Grant Agreement Number: 257528

KHRESMOI

www.khresmoi.eu

D1.7: Prototype and report on semantic indexing and annotation for information retrieval

Deliverable number	*D1.7*
Dissemination level	*Public*
Delivery date	*Due 28.2.2014*
Status	*Final*
Authors	*Angus Roberts, Johann Petrak, Célia Boyer, Ljiljana Dolamic, Allan Hanbury, Michael Dittenbach, Julien Gobeill, Michal Novak*

This project is supported by the European Commission under the Information and Communication Technologies (ICT) Theme of the 7th Framework Programme for Research and Technological Development.

Executive Summary

This deliverable is the final report on the semantic indexing and annotation efforts within the Khresmoi project. Documents indexed within Khresmoi are semantically processed in two ways. In the first, information about the health content of sections within the document is added. In the second, links, or annotations, are added between terms in the document and concepts in the Khresmoi Knowledge Base. These two types of semantic knowledge are indexed alongside the full document text, within the two Khresmoi search engines, Lucene and Mimir. Together, these semantically enriched indices are intended to help achieve the primary goal of Khresmoi: to improve the performance of information retrieval, and to provide a richer search experience for end-users.

This report describes the crawling of documents, their processing (to add section information and to add annotations), their semantic indexing, and the ranking search results retrieved from these indices. The computational machinery and processes are described for two indices, as used in the general public "Khresmoi for Everyone" use case; in the Khresmoi medical practitioner use case; and in the radiology use case. In each of these, the indices are exposed to user search via different user interfaces. This report concludes by describing a specific user interface written to make maximum use of the semantic indices in the "Khresmoi for Everyone" use case.

Table of Contents

List of Figures

List of Tables

Abbreviations

API	Application Programming Interface
BM25	BM25, a document ranking model
CME	Continuing Medical Education
CSV	Comma Separated Variable (a file format)
CUI	Concept Unique Identifier (an identifier used within UMLS)
EDT	Eastern Daily Time
ETL	Extract, Transform, Load (data processing steps)
FAQ	Frequently Asked Questions (a document listing common questions and answers)
FTP	File Transfer Protocol (a standard way of transferring files)
GATE	General Architecture for Text Engineering
GNU	GNU is Not Unix (a software writing and licensing organisation)
HES-SO	Haute Ecole Specialisee de Suisse Occidentale (University of Western Switzerland)
HON	Health On The Net
HONCode	HON Code of Conduct
HTML	Hyper Text Markup Language (the standard code used to write web pages)
HTTP	Hyper Text Transfer Protocol (the standard way of transferring web pages)
IANA	International Assigned Numbers Authority
ID	Identifier
IDDM	Insulin Dependent Diabetes Mellitus
IDF	Inverse Document Frequency
IR	Information Retrieval
ISO	International Standards Organisation
JAPE	Jave Pattern Engine (a pattern matching language used in GATE)
JSON	JavaScript Object Notation (a file format with standard syntax)
K4E	Khresmoi for Everyone
MD5	MD5, a message-digest algorithm (used to check file integrity)
MeSH	Medical Subject Headings
MG4J	Managing Gigabytes for Java (a search engine)
ML	Machine Learning
MTR	MeSH terms ratio
NE	named Entity (a term in text referring to some named thing)
NGO	Non-Governmental Organisation
NLM	National Library of Medicine
OWL	Web Ontology Language (a standard language for ontologies on the Web)

OWLIM	OWLIM, a semantic repository with support for OWL
PDF	Portable Document Format (a standard document file format)
PMID	PubMed ID
POS	Part Of Speech
PR	Processing Resource (used to refer to a GATE component)
SPARQL	SPARQL Protocol and RDF Query Language (a query language for semantic repositories)
SVM	Support Vector Machine
TF	Term Frequency
TF-IDF	Term Frequency / Term Frequency
TUI	Term Unique Identifier (an identifier used within UMLS)
UMLS	Unified Medical Language System
URI	Uniform Resource Indicator (used to identify resources on the Internet)
URL	Uniform Resource Locator (a type of URI used to locate resources
XML	Extensible Markup Language (a file format with standard syntax)
ZIP	ZIP file (a compressed file format)

1 Introduction

The Khresmoi project is building a multi-lingual search and access system for biomedical information and documents [13]. A key idea behind the project has been the use of *semantic indexing* to improve the performance of information retrieval, and to provide a richer search experience for end-users. A semantic index adds a layer of meaning to the indexed text, by linking to a conceptual structure, such as a classification, taxonomy or ontology. Within Khresmoi, this conceptual structure is provided by the Khresmoi Knowledge Base [18]. Linkage is achieved by annotation of the text: marking portions of the text as having given properties and features.

