university of groningen / faculty of science and engineering

Knowledge Mining in Scientific Articles

Bachelor's thesis

Student:
Andreea-Ioana Dan

First supervisor:
Dimka Karastoyanova

Second supervisor:
Mirela Riveni

External supervisor:
Dennis Mulder

July 25, 2022

Abstract

Objective: To investigate and create a system that would facilitate the conversion of data from an unstructured form (medical scientific articles) into a structured form (graph data structure). Identify a schema that allows the creation of the graph database. Identify possible use cases and limitations of the system.

Materials and methods: 5 scientific articles that tackle the subject of olfactory and gustatory dysfunctions against the recently studied coronavirus disease (Covid-19) are used as an initial data set. We will analyze the following sections: Introduction, Methods, Results, Discussion and Conclusion. The technology used will be based on the cloud technologies offered by the Microsoft Azure platform.

Results: We have successfully identified relationships between medical entities within the articles and created graph representations of these. These graphs reflect the overall content of the articles in a structured manner. Further on, these graphs can be queried depending on the specific use case that they are used for.

Discussion: A graph database has been created for the use within several use cases, among which information verification and/or machine learning applications. However, a number of issues and limitations have been discovered and discussed throughout the development process. Therefore, there are many opportunities for improvement, such as structural standardization for scientific articles and solving of previous limitations.

Conclusion: The goals have been reached by using the aforementioned cloud technologies. We are confident that this proposal can grow further and have a real impact on society and in the research environment.

Contents

1 Introduction

When looking at the trends of data growth over the past years, we can see that data-driven services have become more in demand. A world development report from the World Bank Group in 2021 stated that in 2018 there were approximately 100,000 petabytes (PB) of digital data flows per month, that represented approximately 6,000 billion US dollars in service. To have a broader overview, let us look at the previous years. In 2015, there have been approximately 80,000 PB digital data flows per month, in 2012 approximately 70,000 PB and in 2009 approximately 60,000 PB [1]. Respecting the previous trend, we can expect growth in the next years. In 2020, the amount of data processed and used worldwide has quickly reached 59 zettabytes (ZB). Predictions over the next 5 years project data amounts to at least double its amount, reaching almost 149 zettabytes [2].

Even though the aforementioned data refers to overall data present globally, the data produced by scientific research is growing steadily [3]. For example, in the past years, during the pandemic, there has been a surge of research articles in the medical sphere. Between the 1^{st} of January 2020 and 30^{th} of June 2020, there have been at least 23,634 unique documents published around Covid-19 [4].

Thus, the amount of data held by scientific articles nowadays is already more than humans can individually ingest. Expectations are that these amounts will only continue to grow. Therefore, there is a need of automation of processing and mining the data represented by these articles.

1.1 State of the art

Current Situation

The healthcare sector is one of the biggest generators of data. Once Big Data has come into play and became popular, the healthcare industry has proved to be a globally impactful source of data [5]. There is a rising need for automation of data extraction, processing but also understanding [6]. In this document we will propose an approach that focuses on data extraction and processing from medical scientific articles, although the healthcare industry offers numerous opportunities for applying automation. Tightly connected with improving the patient experience, precision and personalized medicine is making its way to becoming a standard of treatment. In order to apply such an approach in medicine, decisions have to be taken in a data-driven mode. However, in order to offer support for data-driven recommendations, we require a way in which we can process and summarize data. In the past years, several statistical and analytical methods have been used which have been partially successful. Nonetheless, in order to extract and gain knowledge, we need to teach an algorithm to understand the data, not only to cite it [7] because just citing it, still leans on the healthcare providers and other involved stakeholders to filter the available data.

In current times, knowledge mining is usually applied on structured data, such as databases, this process is called Knowledge Discovery in Databases (KDD). The data in such databases is either extracted from sensors, is human generated, or stemming from other sources [8]. On the other hand, knowledge mining in unstructured data sources (e.g. scientific articles) is mostly focused on written text study, creating links between facts and identifying patterns [9].

Current Market

Let us turn our attention to technology currently available that would allow us to answer questions like "How can we mine the data from scientific articles?" or "Can we design a program that converts unstructured data to structured data?". There are several tools available that can be used for text mining solutions. These include Vizcontrols created by the company Inxight [10], that make use of visualizations to showcase the links between facts (data points). There are also tools that make use of the cloud environment, such as: IBM's Watson Natural Language Understanding tool [11]; Google's Natural Language Understanding tool [12]; Microsoft's Azure Cognitive Services [13] and many others that make use of Natural Language processing technologies. More focused on Text Mining in Scientific Articles is the CAT (Content Analysis Toolkit) by Indutech Ltd. This tool extracts information, clusters it, creates fact connections and then presents a visualization view that can be seen using Excel [9, 14].

Conclusion

To conclude from the previous findings about the current state of the art, most available technologies focus on extracting data for immediate visualization or conclusion generation. However, we have found less research results that describe a process or a system that is focused on transforming unstructured data into structured data in order for it to be used in long term applications in comparison with simple data extraction research.

Therefore, in our research project we shall continue our research of existing approaches for the aforementioned transformation whilst developing our own implementation. Future views for text mining hide an abundance of opportunities for improvement and additional features. Some of these opportunities can include: adding new features such as research results enhanced by semantic analysis or multilingual data extraction [15] but also improving existing features with a focus on increasing the degree of result consistency and reliability [16].

1.2 Purpose

The purpose of this project is to devise an automated way to ingest unstructured data, in the form of medical scientific articles, and obtain graphs that reflect the overall content of the ingested data. With this system, we aim to answer the following research questions:

1. *Can we **design** a program that is able to extract unstructured data and transform it into a persistent structured data source?*
2. *How can a scientific article in the field of healthcare/health/medicine research be summarized in a **structured database schema**?*
3. *What are the **applications** of such a program?*
4. *What are the **limitations** of such a program?*

In order to respond to the aforementioned questions, we propose the system flow found in Figure 1 where the user is then able to query the graph database as required by their current use case:

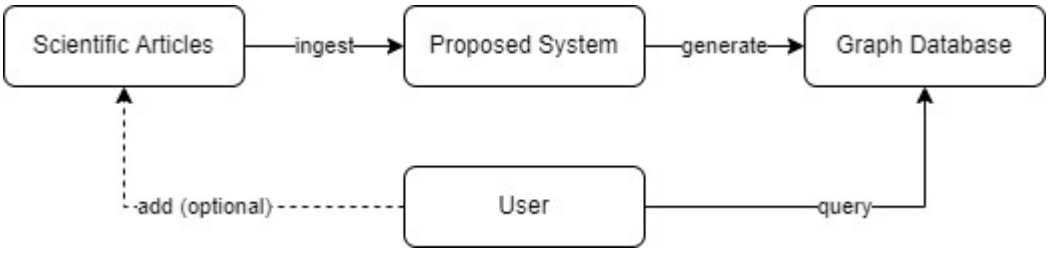

Figure 1: Overall flow of proposed system

We are aware of the size of the impact that this system can have, but also of the challenges and versatility of it. Therefore, our intention for this project is to create the foundation of a more complex future project. There is a heterogeneous pool of use cases that we assume for this system. More about the use case options will be presented in the Discussions section.

It is important to mention that the scope of the project is the ingesting of medical scientific articles, processing them and generating the graph database. Database querying done by the user is out of scope of the current project due to the fact that it is highly dependent on the use case.

2 Materials and methods

Given that the amount of data represented by scientific articles is growing each year, almost 8% to 9% each year and only the articles resulting from biomedical studies represent approximately 1 million papers each year (submitted to PubMed) [17] we decided to make use of cloud technologies.

2.1 Cloud Computing

The main advantages of cloud computing derive from the fact that it makes use of servers connected to the internet at different remote locations. The cloud providers are in charge of storing, managing and/or processing the data offered by their clients. Thus, the client is able to rent the required server capacity, use it to its intended purpose and then dispose of it back to the provider. This automatically results in lowered costs for hardware investments for enterprises [18].

The advantage of renting a server also contributes to the scalability aspect of cloud computing. Most providers offer elastic cloud options where the memory and processing power of the servers is allocated depending on the resources needed by a client at a specific time. This is possible due to the Virtualization technology. By using the Virtualization technology, the physical barriers between isolated resources are removed and the management of several required resources can be treated as a single entity. This virtualization is made possible by using Hypervisor technology [19].

Currently, there are several big cloud providers: Microsoft Azure Cloud, Amazon Web Services (AWS), Google Cloud and IBM Clouds. Other cloud providers are also emerging [20]. These providers offer different types of services, such as: IaaS (Infrastructure as a Service), PaaS (Platform as a Service) and SaaS (Software as a Service). For our project, we will be focusing on using data storage services (PaaS) and computing/processing services (SaaS).

We are expecting our input to be represented by high amounts of data and the output data to require less storing space but possibly a higher computing power. Thus, the low costs for hardware, the scalability together with large storage possibilities, and the availability of complex cloud computing services, contributed to our decision to opt for cloud computing technologies for this project. Due to present circumstances where we have received free access to Microsoft Azure Cloud Computing Services, we will be focusing on their specific technologies.

