detecting the paper's language and was keeping only the ones written in English. The second one was passing through all the JSON files and was omitting those that were missing either the title or the abstract.

After cleaning our input data, we performed entity extraction, by using both DBPedia and Scispacy BioMedical Model. We stored our entities in 2 different lists, one for each source.

DBPedia is a knowledge base that extracts knowledge from the 111 different language versions of Wikipedia, in a structured format. With DBpedia one can semantically query relationships, characteristics, and references to other datasets that are associated inside Wikipedia resources. The "DB" stands for database. [40]

SciSpacy is a python package which is related to Spacy and is mostly used for biomedical, clinical, and scientific data processing. SciSpacy facilitates entity extraction, entity linking and Named Entity Recognition (NER). More specifically en_ner_bionlp13cg_md model was deployed, a spacy NER model trained on the biomedical corpus. This model uses biomedical POS (part-of-speech) taggers which perfectly fits our case. [41]

The next step was to find the entities' relationships and create the triples using both these entities and their relationships. The entities were mapped to their respective paper ids with the following relations:

has_title: to extract this relation, our script was taking every item (word) from our object title and was searching for it against the list of entities that were created for each one of these sources in previous steps. In addition, Spacy library was used to extract the nouns and pronouns of each title.

Final triple schema: [item, 'has_title', paperid]

has_author: in this relation, the script was extracting the author's first and last name, combining them into one single value, called 'name'.

Final triple schema: [name, 'has_author', paperid]

has_abstract: same logic as to 'has_title' was followed. Nouns and pronouns were extracted using the pre-defined Spacy's function for entity extraction. In addition, each word inside the abstract was searched against DBPedia's and SciSpacy's list of entities.

Final triple schema: [item, 'has_abstract', paperid]

has_body: same logic as to 'has_title' and 'has_abstract' was applied for this relation. DBPedia, SciSpacy and Spacy were used to obtain the entities.

Final triple schema: [item, 'has_body', paperid]

has_reference: again, as to 'has_title', 'has_abstract' and 'has_body', we used the exact same way to create this relation with its corresponding entities.

Final triple schema: [item, 'has_reference', paperid]

After extracting both the entities and their relations, the entities were cleaned using the common words corpus, so the most common words to be removed and the least frequent to be kept. This way the "medical domain" specificity of the model was preserved.

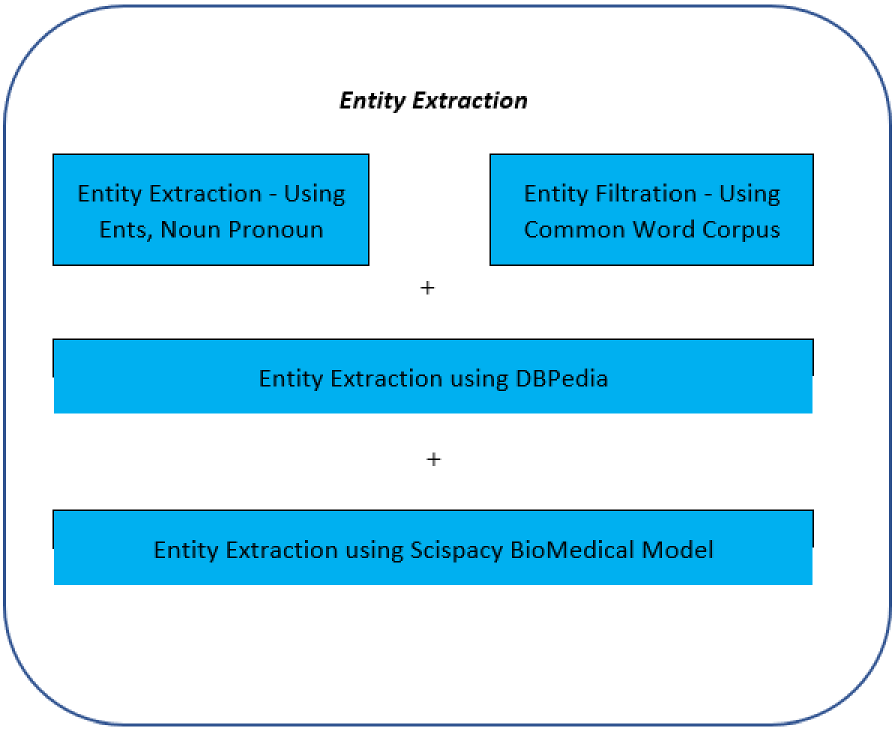

Entity Extraction
Entity Extraction - Using Ents, Noun Pronoun
Entity Filtration - Using Common Word Corpus
+
Entity Extraction using DBPedia
+
Entity Extraction using Scispacy BioMedical Model

Figure 10 Entity Extraction

3.3.2 MeSH enabled Knowledge Graph creation

In the second part of the developed model, MeSH ontology was used for entity extraction. In more detail, the triples were expanded. For each entity in every triple, the top 5 related similar entities were searched and defined from the MeSH ontology and were then added to the triple. Apart from these 5 extra entities, the relations existing between them were also added to the triple. This way the knowledge graph not only did include entities that were referenced in the documents but also included related words and terms that are furthermore domain specific.

