



UNIVERSIDAD NACIONAL DE COLOMBIA

# A knowledge-based approach to information retrieval in collections of textual documents of the biomedical domain

**Luis Alejandro Riveros Cruz**

National University of Colombia  
Engineering Faculty  
Systems and Industrial Engineering Department  
Bogotá, Colombia  
2015

# A knowledge-based approach to information retrieval in collections of textual documents of the biomedical domain

Luis Alejandro Riveros Cruz

A dissertation submitted in partial fulfillment of the requirements for the degree of:  
**Master of Sciences in Systems and Computer Engineering**

Under the guidance of:  
Ph.D. Fabio A. González

Research line:  
Information Retrieval  
Research Group:  
MindLab

National University of Colombia  
Engineering Faculty  
Systems and Industrial Engineering Department  
Bogotá, Colombia  
2015

To my beloved parents  
Maria Cruz and Alberto Riveros

# Abstract

The exponential growth in the amount of available data has posed new challenges to researchers. Search on such amount of data is a difficult task which turns even harder when the data belongs to a specific domain which has its own terminology and requires some background knowledge.

Traditional information retrieval systems are based on keywords. In this kind of systems the output for a given query is a ranking of the documents that match the keywords. This model works well in scenarios with few documents or if the system achieves a high performance ensuring that the first results contain the most relevant documents. However, in most cases the collections are huge and the retrieval results are an endless list of documents that must be scanned manually.

This work proposes an information retrieval approach which incorporates domain specific knowledge from an ontology within the traditional information retrieval model in order to overcome some of its limitations. The domain knowledge is used to add semantic capabilities and to provide the user with an enriched interface which includes metadata about the retrieved results, thus facilitating its exploration and filtering.

**Keywords:** information retrieval, knowledge bases, ontology

# Resumen

El acelerado crecimiento en la cantidad de datos disponibles ha traído consigo nuevos retos para los investigadores. Buscar información en este gran volumen de datos, es una tarea difícil, que se torna aún más compleja cuando los datos pertenecen a un dominio específico el cual tiene su propia terminología y requiere un conocimiento previo.

Los sistemas de búsqueda tradicionales son basados en palabras clave. En este tipo de sistemas la respuesta a una consulta es dada en forma de una lista ordenada de documentos que contienen las palabras en la consulta. Este modelo funciona bien en escenarios en los cuales hay pocos documentos o si el sistema puede alcanzar una precisión muy alta garantizando que los primeros resultados contienen los documentos requeridos. Sin embargo, en la mayoría de los casos las colecciones de documentos son muy grandes y los resultados son una lista interminable de documentos los cuales deben ser explorados manualmente.

Este trabajo propone una aproximación a la búsqueda de información, que incorpora conocimiento del dominio proveniente de una ontología, dentro del modelo de búsqueda tradicional con el fin de superar algunas de sus limitaciones. El conocimiento del dominio es usado para adicionar capacidades semánticas y para proveer a los usuarios con una interfaz enriquecida la cual incluye meta-datos acerca de los resultados, facilitando su exploración y filtrado.

**Palabras clave:** búsqueda de información, ontología

# Contents

<b>1. Introduction</b>	<b>2</b>
1.1. Problem definition . . . . .	2
1.2. Objectives . . . . .	3
1.3. Results and contributions . . . . .	3
1.4. Document structure . . . . .	4
<b>2. Background</b>	<b>5</b>
2.1. Knowledge sources . . . . .	5
2.1.1. SNOMED-CT . . . . .	6
2.2. Information retrieval . . . . .	8
2.2.1. The inverted index . . . . .	8
2.2.2. Term normalization process . . . . .	9
2.2.3. The information retrieval process . . . . .	11
2.2.4. Information retrieval process extensions . . . . .	12
2.2.5. Performance evaluation . . . . .	15
2.2.6. Information retrieval models . . . . .	16
2.2.7. Knowledge based information retrieval . . . . .	17
2.2.8. Knowledge based information retrieval in the biomedical domain . . . . .	19
2.3. Automatic term mapping . . . . .	22
2.3.1. Automatic term mapping challenges . . . . .	22
2.3.2. Performance evaluation . . . . .	24
2.3.3. Automatic term mapping in the biomedical domain . . . . .	24
2.4. Semantic similarity measures . . . . .	26
2.4.1. Path based measures . . . . .	26
2.4.2. Information content based measures . . . . .	28
<b>3. Proposed approach</b>	<b>30</b>
3.1. KBMed . . . . .	30
3.1.1. Query schemes . . . . .	31
3.1.2. Knowledge based retrieval process . . . . .	34
3.1.3. Aided query refinement . . . . .	35
3.1.4. Results visualization and exploration . . . . .	36

---

3.1.5. System architecture . . . . .	39
3.2. Term-mapper . . . . .	44
3.2.1. Dealing with language variability . . . . .	44
3.2.2. Ambiguity resolution . . . . .	45
3.2.3. Term-mapper algorithm . . . . .	46
<b>4. Evaluation</b>	<b>51</b>
4.1. Term-mapper performance . . . . .	51
4.1.1. Semeval dataset . . . . .	51
4.1.2. Experimental setup . . . . .	52
4.1.3. Results . . . . .	53
4.2. Knowledge based retrieval model results . . . . .	56
4.2.1. Clef dataset . . . . .	56
4.2.2. Experimental setup . . . . .	56
4.2.3. Results . . . . .	56
<b>5. Conclusions</b>	<b>58</b>
5.1. Conclusions . . . . .	58
5.2. Future work . . . . .	58
<b>A. Software documentation</b>	<b>60</b>
A.1. Term-mapper documentation . . . . .	60
A.1.1. Term-mapper command line interface . . . . .	60
A.2. REST knowledge services interface documentation . . . . .	60
A.2.1. Concepts endpoints . . . . .	61
A.2.2. Knowledge based retrieval engine endpoint . . . . .	62
A.2.3. Hierarchy endpoints . . . . .	62
<b>Bibliography</b>	<b>66</b>

# 1. Introduction

The available biomedical literature has grown exponentially in the last decades. This literature is of great importance because it can be exploited by researchers and healthcare professionals in many ways. But search information on such amount of data is a difficult task which turns even harder when the data belongs to a specific domain which requires a background knowledge.

As an example PubMed has grown exponentially in the last two decades [24] and now contains around 24 millions of citations. These citations corresponds to research articles which are semi-structured, mostly composed by free text and images, and abundant of domain specific information.

Under this scenario terminological resources and ontologies can help to reduce the knowledge gap, giving better results to final users. In the same way, areas like information extraction and retrieval, will play an increasingly relevant role in order to enable the possibility of searching and taking advantage of the information contained in that data.

This work proposes a model which incorporates domain specific knowledge within the traditional information retrieval approach in order to overcome some of its limitations. This knowledge is obtained from terminological resources and ontologies and then associated with the data by means of information extraction techniques.

## 1.1. Problem definition

Traditional information retrieval systems are based on keywords. In this kind of systems the output for a given query is a set of the documents which match the keywords, sorted in decreasing order according to a notion of similarity. Unfortunately this is not the case for real world problems..

As stated in [15] up to 30% of PubMed queries result in 100 or more hits. Manual scan of such amount of information is an intensive labor which can be frustrating, especially if your information needs are not clear and you have to start from scratch every time you need to

reformulate your query to narrow the obtained results.

This work tries to overcome some of these problems through the development of a knowledge based information retrieval system which provides the following benefits:

- Semantic search capabilities using concepts not only keywords
- Aided query refinement based on a domain specific knowledge source
- Visualization of metadata about the results by using the information on a domain specific knowledge source.

## 1.2. Objectives

**General objective:** To develop a knowledge-based information retrieval system for collections of textual documents in the biomedical domain.

**Specific objectives:**

- To propose a method and develop a software tool that performs the extraction of concept mentions within free text.
- To adapt and implement a knowledge-based strategy to measure the similarity in the retrieval system.
- To propose and implement a knowledge-based strategy to visualize and explore the results in the retrieval system.
- To propose and implement a knowledge-based strategy to perform assisted query refinement in the retrieval system.
- To build a prototype of the system.

## 1.3. Results and contributions

The results and contributions of this work can be summarized as follows:

1. A functional prototype of a knowledge-based information retrieval system built using open source software technologies.
2. A new method for automatic term mapping (ATM) which can be used in different languages, with any knowledge source and that does not depend on manually built resources.

3. A publication about the ATM tool (Term mapper) which implements the method mentioned before, as a part of Semeval 2014 analysis of clinical text shared task: Riveros, Alejandro, et al. "MindLab-UNAL: Comparing Metamap and Term-mapper for Medical Concept Extraction in SemEval 2014 Task 7in Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pages 424 – 427, Dublin, Ireland, August 23-24, 2014.

## **1.4. Document structure**

This document is organised as follows. Chapter 2 introduces the theoretical bases and concepts involved in this work. Chapter 3 presents the method and describes in detail the prototype and other tools developed for this work. Chapter 4 shows the experimentation and the obtained results. The final chapter contains the conclusions and discusses future work.

## 2. Background

This chapter presents an overview of the concepts and theoretical foundations required to introduce this work. It includes a brief section about knowledge sources and Snomed-CT. It also includes detailed sections about information retrieval and automatic terminology mapping along with a description of information content based semantic similarity measures.

### 2.1. Knowledge sources

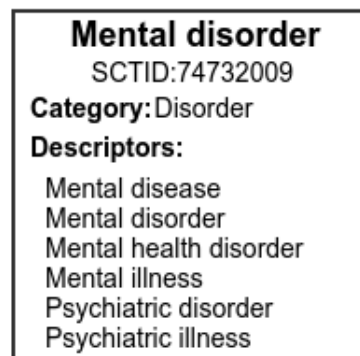
There are many types of resources in the context of knowledge engineering which are referred as knowledge sources (KS). In general a KS is a compendium of information about a general or specific domain but the type, structure and the level of detail in this information, can vary from one source to another. In the context of this work we found many resources that are referred as KS, below we enumerate the most relevant categories:

- Lexicons: contain a list of the lexical units a.k.a lexemes available in a language.
- Terminologies: contains a list of the terms associated to a specific domain.
- Thesaurus: contains information about lexical units, including its synonyms and similar words.
- Taxonomies: contains a categorization or classification of the entities or concepts in a domain.
- Hierarchies: contains a representation in levels of the entities or concepts in a domain organized according to a hierarchical relation, is common to see relations such as “IS-A”.
- Ontologies: contains a formal definition or specification of classes, attributes and relations of the entities or concepts of a particular domain, along with the instances of those entities and their relations.

This categories are not exclusive and there are many examples of knowledge sources that can be categorized in many types simultaneously.

### 2.1.1. SNOMED-CT

domain with versions for languages as English and Spanish, among others. It can be categorized as either a terminology, thesaurus, taxonomy, hierarchy or ontology because it contains information that match all these categories. The version 2012 of Snomed-CT which is used in this work, contains 296235 active concepts, 1446149 relations and 772346 descriptors from which around 7000 corresponds to abbreviated forms, those descriptors can be seen as synonyms by which you can refer a concept as shown in figure 2.1.



**Figure 2.1.:** Snomed-CT concept with associated descriptors

The concepts are classified in 40 categories, around 84% of the concepts are concentrated in seven main categories (2.2).

The relations are organized in 65 different types(2.3). The most notable type is the “IS A” relation with around 540000 instances which are used to define a hierarchy of concepts. This hierarchy has a maximum depth of 20 levels with 87534 inner concepts and 208701 leaf concepts.

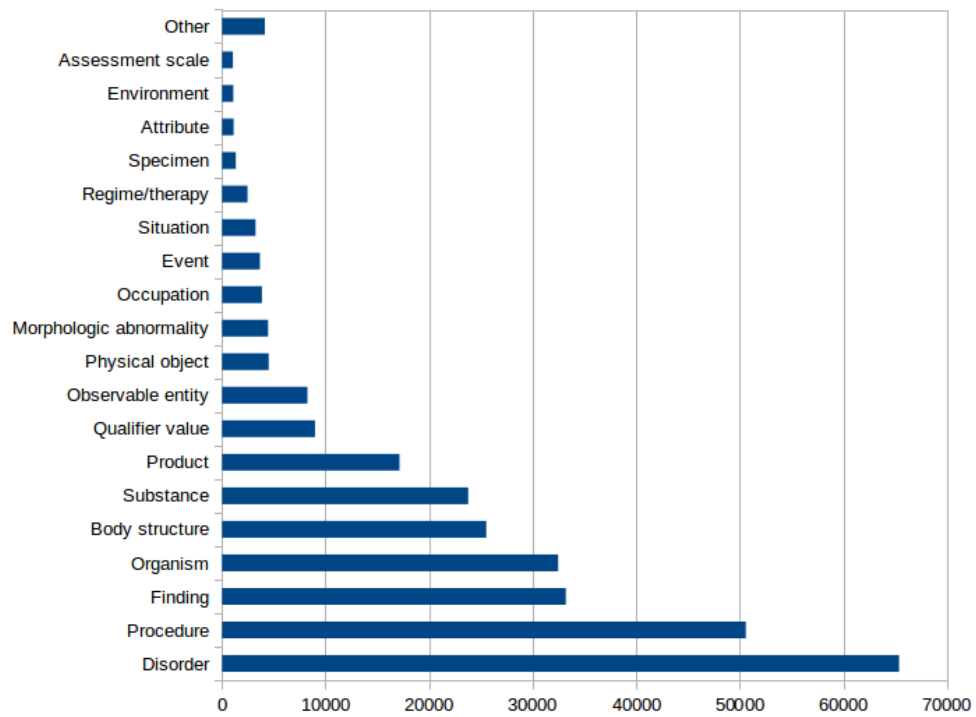


Figure 2.2.: Snomed-CT total concepts by category

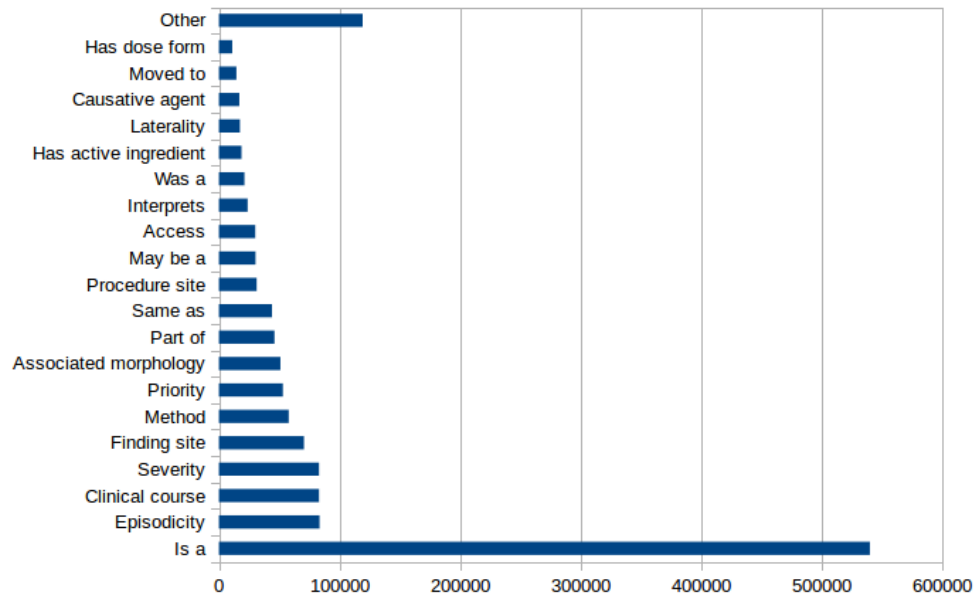


Figure 2.3.: Snomed-CT total relations by type

## 2.2. Information retrieval

The amount of available data has grown exponentially in the last decades, these data ranging from unstructured plain text files to structured databases, passing for repositories that contains multimodal data, i.e. text, images, audio, video, etc. This brings us a great opportunity in the sense that we can explore all the information contained in the data, but also comes with great challenges because we need to create adequate tools to achieve that.