The body of this report describes the prototype software used for semantic indexing and annotation in Khresmoi. In order to place this in context, this introductory section describes the motivation for the work with examples of its use (in Section 1.1), gives an overview of those parts of the Khresmoi architecture covered by this report (in Section 1.2), and then summarises the body of the report and prototype availability (in Section 1.3).

This document does not report evaluation of the prototype software, which will be covered by the future Khresmoi deliverables D1.8 "Report on results of the WP1 second evaluation phase", D7.3 "Meta-analysis of the second phase of empirical and user-centered evaluations" and D10.3 "Report on the extensive tests with the final search system".

1.1 Motivation: examples of use

Linking text to a semantic layer enables intelligent indexing and search over that text. To illustrate this motivation, we will describe several examples of increasing complexity. Our simplest example is that of synonymy, i.e. the use of different words or terms to describe the same concept. Dealing with synonymy means that searches for "IDDM", "type 1 diabetes" and "insulin dependent diabetes" will all return the same results. Each of the searched terms is linked to the concept for Type 1 Diabetes in the Khresmoi knowledge base, and it is this that is searched for in the text. This may be achieved in IR without recourse to indexing against a complex conceptual structure.

Our next example is that of taxonomy, i.e. the hierarchical class or "is-a" relationships between concepts. Dealing with taxonomy means that a search for "heart disease" should return results for both "angina" and "myocardial infarction". This is achieved by mapping the search term to the matching concept in the knowledge base, and then expansion of this concept to all sub-classes. This illustrates the use of hierarchical relations between concepts in search. Use of a flexible knowledge base within Khresmoi means that we need not restrict semantic relations to that of taxonomy. The Khresmoi indexes, via the Khremsoi knowledge base, can search along any represented relation, and can be applied to other relations. For example, a query for a piece of anatomy could also retrieve documents mentioning all parts of that piece of anatomy.

The previous examples are all used within Khresmoi, to drive information retrieval in response to user queries, in the medical practitioner [25] and radiology use cases [14].

In addition to this, the semantic indexing is also used by the Semantic Search Interface in Khresmoi for Everyone (K4E), as described in Section 6. This interface makes greatest use of semantics within Khresmoi, allowing the user to explore the knowledge base in order to construct queries. A user may, for example, ask to search for signs or symptoms that are treated by aspirin. The knowledge base can be used to expand this to a list of concepts that satisfy the query, which can then be used to search the index. This expansion would allow a document about cerebo-vascular accidents to be retrieved, even though it uses the layperson term "stroke", and does not mention aspirin.

Semantics has motivated other development work in Khresmoi. Imagine a user who is interested in "Diabetes", but only in the symptoms of this illness. Using the K4E search engine gives the user the possibility of direct access to such types of information, without having to go through large amount of text about diabetes in general. This is enabled by the section classification module within K4E pipeline, which classifies document sections according to the type of health information they cover. Additionally, development of a relevancy score module, which scores the health relevancy of document, has allowed presentation of documents having greater health related content, first in the list of results. Section classification and relevancy scoring are described in Section 3.1.

1.2 Architecture overview

Figure 1 shows an overview of the full Khresmoi architecture. This deliverable covers the components within the yellow outline in the figure. The Crawler crawls both HONcode certified websites and physician-specific websites, in two separate processes. Extract, Transform and Load (ETL) is carried out by a single component, in to a single data repository, using CouchDb. This provides a unified data retrieval interface to the rest of the Khresmoi architecture, irrespective of the source of the crawl. The resulting crawl is indexed by two components. *Khresmoi for Everyone (K4E)* makes use of a Lucene Solr index, while *Khresmoi professional (KPro)* makes use of a Mimir index using GATE annotations. The radiology reports for *Khresmoi Radiology* are indexed by a specialised GATE and Mimir pipeline. The deliverable also reports on the approaches used to improve ranking of search results in both Lucene and Mimir.