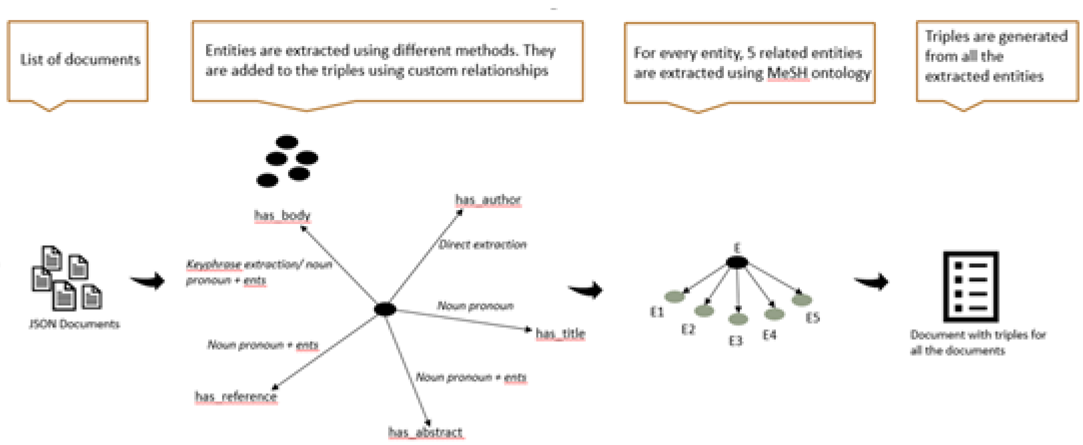

List of documents
Entities are extracted using different methods. They are added to the triples using custom relationships
For every entity, 5 related entities are extracted using MeSH ontology
Triples are generated from all the extracted entities
JSON Documents
has_body
has_author
Direct extraction
Keyphrase extraction/ noun pronoun + ents
Noun pronoun
has_title
Noun pronoun + ents
has_reference
Noun pronoun + ents
has_abstract
E
E1
E2
E3
E4
E5
Document with triples for all the documents

Figure 11 MESH Enabled Triple Extraction for KG Creation

There is no doubt that these preliminary steps were challenging. Despite the open data that was accessible, its huge volume made it difficult to process. In addition, building a domain ontology from scratch can be demanding, as one can meet multiple and diverse types of entities and constraints.

3.3.3 BERT Model

In the third part of the code, BERT model was used as a pre-trained set of text encoders. We were unable to run Bert as a service, so we had to take the second option, which is the pretrained model.

First, we installed pytorch interface for BERT, by Hugging face. This is a library that achieves a good mix between high-level APIs and tensorflow code. Then we used transformers library to import BertTokenizer and BertModel for token and text classification having as an ultimate purpose to extract BERT embeddings.

The titles and the abstract of the JSON articles were used as an input for this task. BERT's pretrained model is using two special tokens *[SEP]* and *[CLS]* to "understand" the separation of the sentences.

For example, if the input text was this:

"*To Get Vaccinated, or Not to Get Vaccinated, That Is the Question: Illness Representations about COVID-19 and Perceptions about COVID-19 Vaccination as Predictors of COVID-19 Vaccination Willingness among Young Adults in The Netherlands"*

After the tokenization would be turned into this:

[CLS] To Get Vaccinated, or Not to Get Vaccinated, That Is the Question: Illness Representations about COVID-19 and Perceptions about COVID-19 Vaccination as Pre-

dictors of COVID-19 Vaccination Willingness among Young Adults in The Netherlands [SEP]

But if the text was more than one sentence, it would have been something like the following:

[CLS] To evaluate the association between ventilator type and hospital mortality in patients with acute respiratory distress syndrome (ARDS) related to COVID-19 (SARS-CoV2 infection), a single-center prospective observational study in France. [SEP] [CLS] We prospectively included consecutive adults admitted to the intensive care unit (ICU) of a university affiliated tertiary hospital for ARDS related to proven COVID-19, between March 2020 and July 2021. [SEP] [CLS] All patients were intubated. [SEP] [CLS] We compared two patient groups defined by whether an ICU ventilator or a less sophisticated ventilator such as a sophisticated turbine-based transport ventilator was used. [SEP]

After separating the sentences, BERT tokenizer was used. Before tokenizing a word, the tokenizer verifies that the entire term is present in the vocabulary. If not, it attempts to split the word into the most extensive sub words in its lexicon before, as a final resort, breaking it down into individual characters. After tokenizing the text, the next step was to create a list of vocabulary indexes through a list of tokenized strings.

By completing this step, our text was finally ready to extract the embeddings. For this task, pytorch was used to convert python lists into torch tensors. Then, *bert-base-uncased* model was loaded, a deep neural network with 12 layers and the word embeddings were extracted.

We applied the *bert-base-uncased* model also to an input query:

```
query="What do we know about COVID-19 risk factors?"
```

The next step was to apply cosine similarity on the embeddings of the input query and the documents to get the relevant documents in order. Unfortunately, this step wasn’t completed with success and due to limitation of time, this is where the experimental part of this dissertation stopped.

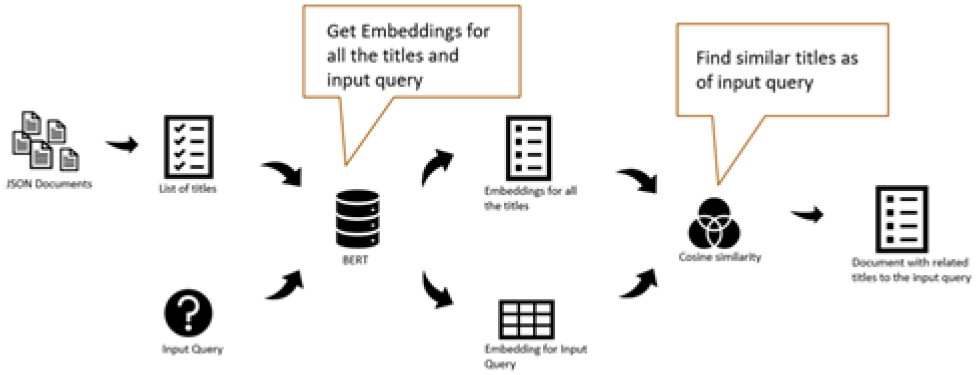

Get Embeddings for all the titles and input query
Find similar titles as of input query
JSON Documents
List of titles
Input Query
BERT
Embeddings for all the titles
Embedding for Input Query
Cosine similarity
Document with related titles to the input query

Figure 12 Bert Training

3.3.4 Next Steps

What was planned to do next in this thesis was paragraph extraction using the created Knowledge Graph in step 3.3.1 and BERT model in step 3.3.2. The vision was to be able to query the knowledge graph or in simple words to form some questions and get as a result the most relevant paragraphs from the used documentation.