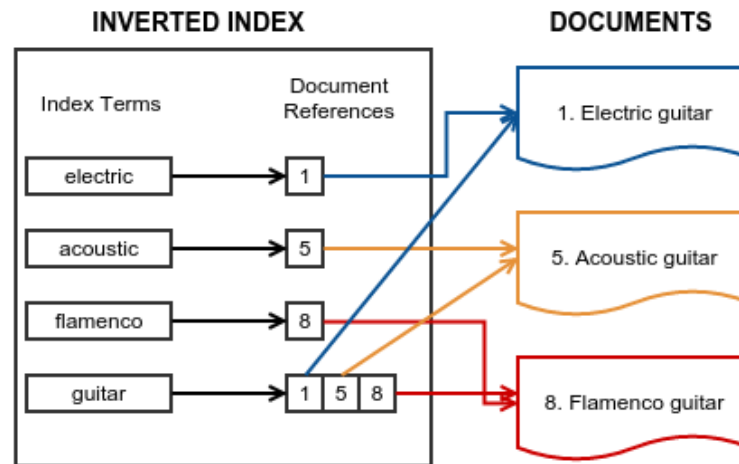
One of the main problems that arises when you have such amount of data is  $\hat{A}$ How you can handle this data to find something and satisfy a particular information need?, this is the question that information retrieval (IR) tries to answer. Information retrieval is the area of Computer Science that study the problem of search and access to information, their main goal is formulate the models and develop the tools that allow users to solve his information needs in a precise and efficient way.

Under the current circumstances in which we talk about repositories that contains terabytes of data and need to answer millions of search queries per day, is a fact that information retrieval is an active research area, which plays an increasingly important role in the future [41].

### 2.2.1. The inverted index

In the beginning of the computing age, before the rise of internet, the amount of data was small and usually stored locally within mainframes. In that scenery, when an user needs to find some information, he usually makes use of tools like GREP (Globally Search a Regular Expression and Print), which performs a sequential search for a pattern usually a regular expression, within a file or a directory structure. This sequential scanning approach makes sense when you have small volumes of data and do not look for information very frequently, but definitely does not scale and becomes infeasible when you are working with large amounts of data that needs to be searched often.

To deal with large data collections was necessary to develop a new data structure that allows fast searches without waiting a long time while a full scan is performed. That is the moment when the so called inverted index born, a data structure specifically crafted to manage fast searches over large amounts of data. The idea behind the inverted index is similar to the glossary index present in some books, which allows the reader to quickly look for the pages that contain certain words. In the case of an inverted index the words are called terms or index terms and the pages are documents identifiers, that can corresponds to files, web pages, databases records etc.



**Figure 2.4.:** The most simple version of an inverted index, when you have terms associated with a list of document identifiers in which the term appears

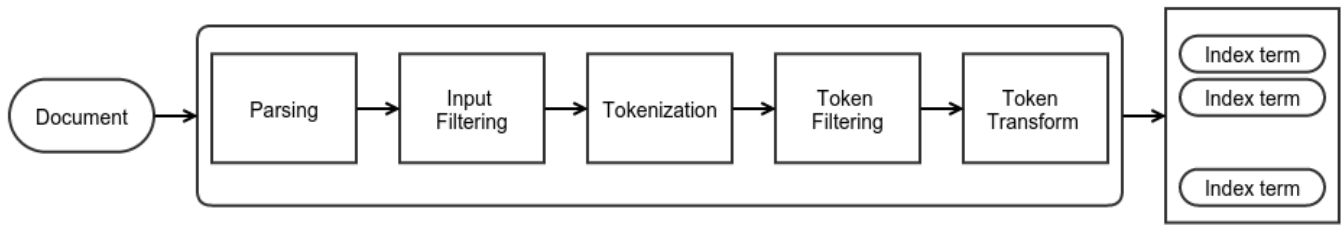
Usually, the inverted index is built in a batch job previous to its usage. In this indexing process the whole data must be scanned, passing for a term normalization process. Although it is a computer intensive task, it is performed only once, whereby its cost can be accepted. In the case of dynamic data collections, the index can be easily updated without incur in a high computational cost. Once the inverted index is generated, the lookups can be performed in constant time, improving enormously the performance of the full scan method.

Also and given the flexible nature of an inverted index, you can use it to store information that goes far beyond the presence of a term in a document. In modern implementations is common to have information about the document structure i.e. the document portion in which the term appears; detailed positional information of term within a document; the total frequency and the per document frequency of a term; and also metadata of the document i.e. total length etc.

### 2.2.2. Term normalization process

One of the challenges that you face when deal with data in the form of free text is the term normalization process. In this step the input data is transformed into index terms. This is a critical step which can affect the match process against the inverted index.

This normalization process can be divided in five main steps:



**Figure 2.5.:** The term normalization process

1. **Parsing:** this can be viewed as a preprocessing step, in which the input data is taken and analyzed to extract the portion of interest for searching purposes. This is a necessary step, because in real word sceneries exists many data sources each with many distinct data formats i.e. plain text files, pdf files, xml files, data streams, web pages etc, so you need to remove the format related data to focus in the content.
2. **Input filtering:** in this step the input is filtered to remove portions of data that can produce noise in the next steps. Usually this include to remove strange characters and punctuation, but this depends entirely on the desired results, i.e. typically you remove all not letters but if you want to look for emails you need to preserve some special characters that are constituent of an email addresses like “@” and dots.
3. **Tokenization:** in this step the input text is splitted in its constituent parts (tokens). It is habitual to split the input in any sequence of one or more whitespace like characters and that the tokens corresponds to words, but as in the filtering step this depends of the desired results.
4. **Token filtering:** in this step the tokens are filtered to preserve only those significant enough to be stored in the index, it is usual to remove very short or long tokens, and also to remove the so-called “stop words”, because of its high frequency i.e. English articles like: a, an, the, etc, that appear in almost every document and hence can’t be used as discriminative terms.
5. **Token transformation:** in this final step the tokens are transformed in order to increase the probability of a match to terms in the index. Usually the tokens are converted to a particular case either lower or upper; in languages with latin accents is usual to perform a folding i.e. change the accented words by their unaccented counterparts, to avoid that spelling mistakes affects the match.

Also and with the inclusion of specialized tools like stemmers you can perform morphological transformations. The stemmers goal is to reduce the tokens to its stem or root form, this is a simple way to deal with morphological variations, i.e. “fish” is the stem of the words: fish,

fishes and fishing, so if you perform a lookup over the index for any of those three words, you obtain matches for all of them.

### 2.2.3. The information retrieval process

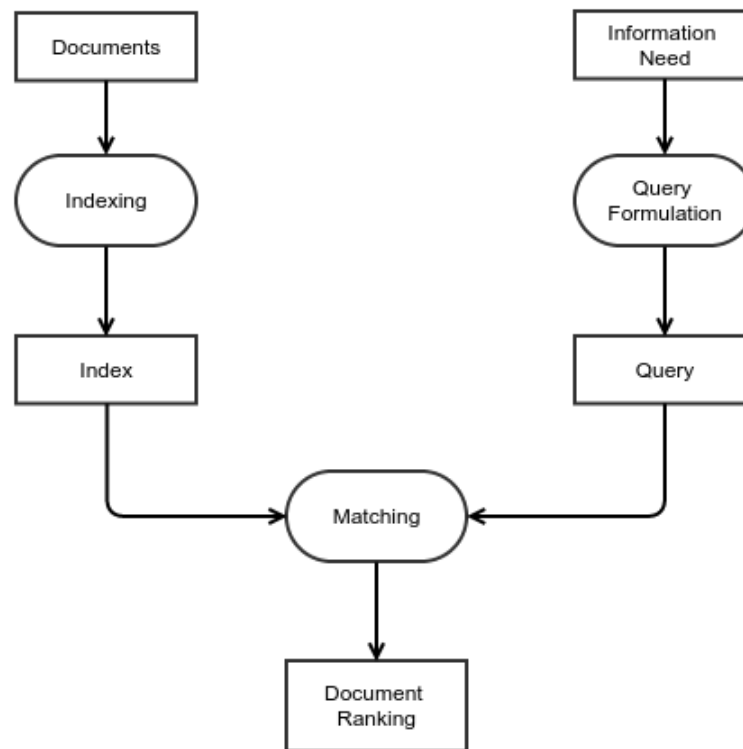


Figure 2.6.: An overview of the general information retrieval process.

The most typical information retrieval process consists of the following three steps [12]:

1. **Indexing step:** The goal of this process is to build the inverted index, this step is usually executed in a batch job previous to any interaction with the system. It starts with the raw data and through the application of the term normalization process described before, obtains the terms that will be included in the index and also extract all the required metadata i.e. frequency counts, positional information, etc.
2. **Query formulation step:** this is the step in which the user information need is transformed into a query, the goal of this process is to obtain a query that suits the requirements of the system, i.e. can be matched against the index, this usually is guaranteed by apply the same term normalization process to both index and query

terms. The resulting query can range from a simple single term to a more elaborated composed of many terms with relations between them. In modern full text search engine implementations is common to see queries that allows the usage of boolean operators, ranges, proximity between terms, regex like terms and even fuzzy matching.

3. **Matching step:** in this step the query is executed against the index, as result of this a subset of documents that match the query are selected. In the case of large collections the number of documents that match the query can exceed for long the capacity of a human to scan it manually [26] i.e. one by one. So a final scoring step is required in order to sort matching documents according to its relevance to the given query, after this we obtain a document ranking that represents the system answer to the user information need.

#### **2.2.4. Information retrieval process extensions**

The process described in the previous section is the core of any retrieval system, however, there are many alternatives that extend or even redefine this process. The most popular extensions are described below.

##### **Relevance feedback**

The idea of relevance feedback is to involve the user judgements about the relevance of the returned results, in order to construct a better representation of the user information need and then formulate a better query, this feedback and reformulation steps can be repeated many times until the information need is satisfied.

Relevance feedback is based on the notion that in some sceneries it is difficult to formulate good precise queries, but in the most cases its easy to judge the relevancy of particular documents. A typical scenario for this is an image search engine in which it is difficult to find the right query terms but once images are showed its easy to see if they are relevant or not.

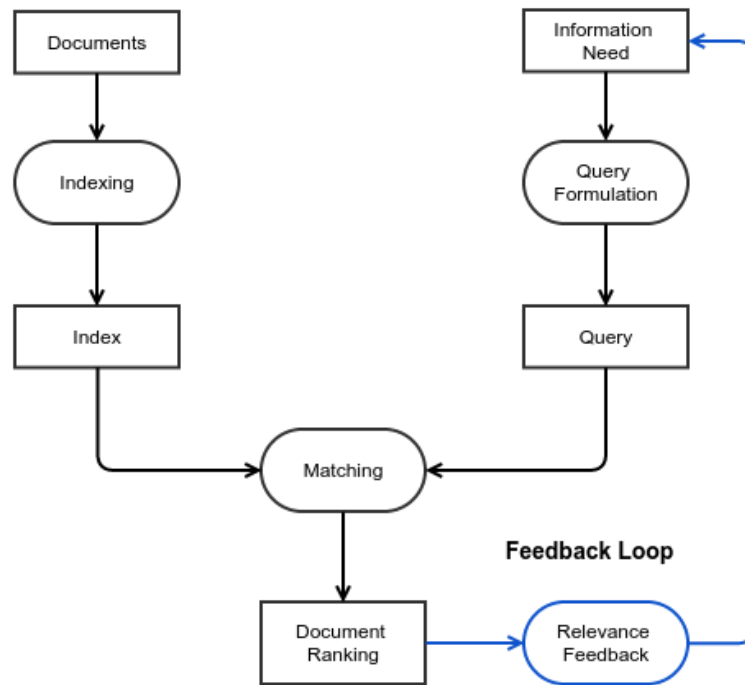
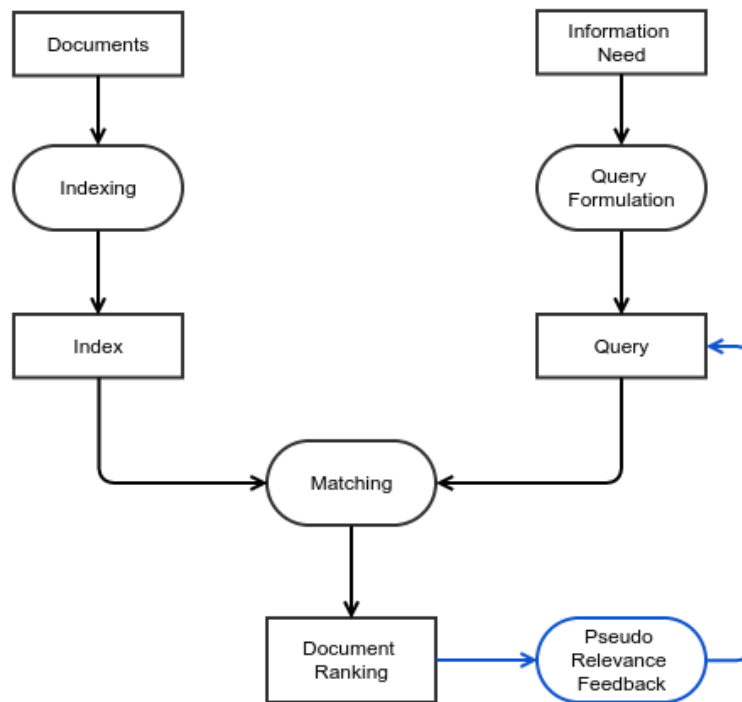


Figure 2.7.: Information retrieval process with relevance feedback

### Pseudo-Relevance feedback

One criticism against the relevance feedback is that given that the user intervention is required the entire process may become biased and costly. To cope with this problem, a new automatic approach called pseudo-relevance feedback was proposed [7].

The idea of this approach is to assume that the top  $k$  documents retrieved by the original query are relevant, then take these documents and extract the most representative terms within them and use those terms to expand the original query. This method has proven to be very effective to deal with short queries [41].



**Figure 2.8.:** Information retrieval process with pseudo-relevance feedback

### 2.2.5. Performance evaluation

To measure the performance of an information retrieval system it is necessary to have at least one query along with their associated ranked results, this allow us to classify the documents in a collection as relevant or irrelevant, and then compute the following measures.

1. Precision: which corresponds to the proportion of retrieved documents that are relevant over the total documents retrieved.

$$precision = \frac{|\{relevant\_documents\} \cap \{retrieved\_documents\}|}{|\{retrieved\_documents\}|} \quad (2.1)$$

2. Recall: which corresponds to the proportion of relevant documents retrieved over the total relevant documents.

$$recall = \frac{|\{relevant\_documents\} \cap \{retrieved\_documents\}|}{|\{relevant\_documents\}|} \quad (2.2)$$

3. F-measure: a measure that summarize the compromise between recall and precision, which is the harmonic mean between both metrics.

$$F - measure = \frac{2 * precision * recall}{precision + recall} \quad (2.3)$$

A well performant retrieval model should not retrieve irrelevant documents, and at the same time it should retrieve all the relevant ones, which will be reflected in a high precision and recall value respectively, however, is a fact that the improvement in one metric degrades the other [41].

An issue with these metrics is that they are set oriented, and therefore does not take into account the order in which the documents are ranked, to avoid this problem we can use one of following metrics, which are ranking aware:

- **Precision@N:** is a variation over the original precision idea, which only takes into account the first N results, to obtain precision values at different ranking levels, typical values for N are 5, 10, 20 or 50.
- **Average precision:** is a metric which takes into account both precision and recall, this metric is computed as the average of the precision in the points where the recall value is 10 %, 20 %, 30 %, etc [35].
- **Mean average precision (MAP):** is a metric derived from the average precision, used to measure the performance over multiple queries, is computed as the mean of the average precision over all queries.

