# Search Mechanism focused on Diseases, Symptoms and Treatments

André Esteves, Francisco Filipe, Helena Montenegro, Juliana Marques
*Information Description, Storage and Retrieval DAPI*
*Master in Informatics and Computing Engineering MIEIC*
*Faculty of Engineering, University of Porto FEUP*
Porto, Portugal
{up201606673, up201604601, up201604184, up201605568}@fe.up.pt

***Abstract*—Nowadays, there are several search engines publicly available for people to use in a global scale, containing information about almost anything we can think about. However, when it comes to health matters, these search engines might not be the best choice. Very frequently there will be documents from unreliable sources containing misleading information that might needlessly worry the average user. To tackle this problem, we have implemented and tested information retrieval systems and an ontology based on semantic web, which we built based on a dataset centered on diseases, symptoms and treatments. The preparation of the dataset involved the selection of data sources, data retrieval, cleaning and enrichment and data characterization. Afterwards, we developed and compared three information retrieval systems: a simple system, a system with improvements to the indexing process and a system with improvements to both the indexing and querying processes, which led us to conclude that the quality of the respective results is limited by the capacity of the query to express an information need and that both the indexing and querying processes need to be carefully developed in order to achieve a system of quality. Finally, we developed an ontology to represent the knowledge gathered in the dataset.**

***Index Terms*—Information Retrieval, Diseases, Treatments, Symptoms**

## I. Introduction

Search engines such as Google are used by billions of people on a daily basis to retrieve information about various subjects, including health matters. When a person is worried about a symptom and searches for it on the web, many documents are retrieved, including some from unreliable sources, such as Yahoo answers where someone asks a question and any person can answer, for example. The average person does not have the knowledge to discern between reliable and unreliable sources of information, easily believing any information that comes up even when the sources are not specified, just because the content seems legitimate. When it comes to health, unreliable and exaggerated information can lead to anxiety and panic in patients.

There are search engines that specialize in obtaining health information, such as MedWorm [1], however, these engines focus on retrieving biomedical texts and articles published in the health scientific community, which are useful for health specialists, but not for the average user, due to difficulties in understanding the content of the documents and unfamiliarity with the technical terms used in these. A normal user needs a simple and straightforward mechanism that, rather than focusing on advances in the medical community or overly specific documents, shows information about diseases and respective symptoms and treatments.

As there is currently no search mechanism that allows a person to easily obtain reliable information about health matters, the goal of this project is to develop one, focusing on diseases, treatments and symptoms.

This article serves to describe the first steps in this project, which regards the preparation of the data set and the development of an information retrieval system. Section II describes the pipeline implemented for the process of preparing and characterizing the data. Section III is focused on the system results and retrieval tasks. Section IV describes the process of development and comparison between the information retrieval systems that we created. Section V describes the development of an ontology, in the context of semantic web, culminating in a comparison between relational databases, information retrieval and semantic web. Finally, we have Section VI which summarizes the main findings in this work.

## II. Data Preparation

For the process of preparing the data, we developed a pipeline Fig. 1 divided in 5 processes: data collection, storage, cleaning, enrichment and characterization.

### A. Data Collection

The data set was obtained from two sources: Wikidata [2] and Wikipedia [3].

*1) **Wikidata**:* Wikidata is a large database with structured data, containing information of various subjects, including diseases. This data can be copied, modified, and distributed, even for commercial purposes, without needing permission, under the license "Creative Commons Public Domain Dedication 1.0" [4]. However, information from Wikidata can be modified by anyone, without needing to be verifiable against authoritative sources and is, therefore, unreliable. Furthermore, after analyzing the obtained data we arrived at the conclusion that it is severely lacking, with many diseases that do not have information about symptoms or even health specialties.

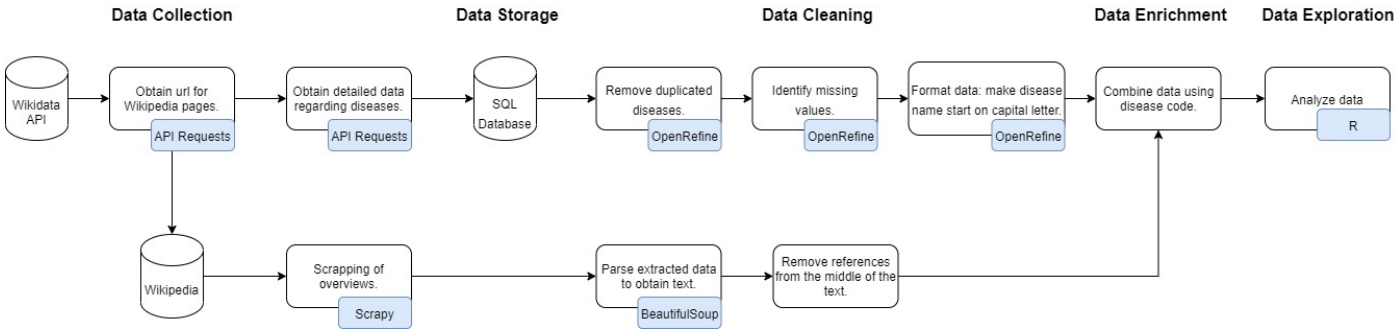


Fig. 1. Data pipeline diagram.

*2) **Wikipedia**:* Wikipedia is an encyclopedia that contains unstructured textual data which is free to be shared and adapted for any purpose, including commercial purposes, as long as appropriate credit is given, under the license "Creative Commons Attribution-ShareAlike 3.0 Unported" [5], [6]. Although anyone can edit Wikipedia pages, Wikipedia has a policy that states that any alterations or additions must be verifiable against an authoritative source, which makes it reliable as a data source [7].

Data from Wikidata was obtained in JSON format through its API. To retrieve diseases, we fetched objects that were instances of the class "disease" on Wikidata. While doing this we realized that there were several duplicated diseases, for example, Alzheimer's disease had more than 10 entries on Wikidata, each of them numbered and that were instances of "Alzheimer's disease". We noticed that only the original diseases, which are not instances of other diseases, had Wikipedia pages, which led us to only fetch diseases that were associated to Wikipedia pages. Along with information about the diseases and respective characteristics, we also obtained the URL for the respective Wikipedia page, which allowed us to obtain overviews of diseases, symptoms, treatments and of the other classes, through crawling and scraping, using the tool Scrapy [8].

The use of Wikidata, which possesses unreliable and incomplete data, is a limitation that risks the reliability of the search mechanism being developed.

### *B. Retrieved Data*

With the previous data sources, we retrieved data about diseases and many other concepts associated to the disease, such as their medical specialties, symptoms, possible treatments, causes and drugs used in their treatment. The conceptual model (Fig. 2) represents the entities of interest and the relations between them.

The Disease class is the core of our data set and is related to all the other classes of the domain. This class stores all the relevant information about a disease and it's attributes are:

- **code** - a unique attribute that identifies a disease.
- **name** - the disease's name.
- **description** - a small text that describes the disease.

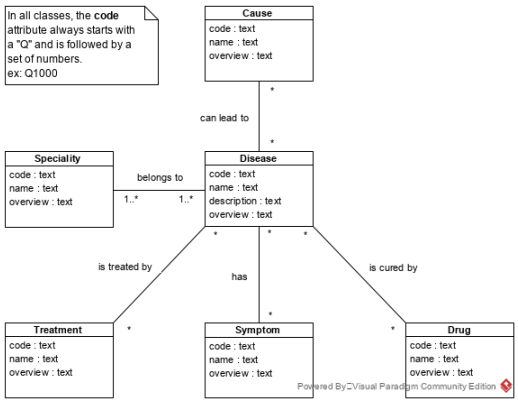


Fig. 2. Conceptual Model.