The higher plan was to export the entities from the 3 following queries:

1. How many nucleotides are there in SARS-CoV2 RNA?
2. When to vaccinate kids?
3. What do we know about COVID-19 risk factors?"

Having our triples already created from step 3.3.1, we would just have to search for the entities of the abovementioned input queries against our triples, define the most relevant papers (since their ids are part of the triples) and then get the most relevant paragraphs that answer these questions. To do this, we planned to use Fuzzy Matching.

Fuzzy matching (FM), sometimes referred to as fuzzy logic, approximate string matching, fuzzy name matching, or fuzzy string matching, is a machine learning and artificial intelligence method that locates components in data table sets that are similar but not identical. Fuzzy matching algorithms attempt to measure how closely two strings are related to one another as opposed to simply looking at equivalence between two strings to see if they are the same.

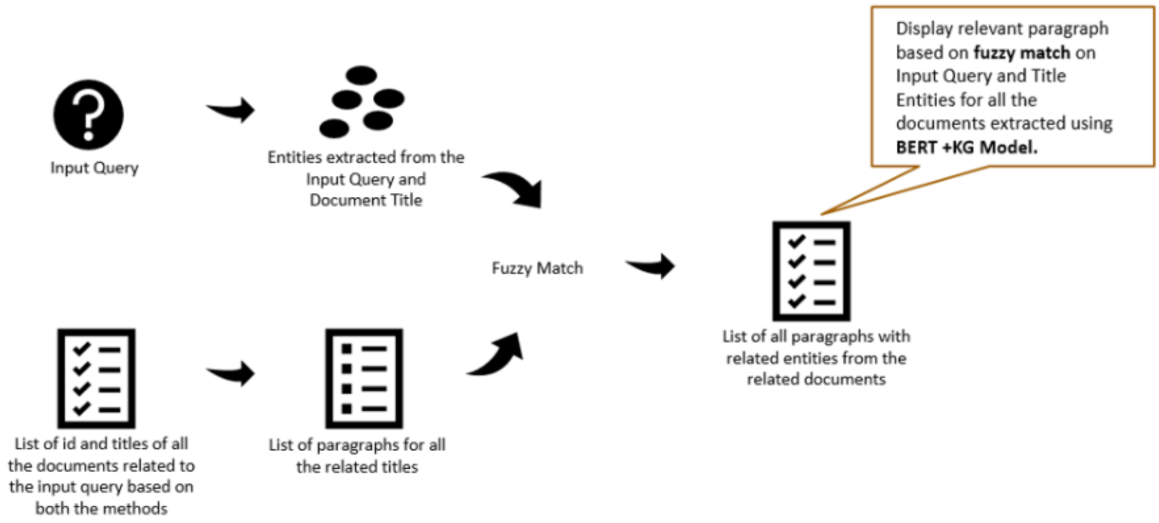

Input Query
Entities extracted from the Input Query and Document Title
Display relevant paragraph based on **fuzzy match** on Input Query and Title Entities for all the documents extracted using **BERT +KG Model.**
Fuzzy Match
List of all paragraphs with related entities from the related documents
List of id and titles of all the documents related to the input query based on both the methods
List of paragraphs for all the related titles

Figure 13 Get the list of all paragraphs

After getting all the relevant paragraphs, our plan was to visualize the graph using Neo4j, an open-source, NoSQL, native graph database. However, Neo4j cannot understand natural language, so one can communicate with Neo4j by using the query language “Cypher”. Cypher is strong and adaptable, and it can be used for a variety of tasks, such as:

- Extract the paragraphs
- Produce data tables from the graph's entities and relations.
- Run the graph via algorithms like PageRank and Shortest Path to count, filter, and aggregate data.

Cypher recognizes and benefits from data relationships. In order to uncover previously undiscovered linkages and clusters, it follows connections in any direction. Compared to complex SQL joins, cypher queries are much simpler to write. This Cypher query's SQL equivalent can be compared.

The syntax used by Cypher is of the ASCII-art variety, with (nodes)-[:ARE CONNECTED TO]->(otherNodes) employing rounded brackets for circular (nodes) and -[:ARROWS]-> for relationships. One can create a graph pattern over their data when they write a query.

Let’s take the example below:

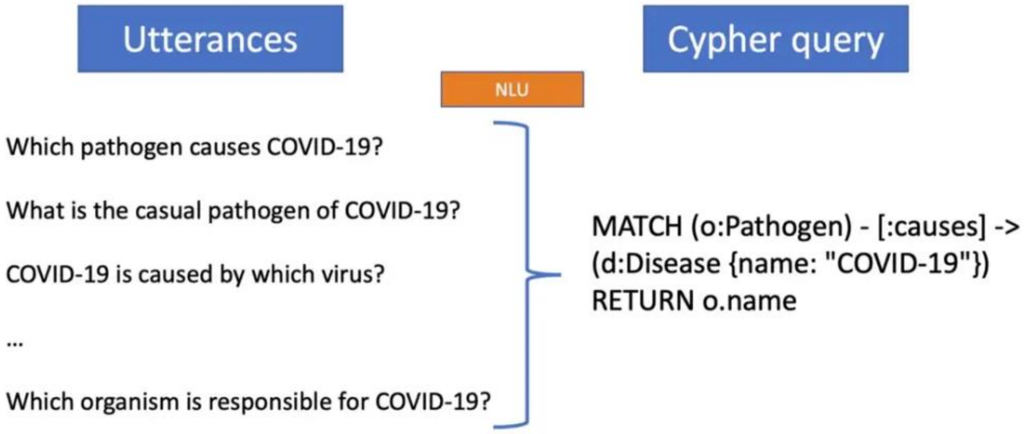

Utterances
Cypher query
NLU
Which pathogen causes COVID-19?
What is the casual pathogen of COVID-19?
COVID-19 is caused by which virus?
...
Which organism is responsible for COVID-19?
MATCH (o:Pathogen) - [:causes] ->
(d:Disease {name: "COVID-19"})
RETURN o.name