2.2 Microsoft Azure Cloud Computing Technologies

According to Microsoft, Azure is a cloud platform that contains over 200 products and cloud services that aim to build, run and manage applications throughout a global network of privately managed and owned data centers. Azure offers a collection of services that span across several fields: database, computing, analytics, storage, security, networking etc. [21, 22].

For this project, we will be making use of the following Azure technologies:

1. Azure Blob Storage
2. Azure Cognitive Search
3. Azure Functions
4. Azure Cognitive Services - Text Analytics for Health
5. Azure Cosmos DB - Gremlin API

Next, we will present a short overview about each technology and the reasoning behind the choice of their usage:

Azure Blob Storage

Azure Blob Storage is part of the solutions offered by Microsoft as part of the Azure Storage. In Azure Storage, different types of data can be stored as: Blobs, Queues, or Tables. For our project, we chose block blobs. The word blob stands for a binary large object. Such objects simply store binary data in any shape. Examples of such objects are: portable document format (PDF) files, Microsoft Office format files (e.g. .doc, .xlsx, .pptx etc.), images, videos, comma separated value (CSV) files, archive (e.g. zip) files, virtual hard drives (VHD) etc. [22, 23, 24].

Given that the majority of research articles can be found on the internet in a downloadable format as a PDF file, we found that the Blob Storage is most suitable for this kind of files.

Azure Cognitive Search

Searching and extracting content from the PDF files that we previously stored in the Blob storage can become quite complex. However, we made use of the Cognitive Search in order to facilitate this process. The Cognitive Search provides deep integration at the content layer but it also extracts and deduces information resulted from the text.

The Cognitive Search service imports the content of the uploaded blob files and considers them to be search documents. These documents are then indexed with *inverted indexes*, that are capable to be filtered by user-managed conditions or queried. The inverted index is important because it maps the content found to its original location in a given document. Therefore, a pipeline is established that takes the files from a single blob container and returns the output as a Cognitive Search index that is stored separately from the blob container. However, the indexing operations can be re-run anytime in order to refresh the index of each document [24].

The aforementioned pipeline that takes input as the blob files and returns the index is led by an indexer, as depicted in Figure 2 and discussed next:

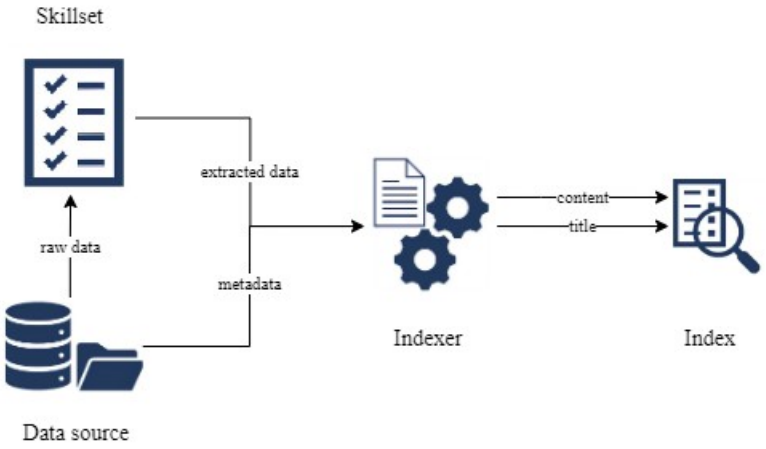

Figure 2: Overview of Cognitive Search pipeline

An indexer is a data-source-aware subservice that is programmed to sample data, read metadata, retrieve and serialize data from raw binary formats and store them into structured JSON documents (the index). This indexer crawls through the documents in order to extract the searchable content and store it in the inverted index, using field-to-field mapping between the blobs and the index. This technique is called "pull model", as the indexer pulls the data in the indexer without the user having to specifically design the code that extracts and stored data in the index.

In the pipeline, as seen in Figure 2, after the connection between the search service and the storage has been established, the indexer starts with opening the respective files and extracting the contents. This stage is called **"document cracking"**. In the case of PDF files, the indexer extracts the text (raw data), metadata and optionally, images [24, 25]. Between the "document cracking" and the data processing, the **field mapping** from the indexer to an index has to be defined. This is achieved by connecting the source field to the destination field within the indexer definition. Then, a custom defined **skillset** can be used. This varies from case to case, in the case that there are images in the given files, optical character recognition (OCR) can be used to extract text from images. It can also involve key phrase extraction, text translation etc. After obtained the data from the "document cracking" and from the skillset processing, the output field mapping takes place, where the content of the document is mapped to the destination fields from the index [25].

The current files that are considered to be input for our system are in a PDF format. The data that is required to be extracted is mainly the overall content and the title. Furthermore, it is expected that the amount of articles to reach a high number after exiting the prototyping phase. Therefore, because the Cognitive Search offers the possibility to extract raw content and metadata from numerous PDF files, it has been selected to play an important part in the proposed system's architecture.

Azure Functions

The Azure Functions service is a SaaS serverless solution that allows less infrastructure maintenance and a focus on code functionality. This means that code can run without having having to continuously handle its deployment and servers on which in runs. Essentially, the responsibility of the programmer is just to create the code that implements the logic of the program. This block of code is referred to as "a function". Given that our project is architected to make use of the functions only once in the pre-processing phases, the fact that this service offers a "compute on-demand" option, the choice has been taken also by considering the resource usage for these functions. In the pre-processing phase, the resource need is higher and then the quota decreases as the data enters the next phase [26].

Text Analytics for Health

The Text Analytics for Health service belongs to the package of services called Azure Cognitive Service for Language. These services provide Natural Language Processing (NLP) capacity in order to understand and analyze text [27].

Text Analytics for Health is capable of extracting medical information from unstructured texts and labeling them according to the vocabularies defined by the UMLS® Metathesaurus® [28]. For examples of extracted entities and relationships, refer to the Implementation subsection of this paper, that discusses the Text Analytics for Health.

The unified medical language system (UMLS®) Metathesaurus® is one of the largest dictionaries in the biomedical sphere. It is able to mine knowledge from text by extracting **concepts** classified by semantic type but also by extracting **relationships** (hierarchical and non-hierarchical) between the identified concepts. The main program that is used to populate such a thesaurus is MetaMap, developed at the National Library of Medicine (NLM) from the United States of America. This program enables a knowledge intensive approach that focuses on natural language processing, symbolic and computational linguistic techniques. The

MetaMap is available for free use as long as the UMLS® License Agreement is valid between the involved parts. MetaMap has been proved to be a valuable tool for discovering medical concepts in text. However, improvements are recommended in several performance areas, such as: idiosyncratic text detection (e.g. chemical names, acronyms, abbreviations, numeric amounts etc.) or resolution of ambiguity. These issues represent the focus of future refinement work [29].

Thus, there is a high degree of transparency regarding the limitations and improvement options of such a dictionary. This is directly communicated throughout the documentation of the Text Analytics for Health as well. Therefore, it is important to mention that this service is a capability provided "AS IS" and "WITH ALL FAULTS", highly recommended that this service not be used as a single decision point when considering diagnosis and/or treatments. It is also recommended that this system not replace the medical advice of a human party but used as an adjuvant system in the process of medical reasoning and decision-making [28].

Therefore, considering that the UMLS® Metathesaurus® has been evolving throughout the past decades, it has grown to a substantial size and accuracy in detecting medical entities and relationships, the architectural decision has been made to involve the Text Analytics for Health service in the pipeline of the proposed system. However, for our proposed system we maintain the previous disclaimer that includes the possibility of inaccuracies in the obtained results.

Azure Cosmos DB - Gremlin API

Azure Cosmos DB is a fully managed NoSQL database. Cosmos DB has immense benefits when considering the impact that it has on the final product. It has a small response time, automatic and instant scalability depending on the need but also high speed regardless of amount of data. Because Cosmos DB is a multi-model database service, it supports key-value graphs and column-family data models. When storing information in a graph structure, the number of vertices and edges can reach billions. In order to use the graph database service, the Gremlin API is enabled. This is due to the fact that Cosmos DB is highly compatible with Apache TinkerPop's applications and libraries, including their graph traversal language, Gremlin. By using the Gremlin language to query, populate and analyze the graphs in batches, an online analytics process (OLAP) is established [30].

Therefore, Cosmos DB has been chosen to represent the Graph Database in the last step of the system flow (as seen in Figure 1). The graph structure has been chosen due to the fact that we are implementing a **structural analysis** of a text, resulting in entities connected by relationships. In the resulting semantic network, the entities will be represented by nodes and the relationships will be represented by edges between the nodes. This **semantic network** can be visually represented by a graph. It can further be used in queries and traversals in order to retrieve and/or validate connections between entities [31].