- **overview** - a detailed text about the disease extracted from Wikipedia.

Furthermore, our domain has 5 more classes, all related to the Disease class.

- **Cause** - Class that represents all the causes that can lead to disease.
- **Speciality** - Class that represents all the specialities associated with a disease. All the diseases have at least one speciality.
- **Treatment**- Class that represents all the treatments that can be used to treat a disease.
- **Symptom** - Class that represents all the symptoms a disease can have.
- **Drug**- Class that represents all the drugs that can be used to cure a disease.

All these classes have 3 attributes, which are:

- **code** - a unique attribute that identifies the instance.
- **name** - the instance's name.
- **overview** - a detailed text about the instance extracted from Wikipedia.

### *C. Data Storage*

We stored the data collected in a local SQL database, using Microsoft's Azure SQL Database [9].

### D. Data Cleaning

The data extracted from Wikidata contained a lot of diseases without any connections to other classes, such as symptoms, treatments, and so on. We made the decision to remove all diseases that contain less than two connections to other classes. After that, we removed all symptoms, treatments, health specialties, causes and drugs that were not connected to any remaining disease. This was achieved by performing DELETE statements on the database in Azure Data Studio [10]. We also capitalized the first letter of the name of diseases, symptoms, treatments, causes, health specialties and drugs, using UPDATE statements on the database. By making the code of the diseases retrieved from Wikidata unique, we ensured that there were no duplicate values in the database.

The scraped data obtained from Wikipedia was in HTML format. The data was parsed using the tool BeautifulSoup [11] to extract the text. We then proceeded to remove special characters and references, using Python.

### E. Data Enrichment

The data from Wikidata was enriched with data from Wikipedia. During the data collection process, we made sure that all the diseases collected in Wikidata had a Wikipedia page. The data was easily joined, using the code of diseases and of the other classes, with UPDATE statements on the SQL database.

### F. Data Characterization

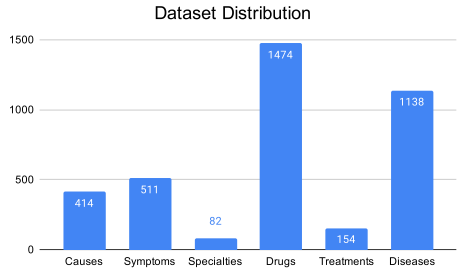


Fig. 3. Bar graph with data set distribution.

The data set distribution (Fig. 3) shows that the largest classes present are **Drugs** (approximately 40%) and **Diseases** (approximately 30%). The smallest class is the one that concerns **Specialities** (approximately 0,02%).

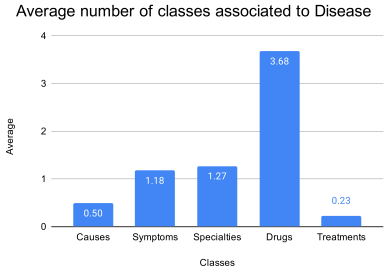


Fig. 4. Bar graph with average number of classes associated to disease.

The average number of classes associated with a Disease graph Fig. 4 shows that there are less causes and treatments associated to diseases than the remaining classes. There is more information about drugs on diseases, since a disease has, on average, around 3.7 drugs.

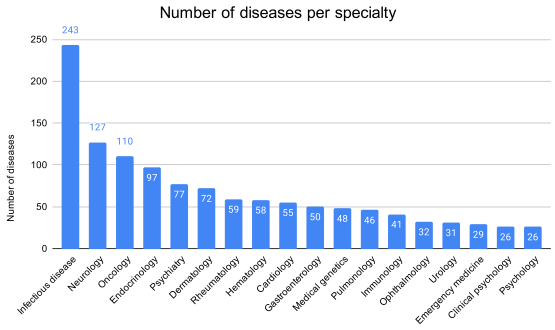


Fig. 5. Bar graph with number of diseases per specialty.

When it comes to the number of diseases organized by specialty (Fig. 5), it's possible to see that the specialty associated with the most diseases is **infectious diseases** (243 diseases), and the specialty with the least number of diseases is **psychology** (26 diseases).

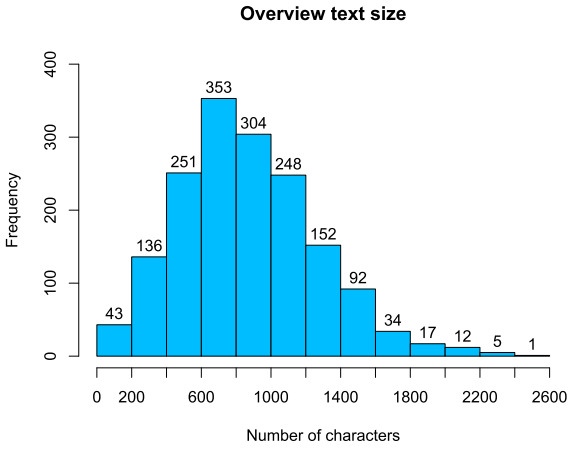


Fig. 6. Histogram with overview text size.

When it comes to the size of the overview texts (Fig. 6) of the documents, most of the overviews have about 600-1,200 characters (0,56 %). About 10% of the overviews have more than 1,400 characters and only 0,3% have less than 200 characters.

## III. Information Needs

### A. System Results

There are 1,648 documents in the system, distributed in three classes: Disease, Treatment and Symptom, as shown in Fig. 7.

*1) **Disease**:* The document Disease contains a small overview of two paragraphs about the disease and a list of symptoms, treatments, causes, drugs and health specialties associated with the disease, if these exist. The user will be able to search for the disease by keywords associated with it, present in the overview, and by symptoms, treatments, causes and health specialties.

*2) **Treatment**:* The document Treatment contains an overview of the treatment and a list of diseases that can be cured by the treatment. A user will be able to search for the treatment by the name of a disease or by a keyword present in the name or overview of the treatment. Not all treatments in the system will be a document, only the ones that contain overviews.

*3) **Symptom**:* The document Symptom contains an overview of the symptom and a list of diseases that can have the symptom. This document, similarly to the treatments, should appear when a user searches by a disease or a keyword present in the name or overview of the symptom. Not all symptoms in the system will be a document, only the ones that contain overviews.

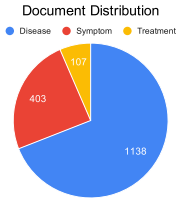


Fig. 7. Pie chart with document distribution

### B. Retrieval Tasks

The expected return value of the retrieval tasks are documents which represent diseases, treatments and symptoms as these are the focus of the project. These documents will be retrieved based on the title's content and in the respective overview's content.

Some possible search tasks include:

- Retrieve diseases that fall under a specific medical specialty.
- Retrieve diseases that contain a certain symptom.
- Retrieve symptoms associated with a certain disease.
- Retrieve treatments associated with a disease.

Examples of queries in this system may include:

- Which medical specialty should I visit when I feel eye irritation?
- What disease may I have if I have cough?
- Which treatments are used for cancer?
- What medication should I take for headaches?

## IV. Information Retrieval

We developed three information retrieval (IR) systems using Apache Solr [12]. System A is a simple version, System B contains improvements to the indexing process and System C contains improvements to the querying process. Using Apache Solr, we have defined 3 cores, one for each system, where each core contains the three types of documents mentioned in the previous section. As the first step of the indexing process, relative to data acquisition, we indexed these documents as JSON files, which were obtained by extracting them from the SQL database that we had previously defined.