## 2.2.6. Information retrieval models

An information retrieval model describes the abstractions and the mathematical foundations used to implement an information retrieval process, it can be viewed as a blueprint that helps to share the ideas and formalize the terminology [12].

### Vector Space Model (VSM)

This is the most significant model within the area and it is the base for the implementation of the majority of the search engine implementations available today, it was proposed by Salton in 1975 [34].

The idea of this model is to represent documents as vectors in a Hilbert space, this is based on the notion that the best indexing space is the one in which each term lies as far away from the others, and a Hilbert space meets this requirement given that its dimensions are mutually orthogonal.

Suppose you have  $n$  index terms, each term corresponds to a dimension within the Hilbert space, so the space has  $n$  dimensions and therefore both documents and queries are represented by vectors of size  $n$  as follows:

$$d = (w_{t1}, w_{t2}, \dots, w_{tn})$$

Where  $w_{ti}$  corresponds to the weight associated to the  $i$ -*esim* term for the document, in the case of the query the weight is always one.

Given that documents and queries are represented in the same way, the match process consists in compare the vector that represents the query, against the vectors that represents the documents using a measure of similarity.

The similarity measure proposed is cosine also known as the normalized inner product, which can be thought as the cosine of the angle that separates the two vectors, this measure is 0 if vectors are orthogonal and 1 if the vectors are the same regardless of its magnitude, in any other case its value is between (0, 1).

$$\cos(\vec{d}, \vec{q}) = \frac{\vec{d} * \vec{q}}{\|d\| * \|q\|} = \frac{\sum_{i=1}^n \sum_{j=1}^n d_i * q_j}{\sqrt{\sum_{i=1}^n d_i^2} * \sqrt{\sum_{i=1}^n q_i^2}} \quad (2.4)$$

Once the similarity values are computed the documents are sorted in decreasing order based on this value.

As a measure of term weight, this model propose the usage of  $TF - IDF$ , which is the product of two values:

- **TF(Term Frequency):** which measure the number of times the term appears in the document, this under the assumption that frequent terms in a document tend to be more important.
- **IDF(Inverse Document Frequency):** which measure the term specificity, which is computed by taking the logarithm of the quotient between the total number of documents  $N$  and the document frequency  $df$ , which is the number of documents containing the term.

$$TF - IDF = tf * \log\left(\frac{N}{df}\right) \quad (2.5)$$

This way  $TF - IDF$  shows a balance between the local term frequency and the global document frequency.

Good points of this model are its ease of implementation and well grounded and structured foundations, downsides comes from the strong assumptions about the representation space, i.e. the orthogonality between dimensions which implies that terms can only be compared with themselves, so the similarity between any pair of documents that does not share terms is zero, regardless of whether the documents can be similar in some sense.

To overcome this problem some methods were proposed, the most relevant ones are the Generalized Vector Space Model [43] and Latent Semantic Indexing [9], these methods tries to include term dependencies within the VSM model by exploiting the correlation between terms obtained from the data, however, these methods were not very successful because they bring small benefits at high computational cost.

### 2.2.7. Knowledge based information retrieval

The goal of semantic search is to go beyond term matching through the use of real world knowledge, in order to have a better understanding of document content and user information needs and thus improve the retrieval performance.

Knowledge based information retrieval refers to a branch of semantic search where the traditional retrieval process is reassembled adding to it semantic capabilities, but the objective

remains the same i.e. you still want to retrieve documents. This is opposed to semantic search in the context of the semantic web, in which the objective is to find information or answer questions about real world entities regardless of the origin of the information as described in [13].

The real world knowledge is taken from external knowledge sources which can be generic or specific to a domain and which come in a variety of flavors depending of its structure, contents and level of detail as described in section 2.1.

Two typical scenarios in which this knowledge can be used to improve retrieval performance are in the management of term synonymy and polysemy [12]. In the first scenario, queries can be expanded adding the synonyms of the original query terms improving recall. In the second scenario, when a query term has multiple meanings, the system can suggest the possibilities and leave to the user the selection of the right sense improving precision.

Mangold [25] establishes a categorization for semantic search approaches according to seven dimensions, as follows:

1. Architecture: refers to the architecture of the system.
  - Standalone: self contained solution does not use external resources.
  - Meta-search: use external resources to provide semantic search capabilities i.e. full text search engines.
2. Coupling: refers to the coupling between the knowledge source and the data.
  - Tight: special purpose KS, in the major of the case the KS is developed specifically for the solution and the data is manually annotated.
  - Loose: apply for general domains, use resources already existing i.e. WordNet [28]
3. Transparency: with respect to the user, i.e. the user take part of the retrieval process and is aware of that.
  - Transparent: user is not aware
  - Interactive: the users interact with the system i.e. to solve ambiguity and narrow queries.
  - Hybrid: merge both and example of this can be a system with offers simple and advanced search modes.
4. User context: use the user to context to narrow and disambiguate queries:

- Learning: from user behaviour.
  - Hard coded: offer a predefined set of queries called question-categories, the category is selected by either user or user context examples of question categories are “location of ...”.
5. Query modification: refers to the method used to improve queries.
    - Manually: suggest query modifications based on data from the knowledge source, but leave the query modification to the user.
    - Query rewriting: the system automatically modify the query, by augmenting, trimming or substitute a portion of the query according to some predefined rules.
    - Graph based: merge both documents and knowledge source in a unique graph from which take information to modify the query.
  6. Ontology properties: refers to the set of KS properties exploited by the system
    - Anonymous properties: use the association or relation regardless of its type.
    - Standard properties: use relations like synonym, hyponym, instance of etc.
    - Domain specific properties: use domain specific relations i.e. processor architecture for a computer description.
  7. Ontology technology: RDF, OWL,DAML or others.

A survey of the systems that implements this approach can be found in [25]. The focus of this work is in knowledge based approaches within the biomedical domain which are described in the next section.

### **2.2.8. Knowledge based information retrieval in the biomedical domain**

The available biomedical literature has grown exponentially in the last decades, as an example in 2014 the MEDLINE database contains around 24 millions of citations. This literature is of great importance because it can be exploited by researchers and healthcare professionals in many ways. Search information on such amounts of data is an increasingly difficult task and services like PubMed only provide basic search capabilities which are not enough on such scenario, as an example over one third of PubMed queries result in 100 or more hits [15] that is much information to be scanned manually.

To cope with this problem in the last years new knowledge based information retrieval approaches have been developed, these approaches tries to overcome the limitations of PubMed and go beyond term matching, by means of the use of domain specific knowledge that

comes from a variety of resources.

One of the first approaches is XplorMed [31], this system clusters the results obtained from PubMed with MESH (Medical Subject Headings) main headings and allows to filter the results according to these clusters, additionally it performs keyword extraction from results abstracts and allows to iteratively refine the query using the extracted keywords.

Textpresso [17] presents another approach in which the user can define the scope of the search at sentence, abstract or full text level. It also offers semantic metadata that can be used to filter the retrieved atoms. The semantic metadata consists of terms from an ontology, specifically built for the system, which are automatically extracted from the text using a simple approach based on case folding and light stemming.

GoPubMed [11] uses the GO (Gene Ontology) [3] to provide a nice interface to navigate PubMed search results. The results are enriched with semantic metadata in the form of GO and MESH terms which are previously identified using a method based on local sequence alignment. This semantic metadata can be used to filter results or narrow queries iteratively. The interface highlights both search hits and semantic metadata and provides context information obtained from the corresponding ontology.

MEDIE [29] presents a hybrid approach between natural language processing and knowledge based information retrieval in which the idea is to identify abstracts which contain triplets in the form subject, verb and object, where the subject or the object corresponds to terms categorized as diseases within UMLS (Unified Medical Language System). Later these triplets can be used to perform semantic queries i.e. this question ¿ What causes lung cancer? can be represented by a semantic query that matches “cause” as verb and “lung cancer” as object.

MedEvi [19] provides ten biological categories i.e. drug, gene, etc which can be used along with typical terms in the formulation of semantic queries, it also suggests candidate query terms by performing an information extraction step for entities like genes, proteins and GO terms over the initial query results.

GeneView [42] integrates the output of ten different information extraction tools and aligns its results with multiple knowledge sources like MESH, chemSpot, drugBank, dbSNP, etc. The resulting semantic metadata is used to provide a rich visualization of retrieved results which include context information about the present entities and their relations.

These approaches can be categorized according to the seven dimensions defined in the previous

section as follows:

	XplorMed [31]	Textpresso [17]	GoPubMed [11]	MEDIE [29]	MedEvi [19]	GeneView [42]
Architecture	Metasearch	Metasearch	Metasearch	Standalone	Standalone	Metasearch
Coupling	Loose	Tight	Loose	Loose	Loose	Loose
Transparency	Interactive	Interactive	Hybrid	Interactive	Interactive	Hybrid
User context	-	Hard Coded i.e. scope	-	-	Hard coded i.e. concept category	Hard coded i.e. concept instance
Query modification	Manually	Manually	Manually	-	Manually	-
Ontology structure	Standard	Standard	Standard	Standard	Standard	Standard
Ontology technology	Other MESH	Other based on GO	Other GO	Other UMLS	Other GO	

**Table 2.1.:** Categorization of knowledge-based IR approaches within the biomedical domain

## 2.3. Automatic term mapping

Automatic term mapping (ATM) is an information extraction (IE) subtask, its goal is to identify term mentions within text and associate those mentions with an unique entry within a terminology. It differs from tasks like automatic term recognition (ATR) and entity recognition (ER) because it tries to establish the exact term identity not only the presence of a domain term or a category.

As an example take the following sentence and assume you are working in the biomedical domain looking for terms associated with diseases:

“Non-small cell lung cancer in young adults is a rare...”<sup>1</sup>

In the context of an ATR task the expected output must differentiate between text portions that contains domain terms and those that do not.

`<domain-term>Non-small cell lung cancer</domain-term>` in young adults is a rare.

In the context of an ER task the expected output must mark text portions with a broad category label.

`<DISEASE>Non-small cell lung cancer</DISEASE>` in young adults is a rare.

In the context of an ATM task the expected output is much more specific and tries to associate a text portion with an term instance, in this example the text portion is associated with the SNOMED-CT concept with ID 254637007 which belongs to the category disease.

`<SCTID:254637007>Non-small cell lung cancer</SCTID:254637007>` in young adults is a rare.

ATM is a critical step needed to exploit the information contained in ontologies and terminologies in task such as information retrieval and semantic integration, however, due to its complexity it still is a bottleneck in the process [21].

### 2.3.1. Automatic term mapping challenges

ATM methods can range from simple ones which only deal with exact matching on single terms, to more sophisticated approaches in which you have fuzzy matching and can manage

---

<sup>1</sup>[Ahttp://www.ncbi.nlm.nih.gov/pubmed/25725079](http://www.ncbi.nlm.nih.gov/pubmed/25725079)

non continuous and non ordered terms, expand synonyms, detect abbreviations and deal with ambiguity resolution. But independently of the approach ATM methods always face three main challenges which are discussed below.

### Language variability

The high degree of variability in natural language text [27] brings a big challenge for ATM methods because it makes hard to match text and terminological resources, this language variability can be categorized as follows:

- Orthographic: are variations given by either orthographic rules or orthographic errors, as example the terms “cauterization of Bartholin’s gland” and “cauterization of Bartholin gland” refers to same term with a small difference in its orthography.
- Morphological: are given by inflectional or derivational variations of a lexical unit, as example the terms “transcription intermediary factor-2” and “transcriptional intermediate factor-2” refers to the same term, but with different morphological variations i.e. intermediary and intermediate.
- Lexical: refers to variations in the lexical units used to express a concept, as example the terms “bacterial septicemia” and “bacterial sepsis” represent the same concept using different lexical units.

The use of abbreviations also brings a challenge which is particularly critical in the case of the biomedical text due to its richness in acronyms and abbreviations [20]. The issue with the abbreviations comes from the fact that they are constantly changing and hard to detect because its generation does not follows any pattern, so a significant effort is required to maintain terminological resources up to date [40].

### Ambiguity

Another challenge for ATM methods is ambiguity, which refers to the scenario in which a lexical unit, word, stem or token has multiple meanings or senses. The two main scenarios of ambiguity in an ATM task are:

- Ambiguity due to homonymy when a word has multiple meanings, as an example within SNOMED-CT(r) the word radius appears in the following terms:
  - SCTID281272010 Radius (Body Structure)
  - SCTID735563018 Radius (Qualifier Value)

Which refers to a bone and a geometric property respectively, ideally and ATM method must perform disambiguation and choose the right sense to obtain a correct mapping.

- Ambiguity due to abbreviations or acronyms which corresponds to more than one long form or expansion, as example the acronym "MR" can refer to "Mitral Regurgitation" or "Magnetic Resonance".

### **Terminology completeness**

The performance of ATM methods always will be bounded by the richness and completeness of the terminology resources used. You can have sophisticated ATM methods but without the appropriate terminological resources they are useless. It is important to note that the biomedical domain is the domain with the most rich terminological resources available [30].

### **2.3.2. Performance evaluation**

To measure performance of ATM task you need almost one document along with a set of annotations which associate text portions with instances in a terminology using an unique identifier. This allows to classify resulting annotations as either correct or incorrect and then use traditional IR metrics like precision, recall and f-measure which are described in section 2.1.5.

### **2.3.3. Automatic term mapping in the biomedical domain**

ATM methods have been of particular interest within the biomedical domain, mainly because ATM is required as a previous step for tasks that aim to exploit the rich knowledge sources available for this domain. However, although much research work has been done in this area, there are very few available tools to perform ATM [4].

MetaMap [2] is the gold standard it was developed by the National Library of Medicine (NLM) to map free text to concepts within the UMLS (Unified Medical Language System) metathesaurus, it is widely used for many tasks including the indexing of MEDLINE abstracts [45].

The approach of MetaMap is described as a linguistic intensive approach in which the input text is processed through an analysis pipeline, which perform task such acronym detection, sentence detection, tokenization, lexical variant generation, candidate scoring and disambiguation.

The main advantages of MetaMap are its configurability and ease of installation and usage, the drawbacks are they strong coupling with UMLS and other handmade resources, its

slowness, poor scalability capabilities and that only works for english [2].

MGrep [8] is a command line tool that extends unix grep behaviour by allowing the matching across multiple lines, it is used to empowers the Open Biomedical Annotator (OBA) service [16], when evaluated it shows high precision rates at the cost of recall [6], its advantages comes from speed and ease to use.

Peregrine [39] propose a lightweight approach that process input text through an analysis pipeline which perform tokenization, variant generation, filtering and simple disambiguation to normalize gene names and them match it against terminology entries.

## 2.4. Semantic similarity measures

Semantic similarity measures (SSM) refers to a group of metrics used to evaluate the degree in which a pair of concepts are similar, these metrics were designed with the objective of imitate human judgments about similarity [30]. As SSM try to imitate human judgments, they performance is evaluated by the positive correlation against manually build judgments.

SSM are a particular case of semantic relatedness measures (SRM) because it focused in concepts of the same type which are contained within the same “IS A” hierarchy, while SRM use relations other than “IS A” to measure a more general notion of similarity between concepts of different types.

### 2.4.1. Path based measures

The first notion of similarity within a hierarchy is based in the idea that the similarity between a pair of concepts is inversely proportional to the length of the path that connects them within the hierarchy, as can be seen in figure 2.9 the path length is obtained by counting the number of edges without take into account its direction.