2.3 Implementation

In this subsection, we will present the way in which we applied the presented technology. In order to answer the first research question that was posed, *"Can we design a program that is able to extract unstructured data and transform it into a persistent structured data source?"*, we have implemented the steps described in the following paragraphs. The answer to the research question is represented by the architecture that was obtained.

The used technology stack includes the Visual Studio Code (VS Code) code editor and the Python programming language. The reason behind using VS Code is that it is a light-weight editor that is intended to be used directly with cloud applications. An advantage of VS Code is that it integrates with Azure via the specifically designed extensions (e.g. for Azure Functions) [32].

The code base for this project can be found on a public GitHub repository at the following link: *GitHub Repository - Knowledge Mining in Scientific Articles*

In order to ease the understanding of this subsection, we are presenting the system architecture diagram (Figure 3) that will be described further on:

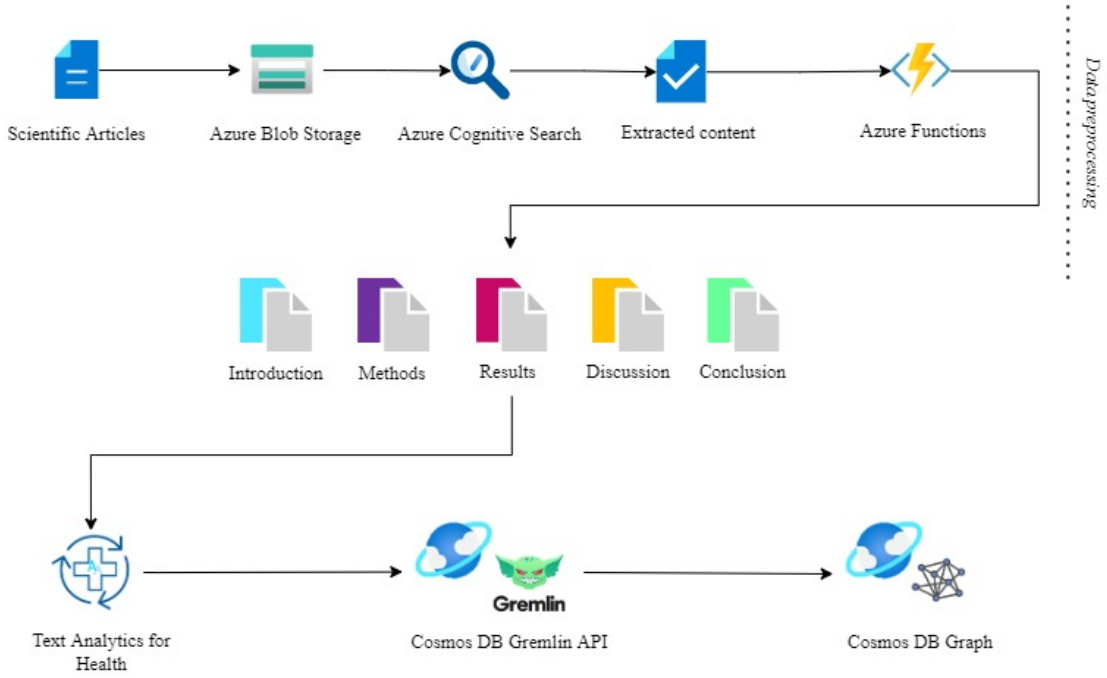

Figure 3: System Architecture

The raw data, represented by the articles, will be stored and processed to extract their content. Then, the content will then be separated into specific sections: Introduction, Methods, Results, Discussion and Conclusion. Finally,the health entities will be identified and relationships between them found. These entities and relationships will then be reflected in graphs. More detailed explanations will follow in the continuation of this subsection.

2.3.1 Article selection

To create our dataset that will be ingested by the system, we researched a number of articles that can be used as an initial dataset. For this, we made use of free searchable databases, such as PubMed® but also ScienceDirect® that provided access to peer-reviewed medical journals. All articles have been manually selected and currently, they tackle the same issue, the olfactory and gustatory dysfunctions that are caused by Covid-19. For the current prototype, we have used 5 articles. However, for extending the applicability of the proposed system, more articles from the same area and articles from other specific areas (e.g. Covid-19 vaccine and/or cancer) would be added. Once the system has been fed with sufficient articles, the step of adding further data to the system by the user becomes optional.

2.3.2 Article storage

After article selection, the articles are present on our local device. Therefore, in order to move them to they cloud, they are stored in a single Blob Storage container, as seen in Figure 4. Containers are organized such that each container is populated by articles that tackle the same issue, so that we can facilitate the separation of concerns in the future. For example, one container can contain articles that treat the olfactory and gustatory dysfunctions and another container would contain articles that talk about the Covid-19 vaccine.

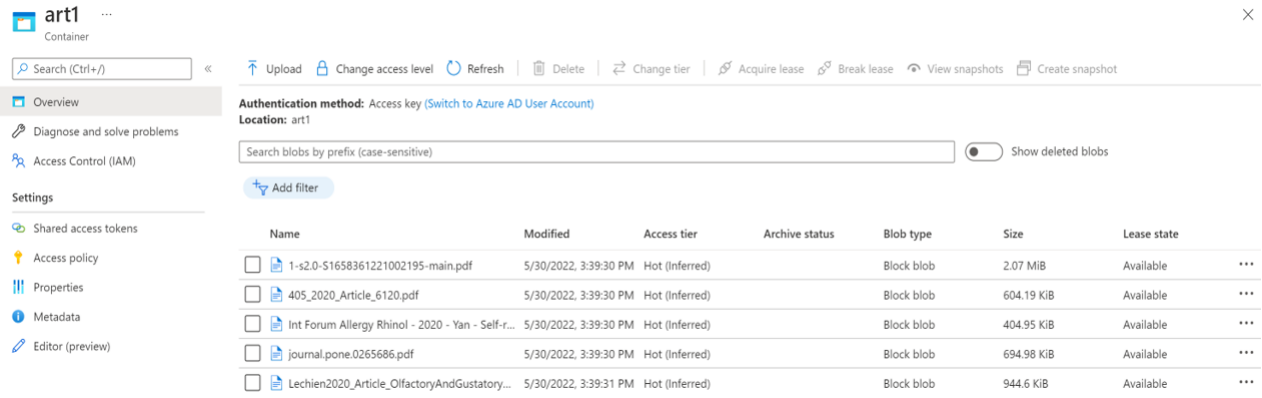

Figure 4: Blob Storage container *art1* containing articles about the olfactory and gustatory dysfunctions in relation with COVID-19

2.3.3 Content Extraction

After storing the dataset in the cloud, the Cognitive Search extracts the data and metadata from the files. Without applying any skillset to the cracked documents, the Cognitive Search is capable of extracting the written content from the files. As an embedded feature, without applying any skillset, specific metadata can be extracted from PDF files. In the current prototype, the title metadata has been extracted by using `metadata_title` as a source field name. However, in the future developments of the system, properties such as `metadata_creation_date`, `metadata_author` and/or `metadata_language` can be mapped [33].

For the data extraction we followed a basic workflow, involving:

1. **Data source creation**: Establishing connection between the Blob Storage and Cognitive Search data source.
2. **Skillset creation**: A skillset blueprint has been created. However, currently it is empty.
3. **Indexer creation**: Defining field mappings and setting connections between data source, index and indexer.

```
    indexer_payload = {
        "name": indexer_name,
        "description": "Indexer to lead article data extraction",
        "dataSourceName": datasource_name,
        "targetIndexName": index_name,
        "skillsetName": skillset_name,
        "fieldMappings": [
            {
                "sourceFieldName": "metadata_storage_path",
                "targetFieldName": "id",
                "mappingFunction":
                {"name": "base64Encode"}
            },
            {
                "sourceFieldName": "content",
                "targetFieldName": "content"
            },
            {
                "sourceFieldName": "metadata_title",
```

```
                "targetFieldName": "title"
            }
        ]
        ...
    }
```

4. **Search**: Querying the index via REST API calls in order to retrieve the content and title.

```
    r = requests.get(endpoint + "/indexes/" + index_name +
    ↪  "/docs?&search=*&$count=true&$select=content", headers=headers,
    ↪  params=params)
    t = requests.get(endpoint + "/indexes/" + index_name +
    ↪  "/docs?&search=*&$count=true&$select=title", headers=headers,
    ↪  params=params)
```

2.3.4 Section Extraction

The content stored in the index represents the text from the scientific articles as resulted from the recognition processes, as it can be observed in Figure 5. The content is organized in a `.json` format. In this paper, we are presenting examples resulted from analysing the article written by Hintschich et al. ([34]) about the connection between olfactory and gustatory dysfunction after Covid-19 infection.


```
{'@odata.context':
"https://cognsearchbasicth.search.windows.net/indexes('cogsrch-py-
index')/$metadata#docs(*)", '@odata.count': 5, 'value': [{'@search.score':
1.0, 'content': '…could not be\n\nobserved in acute COVID-19 [42] and suggests
that a chronic post-COVID-19 hyposmia\n\nmight secondarily impair gustatory
function.\n\nConclusion\n\nThis study confirmed that olfactory dysfunction can
be a chronic symptom even eight months\n\nafter COVID-19. Interestingly,
hypogeusia, a rare symptom during the acute infection, has\n\nbeen
psychophysically confirmed in 32% of the study population. This must be seen
as a fre-\n\nquent symptom of post-COVID-19 condition and might be partly due
to a decreased central\n\nnervous amplification.\n\nSupporting
information\n\nS1 Data.\n\n(XLSX)\n\nAuthor Contributions\n\nCon…'}]}
```