To evaluate and compare the three systems we developed two information needs, following the TREC [13] structure:

- **Title**: Drugs used for symptoms
  **Description**: What drugs are used to cure cough?
  **Narrative**: Relevant documents describe diseases that contain the symptom, in this case, cough, and that mention drugs used. Documents about the symptom are only relevant if there is a mention of types of drugs that are used to tackle the symptom.
  **Queries**: [drugs treat cough] and [drugs cough]
- **Title**: Symptoms associated with a medical specialty
  **Description**: What medical specialty should I visit when I have cough?
  **Narrative**: Relevant documents describe diseases that contain the symptom, in this case, cough, each of these documents has a medical specialty associated.
  **Queries**: [medical specialties related cough] and [medical specialties cough]

We developed a schema (Fig. 15), that contains the following fields, common to all three systems:

- Fields that belong to the document Disease:
  - **disease_description**: small textual description of the disease.
  - **disease_name**: the disease's name.
  - **disease_overview**: large textual description of the disease.
  - **drug**: list of drugs that are used to treat the patient.
  - **cause**: list of causes that can cause the disease.
  - **specialty**: list of medical specialties associated with the disease.
  - **symptom**: list of symptoms associated with the disease.
  - **treatment**: list of treatments used to treat the patient.
- Fields that belong to the document Symptom:
  - **symptom_name**: the symptom's name.
  - **symptom_overview**: large textual description of the symptom.
  - **symptom_disease**: list of diseases that are associated to the symptom.
- Fields that belong to the document Treatment:
  - **treatment_name**: the treatment's name.
  - **treatment_overview**: large textual description of the treatment.
  - **treatment_disease**: list of diseases that are treated by the treatment.

None of the fields were marked as required to enable the indexing of the three different documents in the same core. One example of results obtained from in these information

retrieval systems, where the three types of documents indexed were retrieved, can be found in Fig. 14.

### A. System A

In system A, no changes were made to the indexing or querying processes. All of the fields in the documents used Solr's field type *text_general*, which, during the indexing process, obtains the indexes by separating the tokens using white spaces and punctuation as delimiters and turning each token into lowercase. Stop words were ignored. By analyzing the first 20 results for each query mentioned in the information needs, we obtained the results presented in Table I. The mean average precision values calculated for each information need are:

- MAP(Drugs used for symptoms) = 0.40
- MAP(Symptom associated with medical specialty) = 0.42

The mean average precision for the overall system, taking into consideration both information needs is:

- MAP(System A) = 0.41

TABLE I
RESULTS FOR SYSTEM A

| Query | Documents Retrieved | Relevant Docs Retrieved | Average Precision |
|---|---|---|---|
| [drugs treat cough] | 146 | 5 | 0.26 |
| [drugs cough] | 118 | 10 | 0.54 |
| [medical specialties related cough] | 365 | 7 | 0.23 |
| [medical specialties cough] | 279 | 15 | 0.61 |

Looking at these results, we can conclude that the query [drugs cough] is better at retrieving relevant documents than the query [drugs treat cough], since it retrieved more relevant documents and it has a higher value of average precision. The presence of the word "treat" in the first query resulted in the retrieval of more documents about treatments, which were completely unrelated to the topic, worsening the results of this query. For the second information need, the query [medical specialties cough] obtained better results than [medical specialties related cough], with a higher average precision and higher number of relevant documents that were retrieved.

We can conclude from these results that the query used to express the information need influences the results. The results provided by the information retrieval system are limited by the capacity of the query of expressing the information need. To create better queries, we should remove tokens that do not provide useful information about the information needs, like "related" or "treat".

The graph on Fig. 8 presents the interpolated precision-recall curve obtained from the results of each query in this system. For the recall, we considered the number of relevant documents on the whole collection to be the maximum number of relevant documents that were retrieved for each information need on all systems. In this graph we can see that, in the query [drugs cough], corresponding to the red line, the relevant documents were ranked higher, since precision is 1 for lower levels of recall. The query [medical specialties cough], represented by the green line, was able to retrieve a higher percentage of relevant documents, since its precision only drops to 0 on very high recall values, around 90%.

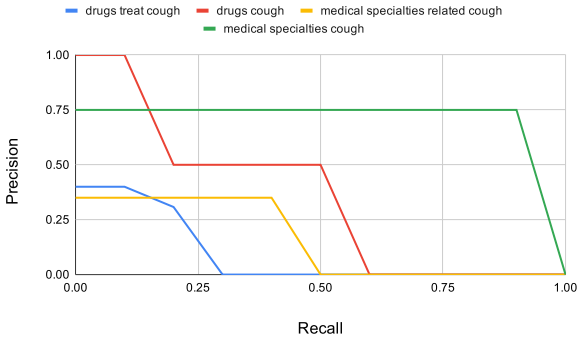


Fig. 8. Interpolated Precision-Recall Graph for System A's results.

### B. System B: Indexing Process

In system B, we developed improvements to the indexing process with the implementation of a customized field type (Fig. 16) which was applied to the previous schema in all of the fields that were previously defined as *text_general*. With the new field type, the indexes are obtained through a process of tokenization, which separates tokens using white spaces and punctuation as delimiters, followed by the operations:

- Ignore stop words, which are words with no discriminating power, that appear in every document of the collection;
- Turn tokens into lowercase;
- Remove possessive ('s);
- Stemming using the Porter Stemming Algorithm.

By analyzing the first 20 results for each query mentioned in the information needs, we obtained the results presented in Table II. The mean average precision values calculated for each of the information needs are:

- MAP(Drugs used for symptoms) = 0.56
- MAP(Symptom associated with medical specialty) = 0.31
- MAP(System B) = 0.435

The mean average precision for the overall system, taking into consideration both information needs is:

- MAP(System B) = 0.435

TABLE II
RESULTS FOR SYSTEM B

| Query | Documents Retrieved | Relevant Docs Retrieved | Average Precision |
|---|---|---|---|
| [drugs treat cough] | 253 | 7 | 0.44 |
| [drugs cough] | 163 | 8 | 0.67 |
| [medical specialties related cough] | 473 | 6 | 0.20 |
| [medical specialties cough] | 392 | 13 | 0.43 |

Regarding the first information need, compared with the previous system, this system was capable of obtaining more relevant results with a higher ranking, as can be seen by the higher average precision values for both queries, as well as the mean average precision value. In the second information need, however, the results worsened, with lower mean average precision value. This may be explained by the fact that the word ”medical”, after passing through the stemming step, becomes ”medic”, which is not necessarily associated with medical specialties. We observed that this word alone retrieves 329 documents in this system. These results support the idea that the quality of the information system is limited by the quality of the query.

In both information needs, as expected, the system retrieved a higher number of documents.

The graph on Fig. 9, clearly shows that the queries for the first information need, represented as the red and blue lines, resulted in a higher ranking of relevant documents, starting with precision at 1 for recall values of 0, which means that the first result obtained on both these queries was relevant. The queries for the second information need, which are the green and yellow lines, had relevant documents more evenly distributed in the results list.

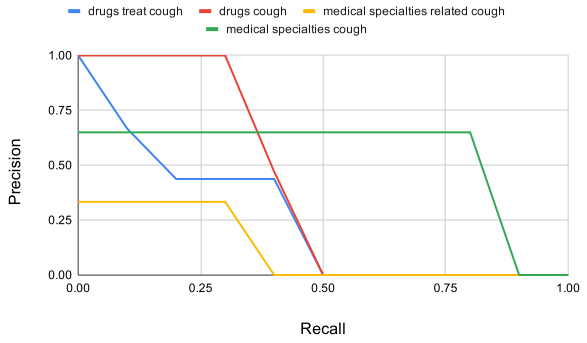


Fig. 9. Interpolated Precision-Recall Graph for System B’s results.