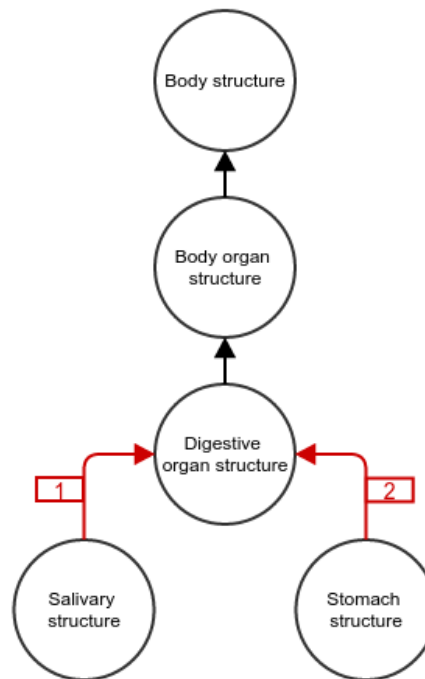
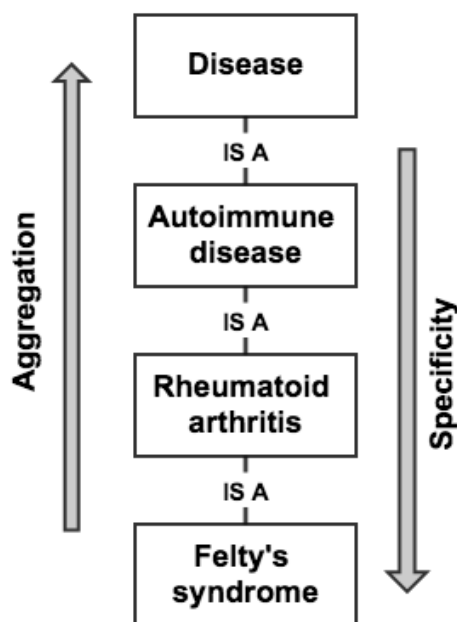


Figure 2.9.: Similarity by path length

This notion of similarity has a problem it consider all paths within the hierarchy equal, that is not a good idea because it ignores the fact that more general concepts are found in hierarchy first levels while specific ones are found in leafs as can be seen in figure 2.10, so the weight of the paths in the most depth levels cannot be considered equal to the weight of the paths in the first levels.



**Figure 2.10.:** IS-A hierarchy fragment from Snomed-CT

To deal with this problem Wu [44] and Leacock [22] propose measures that include the depth in the hierarchy within its computation. These measures use the depth of the least common subsumer (LCS) concept as scaling factor to assign more weight to paths on the deepest levels as can be seen in figure 2.11. Where the LCS corresponds to the most specific concept which is an ancestor of the concepts which are compared.

Another issue with path based measures is given by the non uniformity in the structure of the hierarchy, which is evidenced by the fact that concrete concept i.e. leafs can be found at any level, and therefore the specificity of a concept cannot be associated only with its hierarchy depth. To deal with this problem a new kind of measures that take into account factors other than path length and depth level were developed one of these approaches will be described in the next section.

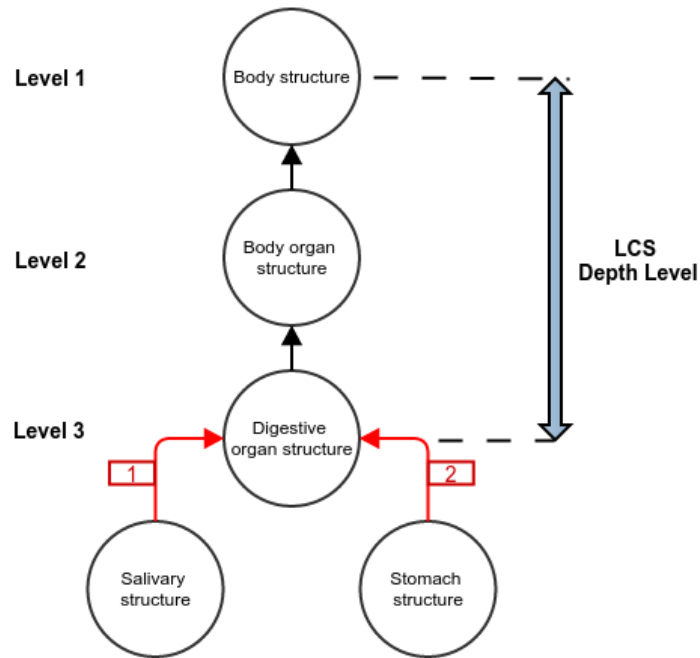


Figure 2.11.: Path based scaled by depth level

## 2.4.2. Information content based measures

### Information content

Information content (IC) value provides a measure of the amount of semantic content carried by a concept [36], general concepts have an IC value close to zero, while specific or concrete ones have a value close to one. IC values are used to deal with some of the drawbacks of the path based SSM described before. In particular IC values are used to build a new set of SSM that are aware of the specificity of the concepts and produce results which are close to human similarity judgments.

There are many methods used to compute IC, some of them used a corpus based approach like the proposed by Resnik et al [33], however, these approaches suffer of problems of scalability and are data dependent [37]. To overcome these problems new methods characterized by rely solely in the structure and information contained in ontologies and hierarchies were developed this kind of methods are referred as intrinsic methods.

The knowledge modelling process used in the construction of hierarchical knowledge sources, in which new abstract inner concepts are introduced to generalize concrete ones can produce as a result that a great portion of the concepts in a hierarchy corresponds to abstract ones i.e. around 21% of WordNet [28] nodes corresponds to inner ones [10]. Moreover, the set of

leaves in a hierarchical knowledge source should accurately define and cover all the scope of the modeled domain. Starting from these two arguments Sanchez et al. [37] propose a new IC intrinsic computation method which avoid inner nodes and focus only on leaves when compute IC values, the method is as follows:

Lets  $C$  denotes the set of concepts in an hierarchical knowledge source, then define the following subsets over  $C$ :

$$leaves(a) = \{l \in C | l \in hyponym(a) \wedge l \text{ is a leaf}\} \quad (2.6)$$

$$subsummers(a) = \{s \in C | \text{where } a \text{ is hierarchical specialization of } s\} \quad (2.7)$$

Then the IC value for a concept  $a$  is computed as:

$$IC = -\log \left( \frac{\frac{|leaves(a)|}{|subsummers(a)|} + 1}{max\_leaves + 1} \right) \quad (2.8)$$

and max-leaves as the number of leaves corresponding to the root node.

### Lin's semantic similarity measure

Lin et al [23] introduce a general notion of similarity based on an information theoretic approach which is applicable in a variety of domains and problems, such notion is applied to define Lin's SSM as follows:

$$sim_{Lin}(a, b) = \frac{2 * IC(LCS(a, b))}{IC(a) + IC(b)} \quad (2.9)$$

Where IC refers to the IC content value and LCS refers to the least common subsumer of the respective concepts.

Lin's SSM with IC values computed using the intrinsic approach described in the previous section has been tested, the results obtained shows a correlation of 0.85 in a general domain [37] and a correlation of 0,79 in the biomedical domain using SNOMED-CT as hierarchical knowledge source [36].

## 3. Proposed approach

This chapter describes the system proposed in this work called KBMed a knowledge-based information retrieval system. This chapter also includes a detailed description of Term-mapper an ATM tool developed in this project to automatically extract the semantic metadata required by KBMed.

### 3.1. KBMed

KBMed is a knowledge based IR system which extends the traditional IR process by adding a semantic layer which makes use of semantic metadata obtained by means of IE techniques, to provide query suggestions and result exploration capabilities in the top of a full text search engine. It also exploits this metadata to construct a knowledge based retrieval approach which tries to overcome some of the limitations of the traditional keyword based retrieval.

Using the categorization introduced in section 2.2.7 KBMed is classified as shown in the table 3.1. There are some features of KBMed that go beyond the classes used in this categorization, these features include the knowledge based retrieval approach and the possibility of using the ontology structure to navigate the results.

Architecture	Meta search engine
Coupling	Loosely couple
Transparency	Hybrid
User context	Hard coded i.e. concept instance
Query modification	Hard coded i.e. concept instance
Ontology structure	Standard
Ontology technology	Other

**Table 3.1.:** KBMed categorization according to Mangold classes [25]

This section describes KBMed features including query schemes, aided query refinement

strategies, results exploration and the knowledge based retrieval process used to build the semantic ranking.

### 3.1.1. Query schemes

KBMed is a metasearch engine which adds a semantic layer in the top of a full text search engine, given this it provides a variety of query schemes that range from traditional keyword search to a more elaborated knowledge based scheme based on semantic metadata, each of these retrieval schemes along with the associated retrieval process is detailed in the following subsections.

As a previous step to the retrieval processes in all the query schemes, it is necessary to perform a query formulation process, in the case of KBMed this process is quite simple and consist on choose all the different keywords and concepts entered by the user, then the keywords are combined using the “OR” operator, and the concepts are combined using the “AND” operator, thus obtaining a keyword query and a concept query respectively.

#### Query by keywords

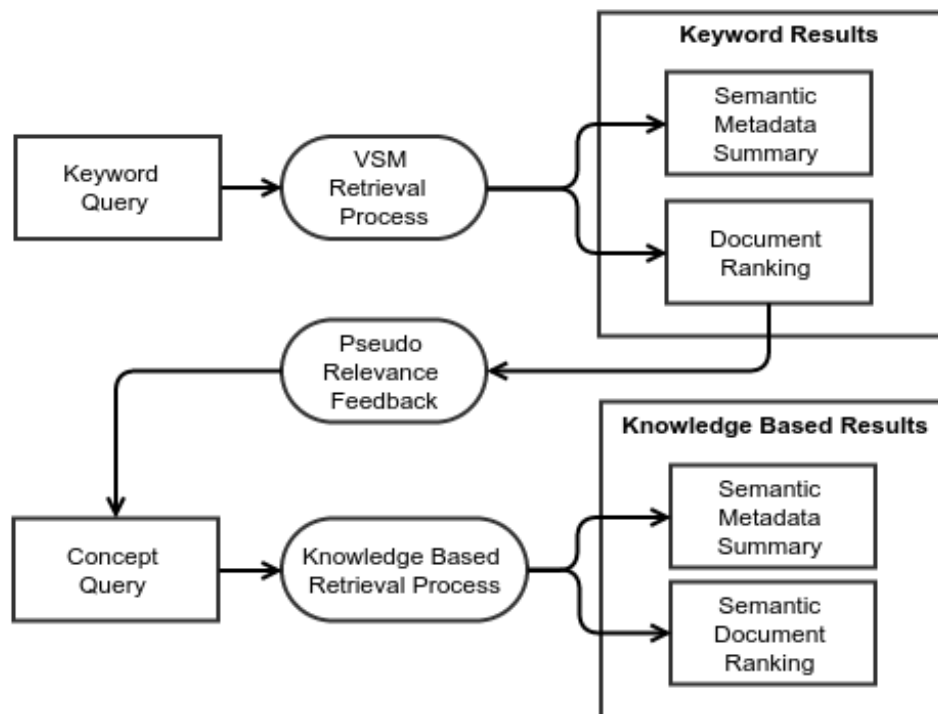
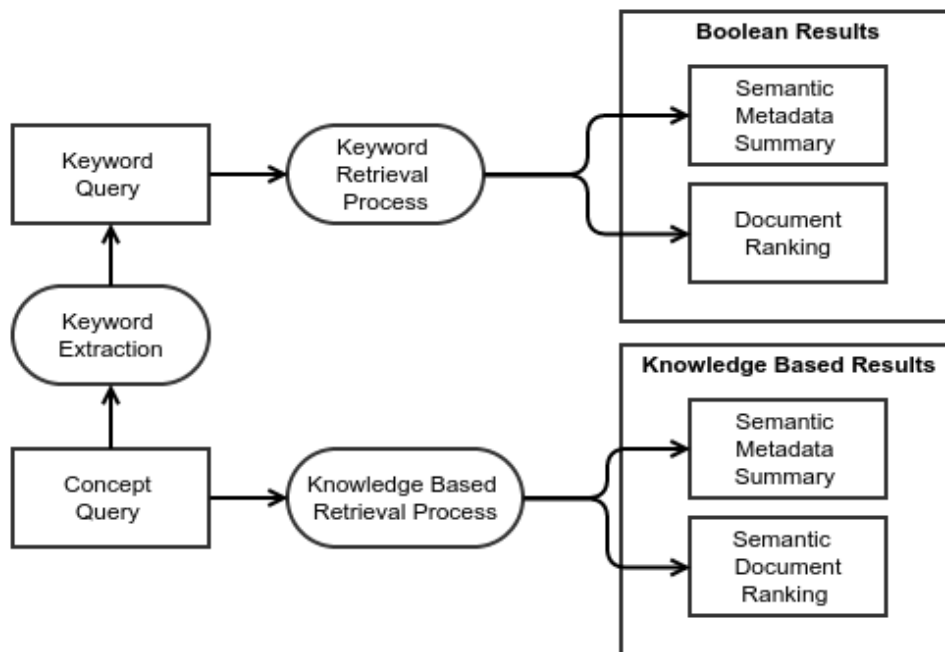


Figure 3.1.: KBMed retrieve by keywords process

In this scenario the user enters one or more keywords which represent its information need, from those keywords a keyword query is formulated and then the retrieval process in figure 3.1 is executed as follows:

1. Keyword query is used to perform a standard VSM retrieval process, obtaining as results a ranking of relevant documents along with its semantic metadata.
2. Then a pseudo-relevance feedback process is executed using the top 10 documents in the ranking, from this documents the most relevant concepts are extracted using  $TF-IDF$  weight average as criteria, this subset of relevant concepts is used to formulate a concept query.
3. Then the concept query is used to perform a knowledge based retrieval process, obtaining as result a semantic ranking of relevant documents along with its semantic metadata. The knowledge based retrieval process will be detailed in section 3.1.1.2.

### Query by concepts

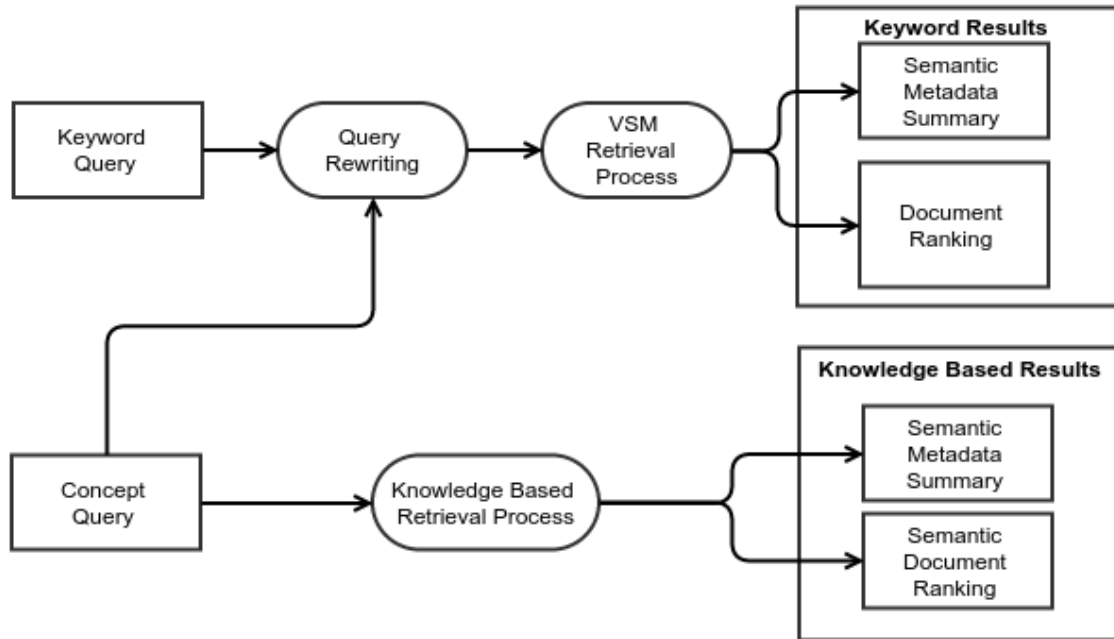


**Figure 3.2.:** KBMed retrieve by concepts process

In this scenario the user enters one or more concepts which represent its information need, then the retrieval process in figure 3.2 is executed as follows:

1. The concept query is used to formulate a keyword query, this keyword query is built using the words in the terms associated to the concepts in the original query.
2. Then a standard VSM retrieval process is executed obtaining as results a ranking of relevant documents along with its semantic metadata.
3. Simultaneously the concept query is used to perform a knowledge based retrieval process, obtaining as result a semantic ranking of relevant documents along with its semantic metadata.