Figure 14 Example of Cypher query

Here we can see 2 nodes with the labels “Pathogen” and “Disease” that are connected with the relationship “causes”. We can observe the letters “o” and “d” next to the labels “Pathogen” and “Disease” respectively. This is because all the nodes of type “Pathogen” (labeled as Pathogen) are designed to be accesses with the letter “o” and similarly, all the nodes of type “Disease” (labeled as Disease) are designed to be accessed with the letter “d”. Inside the Disease, we can see the term {name: “COVID-19”}. This means that one of the elements this node contains, has the name COVID-19. The direction of the arrow shows us that the Disease is derived from the Pathogen. Using the word match, the query is trying to allocate the COVID-19 element inside the Disease node and to find its relation to an element inside the node “Pathogen”. Then, with the phrase “return o.name” it will finally get the name of this related element, meaning the name of the pathogen that causes the disease of COVID-19.

A similar framework is meant to be used in our knowledge graph, to find and visualize the connections between the input queries mentioned above and paragraphs from our documents. Unfortunately, due to limitations of time and lack of Cypher language knowledge this task was not performed.

3.3.5 Haystack

In addition to our effort to construct and query a Knowledge Graph, we also developed a question-answering model using Haystack. Haystack is an open-source framework that can be used to create search engines that can handle massive document collections with intelligence. The application of question answering, retrieval, and summarization to real-world contexts has been made possible by recent developments in NLP, and Haystack is intended to be the link between academia and business. It uses NLP to select components that can be used for question-answering and all transformed-based models, including BERT. It is scalable since it can manage millions of documents and deploy them using a REST API. [42]

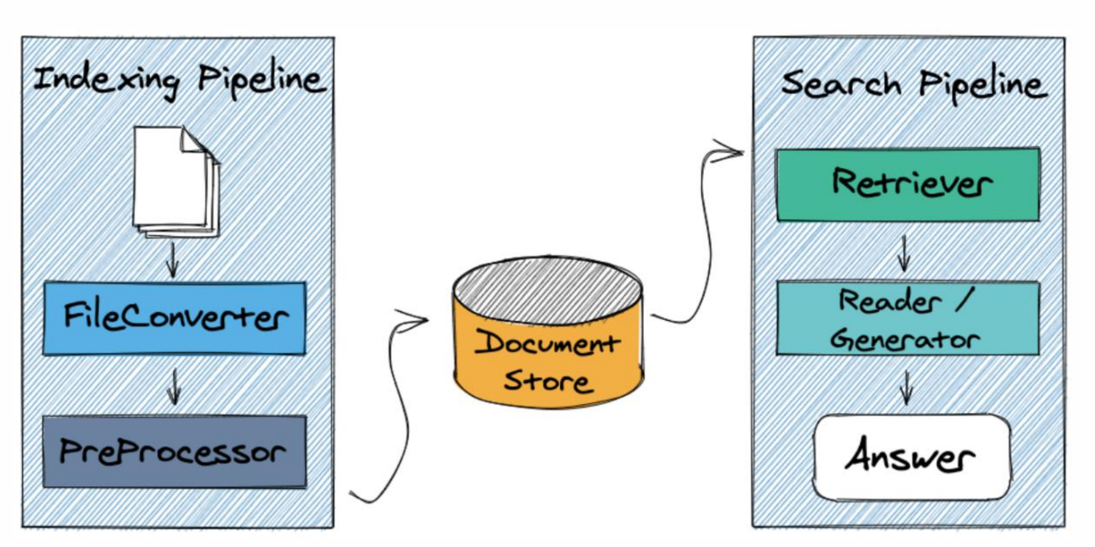


Figure 15 Haystack QA

Elasticsearch was used in the first steps of building the QA system. It is an engine for searching and analytics, open and free to use and capable of handling all kinds of data, both structured and unstructured. The following steps were to create a Document Store instance, convert our files into texts, clean and split the texts and then write them back to the Document Store. The same JSON files from CORD-19 dataset were used in this task.