Overall, although this system obtained a slightly higher mean average precision value (0.435) than the previous system (0.41), the difference is not very significant. As such, it’s not possible to assert with certainty which system is better.

### C. System C: Querying Process

In system C, we developed improvements to the querying process by adding weights to boost some of the fields. These improvements were added on top of the system B, which means that, in comparison to system A, system C has improvements to both the indexing and the querying processes.

To achieve this purpose of improving the querying process, we used the eDismax query parser [14]. We applied two different sets of weights to the fields, to compare and see which set provides better results. The weights are expressed in Table III. The remaining fields that do not appear on this table were given weight values of 1.

TABLE III
WEIGHTS APPLIED TO FIELDS FOR QUERY ”DRUGS USED FOR SYMPTOMS”

| Field | Weight 1 | Weight 2 |
|---|---|---|
| disease_name | 5 | 5 |
| symptom_name | 3 | 3 |
| treatment_name | 3 | 1 |
| treatment_disease | 2 | 1 |
| symptom_disease | 2 | 1 |
| specialty | 2 | 1 |
| symptom | 4 | 3 |
| treatment | 4 | 1 |

In the first set of weights, we gave more importance to the diseases’ name and equal importance to both symptoms’ and treatments’ names. In the second set, we gave more importance to symptoms than to treatments, expecting to achieve better results since both the information needs we developed are related to symptoms. We obtained the results present in Table IV. Note that the number of documents retrieved is not present in this table since it is the same as the number of documents retrieved for these queries on System B. System C changes the weights of the fields, which only changes the ranking of the retrieved documents.

TABLE IV
RESULTS FOR SYSTEM C

| Query | Set of weights | Relevant Docs Retrieved | Average Precision |
|---|---|---|---|
| [drugs treat cough] | 1 | 12 | 0.56 |
| [drugs cough] | 1 | 12 | 0.54 |
| [drugs treat cough] | 2 | 14 | 0.67 |
| [drugs cough] | 2 | 17 | 0.63 |
| [medical specialties related cough] | 1 | 11 | 0.41 |
| [medical specialties cough] | 1 | 13 | 0.54 |
| [medical specialties related cough] | 2 | 12 | 0.59 |
| [medical specialties cough] | 2 | 16 | 0.63 |

The mean average precision values obtained for each of the set of weights is:

- MAP(Set of weights 1) = 0.51
- MAP(Set of weights 2) = 0.63

As we can see, the second set of weights, which gave more importance to symptoms rather than treatments, obtained better results, with a higher number of relevant documents retrieved and a higher mean average precision. As such, we will use this set of weights to compare this system with the other systems. In both information needs, system C was able to obtain more relevant documents when compared to the previous systems, and achieved higher average precision values.

The graph on Fig. 10 shows the interpolated precision-recall values obtained for the queries, using the second set of weights. For the queries [drugs cough], in red, and [medical

specialties cough], in green, the curve is a straight line, where precision never drops to 0, which means that in this system, these queries were able to obtain the maximum of relevant documents of all the systems. Even in the other queries, we can see that the precision only drops to 0 on very high values of recall, surpassing 75%. These results lead to the conclusion that system C has a higher capacity to retrieve relevant documents than the other systems.

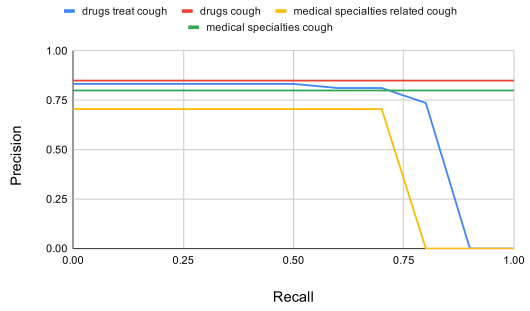


Fig. 10. Interpolated Precision-Recall Graph for System C's results.

### D. Comparison between systems

To compare between the systems, we calculated the average of the precision-recall curves for each information need and system, to evaluate the capacity of each system to obtain relevant documents.

Fig. 11, represents the interpolated precision-recall curves for each system in regards to the first information need. In this graph, we can see that System A provided the worse results, with lower values of precision for each level of recall. System B, which has an improved indexing process, shows an improvement in results when compared with the previous system, which can be seen by the larger area under the curve for this system. System B possesses an higher capacity to rank relevant documents higher than System A. System C, which contains an improved querying process on top of the improved indexing process seen on System B, has the highest capacity to retrieve more relevant documents, since its curve only drops to zero once the recall is higher than 0.8. Therefore, System C provided the best results out of all the systems.

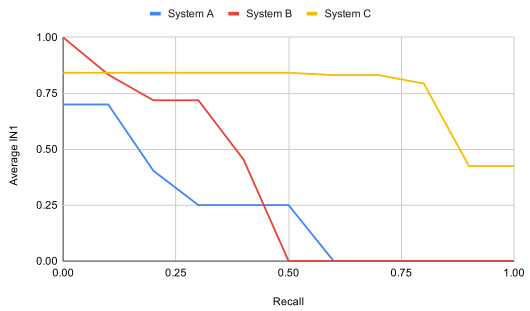


Fig. 11. Average Interpolated Precision-Recall Graph for information need 1.

Fig. 12 represents the interpolated precision-recall curves for each system in regards to the second information need. In this graph, we can see that System A provided better results than System B, since it possesses higher values of precision for all levels of recall. System C provided the best results, with even higher precision values.

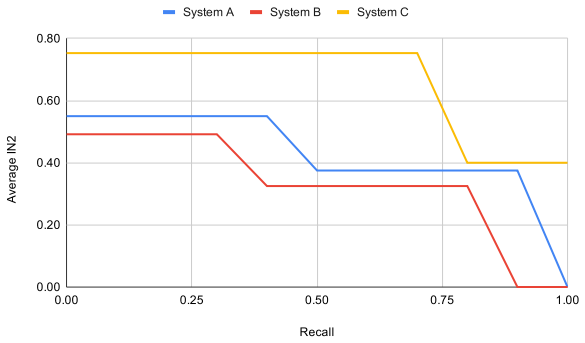


Fig. 12. Average Interpolated Precision-Recall Graph for information need 2.

We can conclude from both these graphs that the system which had improvements to both the indexing and querying processes, system C, obtained the best results. As such, to develop an information retrieval system with quality, there is a need to develop the indexing process, in order to retrieve as many documents that may be relevant as possible, as well as the querying process, in order to obtain the most relevant documents of the ones retrieved higher in the ranking.

## V. Semantic Web

Semantic Web aims to create a shared space of data, with meaningful information, that can be navigated and understood by both humans and machines, as opposed to relational databases, which focus more on the structure of the data rather than its meaning. It relies on the concept of Linked Data, which relies on the use of Uniform Resource Identifiers (URI) to identify and link resources, allowing to find new information when we look up a resource.

In this context, ontologies can be used to represent knowledge. In this section we will review existing ontologies in the health domain as well as present the ontology that we have developed.

### A. Review of Existing Ontologies

Since this work's theme revolves around medical terms, our research for ontologies of interest was based in an open repository for biomedical ontologies called Bioportal [15]. We found a few ontologies that are related to diseases, such as the Human Disease Ontology [16], or symptoms, such as the Symptom Ontology [17]. Out of all the ontologies we found, the one that is the most complete is the MedlinePlus Health Topics ontology [18].

MedlinePlus is a service provided by the National Library of Medicine, which belongs to the National Institutes of Health

(NIH), in the United States of America, developed with the goal of providing reliable information for public access. This service provides an ontology called Health Topics, which contains information about symptoms, treatments, diseases and any other topic related to health.