### Query by both keywords and concepts



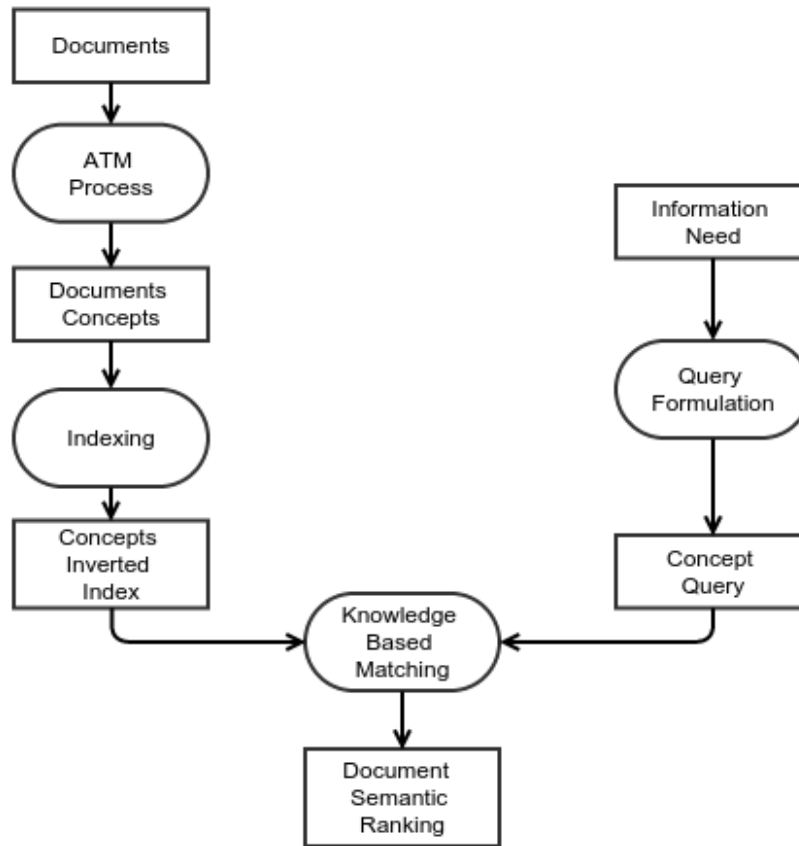
**Figure 3.3.:** KBMed retrieve by both keywords and concepts process

In this scenario the user enters one or more keywords and one or more concepts which represent its information need, from that input a keyword query and a concept query are formulated, then the retrieval process in figure 3.3 is executed as follows:

1. Keyword query is rewritten adding to it the words present in the terms associated with the query concepts.
2. Then a standard VSM retrieval process is performed, obtaining as results a ranking of relevant documents along with its semantic metadata.

3. Simultaneously the concept query is used to perform a knowledge based retrieval process, obtaining as result a semantic ranking of relevant documents along with its semantic metadata.

### 3.1.2. Knowledge based retrieval process



**Figure 3.4.:** Knowledge based retrieval process

The knowledge based retrieval process implemented in KBMed is an adaptation of the VSM which redefine two aspects to make it compatible with a knowledge based search scenario.

The first aspect is the document representation which is based on concepts instead of words, that means that a document is represented by a vector of size  $n$  as follows:

$$d = (w_{c1}, w_{c2}, \dots, w_{cn}) \quad (3.1)$$

Where the values  $w_{c_i}$  to the weight associated to the  $i$ -esim concept for that document or zero if the document does not contain the concept, and  $n$  corresponds to the total number of different concepts identified in the collection.

It is important to note that the concepts used to represent the documents, corresponds to concepts in a knowledge source, which are previously identified by means of an ATM process executed by term-mapper a tool that will be described in a section below, and the weight associated which each concept is computed using a TF-IDF weighting scheme.

The second aspect is the similarity measure used to rank the documents according to its relevance for a given query, the measure used by this model is a variation of the extended cosine similarity, this measure was originally proposed to extend cosine similarity adding to it the capacity to handle term dependencies [43].

In this work this measure is extended as in [14] to handle concept semantic similarities by replacing the term correlations factor by the  $\text{sim}(c_i, c_j)$  factor which corresponds to the lin's semantic similarity computed over the "IS A" hierarchy of the knowledge source, this way it allows to compute a pair wise semantic similarity measure between the documents and the query.

$$\text{ext-cos}(d, q) = \frac{\sum_{i=1}^n \sum_{j=1}^n d_i * q_j * \text{sim}_{lin}(\text{concept}_i, \text{concept}_j)}{\sqrt{\sum_{i=1}^n d_i^2} * \sqrt{\sum_{i=1}^n q_i^2}} \quad (3.2)$$

It is important to note that the computation of this measure is very expensive in terms of computation time, because it requires to compute the product between the weight of all the concepts in the query and all the concepts in every document in the collection and its respective semantic similarity, thus it is infeasible for large collections.

To deal with this problem we propose to use an approximation to this measure, which only takes into account a subset of the documents, this subset consists of the documents which contains at least one of the  $k = 50$  most similar concepts to any of the query concepts. This way the measure is computed against the documents in the subset not the whole collection.

### 3.1.3. Aided query refinement

Within KBMed there are two strategies used to help users to formulate and narrow its information needs and thus build better queries. The first strategy is an input field with autocomplete capabilities which offer suggestions for words and concepts based on the characters introduced in the query field as can be seen in figure 3.5. This way the user can easily

find a concept which represents its information need and then formulate a concept query.

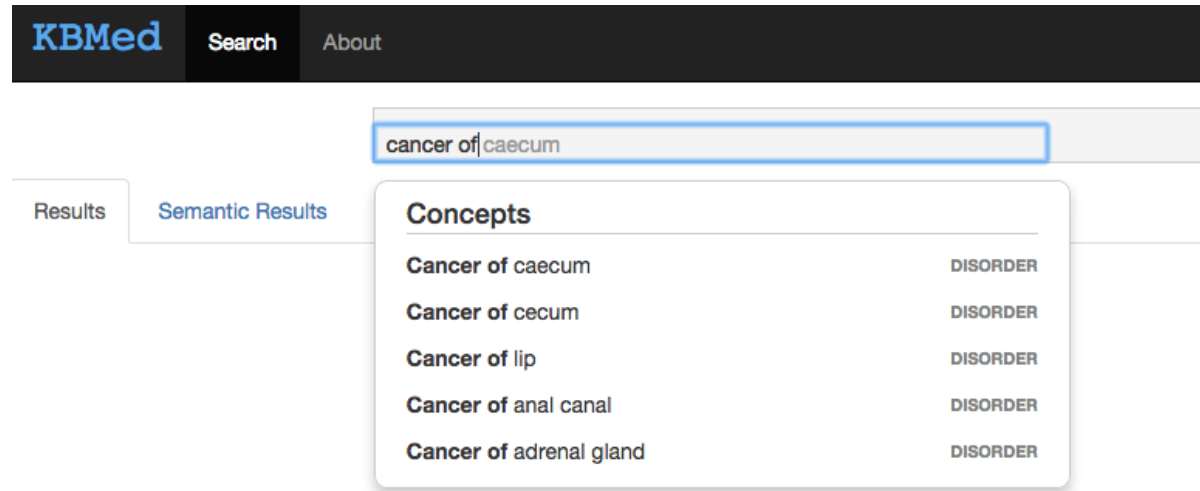


Figure 3.5.: KBMed concept suggestion based on prefix

The second strategy is posterior to a query and use the query concepts to suggest more specific concepts in the knowledge source hierarchy, punctually it suggest the direct childrens of the query concepts as can be seen in figure 3.6.

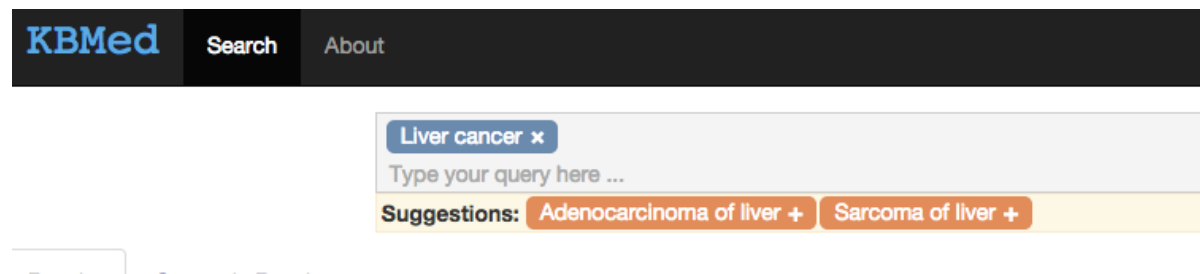


Figure 3.6.: KBMed concept suggestion based on hierarchy

### 3.1.4. Results visualization and exploration

KBMed include many features that improves the results visualization and exploration as can be seen in figures 3.7 and 3.8, such features include highlighting of both keyword and concepts matches, detailed metadata about concept annotations and faceting filtering using the knowledge source data.

The keywords highlighting functionality is useful because the user can quickly discard not relevant results by looking the context of the match which is provided as a text snippet. The concept annotations highlighting provide a quick way to find concepts in the text, in addition when the user click on an annotation the system shows a tooltip with additional information which include the concept description, its synonyms, and the hierarchical categorization of the concept which can be used to refine the search.

The faceting filtering is designed to provide a summary about the most significant concepts found within the subset of documents that match the query. The significance of the concepts is computed using a measure that compares the ratio between the frequency of the concept annotations in the results and the frequency in the whole collection as described in equation 3.3, this score is used to select the most significant concepts which will be used as facet filters.

$$\textit{significance} = \frac{\left(\frac{\textit{results-frequency}}{\textit{total-results}}\right)}{\left(\frac{\textit{global-frequency}}{\textit{total-documents}}\right)} \quad (3.3)$$

The screenshot shows the KBMed search interface. At the top, there is a navigation bar with 'KBMed', 'Search', and 'About'. Below this is a search bar containing the query 'cancer' and a 'Search' button. The results are displayed in two tabs: 'Results' and 'Semantic Results'. The 'Semantic Results' tab is active, showing a list of concepts with their respective counts:

Concept	Count
<b>BODY STRUCTURE</b>	
Entire breast	4534
Entire prostate	2182
Entire colon	2002
Entire lung	3079
Entire lymph node	1406
<b>PROCEDURE</b>	
Chemotherapy	2780
Radiation oncology	1366
Determination of prognosis	2106
<b>FINDING</b>	
Tumor progression	1496

Below the facets, three search results are shown, each with a title, authors, DOI, PubMed Central link, and a snippet:

- Metastatic colorectal cancer to a primary thyroid cancer**  
 Authors: Swain, Sarah; Serpell, Jonathan; Topliss, Duncan J; Moore, Maggie; Cherk, Martin H  
 DOI: 10.1186/1477-7819-6-122  
 PubMed Central: <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2588591>  
 ressection of oligometastatic metastatic colorectal cancer and chemotherapy. Conclusion Metastatic rectal
- Cancer treatments transform residual cancer cell phenotype**  
 Authors: Harless, William W  
 DOI: 10.1186/1475-2867-11-1  
 PubMed Central: <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3022788>  
 behaviors may have particular relevance to how cancer cells can be predicted to behave after treatments
- The cancer burden and cancer control in developing countries**  
 Authors: Pisani, Paola  
 DOI: 10.1186/1476-069X-10-S1-S2  
 PubMed Central: <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3073194>  
 contribution summarizes some of the priorities in cancer prevention in developing countries and the underlying
- Overexpression of Pim-1 in bladder cancer**  
 Authors: Qiu, Shaopeng; Huang, Bin; Chen, Junxing; Xu, Zhenbo; Jin, Chu; Mao, Xiaopeng; Guo, Shengjie  
 DOI: 10.1186/1756-9966-29-161  
 PubMed Central: <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3012037>  
 a role in different cancer types, however, the function of Pim-1 in bladder cancer is poorly understood

Figure 3.7.: KBMed results visualization

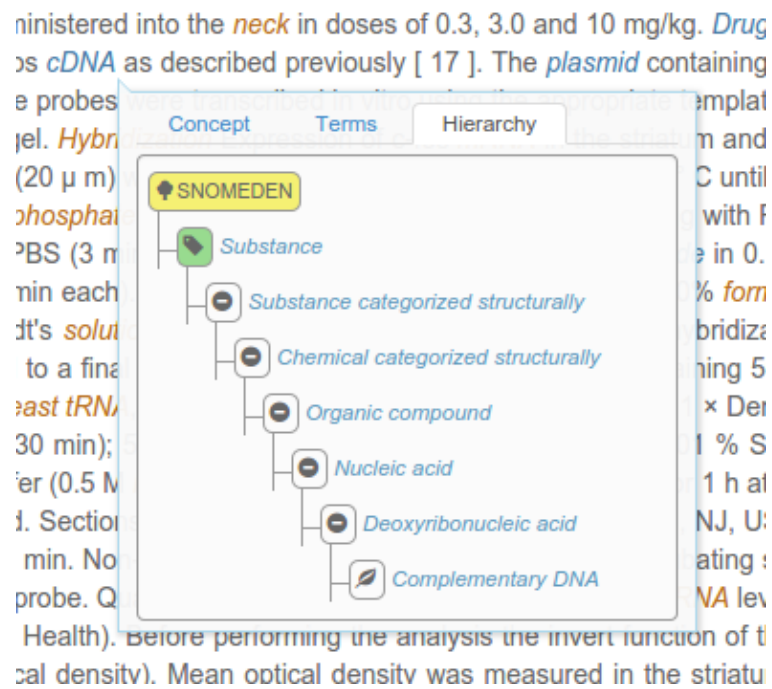


Figure 3.8.: KBMed documents metadata visualization using a tooltip

### 3.1.5. System architecture

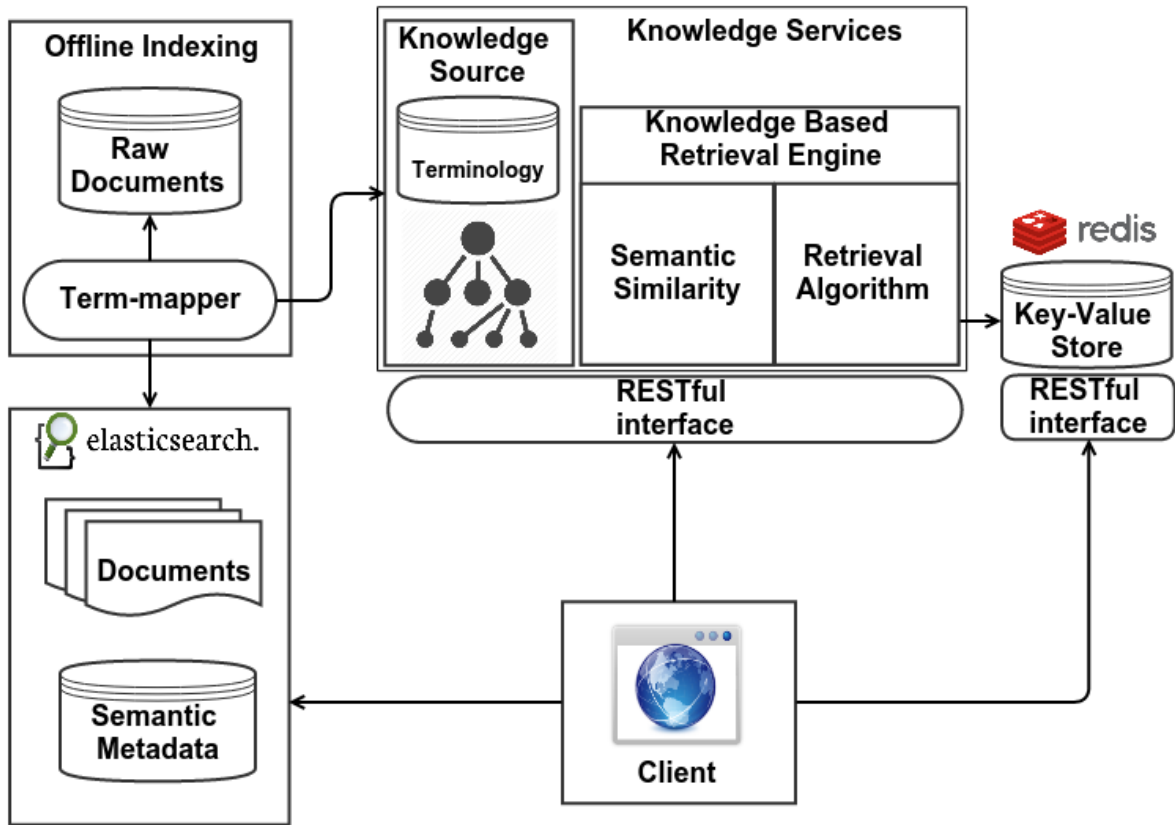


Figure 3.9.: KBMed components diagram

The architecture of KBMed was conceived as a service oriented architecture, in which each component exposes a set of services through RESTful interfaces that use JSON as common idiom. These RESTful interfaces are consumed by a web application which is the face of the system.