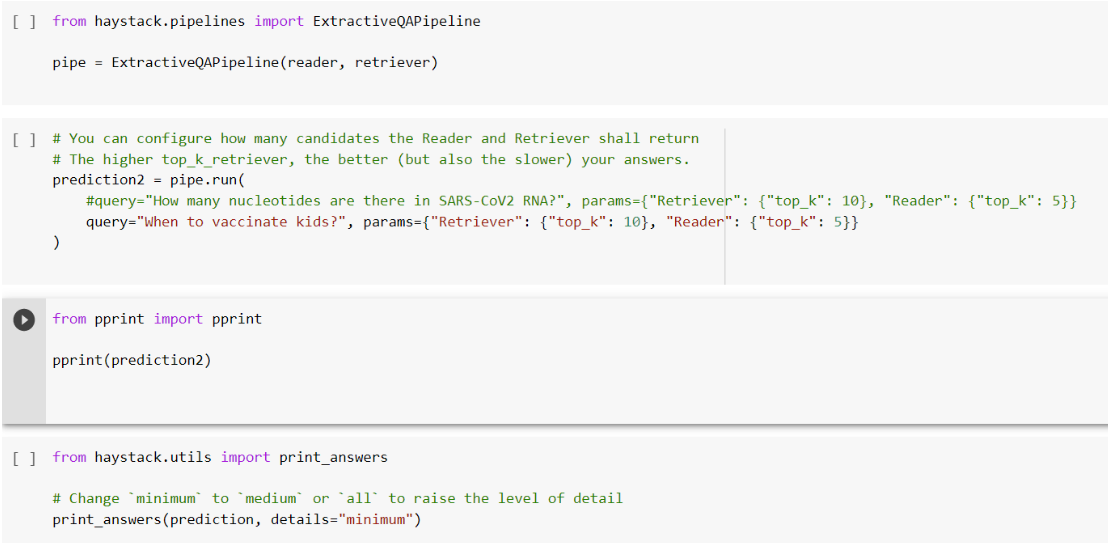
We used BM25Retriver to process smaller units of text and provide an answer when a question is asked. After this, FARMReader was deployed, to read our documents. Finally, ExtractiveQAPipeline was used, that combines a retriever and a reader to answer the question-user query.

As mentioned above, Haystack is very scalable, and one can configure how many candidates the Reader and Retriever shall return. We set up the top_k retriever to 10 and the top_k reader to 5 to get the best answers. You can find in below screenshots 2 of the queries-questions asked among with the best 5 answers received.

Questions:

1. How many nucleotides are there in SARS-CoV2 RNA?
2. When to vaccinate kids?

Code snippet:



```
[ ]  from haystack.pipelines import ExtractiveQAPipeline

     pipe = ExtractiveQAPipeline(reader, retriever)
```

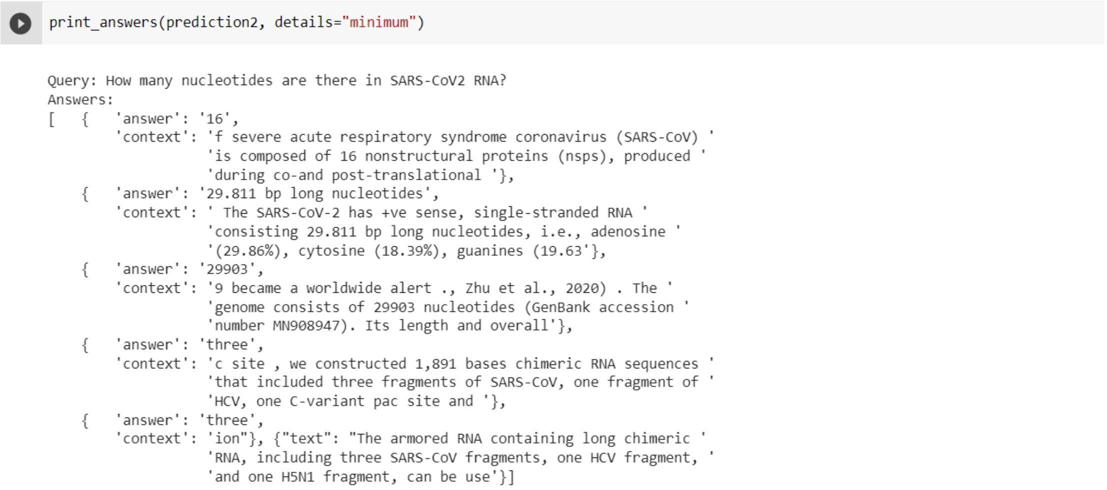
```
[ ]  # You can configure how many candidates the Reader and Retriever shall return
     # The higher top_k_retriever, the better (but also the slower) your answers.
     prediction2 = pipe.run(
         #query="How many nucleotides are there in SARS-CoV2 RNA?", params={"Retriever": {"top_k": 10}, "Reader": {"top_k": 5}}
         query="When to vaccinate kids?", params={"Retriever": {"top_k": 10}, "Reader": {"top_k": 5}}
     )
```

```
     from pprint import pprint

     pprint(prediction2)
```

```
[ ]  from haystack.utils import print_answers

     # Change `minimum` to `medium` or `all` to raise the level of detail
     print_answers(prediction, details="minimum")
```