Figure 5: Content stored in search index. *Disclaimer: Actual section is longer, but it was shortened to facilitate reading*

In order to extract specific sections from the content, custom Azure Functions were used. Thus, we made use of regex functions to extract the sections, as seen in the following code snippet:

```
regex_terms = r'\n('+section_name+r')\n(.|\n)+\n('+next_section_name+r')\n'
```

For this, we assumed that every section is bordered by newlines (\n), thus they are considered to be titles. The section names have been set to have several values, so that a recursive function verifies several possibilities of sections. The extraction function can be seen in the following code snippet:

```
def extract_section(section_name:str, next_section_name: str, content: str,
↪  all: bool) -> str:
    ...
    # Return the section; None if section not found
    result = re.search(regex_terms, content)

    # Recursive steps: try variations of section names
    if result == None:
        variation_current_section_name =
        ↪  variation_current_section(section_name)
        if variation_current_section_name != None:
          result, section_name =
          ↪  extract_section(variation_current_section_name,
          ↪  next_section_name, content, all)
        ...
    else:
        result = clean_result(result,section_name, next_section_name)

    # Base case
    if result == None:
        return "Section NOT FOUND!", section_name.lower()

    return result, section_name.lower()
```

More information about different section possibilities will be presented in the Discussion section.

2.3.5 Medical entity and relationship recognition

In order to store the data in a graph database, we require to obtain **relationships** between **entities**. As mentioned before, we are making use of a Cognitive Service, namely Text Analytics for Health. This service is identifying medical entities from the given text and the relationships between them. First, a client was created: `client = TextAnalyticsClient(endpoint=language_endpoint,credential=ta_credential)`, which was used to analyze the healthcare entities and relationships from the given documents, as it can be seen in the following code snippet:

```
    def health_example(client, text):
        documents = [text]
        poller = client.begin_analyze_healthcare_entities(documents)
        result = poller.result()

        docs = [doc for doc in result if not doc.is_error]

        for idx, doc in enumerate(docs):
            for relation in doc.entity_relations:
                print("Relation of type: {} has the following
                ↪  roles".format(relation.relation_type))
                for role in relation.roles:
                    print("...Role '{}' with entity '{}'".format(role.name,
                    ↪  role.entity.text))
                print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")

        return docs
```

The Text Analytics for Health service supports **concept** identification within different types of **health entity categories** [35]. For example:

1. Anatomy: BODY_STRUCTURE
2. Demographics: AGE, GENDER
3. Examinations: EXAMINATION_NAME
4. External influence: ALLERGEN
5. General attributes: COURSE, DATE, MEASUREMENT_UNIT, MEASUREMENT_VALUE etc.
6. Genomics: VARIANT, MUTATION_TYPE, EXPRESSION etc.
7. Healthcare: ADMINISTRATIVE_EVENT, HEALTHCARE_PROFESSION etc.
8. Medical condition: DIAGNOSIS, SYMPTOM_OR_SIGN, CONDITION_QUALIFIER etc.
9. Medication: MEDICATION_CLASS, MEDICATION_NAME, DOSAGE, MEDICATION_ROUTE etc.
10. Social: FAMILY_RELATION
11. Treatment: TREATMENT_NAME

It also supports inferring different **semantic relations** between concepts [36]. For example:

1. ABBREVIATION
2. BODY_SITE_OF_CONDITION
3. COURSE_OF_CONDITION
4. COURSE_OF_EXAMINATION
5. DIRECTION_OF_EXAMINATION
6. DOSAGE_OF_MEDICATION
7. EXAMINATION_FINDS_CONDITION
8. QUALIFIER_OF_CONDITION
9. TIME_OF_CONDITION
10. UNIT_OF_CONDITION
11. VALUE_OF_CONDITION etc.

In Figure 6 we can see the result of the previous step, section extraction, which is represented by the individual conclusion of the respective article. Underneath the conclusion is the output of the previous code snippet (`health_example() function`), where the entities and relationships are identified. In the figure, we chose only to showcase a part of the output, in order to maintain a clear representation:

```
"This study confirmed that olfactory dysfunction can be a chronic symptom even
eight months after COVID-19. Interestingly, hypogeusia, a rare symptom during
the acute infection, has been psychophysically confirmed in 32% of the study
population. This must be seen as a fre- quent symptom of post-COVID-19
condition and might be partly due to a decreased central nervous
amplification."


~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Relation of type: ExaminationFindsCondition has the following roles
...Role 'Condition' with entity 'hypogeusia'
...Role 'Examination' with entity 'psychophysically'
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Relation of type: ValueOfExamination has the following roles
...Role 'Examination' with entity 'psychophysically'
...Role 'Value' with entity '32'
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Relation of type: UnitOfExamination has the following roles
...Role 'Examination' with entity 'psychophysically'
...Role 'Unit' with entity '%'
```

Figure 6: Text Analytics for Health applied on a conclusion section

2.3.6 Graph creation and population

In oder to attempt answering the second research question that was posed, *"How can a scientific article in the field of healthcare/health/medicine research be summarized in a structured database schema?"*, we will make use of the obtained entities and relationships and we will create a structured data representation in the form of a graph. In order to populate such graphs, we designed and adopted the following graph schema, that represents the answer to the previous research question (Figure 7):

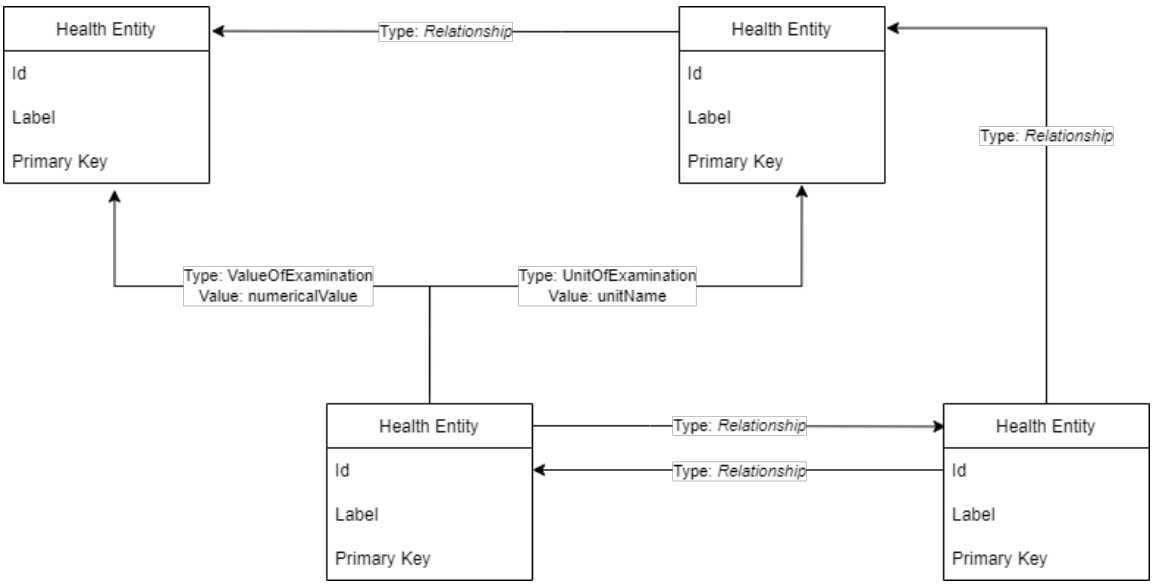

Figure 7: Graph schema; *Relationship* can have different types but will not have a value

Every entity will represent a vertex. Every relationship will represent an edge. However, there are 2 exceptions implemented for the edges. The relationships of type ValueOfExamination and UnitOfExamination are not going to be independent edges, but they will represent attributes of already existing edges. Every edge is going to be **unidirectional** from one entity to another.