The main considerations followed in the design and implementation of KBMed can be summarized as follows:

- Build the entire system using only open source technologies
- Build a system easily scalable that can cope with real scale problems
- Build a system that can be extended to work in many domains and with any KS
- Build a system that can be easily integrated with existing FTS solutions

- Use a basic modern web architecture i.e. rich client which consume backend services

In the following sections a description of the system components along with its role and functionality is given.

## Document and semantic metadata repository

Elasticsearch<sup>1</sup> is an open source search server built in the top of Apache Lucene<sup>2</sup> the most popular full text search library which is available as an open source project.

In addition to its search capabilities Elasticsearch can be used as an efficient document store, this is because Lucene can store whole documents without incur in a significant overhead, this feature is exploited by KBMed to store both the document and the semantic metadata, this way all this information is obtained when a document is retrieved as a part of the answer to a query. A sample of a document stored within Elasticsearch can be seen in figure 3.10.

```
{
  "_id" : "5218f858b07842db15e64830",
  "article" : {
    "pmid" : "20672001",
    "title" : "Metastatic Melanomas Express Inhibitory Low Affinity Fc Gamma Receptor and Escape Humoral Immunity",
    "abstract" : "Our research, inspired by the pioneering works of Isaac Witz in the 1980s, established tha...",
    "fulltext" : "1. Introduction The analysis of different types of tumor biopsies by immunohistochemistry...",
    "doi" : "10.1155/2010/657406",
    "pmc-article-url" : "http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2905727",
    "figures" : [
      {
        "iri" : "DRP2010-657406.001",
        "caption" : " The human Fc gamma Receptors. Structure of Human Fc gamma..."
      },
      {
        "iri" : "DRP2010-657406.002",
        "caption" : " Tumor Fc γ RIIB1 expression inhibits metastatic melanoma growth..."
      }
    ],
    "authors" : {
      "author" : [
        "Loncar, Shannon M.",
        "Fridman, Wolf Herman",
        "Sautes-Fridman, Catherine"
      ]
    }
  }
}
```

Figure 3.10.: Example document within ElasticSearch

It is important to note that some parts of the semantic metadata are stored within the index in a way that they can be searchable, this allow to search documents annotated with certain concepts, this is done by add the set of distinct annotations as an independent field within the index.

<sup>1</sup><http://www.elasticsearch.org>

<sup>2</sup><https://lucene.apache.org>

```
{
  "mappings" : {
    "mappings_by_field" : {
      "title" : [
        {
          "id" : "55522006",
          "spans" : [
            "53-55"
          ]
        }
      ],
      "abstract" : [
        {
          "id" : "121122009",
          "spans" : [
            "1000-1004",
            "1207-1211"
          ]
        }
      ]
    },
    "all_mappings" : [
      "55522006",
      "121122009",
      "49616005"
    ]
  }
}
```

Figure 3.11.: Example semantic metadata within Elasticsearch

### Full text search engine

Elasticsearch provides a distributed and scalable solution that offers all the functionalities of Lucene through a RESTful API. Within KBMed we use Elasticsearch to provide the full text search capabilities of the system, and also to enable functionalities like term and concept suggestions, faceted navigation and result highlighting.

### Knowledge source

This component is a Java application which offer a variety of services to access the data in the knowledge source i.e. terms, concepts and relations. It also maintains an in memory representation of the ontology hierarchy in order to allow fast queries over this structure, and also give access to pre-computed values about the information content and similarity between concepts which are required by other components in different process.

### Knowledge based retrieval engine

This component is a Java application that implements the retrieval algorithm described in section 3.1.1.2.

## Knowledge services

This component is a Java application which offers an unified RESTful API interface to access the functionalities in the knowledge source and knowledge based retrieval engine components, this API will be described in detail in the appendix 2.

## Key-Value Store

Redis<sup>3</sup> is an open source key-value store which is fast, lightweight and easily scalable. Within KBMed we use Redis to persist data like:

- Pre-computed information content of the concepts in the knowledge source
- Pre-computed semantic similarity between the concepts in the knowledge source
- An inverted index which maps annotations to documents

It is also used to store volatile data which only is retained for a short time period i.e. a search request, example of this kind of data are knowledge based rankings and the sub-hierarchies used to filter results.

## Web application client

The screenshot shows the KBMed web application interface. At the top, there is a search bar with the text 'cancer of liver' and 'Liver structure' entered. Below the search bar, there are suggestions: 'sarcoma of liver' and 'adenocarcinoma of liver'. The main content area is divided into two columns. The left column is a sidebar with categories: 'BODY STRUCTURE', 'SUBSTANCE', and 'FINDING'. Under 'BODY STRUCTURE', there are items like 'Liver structure' (3232), 'Entire colon' (757), 'Entire lung' (1301), 'Entire breast' (1122), 'Entire trunk of portal vein' (206), and 'Entire bile duct' (169). Under 'SUBSTANCE', there are 'Liver tissue' (538), 'Alanine aminotransferase' (392), 'Bilirubin' (289), and 'Bile' (250). Under 'FINDING', there is 'Liver normal' (368). The right column displays search results for 'Treatment of colorectal liver metastases', 'STAT3 activation in monocytes accelerates liver cancer progression', 'biomarkers for liver cirrhosis and hepatocellular cancer in individuals with fatty liver disease', 'multidisciplinary management of stage IV colorectal cancer with liver metastases', and 'Clinicopathologic and gene expression parameters predict liver cancer prognosis'. Each result includes authors, DOI, and PubMed Central links.

Figure 3.12.: KBMed web application snapshot

This is a web application developed using technologies such as HTML5, CSS3, Bootstrap, Javascript, jQuery, Backbone JS, etc. It is a rich client which invokes through AJAX the

<sup>3</sup><http://redis.io>

RESTful services exposed by the other components in the system and dynamically renders the appropriate views.

It is important to note that this client works in two different ways, depending of the estimated time required by the back-end service to process a request. If the response can be generated immediately by the back-end it works in a near to synchronous way which waits an immediate response.

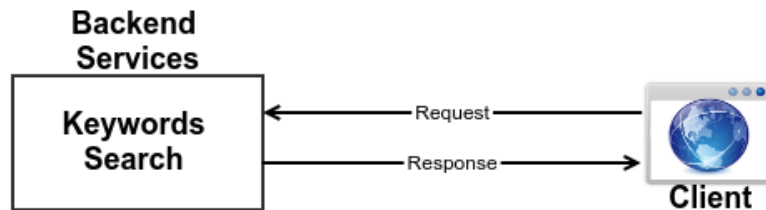


Figure 3.13.: Synchronous processing

In the other case, if is expected that the response takes a long time, the back-end generates a response with a process token which is used later by the web application to obtain the result once this is generated. This way it works asynchronously using Redis as intermediary to decouple long run processing tasks from HTTP requests/response, thus offering a better user experience.

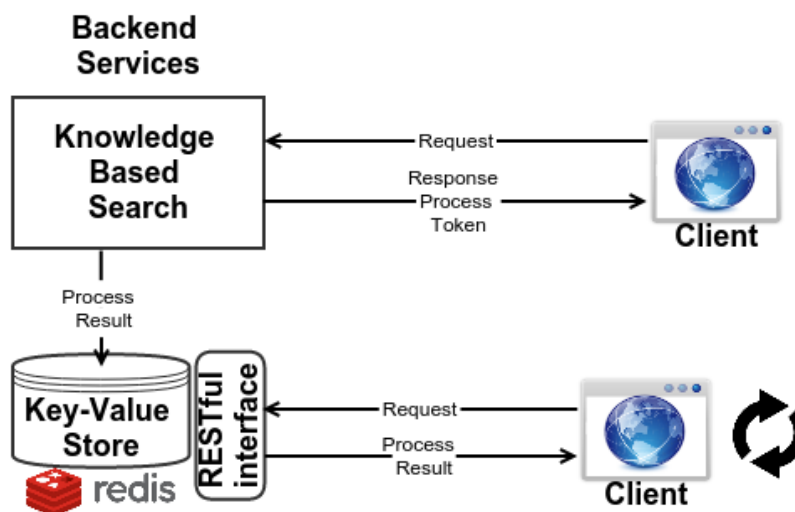


Figure 3.14.: Asynchronous processing

## 3.2. Term-mapper

Term-mapper is an ATM software which was developed with the aim of build a tool that offers a good balance between complexity and speed, that can fill some of the gaps left by available tools, in that sense the main goals pursued while developing Term-mapper were:

- Develop a tool capable of work with many languages initially English and Spanish.
- Develop a tool that can be easily extended to work with any knowledge source.
- Develop a tool that does not rely on manually build resources, this is motivated by the fact that terminology resources are growing and changing constantly [4], so is desirable that methods for ATM does not depends of this kind of resources which are costly and hard to maintain.
- Develop a fast tool that can scale to thousands or even millions of documents.

The ATM method implemented by Term-mapper is inspired by the method of Metamap [1] with some modifications motivated by the goals explained before. In the following sections an overview of the approaches used by Term-mapper for solve ATM challenges is given, after that a detailed description of the mapping algorithm is presented.

### 3.2.1. Dealing with language variability

Language variability is a big challenge for ATM methods, Term-mapper use three different strategies to cope with this problem.

The first strategy tries to cope with acronym mentions, it use a regular expression based recognition of author acronym along with an acronym dictionary, which in combination with the method in [40] are used to identify acronym mentions along with its extended forms. These mentions are directly mapped and therefore skip the remaining mapping process.

The second strategy tries to deal with orthographic variations, it use the spell checking capabilities of Apache Lucene and a previously build lexicon index to identify and correct misspelled words, the lexicon is build using the words in the knowledge source and a set words that comes from an external source, which in our case corresponds to a list of around 250000 words.

The third strategy is the use of an stemmer for the corresponding language in order to deal with morphological variations, in particular we use the Snowball implementation [32].

### 3.2.2. Ambiguity resolution

The ambiguity resolution strategy is based on the work of Schadd et al. [38] which proposes a disambiguation method for ontology alignment based on the construction of virtual documents and standard IR techniques.

The idea of this method is to build a new collection of so called virtual documents which are composed by the information in the knowledge source and are used to represent each concept along with its context. The information used to build the virtual documents includes the concept itself, its description or definition, its synonyms and the set of concepts which are directly connected with him which are referred as neighbors, a simplified representation of a virtual document can be seen in figure 3.15.

<b>290006</b>
Category: disorder
Description: Melnick-Fraser syndrome
Terms: Melnick-Fraser syndrome BOR syndrome Branchio-oto-renal syndrome
Neighbors: Kidney Renal structure Congenital anomaly of kidney Congenital malformation

**Figure 3.15.:** SNOMED-CT virtual document representation example

Once the virtual documents collection and the inverted index are created, when an ambiguous mapping appears a standard retrieval process using a VSM approach is performed, this retrieval process is executed using as query a portion of the sentence which is currently mapped, in particular, we use a portion of up to ten tokens before and after of the span which englobes the ambiguous candidate mapping.

Then the results of the retrieval process are used to select the correct sense, choosing the concept associated with the virtual document which ranks first for the query among the possible concepts senses. If this process is not enough to solve the ambiguity then the most concrete concept is selected using information content as criteria, if the ambiguity persist then all mappings are preserved as possible senses.

### 3.2.3. Term-mapper algorithm

The mapping algorithm implemented by Term-mapper requires a group of index structures that allows for fast lookups over the terminology and the words in a lexicon, these index must be constructed as a previous step or step zero. As shown in figure 3.16 the main inverted index is built starting from the terminology, and the spell index is built using the words from both the terminology and the general lexicon. Finally the virtual documents required for the disambiguation process are generated and indexed using Apache Lucene.

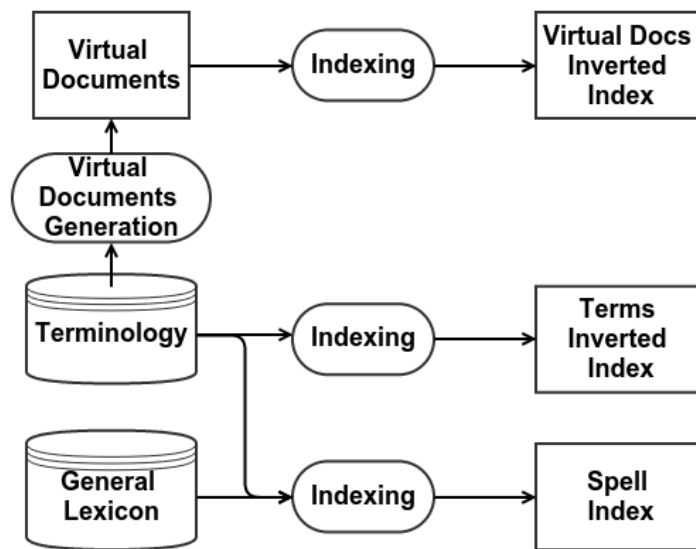


Figure 3.16.: Term-mapper indexing

Once these index are built, Term-mapper is ready to perform the mapping process, this process receives as input a set of documents, each document consist of an identifier and and a set of fields along with its corresponding content. The output of the process consist of a set of annotations for each document field, an annotation consist of a concept or term identifier along with its exact position within the field value.

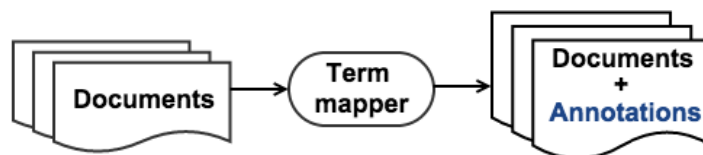


Figure 3.17.: Term-mapper mapping overview

The mapping algorithm in Term-mapper is composed of four main steps as shown in figure 3.18, a detailed description of each one of these steps is given below.