Figure 16 questions

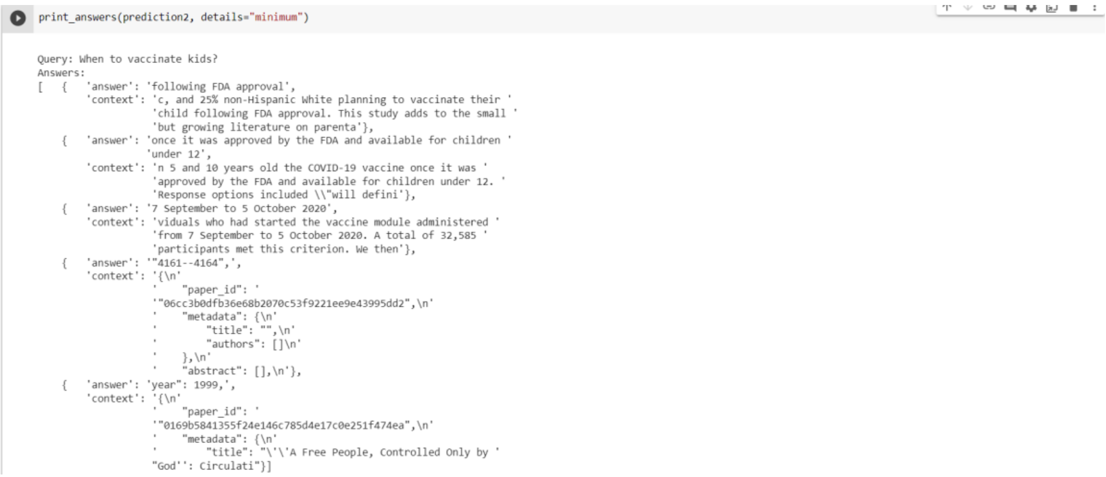
Results:



```
print_answers(prediction2, details="minimum")
```

```
Query: How many nucleotides are there in SARS-CoV2 RNA?
Answers:
[   {   'answer': '16',
        'context': 'f severe acute respiratory syndrome coronavirus (SARS-CoV) '
                   'is composed of 16 nonstructural proteins (nsps), produced '
                   'during co-and post-translational '},
    {   'answer': '29.811 bp long nucleotides',
        'context': ' The SARS-CoV-2 has +ve sense, single-stranded RNA '
                   'consisting 29.811 bp long nucleotides, i.e., adenosine '
                   '(29.86%), cytosine (18.39%), guanines (19.63'},
    {   'answer': '29903',
        'context': '9 became a worldwide alert ., Zhu et al., 2020) . The '
                   'genome consists of 29903 nucleotides (GenBank accession '
                   'number MN908947). Its length and overall'},
    {   'answer': 'three',
        'context': 'c site , we constructed 1,891 bases chimeric RNA sequences '
                   'that included three fragments of SARS-CoV, one fragment of '
                   'HCV, one C-variant pac site and '},
    {   'answer': 'three',
        'context': 'ion"}, {"text": "The armored RNA containing long chimeric '
                   'RNA, including three SARS-CoV fragments, one HCV fragment, '
                   'and one H5N1 fragment, can be use'}]
```

Figure 17 answers



```
print_answers(prediction2, details="minimum")
```

```
Query: When to vaccinate kids?
Answers:
[   {   'answer': 'following FDA approval',
        'context': 'c, and 25% non-Hispanic White planning to vaccinate their '
                   'child following FDA approval. This study adds to the small '
                   'but growing literature on parenta'},
    {   'answer': 'once it was approved by the FDA and available for children '
                  'under 12',
        'context': 'n 5 and 10 years old the COVID-19 vaccine once it was '
                   'approved by the FDA and available for children under 12. '
                   'Response options included \\"will defini'},
    {   'answer': '7 September to 5 October 2020',
        'context': 'viduals who had started the vaccine module administered '
                   'from 7 September to 5 October 2020. A total of 32,585 '
                   'participants met this criterion. We then'},
    {   'answer': '"4161--4164",',
        'context': '{\n'
                   '    "paper_id": '
                   '"06cc3b0dfb36e68b2070c53f9221ee9e43995dd2",\n'
                   '    "metadata": {\n'
                   '        "title": "",\n'
                   '        "authors": []\n'
                   '    },\n'
                   '    "abstract": [],\n'},
    {   'answer': 'year": 1999,',
        'context': '{\n'
                   '    "paper_id": '
                   '"0169b5841355f24e146c785d4e17c0e251f474ea",\n'
                   '    "metadata": {\n'
                   '        "title": "\'\'A Free People, Controlled Only by '
                   "God'': Circulati"}]
```

Figure 18 answers II

4 Conclusions

The goal of this thesis was to give a thorough and approachable introduction to knowledge graphs, the approaches for creating a knowledge graph from documents and the ways that Natural Language Processing can be used to improve question answering for Covid-19 information.

Scientists and domain specialists could leverage this technology to get critical insights from the massive swathes of data made available to the public since the global commencement of the COVID-19 pandemic. Since at least two groups have published COVID-19 KG implementations that may be used in conjunction with one another, this potential is not only hypothetical. This is just the beginning and the following years we will see a ton more work done. [23]

Knowledge graphs are not a novel concept; graph-based knowledge representation has been studied for many years. Instead, it is a catchphrase that Google developed, and that other businesses and academic institutions have adopted to describe various applications of knowledge representation. Applications for knowledge representation are constructed in a variety of ways, from fully handmade knowledge bases to knowledge graphs that are automatically retrieved and processed. [2] Web knowledge graphs can be found free of use or commercial. Apart from Google's first introduced to the public knowledge graph, many other knowledge graphs can be found, mostly created by using semi-structured data/knowledge. DBpedia, Freebase , YAGO can be considered the most well-known publicly accessible knowledge graphs that are using raw text as input (from Wikipedia or other sources). Different refinement techniques have been put forth to further improve the usefulness of these knowledge graphs. These techniques aim to infer and add missing knowledge to the graph or to spot inaccurate information with the use of various NLP techniques. [43]

It is obvious that a COVID-19 KG might possibly play a significant role for physicians, policymakers, epidemiologists, and other domain specialists now striving to gain deeper insight into the issue given the many milestones in applied KG research, both in for-profit and not-for-profit domains. There are undoubtedly more academic organizations working on related projects nowadays. The COVID*GRAPH, COVID-19 Pathophysiology Knowledge Graph, and Yahoo COVID-19 KG efforts, however, already demonstrate the promise and growth of KGs, as it would have been unthinkable just a few years ago to set up a fully-fledged domain-specific KG implementation and public-facing architecture within months of a pandemic. The technology has undoubtedly advanced much in the last few years, even though there are still a lot of obstacles to overcome and opportunities to seize. [23]

5 List of Figures

6 List of Abbreviations

Abbreviation	**Meaning**
AI	Artificial Intelligence
BERT	Bidirectional Encoder Representations from Transformers
DB	Database
ETL	Extract, Transform, Load
FDA	Food and Drug Administration
FM	Fuzzy Matching
IBM	International Business Machines
IE	Information Extraction
ISE	Intelligent Search Engine
JSON	JavaScript Object Notation
KEQA	Knowledge Embedding based Question Answering
KG	Knowledge Graph
KGC	Knowledge Graph Completion
MERS	Middle East Respiratory Syndrome
MESH	Medical Subject Headings
NER	Named Entity Recognition
NLP	Natural Language Processing
NLQ	Natural Language Question
PCQA	Predicate Constraints based Question Answering
PDF	Portable Document Format
QA	Question Answering
RDF	Resource Description Framework
SARS	Severe Acute Respiratory Syndrome
SPARQL	SPARQL Protocol and RDF Query Language
W3C	World Wide Web Consortium

Bibliography

[1] X. Zhao, H. Chen, Z. Xing and C. Miao, «Brain-Inspired Search Engine Assistant Based on Knowledge Graph,» *IEEE Transactions on Neural Networks and Learning Systems,* pp. 1-15, 2021.