At this phase of prototype development we took into consideration several scenarios, one involving the creation of a separate graph per each article. Our program implementation has been conducting using python, the python SDK's provided by Microsoft and REST API calls. However, the graph creation could be done manually via the Azure portal or automated via the .NET SDK, Azure CLI or the Azure Powershell [37]. Therefore, an important feature of our program is also the automation of graph creation using python. For this, a background process runs an Azure Powershell script that created a new graph. In the following code snippet, the launch of the background process for each graph is shown:

```
for index in range(graph_count):
        graph_name = "art_" + str(index)
        # run powershell script
        subprocess.Popen(['powershell.exe', '-ExecutionPolicy', 'Unrestricted',
        ↪  '-File', PATH_TO_PWS_SCRIPTS + '\\create_container.ps1','-graphName' ,
        ↪  "test"])
```

In the following code snippet, the code of the file `create_container.ps1` is presented:

```
# graphName as input parameter
param([string] $graphName)
# -----------------------------------------------
# Variables
$resourceGroupName = "Thesis" # Resource Group must already exist
$accountName = "cosmosdbgremlinstudent" # Must be all lower case
$databaseName = "article-database-cosmos"
$graphRUs = 400
$partitionKeys = @("/pk")
# -----------------------------------------------
# Retrieving existing database
$database = Get-AzCosmosDBGremlinDatabase -ResourceGroupName $resourceGroupName
    ↪  -AccountName $accountName -Name $databaseName
# -----------------------------------------------
# Creating new graph in database
Write-Host "Creating graph $graphName"
New-AzCosmosDBGremlinGraph -ParentObject $database `
    -Name $graphName -Throughput $graphRUs `
    -PartitionKeyKind Hash -PartitionKeyPath $partitionKeys
```

Graph creation as a background process is particularly efficient because it can run in whilst other processes are populating the already existing graphs. Thus, by avoiding sequentiality in graph creation and population, we are decreasing the time complexity needed to create the graph database.

At this point of development, given that all the processes are automated, the only limiting factor of the scalability are the financial resources available. Therefore, we recommend caution when using the graph creation script because it can cause unexpectedly high costs depending on the amount of graphs generated, especially if the tier of Cosmos DB is set high.

3 Results

In this section, we will show the results of the presented implementation.

During our planning phase, one of the goals of the project was the **architecture** of a system capable of ingesting medical scientific articles and using them to generate a graph database that can be queried by a user. Therefore, the architecture presented in Figure 3 in the Implementation subsection represents one of the final results. This architecture has been proven to be a workable solution to the proposed problem because was efficient in ingesting the articles and generating the graph database.

The end goal from our proposed solution is represented by the **graph database**, as it can be seen in Figure 1 from the Purpose subsection. Next, we will present a few images representing that obtained graph visualizations. As a disclaimer for the visualizations, we would like to mention that the visualization tool offered by Cosmos DB is showing the relationships between entities and it can also show the labels of the entities. However, at this moment, it is not able to show the labels and values of different relationships. The labels and values of the relationships are available to be retrieved only through graph querying.

In Figure 8, we are showcasing a graph that has been obtained from the Conclusion section from 5 medical articles from Covid-19 area, focusing on the olfactory and gustatory dysfunctions. These articles can be found in the references section at the following numbers: [34], [38], [39], [40], [41].

Figure 8: Graph obtained from the conclusions of 5 articles

When using this visualization tool, the blue circles represent the nodes, the arrows represent the relationships, the red outline of the middle circle represents that that node is the focus of the visualization and the yellow circles around the nodes signify that there are other connections to that node that are not shown in the current visualization.

In comparison with the previous figure (Figure 8) that has been obtained from 5 articles but only by using one section (Conclusion), we are also presenting the resulting graph that was generated by processing a single article, written by Yan et al. ([38]) about self reported olfactory loss after Covid-19 infection, in Figure 10.

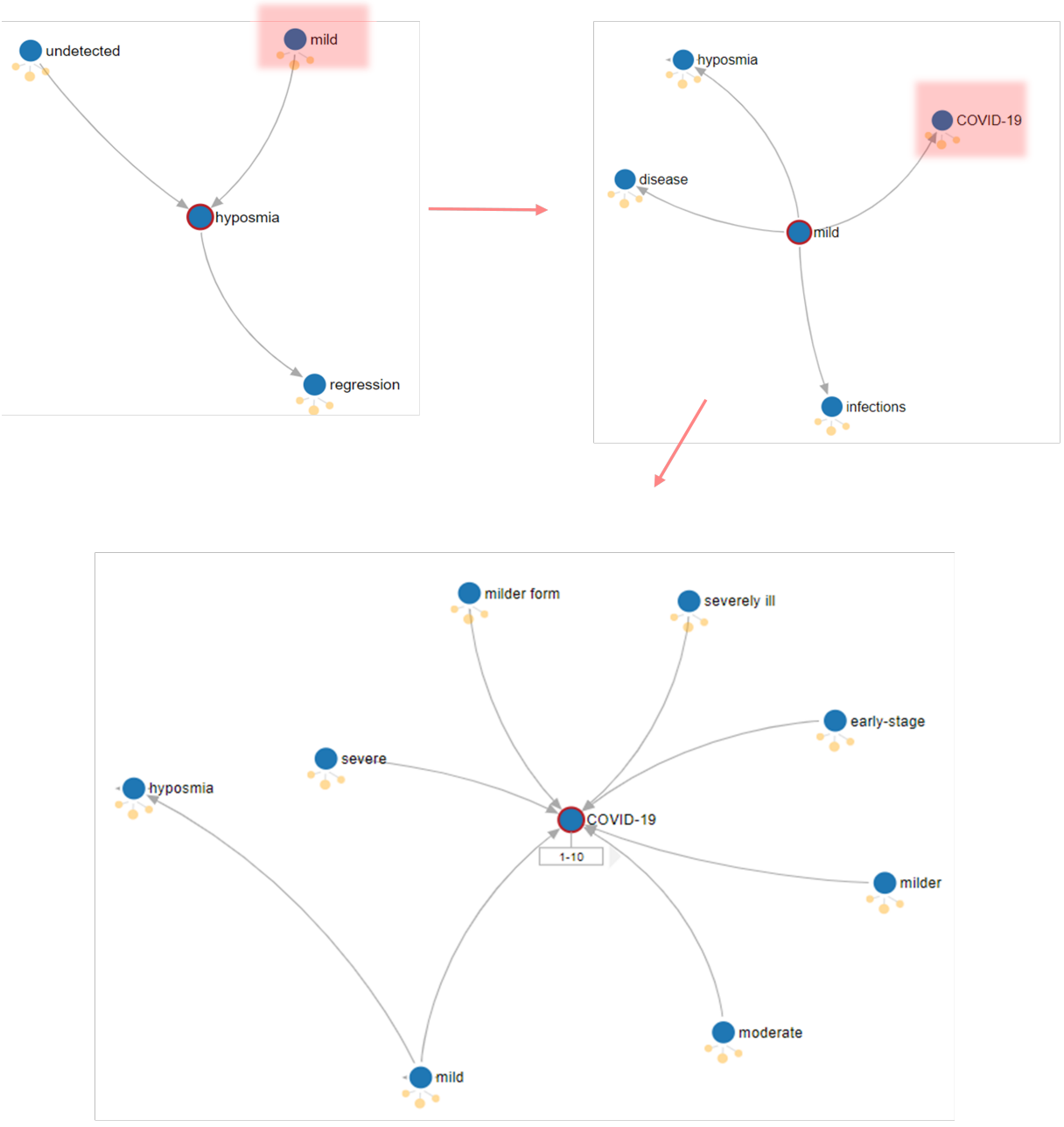

Figure 9: Graph expansion from 1 article

The red faded rectangles are not present in the initial visualization, their role is to help the reader understand what was clicked in order to obtain the next visualization, thus, showcasing the expansion of the graph. The rectangle under the COVID-19 node containing the range 1-10 signifies that this node has more connections that are no shown in the current visualization.

Other examples of graphs that can be expanded obtained from single articles are:

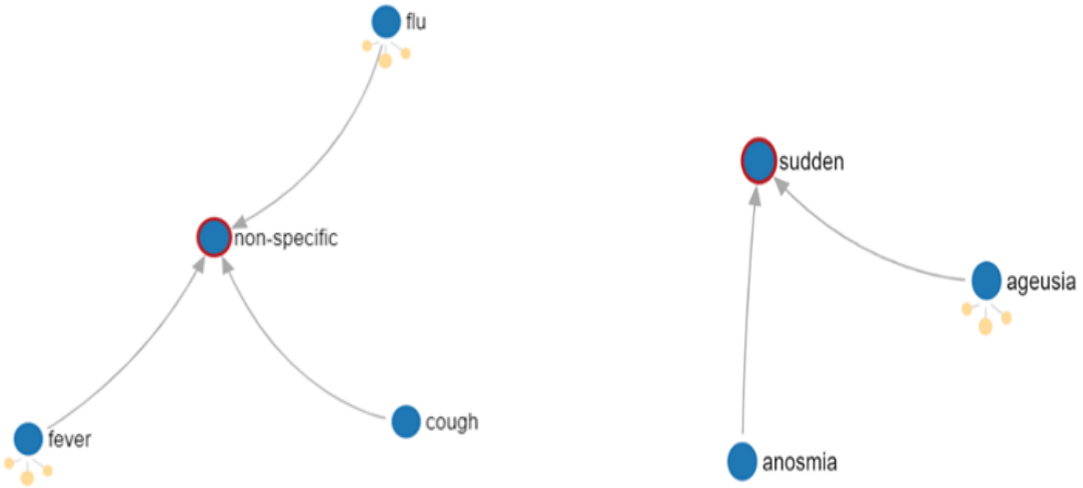

Figure 10: Graphs from one article

However, whilst analysing the obtained graphs, some limitations that could lead to inaccuracies have been observed:

Spelling inconsistencies:
When referring to certain conditions/entities/examinations/etc., same or different authors may use different spelling forms for the same words. In Figure 11 we are presenting some examples of this observation:

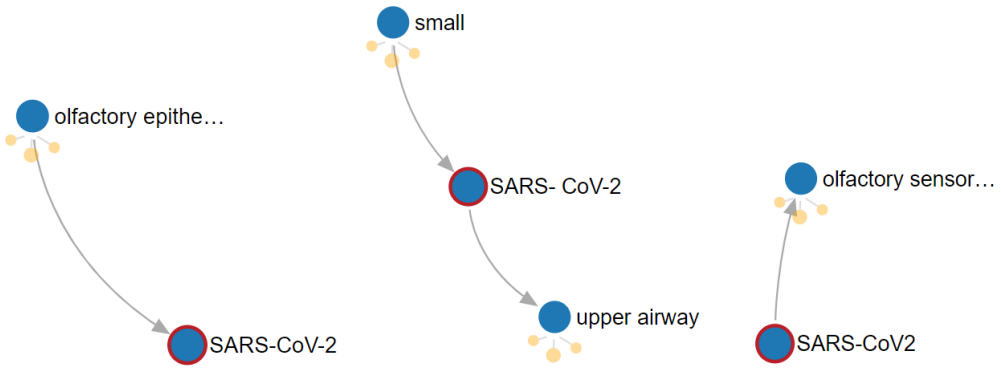

Figure 11: Spelling inconsistencies

Figure 11 depicts examples of different spelling forms of a virus name. According to the World Health Organization, the official name of the virus is "SARS-CoV-2" and it causes the disease called "COVID-19" [42]. In the first graph, the word "SARS-CoV-2" contains hyphens to join acronyms that form the name of the virus. In the second graph, the authors included an empty space in the name "SARS- CoV-2". In the third graph, the word "SARS-CoV2" is used to describe the virus. The Text Analytics for Health will record these entities as separate entities because they will not have identical labels, due to the spelling inconsistencies.

Syllable separation:
Another limitation for our process is the syllable separation. Syllable separation often happens when a certain word does not fit entirely on a certain line. Therefore, it is grammatically correct to separate certain parts of the word by using hyphens. However, when the Cognitive Search reads the content of the files, it will include the hyphen and the space after it in the actual word. Even though the meaning of the word remains unchanged, the identified word will be considered to be a different entity than an entity with the same meaning but without the syllable separation.

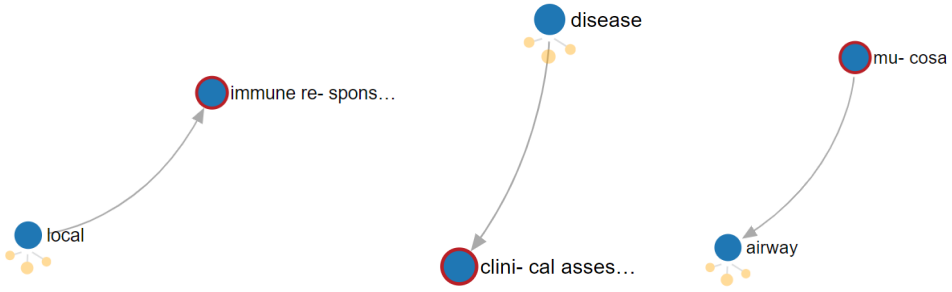

Figure 12: Syllable separation

In Figure 12, the words "immune re- sponse", "clini- cal assessment" and "mu- cosa" are separated in syllables. They will be considered different from "immune response", "clinical assessment" and "mucosa" entities.

Interchangeable words:
Even though the use of interchangeable words is not recommended in scientific writing, it may occur that different words are used to describe similar entities. In Figure 13 we are showcasing some observed examples:

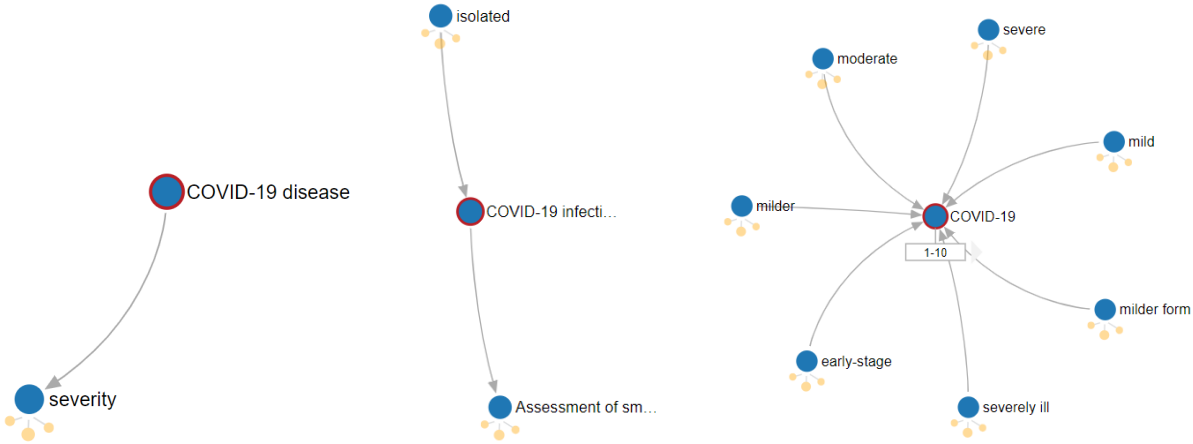

Figure 13: Interchangeable words

According to the World Health Organization, the name of "COVID-19" is used when referring to the disease caused by the SARS-CoV-2 virus [42]. Therefore, it does not need the attribute of disease or words that share the meaning, when using "COVID-19". However, in Figure 13 we can see "COVID-19" being used with interchangeable words, such as "disease" or "infection" but also independently. The meaning of "infection" is used when a pathogen is invading another organism and causing an illness. The meaning of "disease" is used when a part of an organism is not functioning properly and there can be structural or biochemical imbalances. Therefore, even though their meaning are not identical,they sometimes may be used interchangeably.

Abbreviations:
The Text Analytics for Health service has the capacity to identify abbreviations. However, it is important to mention that it has to be clearly indicated in the text body that that entity is an abbreviation, for example "PCR (polymerase chain reaction)". If the acronym "PCR" is used independently, it will be considered as a separate entity. If the abbreviation relationship exists between 2 terms, then the use of either "PCR" or "polymerase chain reaction" will not influence the end result.

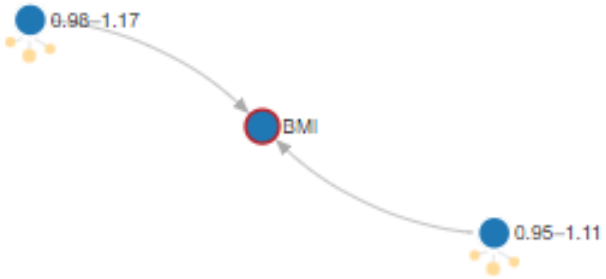

Figure 14: Abbreviations

In Figure 14, we can see the use of "BMI" which stands for "Body Mass Index". However, we can see that there is no relationship to a separate node that states this fact. Thus, when and if another entity "Body Mass Index" will be mentioned in text, it will be independent from this "BMI" node.

Number ranges:
Due to the schema that we have implemented, shown in Figure 7, the numerical values should not be represented by nodes, but by values of already existing relationships. However, it is noticeable in Figure 14 that some numerical values are represented by nodes. This is due to the fact that during the pre-processing phase of the medical entities, ranges such as "0.98-1.17" are not recognized as numerical units.

4 Discussion

Currently, there are many initiatives that strive to bring more automation in the **healthcare industry**. It is often the case that research in this growing industry generates immense amounts of data. In order to be able to retrieve and analyse the novel insights generated by those experiments, one must process that data. This aspect has started to enter the focus of many researchers and developers across the world. It is already known that the impact of improved big data extraction from medical data sources would be significantly positive for, not only providing novel approaches in healthcare but also to facilitate further research [43].

Therefore, we proposed a system that is capable of ingesting medical research articles (published on online authorised repositories), extracting meaningful data regarding the **medical information** within and generating graph databases that reflect that information, that can be further queried by various users. In order to accomplish our proposal, we focused on several research questions. During the development process, we focused on finding solutions for our initial research questions. We designed a system that is capable of ingesting the unstructured data and generate the structured data source that reflects the initial data. In order to do this, we designed a fitting schema that is able to represent the overall connections between the medical entities found in the text.