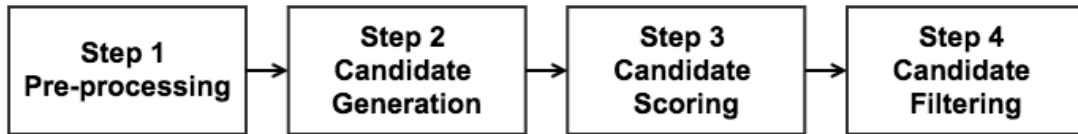


Figure 3.18.: Term-mapper mapping algorithm overview

### Preprocessing

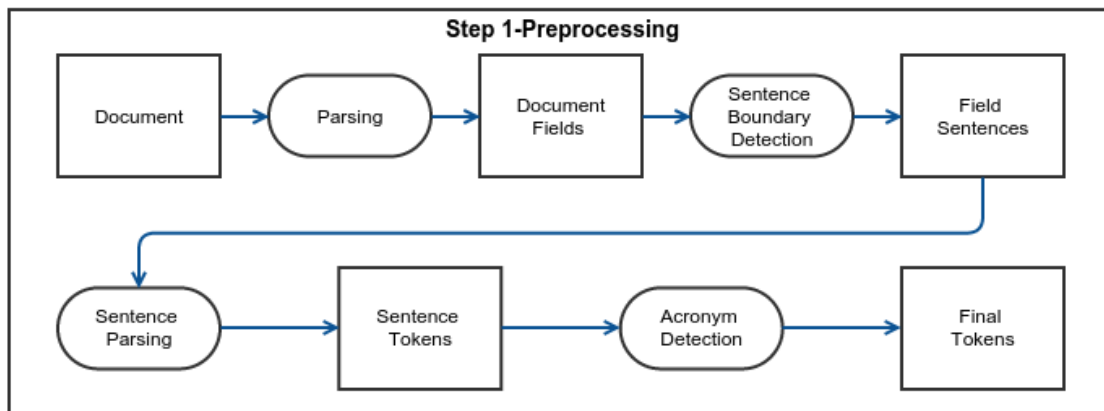
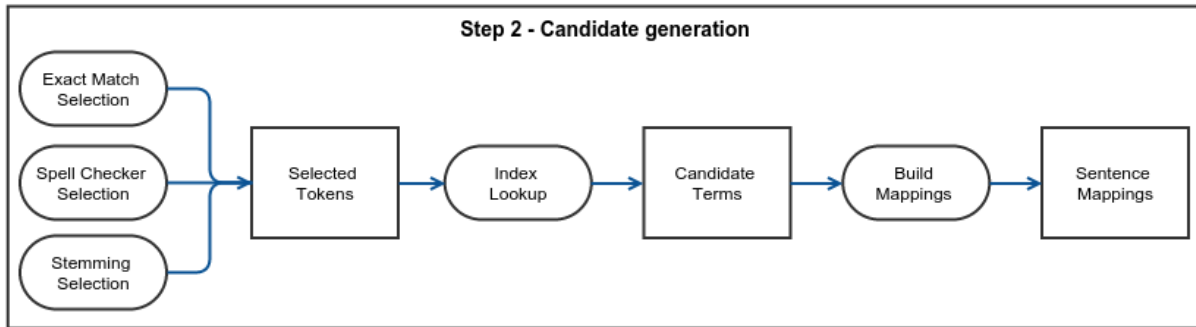


Figure 3.19.: Term-mapper mapping algorithm pre-processing step

In this step the document is loaded and converted to a suitable representation, then for each field in the document a process of segmentation by sentences followed by a term normalization and classification process is executed. The result of this phase is a list of tokens classified either as acronyms from a dictionary, candidate acronyms according to its shape, words, stop words, numbers or alphanumerics.

The candidate acronyms are evaluated if they are already identified as author acronyms they are marked and the process continues, if not they are evaluated trying to identify its long form within the scope of the current sentence, if a match is obtained they are added as author acronyms if not they are discarded and the process continues.

## Candidate generation



**Figure 3.20.:** Term-mapper mapping algorithm candidate generation step

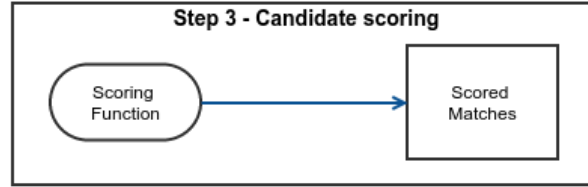
In this step all tokens classified as words in the previous step are used to build candidate terms set. There are three term selection strategies which are used to obtain candidate terms. The first strategy is look for an exact match. The second strategy use a spell checker to deal with typing errors and simple orthographical variations, in this strategy each word that is not contained within the spell index is replaced by the most frequent word among the most similar words found above a threshold of 0.85, the similarity is computed using a normalized score based on the Levenshtein distance. The third strategy use a stemmer to deal with morphological variations.

Then for each term in the candidate terms set a representation of the mapping is constructed, this new representation is more suitable for the subsequent steps and contains the index of the matches in both the term and the sentence along with other data that will be required in future steps.

It is important to note that each candidate term can appear more than one time in a sentence, so one candidate term can generate multiple mappings. The method used to deal with this scenario is based in a windowed scanning of the sentence tokens, that splits a mapping if finds a gap in the match greater than twice the length of the term in tokens.

## Candidate scoring

In this step each candidate mapping is evaluated using a scoring function, the score obtained reflects the quality of the mapping and is used in a later step to discard bad mappings. This function is inspired by Metamap's scoring function [1], but is simpler and do not take into account any language dependant measure. The score is computed as follows:



**Figure 3.21.:** Term-mapper mapping algorithm candidate scoring step

$$score = \alpha_1 * coverage + \alpha_2 * variation + \alpha_3 * dispersion \quad (3.4)$$

The coverage indicates how much tokens in the candidate term are matched by the text chunk:

$$coverage = \frac{\sum_i w_i * t_i}{\sum_i w_i} \quad (3.5)$$

The variation indicates the quality of the matches, penalizing those matches that are only possible if the spelling corrector or the stemmer are used.

$$mapping\_variation = \frac{1}{total\_variation^2} \quad (3.6)$$

Where the total variation is the sum of each token variation which depends of the match type as follows:

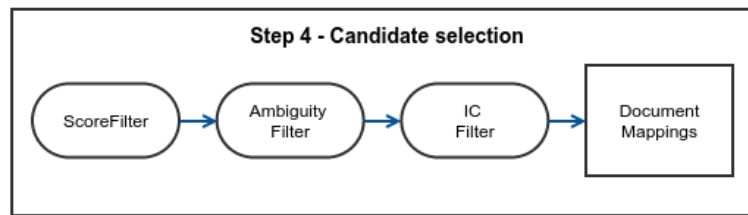
$$token\_variation = \begin{cases} 0 & \text{exact match} \\ 1 & \text{match after spelling} \\ 2 & \text{match after stemming} \end{cases} \quad (3.7)$$

Finally the dispersion indicates the cohesion of the match, penalizing matches which are not in the expected position according to the term order.

$$dispersion = \frac{1}{\sum_i |termPosition_i - centeredSentencePosition_i|} \quad (3.8)$$

### Candidate selection

The score computed in the previous step is used to rank and choose the candidates that will be used as mappings. Ambiguity can occur because a tie in the scores over the same sentence tokens in such case the disambiguation method described in section 3.2.2 is used.



**Figure 3.22.:** Term-mapper mapping algorithm candidate selection step

## 4. Evaluation

This chapter presents the results obtained in the SemEval (Semantic Evaluation) 2014 competition which are used to measure the performance of Term-mapper. It also includes some results about the performance of the proposed knowledge based retrieval approach, it is important to note that an evaluation of all the features of KBMed requires both user validation and the appropriate dataset, but this is out of the scope of the current work.

### 4.1. Term-mapper performance

#### 4.1.1. Semeval dataset

The data for SemEval 2014 analysis of clinical texts task <sup>1</sup>, consists of the ShARe (Shared Annotation Resource) corpus, which contains clinical notes from MIMIC database (Multi-parameter Intelligent Monitoring in Intensive Care). The data were manually annotated for disorder mentions, normalized to a UMLS Concept Unique Identifier (CUI) when possible, and marked as CUIless otherwise. The concepts used in this task corresponds to SNOMED-CT concepts that are mapped within a predefined set of semantic categories within UMLS.

Four types of reports were found in the training corpus: 61 discharge summaries, 54 ECG reports, 42 ECHO reports and 42 radiology reports, for a total of 199 documents, each annotated with several disorder mentions for a total of 5816 annotations from which 4175 corresponds to mapped annotations i.e. have a CUI assigned, and 1641 corresponds to CUI-less annotations.

The annotations are stored in a pipe format which contains the file name, the semantic category of the annotation, the CUI and the exact position of the annotation using character offsets as can be seen in the following example.

---

<sup>1</sup><http://alt.qcri.org/semeval2014/task7>

---

016139-DISCHARGE\_SUMMARY.txt | Disease\_Disorder | C0010520 | 1228 | 1236

**Figure 4.1.:** Pipe annotation example

### 4.1.2. Experimental setup

The analysis of clinical texts task of SemEval 2014 consist of two tasks, the task A is focused on named entity recognition (NER), while the task B is focused on automatic term mapping (ATM).

In the task A the performance is measured using precision, recall and f-measure as metrics and using two different modes to consider an entry as valid, in the first one denominated strict an entry is valid only if its corresponding spans match exactly the actual ones, in the second case denominated relaxed an entry is considered as correct if its spans have an overlap with the actual ones.

In the task B the performance is measured with two different metrics which are described below:

$$strict\_accuracy = \frac{true\_positive}{total\_annotations} \quad (4.1)$$

$$relaxed\_accuracy = \frac{true\_positive}{total\_correct\_spans} \quad (4.2)$$

Using the Java libraries Apache OpenNLP<sup>2</sup> and Maxent [5], a maximum entropy model was implemented for Named Entity Recognition (NER). Two types of classifiers were built: the first one using the library default configuration, and a second one including additional features. The default model includes the following attributes: target word, two words of context at the left of the target word, two words of context at the right of the target word, type of token for target word (capitalized word, number, hyphen, commas, etc.), and type of token for words in the context. For the enhanced model, we included n-grams at character level extracted from the target word, going from two to five characters.

OpenNLP uses the BIO tagging scheme, which marks each token as either beginning a chunk, continuing it, or not in a chunk, therefore, this model cannot identify discontinuous terms. Given this, we excluded discontinuous term annotations from the training data, and trained the model with the resulting corpus.

---

<sup>2</sup><http://opennlp.apache.org>

According to preliminary experiments, the chosen enhanced NER method exhibited low precision, i.e. a high number of false positives. To deal with this problem we calculated a measure for the specificity of a candidate named entity with respect to a specialized corpus, this quantity is based on the weirdness [18] of the candidate words. Those words that are common to any domain will very likely to have a low weirdness score, while those with a high weirdness score. Using around 1000 books from the Guttenberg project as the general corpus, and the terms in UMLS as the specialized corpus, we applied the weirdness measure to those words that according to the NER model, are disorders. By keeping only those with high weirdness measures, we prevent our system from tagging words that are not even medical vocabulary, thus reducing the amount of false positives.

The test corpus consist of 133 discharge summaries. Three submissions were made, the run 0 was intended as a baseline; run 1 used Metamap for UMLS concept mapping and run 2 did this using Term-mapper. Both runs 1 and 2 used the enhanced features for NER and applied the weirdness measure.

For run 0, the documents were processed with Metamap and those concepts mapped to a CUI belonging to one of the desired UMLS semantic types were chosen. Parallel to this, the document was tagged using the default NER model.

Finally, results were merged, preferring Metamap mapping outputs in the cases where a concept was mapped by both tools.

Run 1 differs from run 0 in two steps of the process: the NER model included the enhanced features described previously and its output was filtered, keeping only those concepts whose weirdness measure exceeds 0.7.

Finally, run 2 was equal to run 1, with the difference that Term-mapper was used to map concepts to the UMLS meta thesaurus.

### 4.1.3. Results

The following tables present the official results obtained in both tasks of SemEval 2014 analysis of clinical text.

As can be seen Term-mapper obtain the best results among our runs, these results proves

Rank	Run	Strict precision	Strict Recall	Strict F-measure
1	BEST	0.843	0.786	0.813
31	Term-mapper	0.561	0.534	0.547
32	Metamap	0.578	0.515	0.545
37	Baseline	0.321	0.565	0.409

**Table 4.1.:** Task A official results ranked by strict F-measure

Rank	Run	Relaxed precision	Relaxed Recall	Relaxed F-measure
1	BEST	0.916	0.907	0.911
35	Term-mapper	0.769	0.677	0.720
37	Metamap	0.777	0.654	0.710
40	Baseline	0.439	0.725	0.547

**Table 4.2.:** Task A official results ranked by relaxed F-measure

that Term-mapper is a viable alternative for mapping concepts to clinical terminologies. This result is particularly interesting given that Term-mapper has unique features i.e. it works for many languages, with any knowledge source and does not rely on manually built resources. Additionally Term-mapper is considerably faster than Metamap, as an example to annotate the 133 documents of this task Metamap took 581 minutes while Term-mapper only requires 96 minutes which represents a speedup close to 6 times.

---

Rank	Run	Strict accuracy
1	BEST	0.741
19	Term-mapper	0.461
24	Metamap	0.411

**Table 4.3.:** Task B official results ranked by strict accuracy

Rank	Run	Relaxed accuracy
1	BEST	0.928
11	Term-mapper	0.863
21	Metamap	0.771

**Table 4.4.:** Task B official results ranked by relaxed accuracy

## 4.2. Knowledge based retrieval model results

### 4.2.1. Clef dataset

The data for CLEF (Cross Language Evaluation Forum) 2013 ad hoc image retrieval, consists of articles from the MEDLINE database. These articles are composed by text and images which are delivered to participants in XML and JPG respectively. The total number of documents is 74654 while the total number of images is 306539.

The goal of the ad hoc image retrieval task is to measure the performance of the multi-modal retrieval models proposed by the participants. This performance is measured using a set of 35 queries which are supplied along with a manually built ranking of the first 1000 documents that match each query. The queries are composed by a short text and a small group of sample images, both text and images can be used to execute the retrieval process. The queries are categorized as visual, semantic or mixed, according to the methods that the organizers expect to yield the best results. It is important to note that this task is focused on image retrieval, so the ranking entries must correspond to image identifiers not documents.

### 4.2.2. Experimental setup

As a previous step to our retrieval experiments all the documents in the CLEF collection were annotated with SNOMED-CT concepts using Term-mapper, this annotation task produce the semantic metadata required by some of the retrieval process which will be tested. In total Term-mapper found around of 19 millions of concept mentions in the collection.

Given that some of the retrieval process require that the query was given in the form of concepts instead of keywords, a manual annotation process was performed, this process consist on identify the concepts mentions that appears within each query. As result of this process 33 queries were annotated the other 2 queries does not contain any concept mention that match a SNOMED-CT concept.

As baseline we use the results obtained from a standard VSM retrieval process implemented using Apache Lucene. The performance of all the retrieval experiments was measured using the “trec-eval” tool, from which we obtain values for precision, recall and map for all the CLEF queries.

### 4.2.3. Results

The following table presents the results obtained for each tested retrieval approach. These approaches cover all the query schemes used by KBMed.

Retrieval process	Solved Queries	Relevant Documents Retrieved	MAP
Standard VSM Section 2.2.6.1	35	477	0.1166
KB retrieval by keywords	35	110	0.0237
KB retrieval by concepts	30	82	0.0326
Concept words	33	448	0.1003
Keywords and concepts words	<b>35</b>	<b>526</b>	<b>0.1204</b>

**Table 4.5.:** Retrieval results over CLEF dataset

As can be seen the performance of the methods which use the knowledge based retrieval approach is not as expected, in both cases they produce worse results than the baseline. This can be related with many factors which include the nature of the dataset, the goal of the CLEF task, the completeness of the knowledge source and the performance of the previous mapping task.