[2] J. Yuan, Z. Jin, H. Guo, H. Jin, X. Zhang, T. Smith and . J. Luo , «Constructing biomedical domain-specific knowledge graph with minimum supervision,» *Knowledge and Information Systems volume,* pp. 317-336, 2020.

[3] M. Yani and A. A. Krisnadhi, «Challenges, Techniques, and Trends of Simple Knowledge Graph Question Answering: A Survey,» *Collection Knowledge Graphs for Search and Recommendation,* 30 June 2021.

[4] X. Wu, Y. Tang, C. Zhou, G. Zhu, J. Song and G. Liu, «An Intelligent Search Engine Based on Knowledge Graph for Power Equipment Management,» in *5th International Conference on Energy, Electrical and Power Engineering (CEEPE)*, 2022.

[5] I. Uzunov, «A Global Pharma Company Uses Ontotext's Solution for Semantic Similarity Search in Documents,» Ontotext, [Online]. Available: https://www.ontotext.com/.

[6] S. Sudhahar, «Knowledge Graph Embeddings for Entity, Link Prediction — The Basics,» 11 February 2019. [Online]. Available: https://medium.com/@saatviga/knowledge-graph-embeddings-for-entity-link-prediction-the-basics-4d433e048c0a.

[7] S. Srivastava, M. Patidar, S. Chowdhury, P. Agarwal, I. Bhattacharya and G. Shroff, «Complex Question Answering on knowledge graphs using machine translation and multi-task learning,» in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 2021.

[8] C. B. G. K. J. L. R. C. Z. I. Sören Auer, «DBpedia: A Nucleus for a Web of Open Data».

[9] R. Sirhari, C. Niu and W. Li, «A Hybrid Approach for Named Entity and Sub-Type Tagging,» in *Proceedings of the sixth conference on Applied natural language processing*, 2000.

[10] S. Shin, X. Jin, J. Jung and K.-H. Lee, «Predicate constraints based question answering over knowledge graph,» *Information Processing & Management,* Vol. 56, No. 3, pp. 445-462, 2019.

[11] B. Shi and T. Weninger, «Open-World Knowledge Graph Completion,» *Artificial Intelligence AAAI - Computation and Language ,* pp. 1-8, 2018.

[12] W. Shen, J. Wang and J. Han, «Entity Linking with a Knowledge Base: Issues, Techniques, and Solutions,» *IEEE Transactions on Knowledge and Data Engineering,* Vol. 27, No. 2, pp. 443-460, 2015.

[13] I. Roldós, «MonkeyLearn,» 2020. [Online].

[14] H. Paulheim, «Knowledge graph refinement: A survey of approaches and evaluation methods».

[15] C. Niklaus, M. Cetto, A. Freitas and S. Handschuh, «A Survey on Open Information Extraction,» in *27th International Conference on Computational Linguistics (COLING 2018)*, 2018.

[16] K. Munir and S. M. Anjum, «The use of ontologies for effective knowledge modelling and information retrieval,» *Applied Computing and Informatics,* Vol. 14, No. 2, pp. 116-126, 2018.

[17] B. Mohit, «Named Entity Recognition,» in *Natural Language Processing of Semitic Languages*, Mohit, Behrang, 2014, pp. 221-245.

[18] M. Missikoff, R. Navigli and P. Velardi, «he usable ontology: An environment for building and assessing a domain ontology.,» in *The Semantic Web—International Semantic Web Conference*, 2002.

[19] A. Mikheev, C. Grover and M. Moens, «"Description OF THE LTG SYSTEM FOR MUC-7,» in *Proceedings of the seventh Message Understanding Conference (MUC-7)*, 1998.

[20] J. P. Mc Cusker, D. L. McGuinness, J. S. Erickson and K. Chastain, «What is a Knowledge Graph?,» *Authorea.*

[21] F. F. Luz and M. Finger, «Semantic Parsing Natural Language into SPARQL: Improving Target Language Representation with Neural Attention,» 12 March 2018. [Online]. Available: https://arxiv.org/abs/1803.04329.

[22] S. Lukassen, R. L. Chua, T. Trefzer, N. . C. Kahn, M. . A. Schneider, T. Muley, H. Winter, M. Meister, C. Veith, A. . W. Boots, B. P. Hennig, M. Kreuter, C. Conrad and R. Eils, «SARS-CoV-2 receptor ACE2 and TMPRSS2 are primarily expressed in bronchial transient secretory cells,» *Roland,* 2020.

[23] K. L. Y. C. R. R. J. Y. D. B. D. E. K. F. Y. K. R. K. Y. L. Z. L. W. M. P. M. D. M. D. R. Lucy Lu Wang, «CORD-19: The Covid-19 Open Research Dataset,» *National Library of Medicine,* 2020.

[24] A. J. Kumar, C. Schmidt and J. Köhler, «A knowledge graph based speech interface for question answering systems,» *Speech Communication,* Vol. 92, pp. 1-12, 2017.