Within our proposition, we also aimed to answer our third research question, *What are the applications of such a program?*. In order to provide answers for this question, we have identified several possible use cases and actors for such a program:

4.1 Use cases

- **Information verification/ validation**: e.g. Twitter messages.
  Given a text of a tweet, we can try to verify whether the information is accurate/correct. A simple solution for this would be a traversal through the graph to return connections between the entities found in the tweet. If there are results returned, we can assume that there indeed is a connection between the entities mentioned in the tweet. If no results have been returned, then we can say that there haven't been studies that connect the entities mentioned in the tweet. This then may depend on the user whether it can qualify it as misinformation or not.

- **Generic solutions for machine learning applications**: E.g. Estimate rate of success per treatment/ experiment.
  Given a large amount of data used as input to create a persistent graph database, that includes various experiment variables, we can try to predict the rate of success, given certain variables as input. These variables may include size of cohort, treatment applied, condition investigated, duration of treatment, or even more detailed for laboratory research, such as temperature, buffer solutions and/or methods applied. Of course, in order for this to be possible, one must be able to create an algorithm that can estimate the rate of success, depending on the results found in text, but also take into consideration the frequency of occurrence of this success. An important aspect for this use case is to be able to define what "success" means for each experiment.

- **Statistical applications**: E.g. Frequency of similar approaches.
  When goals and plans for a research/experiment are being set, in order to quickly gain insights over the current studies, one might choose to query an existing and updated graph in order to see frequent approaches. This is meant to contribute to the decision of modifying variables or scope of the experiment. For example, one might want to research the impact of Covid-19 in elderly populations. However, this has been a focus point for many teams. Therefore, one could revise the scope of the experiment and focus on less researched areas, such as indirect impact of Covid-19 on child mental development. Of course, this is just a broad overview of a proposed use case.

- **Various industry applications**: E.g. Applications outside the healthcare sphere.
  The designed architecture is simple and flexible throughout different research fields. Even though, for

healthcare applications we are focusing on the Text Analytics for Health service, for other focus areas a custom function may be developed to replace this service in the pipeline. Thus, this system could be applied in various areas, including: geology, chemistry, computing science, psychology etc.

The aforementioned use cases are few proposals that could exist, the remaining options are left open due to their high variation of possibilities. The use cases will be applied by **actors**. Actors that could make use of the system are: academic staff, scientific researchers, doctors, pharmacists, students etc.

The fourth and last research question, *What are the limitations of such a program?*, had the role to maintain transparency during our prototyping process. Throughout our development process we encountered several aspects that require improvement, in order to answer the last research question, we are discussing the discovered limitations:

4.2 Limitations

- **Spelling inconsistencies**: E.g. SARS-CoV-2 or SARS- CoV-2 or SARS-CoV2.
  Spelling inconsistencies lead to the Cognitive Services to identify the medical entities in an inaccurate manner. Results have shown that this has a great impact on the results inaccuracies. This is because when querying the graph to find nodes with a certain label, variations of that label are not taken in consideration. In order to resolve this problem, the content retrieved by the Cognitive Search would have to be processed such that similar words are formatted to be structured identical.

- **Syllable separation**: E.g. Current situation or cur-rent situation.
  The present hyphen that is used to separate certain parts of the word is not impactful for the meaning of the word. Therefore, before undergoing medical entity recognition, the uniformity of words must be ensured, in order to avoid inaccurate results.

- **Interchangeable words**: E.g. disease or infection.
  When considering interchangeable words, we must pay careful attention to context. When referring to specific words such as "SARS-CoV-2" and "2020 coronavirus", they can be interchangeably used. For such situations, a synonym dictionary can be created and used in order to bring more clarity to query results. However, when referring to words with a wider and more complex meaning, such as "disease" or "infection", solutions have to take into consideration the human point of view and intent that was applicable at the time of writing.

- **Abbreviation**: E.g. Polymerase chain reaction or PCR
  Clearly marked abbreviations can have a significant impact on the overall entity identification. Abbreviations allow a user to query the graph by using a wider used term for the same issue. For example, when a user wants to query the graph to assess the overall values of the PCR tests that have been done, the user may simply use the label "PCR" or "polymerase chain reaction". This way, the results that are returned will involve both terms, instead of just the given one. For generally used abbreviations, an abbreviation dictionary can be used. However, if specific or custom made abbreviations are used and not clearly specified, it may lead to inaccurate results.

- **Number ranges**: E.g. 0.98-1.17
  Even though numerical values are stored as the value of the respective relationship, the parser used to assess the type of the value will incorrectly read the range as being not a number. Therefore, this can be solved by adapting the parser to correctly assess ranges.

- **Different article structure**: E.g. Material and Methods / Methods / Methods and Materials
  During the pre-processing phases when the section extraction was developed, we noticed structural inconsistencies in various articles. Namely, one of the most frequent inconsistencies was the different section naming. This led to inaccurate section detection. This aspect has been solved by taking into consideration different options for sections. However, we believe that with a stricter structural standardization for medical scientific articles, inaccuracies resulting from this issue can be significantly reduced.

4.3 Future Work

At the current stage, the proposed and developed system is functional and restricted only by financial resources. However, in order to improve the accuracy of the results, we recommend further work to be done. Among these improvements, we propose the following:

- **Solving limitations**: the previously discussed limitations need to be taken into consideration and solved. We recommend that this step be the first.
- **More data**: in order to obtain more concrete and exhaustive results, we recommend using more articles as an initial data source. Depending on the use case and on the focus area, it is recommended that articles that tackle the focus area be predominantly included.
- **Image processing**: in frequent cases, the medical scientific articles also contain images. In order to extract information from these images, it is possible to include in the Cognitive Search skillset also an AI enrichment skill in order to extract text from images [24]. Furthermore, in order to identify objects or to describe the images, it is possible to create a custom Azure function that includes the Computer Vision Image Analysis service [44]. This function can then be included as an independent custom skill in the existing skillset.
- **Improving medical entity and relationship detection**: given that the Text Analytics for Health service is directly related to the thesaurus provided by the National Library of Medicine, the accuracy with which our system is able to detect the entities and relationships is direction proportional with the accuracy of the provided thesaurus.
- **Use case relation visualization and querying**: as we previously discussed, our system may have applications in different use cases and be manipulated by different actors. However, we must take into consideration the technical expertise of future users. Thus, we propose that this system represent the back-end of a program that is accessed via an user-friendly interface. For example, we propose the graph visualization also show the labels of the relationships and allow the user to directly select them in order to read more information about it. Also, we propose a system to translate human language in runnable Gremlin queries or have readily available queries that do not require technical experience to be modified (e.g. choose variables from a list).

5 Conclusion

The goal of this project was to find an architecture that is capable to ingest unstructured data, in the form of a medical scientific research articles, and through our proposed system, to convert them into a structured data source, such as a graph database. In order to place our system in the sphere of real world applications, we also aimed to identify use cases and actors that can benefit from our system. In order to maintain transparency regarding the development process, we also set the goal to present limitations that might arise.

Cloud technologies have been proved to be very advantageous in the modern world. Not only do these technologies offer more independence in terms of developing, by removing the need of server management but they also offer important scalability and storage capacity. For our proposal, we made use of the Azure Cloud Computing Technologies offered by Microsoft.

Based on our work and research, we created a simple architecture that is able to fulfill the goal of the project. Within the architecture, we also implemented a graph schema that reflects insights about the overall content of an article in a graph data structure. Whilst developing our solution to the proposal, we also identified several use cases and limitations that can be applied to our system.

In conclusion, our work was meant to start building the foundation of a greater system and to start analysing the possibilities and the complications that might arise. We are aware that significant further work has to be done in order to achieve real-life usable results. However, we are confident that more work in this direction will follow.

6 Acknowledgements

I would like to thank my supervisors at the University of Groningen, prof. dr. Dimka Karastoyanova and Mirela Riveni PhD, for being open to my ideas and giving me the opportunity to do this project. I would also like to thank my manager and Chief Technology Officer of Microsoft Netherlands, Dennis Mulder, for supporting me throughout my work, believing in my ideas and making this project possible.

7 References

[1] W. Bank, "World development report 2021: Data for better lives.," 2021. Available at https://www.worldbank.org/en/publication/wdr2021. Accessed: 14.03.2022.

[2] B. Berisha and E. Meziu, "Big data analytics in cloud computing: An overview.," 2021. doi: https://doi.org/10.13140/RG.2.2.26606.95048. *Preprint.*

[3] M. Fire and C. Guestrin, "Over-optimization of academic publishing metrics: Observing goodhart's law in action.," *GigaScience*, vol. 8, 2019. doi: https://doi.org/10.1093/gigascience/giz053.

[4] J. Teixeira da Silva, T. Panagiotis, and M. Erfanmanesh, "Publishing volumes in major databases related to covid-19.," *Scientometrics*, vol. 8, 2021. doi: https://doi.org/10.1007/s11192-020-03675-3.

[5] P. Dutta Pramanik, S. Pal, and M. Mukherjee, *Healthcare Big Data: A Comprehensive Overview*, pp. 72–100. 10 2018. doi: https://doi.org/10.4018/978-1-5225-7071-4.ch004.