This ontology is kept up to date, having been updated very recently, on December 7th, 2020. It's also very inclusive, containing all information that is relevant regarding health, from diseases to symptoms and causes, to all the various types of treatments, including drug treatment, therapy, and surgeries. Overall, the ontology is very well organized, with the diseases grouped according to the body parts they affect, which makes it easy to search for information.

The MedlinePlus ontology is currently being used in the project "SIFR: Semantic Indexing of French Biomedical Data Resources" [19], which investigates the scientific and technical challenges in building ontology-based services to facilitate indexing, mining, and retrieval of French biomedical data.

### B. Our Ontology

We have built our ontology using *Protégé* [20]. In this ontology we have declared the following classes:

- Disease
- Treatment
- Symptom
- Specialty
- Drug
- Cause

To describe all the possible relationships between objects, we created this set of Object Properties:

- hasSymptoms
  - Domain: Disease
  - Range: Symptom
- hasTreatment
  - Domain: Disease
  - Range: Treatment
- hasDrug
  - Domain: Disease
  - Range: Drug
- hasCause
  - Domain: Disease
  - Range: Cause
- hasSpecialty
  - Domain: Disease
  - Range: Specialty

To describe attributes that belong to individuals, we created this set of Data Properties:

- Description
  - Domain: Disease
  - Range: xsd:string
- Overview
  - Domain: All the declared classes (Cause, Disease, Drug, Specialty, Symptom and Treatment)
  - Range: xsd:string

We have also applied existential restrictions to the class Disease, which state that an individual of this class has at least one symptom and one medical specialty, through the keyword "some", as defined in Figure 13. We have also included these restrictions expressed in RDF/XML syntax in Figure 17. We define these existential restrictions as equivalent classes, to declare that not only all diseases have symptoms, but also anything that has symptoms is a disease, which could potentially infer that an individual that is a cause and that has symptoms is also a disease. To exemplify, COVID-19 is a cause of pneumonia and can be introduced into the system as part of the class Cause associated to symptoms such as cough. In this case, the system could infer that, since it has symptoms, COVID-19 is a disease. Note that this example is not present in our dataset.


Fig. 13. Restrictions applied to Disease class.

A visual overview of the ontology can be seen in Figure 18. This visual overview was obtained with the VOWL plugin [21].

### C. Populating the Ontology

To populate the ontology, we have used *Cellfie* [22], a plugin for *Protégé*, to create OWL ontologies from spreadsheets. To do so, we exported the dataset from the SQL database as excel files, so that we could import the different objects as individuals to the ontology. Additionally, we have altered the excel file about diseases to capture all relationships between classes in the data, so that we could import them as object property axioms in the ontology.

The process of importing the data using the *Cellfie* plugin was very slow, leading the program to crash after some time, since there was a high amount of over 20,000 axioms being created. As such, we decided to shorten the amount of data we would import into our ontology. In this process, we chose a set of symptoms ("cough", "diarrhea" and "fever") and imported only the diseases that contained a connection to at least one of these symptoms. We have also removed all the data from treatments, causes, drugs and health specialties that were not associated to the imported diseases.

After populating the ontology, we have used the reasoner *HermiT* [23] to determine whether the ontology is consistent and to infer further information about the ontology. We have found that the ontology was already consistent, as no errors were presented by the reasoner, and the reasoner did not present any inferences.

### D. Queries

In order to evaluate the usability of the ontology, we developed a set of queries, based on the retrieval tasks previously

defined for this domain, using SPARQL [24] as the query language. We started by defining the prefixes present in 6, that we would need to execute queries on the ontology. We decided to name the prefix to the developed ontology as "dis".

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-
    syntax-ns#>\\
PREFIX owl: <http://www.w3.org/2002/07/owl#>\\
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-
    schema#>\\
PREFIX xsd: <http://www.w3.org/2001/XMLSchema
    #>\\
PREFIX dis: <http://www.semanticweb.org/
    helenamontenegro/ontologies/2020/11/
    disease-ontology#>
```

Listing 1. Query Prefixes

In the following sections we will describe in detail each of the queries developed.

*1) Query 1: Retrieve diseases by symptom:* One of the most basic user needs that the ontology can fulfill is to retrieve the diseases that a patient might have if he feels a certain symptom. To exemplify this query, we have decided to use "Cough" as a symptom and we ordered the results by the number of symptoms that a disease has, in order to assign a degree of severity to the disease, assuming that the quantity of symptoms is correlated to the severity of the disease.

```
SELECT (STR(?dis_name) as ?disease_name) (STR(
    COUNT(?symptoms)) AS ?number_symptoms)
WHERE {
      ?disease dis:hasSymptoms ?symptom .
      ?disease rdfs:label ?dis_name .
      ?symptom rdfs:label "Cough"^^xsd:string
          .
      ?disease dis:hasSymptoms ?symptoms

}
GROUP BY ?dis_name
ORDER BY ?number_symptoms
```

Listing 2. Query 1 - Retrieve diseases by symptom

This query achieved results present on Table V.

*2) Query 2: Retrieve drugs used to treat a symptom:* This information need has been used in this paper, in the context of evaluating the information retrieval systems developed in Section IV. The goal is that, given a symptom, retrieve all the drugs that can be used to cure diseases that are associated to the symptom. To exemplify this information need, we used "Headache" as the symptom. In order to not overload the user with information, we have only retrieved the first 6 results.

```
SELECT (STR(?drugs_name) AS ?drug)
WHERE {
      ?disease dis:hasSymptoms ?symptom .
      ?disease dis:hasDrugs ?drugs .
      ?symptom rdfs:label "Headache"^^xsd:
          string .
      ?drugs rdfs:label ?drugs_name
}
LIMIT 6
```

Listing 3. Query 2 - Retrieve drugs used to treat a symptom

TABLE V
RESULTS FOR QUERY 1

| Disease | Number of Symptoms |
|---|---|
| Asthma | 1 |
| Croup | 1 |
| Dirofilariasis | 2 |
| Hantavirus pulmonary syndrome | 2 |
| Pneumocystis pneumonia | 2 |
| Lung cancer | 3 |
| Tuberculosis | 3 |
| Measles | 3 |
| Legionnaires disease | 3 |
| Mercury poisoning | 4 |
| Pertussis | 4 |
| Plague | 4 |
| Ornithosis | 5 |
| Ehrlichiosis | 5 |
| Lassa fever | 5 |
| Behçets disease | 5 |

This query resulted in the a list of drugs, as can be seen in Table VI. In these results we can see one advantage when compared to the results obtained in the information retrieval systems: we can retrieve meaningful information, instead of retrieving full documents where we still need to read the whole document to find the pieces of information that we're interested in.

TABLE VI
RESULTS FOR QUERY 2

| Drug |
|---|
| Chloramphenicol |
| Doxycycline |
| Tetracycline |
| D-penicillamine |
| Troleandomycin |
| Levofloxacin |

*3) Query 3: Fetch symptoms associated with a disease:* This query presents the symptoms derived from a disease. In this example the disease used was "Lassa fever".

```
SELECT (STR(?symptom_name) AS ?symptoms)
WHERE {
      ?disease dis:hasSymptoms ?symptom .
      ?disease rdfs:label "Lassa fever"^^xsd:
          string .
      ?symptom rdfs:label ?symptom_name .
}
```

Listing 4. Query 3 - Fetch symptoms associated with a disease

Table VII presents the symptoms associated with the disease "Lassa fever". In our ontology this disease has 6 symptoms.

TABLE VII
RESULTS FOR QUERY 3

| Symptom |
|---|
| Fever |
| Vomiting |
| Nausea |
| Cough |
| Diarrhea |
| Fever |

*4) Query 4: Fetch diseases under a specific medical specialty:* The goal of this query is to retrieve all diseases that are associated with a specific medical speciality. Along with each disease, it also presents the number of treatments that it has. In this example, we used "Infectious disease" as the medical speciality. To limit the number of information, only 8 results were retrieved.