[25] M. Kejriwal, P. Szekely and C. Knoblock, «Investigative Knowledge Discovery for Combating Illicit Activities,» *IEEE Communications Surveys & Tutorials,* Vol. 33, No. 1, 2014.

[26] M. Kejriwal, Domain-specific knowledge graph construction. Springer, 2019.

[27] M. Kejriwal, «Knowledge Graphs for Social Good: An Entity-Centric Search Engine for the Human Trafficking Domain,» *Proceedings of the First International Conference on Web Information Systems Engineering,* Vol. 8, No. 3, 2020.

[28] M. Kejriwal, «Knowledge Graphs and COVID-19: Opportunities, Challenges, and Implementation,» 23 May 2022. [Online]. Available: https://hdsr.mitpress.mit.edu/pub/xl0yk6ux/release/2?readingCollection=0181d53b. [01 December 2020].

[29] M. Kejriwal, «Knowledge Graphs and COVID-19: Opportunities, Challenges, and Implementation,» *Harvard Data Science Review,* 01 December 2020.

[30] P. Joshi, «Knowledge Graph – A Powerful Data Science Technique to Mine Information from Text (with Python code),» 2019.

[31] A. I. f. A. Intelligence, «spaCy».

[32] X. Huang, J. Zhang, D. Li and P. Li, «Knowledge Graph Embedding Based Question Answering,» *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining,* pp. 105-113, 30 January 2019.

[33] C. Gutierrez and J. F. Sequeda, «Knowledge Graphs: A Tutorial on the History of Knowledge Graph's Main Ideas,» in *CIKM '20: Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020.

[34] T. R. Gruber, «Toward Principles for the Design of Ontologies Used for Knowledge Sharing,» *International Journal of Human-Computer Studies,* Vol. 43, No. 5-6, pp. 907-928, 1995.

[35] s. google.com, «Knowledge Panel Help,» Google, 15 March 2021. [Online]. Available: https://support.google.com/knowledgepanel/answer/9787176?hl=en.

[36] L. Ehrlinger and W. Wöß, «Towards a Definition of Knowledge Graphs,» in *Joint Proceedings of the Posters and Demos Track of 12th International Conference on Semantic Systems - SEMANTiCS2016 and 1st International Workshop on Semantic Change & Evolving Semantics*, At: Leipzig, Germany, 2016.

[37] Y. Duan, L. Shao and G. Hu, «Specifying Knowledge Graph with Data Graph, Information Graph, Knowledge Graph, and Wisdom Graph,» *International Journal of Software Innovation,* Vol. 6, No. 2, pp. 10-25, 2018.

[38] S. Domingos Cardoso, M. . D. Silveira and C. Pruski, «Construction and exploitation of an historical knowledge graph to deal with the evolution of ontologies,» *Knowledge-Based Systems,* Vol. 194, 2020.

[39] Deepset, «Haystack,» [Online].

[40] H. Cho, . D. Ippolito and Y. W. Yu, «Contact Tracing Mobile Apps for COVID-19: Privacy Considerations and Related Trade-offs,» 30 March 2020. [Online]. Available: https://arxiv.org/abs/2003.11511. [24 March 2020].

[41] BilalAbu-Salih, «Domain-specific knowledge graphs: A survey,» *Journal of Network and Computer Applications,* Vol. 185, 2021.

[42] J. Al-Saleem, R. Granet, S. Ramakrishnan, N. . A. Ciancetta, C. Saveson, C. Gessner and Q. Zhou, «Knowledge Graph-Based Approaches to Drug Repurposing for COVID-19,» American Chemical Society, 2021.

[43] L. S. Affendey and A. Mansouri, «Named Entity Recognition Approaches,» *IJCSNS International Journal of Computer Science and Network Security,* Vol. 8, No. 2, pp. 339-344, 2008.

Appendix

1) Question Answering Haystack Framework in Python

```
import logging

logging.basicConfig(format="%(levelname)s - %(name)s -  %(message)s", level=logging.WARNING)
logging.getLogger("haystack").setLevel(logging.INFO)

from haystack.utils import launch_es

launch_es()

import time
time.sleep(30)

import os
from haystack.document_stores import ElasticsearchDocumentStore

# Get the host where Elasticsearch is running, default to localhost
host = os.environ.get("ELASTICSEARCH_HOST", "localhost")
document_store = ElasticsearchDocumentStore(host=host, username="", password="", index="document")

from haystack.utils import convert_files_to_docs, fetch_archive_from_http

from google.colab import drive

drive.mount('/content/drive/', force_remount=True)

import os

doc_dir = "/content/drive/MyDrive/cord19/sample"

docs = convert_files_to_docs(dir_path=doc_dir, clean_func=None, split_paragraphs=True)

print(docs[:3])

# Now, let's write the dicts containing documents to our DB.
document_store.write_documents(docs)

from haystack.nodes import BM25Retriever

retriever = BM25Retriever(document_store=document_store)

from haystack.nodes import FARMReader

reader = FARMReader(model_name_or_path="deepset/roberta-base-squad2", use_gpu=True)

from haystack.pipelines import ExtractiveQAPipeline

pipe = ExtractiveQAPipeline(reader, retriever)
```

```
# You can configure how many candidates the Reader and Retriever shall
return
# The higher top_k_retriever, the better (but also the slower) your
answers.
prediction2 = pipe.run(
    #query="How many nucleotides are there in SARS-CoV2 RNA?",
params={"Retriever": {"top_k": 10}, "Reader": {"top_k": 5}}
    query="When to vaccinate kids?", params={"Retriever": {"top_k":
10}, "Reader": {"top_k": 5}}
)

from pprint import pprint

pprint(prediction2)

from haystack.utils import print_answers

print_answers(prediction, details="minimum")
```