The results obtained by the approaches which use the knowledge source to formulate and expand queries are better, the first scenario in which the words in the concepts are used to formulate a keyword query shows a performance which is close to baseline, while the scenario which extends the keywords query by adding the words in the concepts outperforms the baseline.

# 5. Conclusions

## 5.1. Conclusions

The proposed automatic term mapping method has shown to be equivalent in terms of performance and significantly faster than Metamap [2], this result is particularly interesting given that Term-mapper approach has many advantages when compared to Metamap, i.e. it works for many languages, can be easily extended to work with any knowledge source and does not rely on manually built resources.

The proposed knowledge based retrieval approach exhibits low performance, this shows that a representation based on concepts along with the match process based on semantic similarity measures is worse than the standard vector space model approach using words and cosine similarity for the CLEF ad hoc retrieval task. It is important to note that this result can be influenced by the performance of the preliminary automatic term mapping task, given that if you do not have accurate mappings the document representation will be noisy, and therefore the comparisons based on such representation are not very reliable.

Despite the low performance of the knowledge based retrieval approach, the system offers many advantages, as an example, the query refinement and result visualization capabilities, are a clear improvement when compared to traditional retrieval systems which only offer a ranking of documents in response to a query.

Finally the results obtained by the methods which use the knowledge source to reformulate and expand the queries using the words in the concepts query, shows good performance and even outperforms the baseline. This is interesting because it demonstrates a way in which the semantic capabilities of the system can contribute to improve the retrieval performance.

## 5.2. Future work

- Improve automatic term mapping pre-processing pipeline by include a context dependent spelling corrector which can improve term selection accuracy.
- Use a custom chunking model to divide sentences in correlated text portions according

to the sentence syntactic structure.

- Extend KBMed query support to allow query by wide categories and improve suggestions strategy using relations other than “IS A”.

# A. Software documentation

## A.1. Term-mapper documentation

### A.1.1. Term-mapper command line interface

Term-mapper can be used from the command line by means of a service wrapper which support the following parameters:

- *-lang* or *-l*, for language choice can be either *en* for English or *sp* for Spanish.
- *-terms* or *-t*, the path to the file which contains the terms to be mapped.
- *-gen-voc* or *-g*, An optional path to a file with additional vocabulary for the spelling corrector.
- *-in-path* or *-i*, An optional path to the directory where the documents to be mapped are located.
- *-out-path* or *-o*, An optional path to the directory where the mappings will be located.
- *-term-select* or *-ts*, An optional parameter which allows to select the term selection strategy can be either 1 for exact-spell-stem, 2 for exact-spell, 3 for exact-stem or 4 for exact.
- *-mongodb-url*, If the documents to map are stored in MongoDB, the url of the collection in the form `mongodb://localhost:27017/db.collection`.
- *-es-url*, If the mappings will be stored in elasticsearch, the url of the Elasticsearch node in the form `http://localhost:9300`.
- *-numThreads* or *-nt*, The number of parallel mapping process to run.
- *-help* or *-h*, to obtain help.

## A.2. REST knowledge services interface documentation

General parameters:

- `knowledgeSourceName`: the name of the knowledge source to use i.e. `snomed-en`

## A.2.1. Concepts endpoints

### Retrieve concept information

- URL: `/knowledgeSourceName/concepts/{conceptIds}`
- Method: GET
- Parameters
  1. `conceptIds`: a sequence or one or more valid concept identifiers separated by comma.
- Response: A JSON array in which each item corresponds to the detailed information about the corresponding concept i.e. id, category, description and preferred term. The array order corresponds to the `conceptIds` parameter order.

```
[
  {
    "id": "74732009",
    "category": "disorder",
    "description": "Mental disorder",
    "prefTerm": "mental disorder"
  }
]
```

Figure A.1.: Concept information JSON response example

### Retrieve a concept summary

- URL: `/knowledgeSourceName/concepts/{conceptId}/summary`
- Method: GET
- Parameters
  1. `conceptId`: a valid concept identifier.
- Response: A JSON object which contains detailed information about the concept, its associated terms, its direct childrens within the “IS A” hierarchy and its path to the root concept.

```

{
  "concept": {
    "id": "74732009",
    "category": "disorder",
    "description": "Mental disorder",
    "prefTerm": "mental disorder"
  },
  "terms": [ ... ], // 7 items
  "pathToRoot": [ ... ], // 4 items
  "childrens": [ ... ] // 34 items
}

```

Figure A.2.: Concept summary JSON response example

## A.2.2. Knowledge based retrieval engine endpoint

### Knowledge based retrieval

- URL: `/knowledgeSourceName/kbretrieval`
- Method: POST
- Parameter: A JSON body with the following form.

```

query : {
  keywords: [
    "keyword1",
    "keywordn",
  ],
  concepts: [
    "conceptId1",
    "conceptIdn",
  ],
}

```

Figure A.3.: KB retrieval JSON body example

- Response: A JSON array which contains the document ranking, each entry in the array contains a document identifier along with the score obtained.

## A.2.3. Hierarchy endpoints

### Retrieve the parents of a concept

- URL: `/knowledgeSourceName/hierarchy/{conceptId}/parents`

- Method: GET
- Parameters
  1. conceptId: a valid concept identifier.
- Response: A JSON object which contains detailed information about the parents of the given concept within the “IS A” hierarchy.

```
[
  {
    "id": "64572001",
    "category": "disorder",
    "description": "Disease",
    "prefTerm": null
  }
]
```

**Figure A.4.:** Concept parents JSON response example

### Retrieve the childrens of a concept

- URL: /{knowledgeSourceName}/hierarchy/{conceptId}/childrens
- Method: GET
- Parameters
  1. conceptId: a valid concept identifier.
- Response: A JSON object which contains detailed information about the childrens of the given concept within the “IS A” hierarchy.

### Retrieve the path to root of a concept

- URL: /{knowledgeSourceName}/hierarchy/{conceptId}/path/root
- Method: GET
- Parameters
  1. conceptId: a valid concept identifier.

```
[
  {
    "id": "192616009",
    "category": "disorder",
    "description": "Childhood or adolescent disorder of social functioning",
    "prefTerm": null
  },
  {
    "id": "56641006",
    "category": "disorder",
    "description": "Axis II diagnosis",
    "prefTerm": null
  },
]
```

Figure A.5.: Concept childrens JSON response example

- Response: A JSON array which contains detailed information about the concepts that appears in the path to the root for the given concept within the “IS A” hierarchy.

```
[
  ▼ {
    "id": "74732009",
    "category": "disorder",
    "description": "Mental disorder",
    "prefTerm": null
  },
  ▼ {
    "id": "64572001",
    "category": "disorder",
    "description": "Disease",
    "prefTerm": null
  },
  ▼ {
    "id": "404684003",
    "category": "finding",
    "description": "Clinical finding",
    "prefTerm": null
  },
  ▼ {
    "id": "138875005",
    "category": "SNOMED RT+CTV3",
    "description": "SNOMED CT Concept",
    "prefTerm": null
  }
]
```

Figure A.6.: Concept path to root JSON response example

# Bibliography

- [1] A R Aronson. Effective mapping of biomedical text to the UMLS Metathesaurus: the MetaMap program. *Proceedings AMIA Symposium*, pages 17–21, January 2001.
- [2] Alan R Aronson and François-Michel M Lang. An overview of MetaMap: historical perspective and recent advances. *Journal of the American Medical Informatics Association JAMIA*, 17(3):229–236, May 2010.
- [3] M Ashburner, C A Ball, J A Blake, D Botstein, H Butler, J M Cherry, A P Davis, K Dolinski, S S Dwight, J T Eppig, M A Harris, D P Hill, L Issel-Tarver, A Kasarskis, S Lewis, J C Matese, J E Richardson, M Ringwald, G M Rubin, and G Sherlock. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nature genetics*, 25(1):25–9, May 2000.
- [4] Michael Bada. Mapping of biomedical text to concepts of lexicons, terminologies, and ontologies. *Methods in molecular biology (Clifton, N.J.)*, 1159:33–45, January 2014.
- [5] Adam L Berger, Vincent J Della Pietra, and Stephen A Della Pietra. A maximum entropy approach to natural language processing. *Comput. Linguist.*, 22(1):39–71, March 1996.
- [6] N Bhatia, N H Shah, D L Rubin, A P Chiang, and M A Musen. Comparing Concept Recognizers for Ontology-Based Indexing: MGREP vs. MetaMap. In *AMIA Summit on Translational Bioinformatics*, 2009.
- [7] C. Buckley, G. Salton, J. Allan, and Singhal Amit. Automatic Query expansion using SMART : TREC 3. *NIST special publication*, pages 69–80, 1995.
- [8] M. Dai, N. H. Shah, W. Xuan, M. A. Musen, S. J. Watson, B. D. Athey, and F. Meng. An efficient solution for mapping free text to ontology terms. In *AMIA Summit on Translational Bioinformatics 21*, 2008.
- [9] S Deerwester, S T Dumais, G W Furnas, T K Landauer, and R Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

- 
- [10] Ann Devitt and Carl Vogel. The topology of wordnet: Some metrics. In *Proceedings of the 2nd International WordNet Conference GWC*, pages 106–111, 2004.
- [11] Andreas Doms, Michael Schroeder, Doms A, and Schroeder M. GoPubMed: exploring PubMed with the Gene Ontology. *Nucleic Acids Research*, 33:W783–6, 2005.
- [12] Davies John Goker Ayse. *Information Retrieval: Searching in the 21st Century*. John Wiley & Sons, 2009.
- [13] R. Guha, Rob McCool, and Eric Miller. Semantic search. In *Proceedings of the twelfth international conference on World Wide Web - WWW '03*, pages 700–709, New York, New York, USA, May 2003. ACM Press.
- [14] Angelos Hliaoutakis, Giannis Varelas, and Euripides G M Petrakis. MedSearch : A Retrieval System for Medical. In *Research and Advanced Technology for Digital Libraries. Springer Berlin Heidelberg*, pages 512–515, 2006.
- [15] Rezarta Islamaaj Dogan, G Craig Murray, Aurélie Névéol, and Zhiyong Lu. Understanding PubMed user search behavior through log analysis. *Database : the journal of biological databases and curation*, 2009, January 2009.
- [16] Clement Jonquet, Nigam H Shah, and Mark A Musen. The open biomedical annotator. In Amia Summit On Translational Bioinformatics, editor, *Summit on translational bioinformatics*, volume 2009, pages 56–60, January 2009.
- [17] Eimear E Kenny, Paul W Sternberg, Hans-Michael Müller, Eimear E Kenny, and Paul W Sternberg. Textpresso: an ontology-based information retrieval and extraction system for biological literature. *PLOS Biology*, 2(11):e309, November 2004.
- [18] Ahmad Khurshid, Lee Gillam, and Lena Tostevin. Weirdness Indexing for Logical Document Extrapolation and Retrieval (WILDER). In *TREC 1999*, pages 1–8, 1999.
- [19] Jung-Jae Kim, Piotr Pezik, and Dietrich Rebholz-Schuhmann. MedEvi: retrieving textual evidence of relations between biomedical concepts from Medline. *Bioinformatics (Oxford, England)*, 24(11):1410–2, June 2008.
- [20] Youngjun Kim, John Hurdle, and Stéphane M Meystre. Using UMLS lexical resources to disambiguate abbreviations in clinical text. *AMIA Symposium*, 2011:715–22, January 2011.
- [21] Michael Krauthammer and Goran Nenadic. Term identification in the biomedical literature. *Journal of biomedical informatics*, 37(6):512–526, December 2004.

- 
- [22] C Leacock and M Chodorow. Combining Local Context and Wordnet Similarity for Word Sense Identification, 1998.
- [23] Dekang Lin. An Information-Theoretic Definition of Similarity. In *Proceedings of International Conference on Machine Learning ICML*, volume 98, pages 296–304, 1998.
- [24] Zhiyong Lu. PubMed and beyond: a survey of web tools for searching biomedical literature. *Database*, 2011, January 2011.
- [25] Christoph Mangold. A survey and classification of semantic search approaches. *International Journal of Metadata, Semantics and Ontologies*, 2(1):23, January 2007.
- [26] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, July 2008.
- [27] A T McCray, S Srinivasan, and A C Browne. Lexical methods for managing variation in biomedical terminologies. *Proceedings of the Annual Symposium on Computer Application in Medical Care*, pages 235–239, January 1994.
- [28] George A. Miller. WordNet: a lexical database for English, 1995.
- [29] Tomoko Ohta, Katsuya Masuda, Tadayoshi Hara, Junichi Tsujii, Yoshimasa Tsuruoka, Junpei Takeuchi, Jin-Dong Kim, Yusuke Miyao, Akane Yakushiji, Kazuhiro Yoshida, Yuka Tateisi, and Takashi Ninomiya. An intelligent search engine and GUI-based efficient MEDLINE search tool based on deep syntactic parsing. In *Proceedings of the COLING/ACL*, pages 17–20, Morristown, NJ, USA, July 2006. Association for Computational Linguistics.
- [30] T Pedersen, S V S Pakhomov, S Patwardhan, and C G Chute. Measures of semantic similarity and relatedness in the biomedical domain. *Journal of Biomedical Informatics*, 40(3):288–299, 2007.
- [31] C. Perez-Iratxeta. Update on XplorMed: a web server for exploring scientific literature. *Nucleic Acids Research*, 31(13):3866–3868, July 2003.
- [32] Martin F Porter. Snowball: A language for stemming algorithms. Technical report, 2001.
- [33] Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. *Arxiv preprint cmp-lg/9511007*, 1:448–453, 1995.
- [34] G. Salton, A. Wong, and C. S. Yang. A Vector Space Model for Automatic Indexing. *Commun. ACM*, 18(11):613–620, November 1975.

- 
- [35] Gerard. Salton and Michael J. McGill. *Introduction to modern information retrieval*. McGraw-Hill, 1983.
- [36] David Sánchez and Montserrat Batet. Semantic similarity estimation in the biomedical domain: an ontology-based information-theoretic perspective. *Journal of biomedical informatics*, 44(5):749–759, October 2011.
- [37] David Sánchez, Montserrat Batet, and David Isern. Ontology-based information content computation. *Knowledge-Based Systems*, 24(2):297–303, March 2011.
- [38] Frederik C. Schadd and Nico Roos. Word-Sense Disambiguation for Ontology Mapping: Concept Disambiguation using Virtual Documents and Information Retrieval Techniques. *Journal on Data Semantics*, pages 1–20, 2014.
- [39] Martijn J Schuemie, Rob Jelier, and Jan A Kors. Peregrine: Lightweight gene name normalization by dictionary lookup. In *BioCreative Challenge Evaluation Workshop*, pages 131–133, 2007.
- [40] Ariel S Schwartz and Marti A Hearst. A simple algorithm for identifying abbreviation definitions in biomedical text. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, pages 451–462, 2003.
- [41] Amit Singhal. Modern Information Retrieval : A Brief Overview. *Bulletin of the IEEE computer society technical committee on data engineering*, 24.4:35–43, 2001.
- [42] Philippe Thomas, Johannes Starlinger, Alexander Vowinkel, Sebastian Arzt, and Ulf Leser. GeneView: a comprehensive semantic search engine for PubMed. *Nucleic acids research*, 40:W585–91, July 2012.
- [43] S. K. M. Wong, Wojciech Ziarko, and Patrick C. N. Wong. Generalized vector spaces model in information retrieval. In *Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '85*, pages 18–25, New York, New York, USA, June 1985. ACM Press.
- [44] Z Wu and M Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 133–138, Morristown, NJ, USA, June 1994. Association for Computational Linguistics.
- [45] a Jimeno Yepes, Jg Mork, and Bb Wilkowski. MEDLINE MeSH indexing: lessons learned from machine learning and future directions. *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium (2012)*, pages 737–741, 2012.