Total time: approximately 13 minutes. (A bit less if we remove all the blue writing)

**Slide 1: (Title and supervisors) (~00:21)**

Hello everyone!

Today I will be presenting my bachelors project, “Knowledge Mining in Scientific Articles”.

Firstly, I would like to mention that this project has been possible with the support of the University of Groningen through professor Dimka Karastoyanova and Mirela Riveni and with the support of Microsoft Netherlands, through their CTO, Dennis Mulder

I’ve been working with supervisor x and y and with z from MS

**Slide 2: (Table of contents) (~00:18)**

My presentation today will have the structure of a classical scientific article, starting with an Introduction, presenting the used Methods and then the obtained Results. I will then end the presentation by Discussing aspects of the project and drawing Conclusions from our work.

**Slide 3: (Introduction) (~01:24)**

Have you noticed that in the past years the amount of information that we are given in our daily lives just keeps increasing and increasing? In comparison with 10 years ago, it’s hard to keep track of everything even on paper.

Well, that trend is also reflected in statistical studies that depict this increase. In May 2022, the Statista Research Department declared that the amount of data created, captured and consumed globally will soon exceed 150 zettabytes, in 2024. We can see that in the past years, since the Covid-19 pandemic, there was a surge of data. However, the same source declares that even though the amount of data is high, the data that remains persistent is approximately 2% of the total.

(~Animation – Click~)

To put it in the context of healthcare

Now, looking at the scientific articles in the Covid-19 sphere in 2020, we can see that there was a huge growth starting that March, when there was already a worldwide spread of the virus. There was immense work put into researching the disease and we must start thinking that we must leverage information and research in a digitalized manner. The amount of data nowadays is too vast to be processed by the human mind and time.

Even though Big Data applications are used in various industries, we will be focusing our attention towards extracting and converting unstructured data from scientific articles from Covid-19 research into structured data form, such as a semantic network.

**Slide 4: (Research questions) (~00:23)**

Therefore, we will attempt to answer the following research questions.

What schema can we use that would be able to store the information from a scientific article?

What system can we create that is able to extract unstructured data from articles and convert it into a persistent data form?

What are the possible applications and limitations of such a system?

**Slide 5: (Proposal) (~00:43)**

In order to answer these questions, we propose the following flow.

We need the raw data, represented by scientific articles. These articles will then be ingested by our proposed system that will generate a persistent graph database that can then be queried by the user.

(~Animation – Click~)

It is important to mention here that the purpose of the project is only creating the structured database. Thus, querying by the user is out of scope for this project. The queries of the database are highly dependent on the use case that the user has chosen. This can represent a new project by itself. However, we will attempt to apply a specific use case as a continuation of this project.

~ Reuse as many existing tech as possible

**Slide 6: (Methods) (~00:03)**

Now, we shall look at the technology and methods used.

**Slide 7: (Main technologies) (~00:22)**

For creating this project, we used Python and Visual Studio Code. The reason why we used these 2 is that they are very well supported by the Azure Cloud Computing Services via specific SDKs and extensions. Our project also relies heavily on the Azure Services, therefore it was important to use technologies that work best with them.

**Slide 8: (Architecture) (~00:28)**

This is the architecture that we designed for our system.

As mentioned previously, we first need our raw data, represented by the articles, these will then be stored and processed to extract their content. Then, we will separate the content into specific sections: Introduction, Methods, Results, Discussion and Conclusion. We will then identify health entities and find relationships between them that we can reflect in graphs.

We will now discuss each step in more detail.

**Slide 9: (Article Selection) (~00:06)**

The scientific articles that were used for this project have been manually chosen:

**Slide 10: (~00:14)**

Few examples can be observed here. They were downloaded from free online libraries, such as ScienceDirect or PubMed.

Currently, they tackle the same issue, the olfactory and gustatory dysfunctions that are caused by Covid-19.

**Slide 11: (Article Storage) (~00:05)**

After selecting the required articles, they have been stored in the cloud.

**Slide 12: (~00:15)**

Given that the file type of most articles is PDF, we stored them in the Blob Storage. This is highly beneficial also because the technology that we will use next (Cognitive Search) accepts the Blob Storage as a valid data source.

However, another advantage of having the raw articles in the Blob Storage is that they can “archived” easily after use.

**Slide 13: (Content Extraction) (~00:08)**

After obtaining and storing the PDF documents, we are making use of the Cognitive Search to extract their content.

**Slide 14: (~00:49)**

To obtain the content from the documents we have to create 4 components: the datasource, the skillset, the indexer and the index. The datasource will be represented by the container in the Blob Storage where we stored the documents. The skillset will be represented by built-in functionality that uses *optical character recognition* to process the text from the document. The indexer will be the component that leads the pipeline. It will take the extracted data and the metadata and then store it in specific areas in the index for each document. The index will then be our gateway for accessing the data about the document.

(~Animation – Click~)

For example, if we want to obtain the content and the title of a document, we can run REST API calls for getting the content and title from the index.

**Slide 15: (Section identification) (~00:06)**

After extracting the content of the articles, we must extract the separate sections.