```
SELECT (STR(?dis_name) AS ?diseases) (STR(
    COUNT(?treatment)) AS ?number_treatments)
WHERE {
      ?disease dis:hasSpecialty ?specialty .
      ?disease rdfs:label ?dis_name .
      ?specialty rdfs:label "Infectious
          disease"^^xsd:string .
      OPTIONAL {?disease dis:hasTreatment ?
          treatment}
}
GROUP BY ?dis_name
ORDER BY DESC (?number_treatments)
LIMIT 8
```

Listing 5. Query 4 - Fetch diseases under a specific medical specialty

Table VIII presents the results of this query, which consist in a column of diseases, displaying the name of the disease, and a column with the number of treatments present in descending order.

TABLE VIII
RESULTS FOR QUERY 4

| Disease | Number of Treatments |
|---|---|
| Amebiasis | 3 |
| Dirofilariasis | 2 |
| Diphtheria | 2 |
| HIV/AIDS | 1 |
| Rubella | 1 |
| Measles | 1 |
| Japanese encephalitis | 1 |
| Measles | 1 |
| Dracunculiasis | 0 |

*5) Query 5: Fetch treatments by disease with symptom 'X':* The aim of this query is to retrieve only the treatments of diseases that contain a certain symptom. As an example, the symptom used in this query is "Fever".

```
SELECT (STR(?dis_name) as ?disease_name) (STR
    (?treat_name) as ?treatments)
WHERE {
      ?disease dis:hasSymptoms ?symptom .
      ?disease rdfs:label ?dis_name .
      ?symptom rdfs:label "Fever"^^xsd:string
          .
      ?disease dis:hasTreatment ?treatment.
      ?treatment rdfs:label ?treat_name
}
ORDER BY ?dis_name
```

Listing 6. Query 5 - Retch treatments by disease with symptom 'X'

The results displayed on table IX consist of two columns that detail all the treatments found for diseases containing "fever" as a symptom in our ontology. The first column contains the disease's name while the second presents the treatments, as they are the goal of this query. To better group the results, they were ordered by disease name.

TABLE IX
RESULTS FOR QUERY 5

| Disease | Treatment |
|---|---|
| Acute lymphocytic leukemia | Immunotherapy |
| Acute lymphocytic leukemia | Hematopoietic stem cell transplantation |
| Acute lymphocytic leukemia | Blood transfusion |
| Behçets disease | Corticosteroid |
| Behçets disease | TNF inhibitor |
| Behçets disease | Interferon alpha 2 |
| Diphtheria | Tracheotomy |
| Diphtheria | Antibiotics |
| Dirofilariasis | Tetracycline antibiotic |
| Dirofilariasis | Surgery |
| HIV/AIDS | Management of HIV/AIDS |
| Japanese encephalitis | Supportive care |
| Measles | Supportive care |
| Poliomyelitis | Supportive care |
| Polymyositis | Intravenous Immunoglobulin |
| Polymyositis | Corticosteroid |
| Rubella | Supportive care |

*6) Query 6: Fetch top 10 most used drugs to treat diseases:* This query shows the top 10 most used drugs in our ontology. To obtain such measure, the usage of a drug was defined as the number of times it appears throughout the different diseases that our ontology contains.

```
SELECT (STR(?drug_name) as ?drugs) (STR(COUNT
    (*)) as ?times_used)
WHERE {
      ?disease dis:hasDrugs ?drug .
      ?drug rdfs:label ?drug_name
}
GROUP BY ?drug_name
ORDER BY DESC(?times_used)
```

```
LIMIT 10
```

Listing 7. Query 6 - Fetch top 10 most used drugs to treat diseases

Table X presents the top 10 most used drugs in our ontology by descending order. They're divided into two columns, with the first being the name of the drug, and the second the number of times it is used (number of diseases containing the drug). The least used drug present in the top 10, Tinidazole, is used to treat 2 different diseases, while Azithromycin, the table leader, appears as a drug capable of aiding in the treatment of 5 different diseases.

TABLE X
RESULTS FOR QUERY 6

| Disease | Treatment |
|---|---|
| Azithromycin | 5 |
| Metronidazole | 4 |
| Erythromycin | 3 |
| Doxycycline | 3 |
| Atovaquone | 2 |
| Ivermectin | 2 |
| Troleandomycin | 2 |
| Paromomycin | 2 |
| Ciprofloxacin | 2 |
| Tinidazole | 2 |

*7) Query 7: Fetch symptoms and overviews of a disease derived from a specific cause:* This query retrieves the symptoms of diseases that derive from a specific cause. Along with the symptom, an overview is also presented. As an example, the cause used in this query is "Babesia".

```
SELECT (STR(?symp_name) as ?symptoms) (STR(?
    symp_overview) as ?overviews)
WHERE {
      ?disease dis:hasCauses ?cause .
      ?cause rdfs:label "Babesia"^^xsd:string
          .
      ?disease dis:hasSymptoms ?symptom .
      ?symptom rdfs:label ?symp_name .
      OPTIONAL { ?symptom dis:overview ?
          symp_overview}
}
```

Listing 8. Query 7 - Fetch symptoms and overviews of a disease derived from a specific cause

Table results on table XI consist of a column of symptoms and other with the respective overview. All symptoms displayed are connected to diseases that have "Babesia" as cause, according to our ontology.

*8) Query 8: Fetch treatments from diseases with more than 'X' drugs:* This query retrieves the treatments of diseases with more or equal to a specific number of drugs associated. In this query, the number of drugs used is 2.