[6] H. Koh and G. Tan, "Data mining applications in healthcare," *Journal of healthcare information management : JHIM*, vol. 19, pp. 64–72, 02 2005.

[7] A. Holzinger, M. Dehmer, and I. Jurisica, "Knowledge discovery and interactive data mining in bioinformatics - state-of-the-art, future challenges and research directions," *BMC bioinformatics*, vol. 15, p. I1, 05 2014. doi: https://doi.org/10.1186/1471-2105-15-S6-I1.

[8] A. Adhikari, L. C. Jain, and B. Prasad, "A state-of-the-art review of knowledge discovery in multiple databases," *Journal of Intelligent Systems*, vol. 26, no. 1, pp. 23–34, 2017. doi: https://doi.org/doi:10.1515/jisys-2015-0154.

[9] A. Riel and P. Boonyasopon, "A knowledge mining approach to document classification," *Asian International Journal of Science and Technology in Production and Manufacturing Engineering*, vol. 2, 07 2009.

[10] "Inxight vizcontrols," Available at http://www.baeza.cl/cursos/visual/ix/index.html. Accessed: 21.02.2022.

[11] "Ibm cloud docs - natural language understanding," Available at https://cloud.ibm.com/docs/natural-language-understanding?topic=natural-language-understanding-about. Accessed: 21.02.2022.

[12] "Google cloud docs - natural language understanding," Available at https://cloud.google.com/natural-language/docs/reference/rest. Accessed: 21.02.2022.

[13] "Microsoft azure - cognitive services - language service," Available at https://docs.microsoft.com/en-us/azure/cognitive-services/language-service/. Accessed: 21.02.2022.

[14] A. Riel and P. Boonyasopon, "Knowledge mining in manufacturing and management," *Asian International Journal of Science and Technology in Production and Manufacturing Engineering (AIJSTPME)*, vol. 3, pp. 19–28, 01 2010.

[15] S. Alwidian, H. Bani-Salameh, and A. Alslaity, "Text data mining: a proposed framework and future perspectives," *International Journal of Business Information Systems*, vol. 18, pp. 127–140, 02 2015. doi: https://doi.org/10.1504/IJBIS.2015.067261.

[16] D. Antons, E. Grünwald, P. Cichy, and T. O. Salge, "The application of text mining methods in innovation research: current state, evolution patterns, and development priorities," *R&D Management*, vol. 50, no. 3, pp. 329–351, 2020. doi: https://doi.org/10.1111/radm.12408.

[17] E. Landhuis, "Scientific literature: Information overload.," *Nature*, vol. 535, p. 457–458, 07 2016. doi: https://doi.org/10.1038/nj7612-457a.

[18] A. Gautam and I. Chatterjee, "Big data and cloud computing: A critical review," *International Journal of Operations Research and Information Systems*, vol. 11, pp. 19–38, 07 2020. doi: https://doi.org/10.4018/IJORIS.2020070102.

[19] M. Nasr and S. Ouf, "Cloud computing: The future of big data management," *International Journal of Cloud Applications and Computing (IJCAC)*, vol. 5, pp. 53–61, 04 2015.

[20] A. Mahmood, O. Assim, and W. Younis, "Services of the cloud providers giants," *International Journal of Computational and Mathematical Sciences*, vol. 5, 08 2021.

[21] "What is azure?," Available at https://azure.microsoft.com/en-us/. Accessed: 10.07.2022.

[22] S. Luma-Osmani, F. Idrizi, S. Ademi, and R. Fetai, "Above the clouds: A brief overview of microsoft azure environments and applications," *Natural Science*, vol. 3, pp. 79–88, 03 2018.

[23] "Introduction to azure blob storage," Available at https://docs.microsoft.com/en-us/azure/storage/blobs/storage-blobs-introduction. Accessed: 10.07.2022.

[24] "Search over azure blob storage content," Available at https://docs.microsoft.com/en-us/azure/search/search-blob-storage-integration?toc=%2Fazure%2Fstorage%2Fblobs%2Ftoc.json. Accessed: 10.07.2022.

[25] "Indexers in azure cognitive search," Available at https://docs.microsoft.com/en-us/azure/search/search-indexer-overviewdocument-cracking Accessed: 11.07.2022.

[26] "Introduction to azure functions," Available at https://docs.microsoft.com/en-us/azure/azure-functions/functions-overview Accessed: 11.07.2022.

[27] "What is azure cognitive service for language?," Available at https://docs.microsoft.com/en-us/azure/cognitive-services/language-service/overview Accessed: 11.07.2022.

[28] "What is text analytics for health in azure cognitive service for language?," Available at https://docs.microsoft.com/en-us/azure/cognitive-services/language-service/text-analytics-for-health/overview?tabs=ner Accessed: 11.07.2022.

[29] A. R. Aronson, "Effective mapping of biomedical text to the umls metathesaurus: the metamap program.," *Proceedings. AMIA Symposium*, vol. 3, pp. 17–21, 2001.

[30] "Introduction to gremlin api in azure cosmos db," Available at https://docs.microsoft.com/en-us/azure/cosmos-db/graph/graph-introduction Accessed: 11.07.2022.

[31] A. Kharlamov, G. Gradoselskaya, and S. Dokuka, "Dynamic semantic network analysis of unstructured text corpora.," *International Conference on Analysis of Images, Social Networks and Texts*, vol. 10716, pp. 392–403, 2018. doi: https://doi.org/10.1007/978-3-319-73013-4$_3$6.

[32] "Azure extensions," Available at https://code.visualstudio.com/docs/azure/extensions Accessed: 11.07.2022.

[33] "Content metadata properties used in azure cognitive search," Available at https://docs.microsoft.com/en-us/azure/search/search-blob-metadata-properties Accessed: 12.07.2022.

[34] C. A. Hintschich et al., "Persisting olfactory dysfunction in post-covid-19 is associated with gustatory impairment: Results from chemosensitive testing eight months after the acute infection.," *PLoS One*, vol. 17(3), 03 2022. doi: https://doi.org/10.1371/journal.pone.0265686.

[35] "Supported text analytics for health entity categories," Available at https://docs.microsoft.com/en-us/azure/cognitive-services/language-service/text-analytics-for-health/concepts/health-entity-categories Accessed: 12.07.2022.

[36] "Relation extraction," Available at https://docs.microsoft.com/en-us/azure/cognitive-services/language-service/text-analytics-for-health/concepts/relation-extraction Accessed: 12.07.2022.

[37] "Create a container in azure cosmos db gremlin api," Available at https://docs.microsoft.com/en-us/azure/cosmos-db/graph/how-to-create-container-gremlin Accessed: 12.07.2022.

[38] C. H. Yan, F. Faraji, D. P. Prajapati, B. T. Ostrander, and A. S. DeConde, "Self-reported olfactory loss associates with outpatient clinical course in covid-19.," *International forum of allergy rhinology*, vol. 10(7), pp. 821–831, 2020. doi: https://doi.org/10.1002/alr.22592.

[39] J. R. Lechien et al., "Olfactory and gustatory dysfunctions as a clinical presentation of mild-to-moderate forms of the coronavirus disease (covid-19): a multicenter european study.," *European archives of oto-rhino-laryngology : official journal of the European Federation of Oto-Rhino-Laryngological Societies (EUFOS) : affiliated with the German Society for Oto-Rhino-Laryngology - Head and Neck Surgery*, vol. 277(8), p. 2251–2261, 2020. doi: https://doi.org/10.1007/s00405-020-05965-1.

[40] S. Ahmad, A. Sohail, M. A. Shahid Chishti, M. Aemaz Ur Rehman, and H. Farooq, "How common are taste and smell abnormalities in covid-19? a systematic review and meta-analysis.," *Journal of Taibah University Medical Sciences*, vol. 17(2), pp. 174–185, 2022. doi: https://doi.org/10.1016/j.jtumed.2021.10.009.

[41] E. Mehraeen et al, "Olfactory and gustatory dysfunctions due to the coronavirus disease (covid-19): a review of current evidence.," *European archives of oto-rhino-laryngology : official journal of the European Federation of Oto-Rhino-Laryngological Societies (EUFOS) : affiliated with the German Society for Oto-Rhino-Laryngology - Head and Neck Surgery*, vol. 278(2), pp. 307–312, 2021. doi: https://doi.org/10.1007/s00405-020-06120-6.

[42] "Naming the coronavirus disease (covid-19) and the virus that causes it," Available at https://www.who.int/emergencies/diseases/novel-coronavirus-2019/technical-guidance/naming-the-coronavirus-disease-(covid-2019)-and-the-virus-that-causes-it Accessed: 13.07.2022.

[43] S. Dash, S. K. S. Shakyawar, M. Sharma, and S. Kaushik, "Big data in healthcare: management, analysis and future prospects," *Journal of Big Data*, vol. 6, 2019. doi: https://doi.org/10.1186/s40537-019-0217-0.

[44] "What is image analysis?," Available at https://docs.microsoft.com/en-us/azure/cognitive-services/computer-vision/overview-image-analysis Accessed: 14.07.2022.