**Slide 16: (~00:18)**

Here we can see an example of how it works. The initial content will be in the form of a json object.

(~Animation – Click ~)

We created a custom function that takes the json object and runs a recursive function over it to identify the required sections. The identification is done using a regex function.

The output of the function is simply the specific section that was required. In this case, we have extracted the Conclusion from an article. In this situation, the section\_name will be “Conclusion” and the “next\_section\_name” recursively considered to have different options, until the right one is found. This is needed because there are frequent inconsistencies when naming the sections within an article.

What does the fct do

**Slide 17: (Medical entity and relationship recognition) (~00:11)**

After extracting the specific section, we apply the Text Analytics for Health Cognitive Service.

**Slide 18: (~00:44)**

This Service can identify medical entities and their relationship in the given text. It is using a Unified Medical Language System created by the National Library of Medicine from the US. It also contains a large biomedical metathesaurus that is organized by concept or meaning. The metathesaurus is also capable of identifying useful relationships between concepts and preserves the meanings from 200 different vocabularies.

Thus, we can observe that in our conclusion, some relations found between certain entities. For example, from the first relation we can see that ExaminationFindsCondition means that there is a unidirectional relationship between the Condition “hypogeusia” and the Examination type “psychophysically”. Furthermore, we can see that the ValueOfExamination of “psychophysically” is 32 and the UnitOfExamination of the same examination is “%”. This relationship and its attributes can also be noticed highlighted in the text.

We use the entities and relationships discovered in these steps to populate our graphs.

**Slide 19: (Graph creation and population) (~00:08)**

Now, we have entities and relationships. So, we can make use of the Gremlin API to populate the graphs.

**Slide 20: (~01:01)**

The graphs will respect the following schema:

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.

(~Animation – Click~)

An interesting aspect that we managed to solve at this step was the lack of a feature within the python SDK for the graphs in CosmosDB. Even if the SDK permitted the population of a graph, it wasn’t facilitating the creation of a graph. This was only available via the C# SDK, PowerShell or Azure CLI. Therefore, we created a workaround that creates an extra process that runs a PowerShell command to create a new graph.

**Slide 21: (Results) (~00:07)**

Now, let’s see what we managed to produce with the proposed system.

**Slide 22: Graphs (~01:00)**

In this slide, we can see 3 examples of graphs that we obtained.

As a short disclaimer, the graphs that we are seeing now are produced by Cosmos DB directly, which does not allow the visualization of the edge’s labels and attributes.

The graphs from the *left side* are graphs obtained from individual articles and their overall content. It is noticeable that the number of connections between entities is small.

On the *right side*, we can see a graph that was obtained by using data from the Conclusions of 5 articles. It is clear that the more data we use, the larger and more exhaustive the graph will become.

The graph representation allows a good and quick understanding of the information because it offers a more structured and summarized visualization. More than that, the obtained graph form makes it easier to query the data found in scientific articles.

**Slide : Discussion (~00:06)**

Now that we have seen the obtained results, let’s move to the Discussion section.

**Slide : Use Cases (~00:54)**

The obtained semantic network can be used in a multitude of applications, here we discuss several:

*Information verification or validation*: for example, given a text of a tweet, we can try to verify whether the information is accurate. 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.

*Machine learning applications* where we could estimate the rate of success per treatment/ experiment also depending on the conditions.

*Statistical applications* where we could identify similar approaches to another treatment/ experiment.

Also, use cases for a system similar to our project do not have to be limited to the healthcare industry. With specific changes, this architecture can be applied to other fields, such as engineering or geology.

**Slide : Limitations (~01:47)**

During the development of our system, we discovered several limitations that we weren’t aware of during the planning phase.

We noticed that scientific articles, even though they follow a very general structure, are still very inconsistent.

Due to spelling inconsistencies, the Cognitive Service responsible for identifying medical entities recognized the term Covid-19 as a different term than Covid- 19, which can lead to inaccurate results.

Also, even though the Cognitive Service has the capability of identifying abbreviations, it can only do so when they are specifically mentioned and explained in text. However, if they are independent, they will be again considered as individual entities. This problem could be solved by introducing *synonym dictionaries* when identifying entities. These dictionaries would have to be as complete as possible to prevent duplication.

Word separation has also proved to be a big issue. When authors separate words in syllables they use the minus sign which is not relevant for the meaning of the word. However, attempts to remove the minus can lead to inaccurate results for other terms where the minus is relevant, for example chemical names.

Another issue that we encountered is the inconsistent article structure. Scientific articles do not follow a standard section naming convention. This means that there are articles where the same section is called in a different way: Methods or Material and Methods, for example.

These are the most common limitations that we identified that could lead to inaccurate results.

**Slide : Conclusion (~00:35)**

To conclude, during this project we managed to answer the research questions that we proposed at the beginning. We created and proved an architecture that can convert unstructured data, such as scientific articles, in a structured data form, such as a semantic network. We also provided and explained the chosen schema for creating the network.

Finally, we identified possible use cases for our system and the most common limitations that can arise.

**Slide : Thank you!**

Thank you for your attention!