```
SELECT (STR(?treat_name) as ?treatments) (STR
    (?dis_name) as ?diseases) (STR(COUNT(?drug
    )) as ?num_drugs)
WHERE {
      ?disease dis:hasDrugs ?drug .
      ?disease rdfs:label ?dis_name .
      ?disease dis:hasTreatment ?treatment .
      ?treatment rdfs:label ?treat_name
}
GROUP BY ?dis_name ?treat_name
HAVING(COUNT(?drug) >= 2)
ORDER BY DESC(?num_drugs)
```

Listing 9. Query 8 - Fetch treatments from diseases with more than 2 drugs

TABLE XI
RESULTS FOR QUERY 7

| Symptom | Overview |
|---|---|
| Fatigue | Fatigue is a feeling of tiredness.It may be sudden or gradual in onset. It is a normal phenomenon if it follows prolonged physical or mental activity, and resolves completely with rest. However, it may be a symptom... |
| Nausea | Nausea is a diffuse sensation of unease and discomfort, often perceived as an urge to vomit. While not painful, it can be a debilitating symptom if prolonged, and has been described as placing discomfort on the chest, ... |
| Headache | Headache is the symptom of pain in the face, head, orneck. It can occur as a migraine, tension-type headache, or cluster headache.Frequent headaches can affect relationships and employment.There is also an increased risk ... |
| Vomiting | Vomiting(also known as puking,throwing up, barfing, emesis, among other names) is the involuntary, forceful expulsion of the contents of one's stomach through the mouth and sometimes the nose ... |
| Fever | Fever, also referred to as pyrexia, is defined as having a temperature above the normal range due to an increase in the body's temperature set point.There is not a single agreed-upon upper limit for normal temperature ... |

Table XII presents 11 treatments of our ontology. The results are divided into three columns, with the first being the name of the treatment, followed by the associated disease name, and the last column presents the number of drugs used. The results are displayed in descending order by the number of drugs.

TABLE XII
RESULTS FOR QUERY 8

| Treatment | Disease | Number of Drugs |
|---|---|---|
| Management of HIV/AIDS | HIV/AIDS | 3 |
| Tracheotomy | Diphtheria | 3 |
| Medication | Amebiasis | 3 |
| Intravenous Immunoglobulin | Polymyositis | 3 |
| Corticosteroid | Polymyositis | 3 |
| Blood transfusion | Amebiasis | 3 |
| Surgical operation | Amebiasis | 3 |
| Antibiotics | Diphtheria | 3 |
| Corticosteroid | Behçets disease | 2 |
| TNF inhibitor | Behçets disease | 2 |
| Interferon alpha 2 | Behçets disease | 2 |

### E. Discussion

The ontology developed, when compared with the relational database that we had previously defined, possesses many advantages:

- **Data integration**: The standard form that organizes data in triplets makes it easy to integrate data from different domains.
- **Inference**: The use of a declarative language such as OWL allows to extract inferences from the data, based on the ontology's restrictions, and check for inconsistencies through the use of reasoners, such as the *HermiT* reasoner [23] used in our ontology.

The main disadvantage in the tools used to build the ontology, *Protégé* [20], when compared to the ones used to build the information system, *Apache Solr* [12] and the ones used to build the initial relational database, *Azure SQL Database* [9], is that *Protégé* is limited in regards to the volume of data that it can handle efficiently, leading us to shorten the dataset that was used. *Apache Solr* is much more efficient, with quick indexing and querying processes.

When comparing the ontology developed with the information retrieval system developed, we need to take into consideration that Information Retrieval (IR) and Semantic Web have different purposes: IR uses keywords to retrieve documents that might be relevant to answer an information need, and semantic web aims to represent useful data that can be processed by machines.

Some advantages of the querying process of ontologies when compared to IR systems are:

- Querying ontologies using SPARQL allows to capture relations between different resources, providing results that contain useful information from different sources, while in IR systems the results of the querying process are documents that are somehow related with the query.
- In ontologies, the results of a query are fields that directly contain the information that the user needs, while in IR systems, the results are documents that need to be processed, or read, to find the pieces of information that the user really needs, as can be seen in the queries to retrieve drugs used to treat symptoms developed for both systems.

The disadvantage in ontologies when compared to IR systems is that querying requires to know languages such as *SPARQL* [24], in order to obtain the pieces of information that we want, which, for a regular user is not intuitive at all. This means that there is a need to develop an interface with predefined queries for the average user to have access to information in the semantic web. In IR systems, the user can simply type words that are related to a topic and obtain results.

Ideally, to achieve the purpose of having information that can be processed by machines and that can be obtained by humans in an intuitive way, information retrieval systems should be built on top of the semantic web.

Regarding the applications of this ontology in the health domain, it can be used for an app that allows a normal user to look up basic information about diseases, symptoms and treatments.

## VI. Conclusion

The purpose of this project was to create a search mechanism that allows a person to easily obtain reliable information about health matters, focusing on diseases, treatments and symptoms.

In the first stage we successfully populated an Azure SQL database by combining information retrieved from both the Wikidata and Wikipedia. This information was later cleaned and enriched resulting in a total of 1,648 documents distributed amongst Diseases, Treatments and Symptoms.

In the second stage, we developed and compared three information retrieval systems that operated over the documents that we have previously defined. In this stage, we arrived at the conclusions that the quality of the results of an information retrieval system is limited by the capacity of the query to express the information need and that to develop an information retrieval system with quality it's needed to develop improvements to both the indexing and querying processes.

In the third and final state of the project, we developed an ontology to represent the knowledge in the dataset, delving into the domain of Semantic Web. We arrived at the conclusions that ontologies have a lot of potential when it comes to the inference of knowledge from data and to verify the consistency of a dataset and that they allow to easily integrate information between different domains, thanks to the standardized way in which the data is represented.

To provide information that can be processed by machines and that can be obtained intuitively by humans, the research in Semantic Web should work towards integrating information retrieval with Semantic Web.

## References


[1] J. Hanan, "Medworm: Medical search engine and rss news," 2016, last accessed 22 November 2020. [Online]. Available: https://medworm.com

[2] "Wikidata. wikidata: Introduction," last accessed 22 November 2020. [Online]. Available: https://www.wikidata.org/wiki/Wikidata:Introduction

[3] "Wikipedia," 2001, last accessed 22 November 2020. [Online]. Available: https://en.wikipedia.org/wiki/Main_Page

[4] "Creative commons - cc0 1.0 universal (cc0 1.0) public domain dedication," last accessed 22 November 2020. [Online]. Available: https://creativecommons.org/publicdomain/zero/1.0/

[5] "Wikipedia. wikipedia: Text of creative commons attribution sharealike 3.0 unported license," last accessed 22 November 2020. [Online]. Available: https://en.wikipedia.org/wiki/Wikipedia:Text_of_Creative_Commons_Attribution-ShareAlike_3.0_Unported_License

[6] "Creative commons - attribution sharealike 3.0 unported license," last accessed 22 November 2020. [Online]. Available: https://creativecommons.org/licenses/by-sa/3.0/

[7] "Wikipedia. wikipedia: Verifiability," last accessed 22 November 2020. [Online]. Available: https://en.wikipedia.org/wiki/Wikipedia:Verifiability

[8] "Scrapy," last accessed 22 November 2020. [Online]. Available: https://scrapy.org/

[9] "Azure sql database," last accessed 22 November 2020. [Online]. Available: https://azure.microsoft.com/en-us/services/sql-database/

[10] "Microsoft. azure data studio," last accessed 22 November 2020. [Online]. Available: https://docs.microsoft.com/en-us/sql/azure-data-studio/what-is-azure-data-studio?view=sql-server-ver15

[11] L. Richardson, “Beautiful soup,” 2004, last accessed 22 November 2020. [Online]. Available: https://www.crummy.com/software/BeautifulSoup/
[12] “Apache solr,” last accessed 22 November 2020. [Online]. Available: https://lucene.apache.org/solr/
[13] “Text retrieval conference (trec),” last accessed 22 November 2020. [Online]. Available: https://trec.nist.gov/
[14] “Apache solr. the extended dismax (edismax) query parser,” last accessed 25 November 2020. [Online]. Available: https://lucene.apache.org/solr/guide/8_7/the-extended-dismax-query-parser.html
[15] P. L. Whetzel, N. F. Noy, N. H. Shah, P. R. Alexander, C. Nyulas, T. Tudorache, and M. A. Musen, “Bioportal: enhanced functionality via new web services from the national center for biomedical ontology to access and use ontologies in software applications,” 2011, last accessed 10 December 2020. [Online]. Available: https://bioportal.bioontology.org/
[16] L. Schriml, “Disease ontology,” last accessed 31 December 2020. [Online]. Available: https://disease-ontology.org/
[17] W. Kibbe, S. Fuentes, and S. Arabandi, “Symptom ontology,” last accessed 31 December 2020. [Online]. Available: http://symptomontologywiki.igs.umaryland.edu/mediawiki/index.php/Main_Page
[18] “Medlineplus: Health topics,” last accessed 10 December 2020. [Online]. Available: https://medlineplus.gov/healthtopics.html
[19] “Sifr project,” last accessed 10 December 2020. [Online]. Available: https://sifr.mystrikingly.com/
[20] M. A. Musen, “The protégé project: A look back and a look forward,” *AI Matters*, vol. 1, no. 4, p. 4–12, Jun. 2015. [Online]. Available: https://doi.org/10.1145/2757001.2757003
[21] S. Lohmann, S. Negru, and D. Bold, “Vowl plugin for protégé,” 2014, last accessed 4 January 2020. [Online]. Available: http://vowl.visualdataweb.org/protegevowl.html
[22] J. Hardi, M. O’Connor, C. Nyulas, and T. Tudorache, “Protégé plugin for creating owl ontologies from spreadsheets,” 2018, last accessed 31 December 2020. [Online]. Available: https://github.com/protegeproject/cellfie-plugin
[23] B. Motik, R. Shearer, B. Glimm, G. Stoilos, and I. Horrocks, “Hermit owl reasoner,” last accessed 6 January 2021. [Online]. Available: http://www.hermit-reasoner.com/
[24] “Sparql query language for rdf,” last accessed 4 January 2020. [Online]. Available: https://www.w3.org/TR/rdf-sparql-query/

# APPENDIX

## INFORMATION RETRIEVAL FIGURES


```
"responseHeader":{
  "status":0,
  "QTime":0,
  "params":{
    "q":"treat cough",
    "rows":"3",
    "_":"1606652377831"}},
"response":{"numFound":92,"start":0,"numFoundExact":true,"docs":[
    {
      "treatment_overview":["Light therapyorphototherapy, classically r
      "treatment_name":["Light therapy"],
      "treatment_disease":["Jet lag"],
      "id":"eeaaccb2-92de-4ffd-9729-7ef43a5a57f9",
      "_version_":1683084862044504067},
    {
      "disease_overview":["Whooping cough, also known aspertussisor the
      "specialty":["Infectious disease"],
      "drug":["Erythromycin",
        "Demeclocycline",
        "Guaifenesin"],
      "disease_name":["Pertussis"],
      "symptom":["Runny nose",
        "Vomiting",
        "Cough",
        "Fever",
        "Apnea",
        "Fatigue"],
      "cause":["Bordetella pertussis"],
      "disease_description":["human disease caused by the bacteria Bord
      "treatment":["Antibiotic"],
      "id":"ed1d4855-c739-422b-9959-e548bb5cc32b",
      "_version_":1683084861747757057},
    {
      "symptom_name":["Barking cough"],
      "symptom_disease":["Diphtheria"],
      "id":"f4f041ab-b643-4900-bc7d-d4377e3da571",
      "_version_":1683084861985783819}]
}}
```


Fig. 14. Example of results of information retrieval systems.

```
<field name="cause" type="text_general" uninvertible="true" multiValued="true" indexed="true" stored="true"/>
<field name="disease_description" type="text_general" uninvertible="true" indexed="true" stored="true"/>
<field name="disease_name" type="text_general" uninvertible="true" indexed="true" stored="true"/>
<field name="disease_overview" type="text_general" uninvertible="true" indexed="true" stored="true"/>
<field name="drug" type="text_general" uninvertible="true" multiValued="true" indexed="true" stored="true"/>
<field name="id" type="string" multiValued="false" indexed="true" required="true" stored="true"/>
<field name="specialty" type="text_general" uninvertible="true" multiValued="true" indexed="true" stored="true"/>
<field name="symptom" type="text_general" uninvertible="true" multiValued="true" indexed="true" stored="true"/>
<field name="symptom_disease" type="text_general" uninvertible="true" multiValued="true" indexed="true" stored="true"/>
<field name="symptom_name" type="text_general" uninvertible="true" indexed="true" stored="true"/>
<field name="symptom_overview" type="text_general" uninvertible="true" indexed="true" stored="true"/>
<field name="treatment" type="text_general" uninvertible="true" multiValued="true" indexed="true" stored="true"/>
<field name="treatment_disease" type="text_general" uninvertible="true" multiValued="true" indexed="true" stored="true"/>
<field name="treatment_name" type="text_general" uninvertible="true" indexed="true" stored="true"/>
<field name="treatment_overview" type="text_general" uninvertible="true" indexed="true" stored="true"/>
```

Fig. 15. System A schema.

```
<fieldType name="custom_text_field" class="solr.TextField">
  <analyzer>
    <tokenizer class="solr.StandardTokenizerFactory"/>
    <filter class="solr.StopFilterFactory" words="lang/stopwords_en.txt" ignoreCase="true"/>
    <filter class="solr.LowerCaseFilterFactory"/>
    <filter class="solr.EnglishPossessiveFilterFactory"/>
    <filter class="solr.PorterStemFilterFactory"/>
  </analyzer>
</fieldType>
```

Fig. 16. System B custom field type.

```
<owl:Class rdf:about="http://www.semanticweb.org/helenamontenegro/ontologies/2020/11/disease-ontology#Disease">
    <owl:equivalentClass>
        <owl:Restriction>
            <owl:onProperty rdf:resource="http://www.semanticweb.org/helenamontenegro/ontologies/2020/11/disease-ontology#hasSpecialty"/>
            <owl:someValuesFrom rdf:resource="http://www.semanticweb.org/helenamontenegro/ontologies/2020/11/disease-ontology#Specialty"/>
        </owl:Restriction>
    </owl:equivalentClass>
    <owl:equivalentClass>
        <owl:Restriction>
            <owl:onProperty rdf:resource="http://www.semanticweb.org/helenamontenegro/ontologies/2020/11/disease-ontology#hasSymptoms"/>
            <owl:someValuesFrom rdf:resource="http://www.semanticweb.org/helenamontenegro/ontologies/2020/11/disease-ontology#Symptom"/>
        </owl:Restriction>
    </owl:equivalentClass>
</owl:Class>
```

Fig. 17. Restrictions applied to Disease class in RDF/XML syntax.

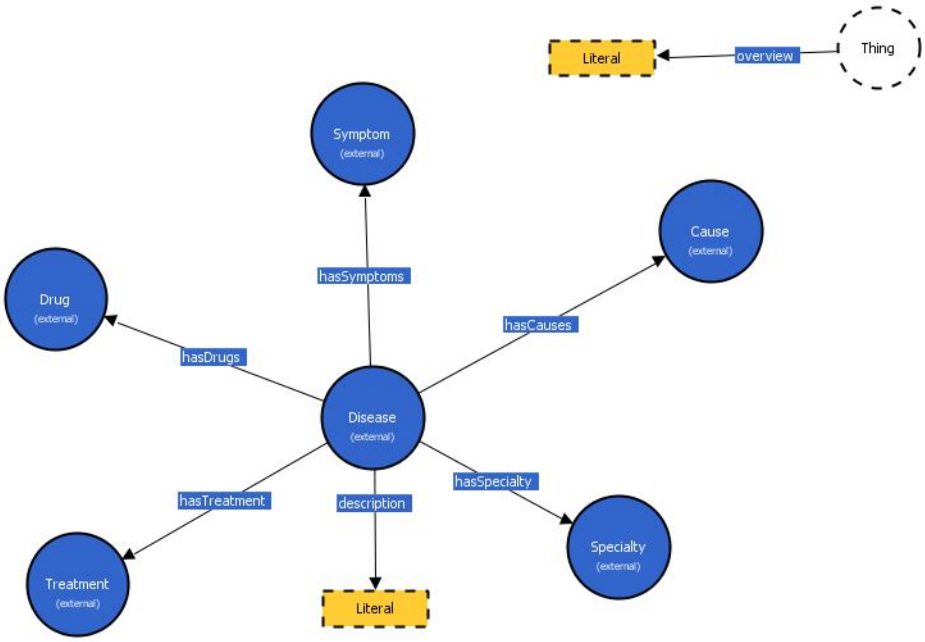


Fig. 18. Ontology Visualization.