The use of two separate indices reflects the two text use cases in Khresmoi, and two possible software deployment patterns. In the first, Khresmoi for Everyone, Khresmoi is building on existing infrastructure at the project partner HON. This is likely to be a common scenario in future Khresmoi uptake, in that new user organisations will already have some search infrastructure in place, and will not want to completely replace this infrastructure. In this case, the existing crawl infrastructure and index (Lucene Solr) is used to support full text search. In order to support additional semantic search, all documents in the existing index are also semantically

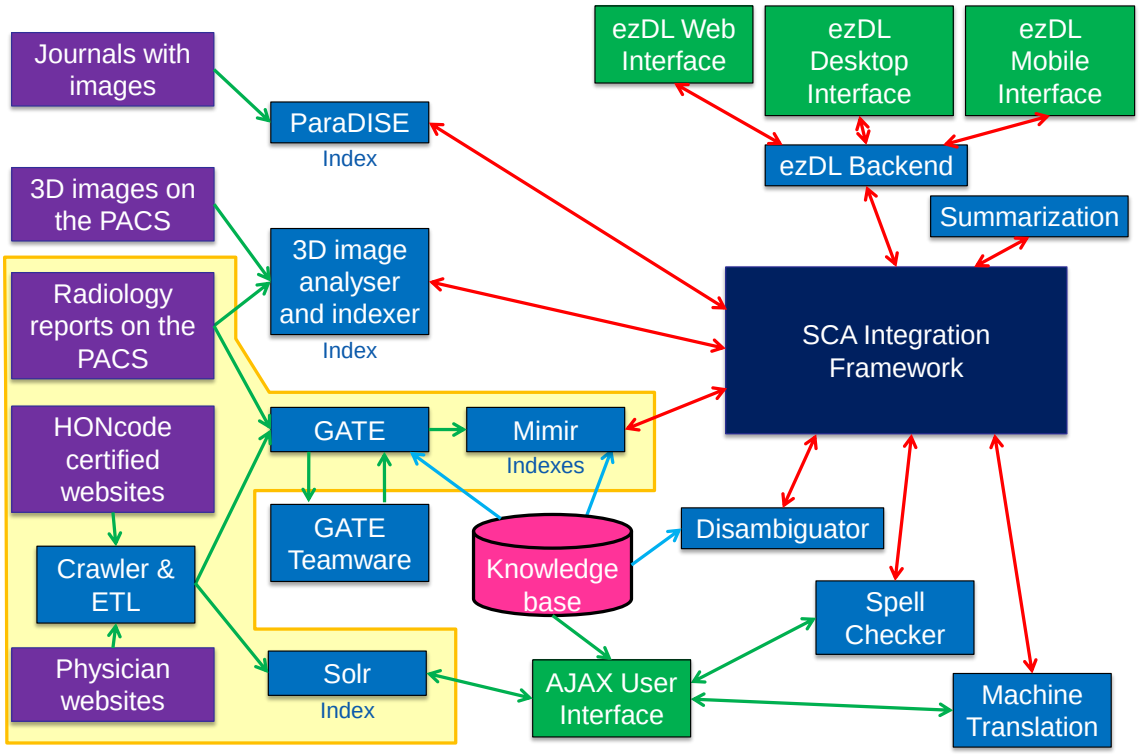


Figure 1: Khresmoi architecture overview

annotated and indexed in Mimir. The final user interface contains components that make use of both indices.

The second use case, Khresmoi Professional, is intended for use by medical professionals. Here, the Khresmoi project is able to deploy search infrastructure from scratch, and there is no need to support full text search from an existing system. The Khresmoi Professional system therefore supports both full text search and semantic search functionality from Mimir. In this use case, an additional crawling component is used to provide material specific to the information needs of medical professionals.

1.3 Summary of the report

The technical work described in this report fits in to the broader Khresmoi architecture, as described above and in [23]. The work described here is instantiated as components within that architecture. The Khresmoi software stack first crawls targeted document sources, as described in Section 2. These are processed in order to add semantic annotation (Section 3.2) and in order to classify document sections and calculate the health relevancy of documents (Section 3.1). The resultant annotated documents are then indexed, as described in Section 4. Khresmoi uses two indexing systems, for different use cases. In the medical practitioner and radiology

use cases, a bespoke semantic indexing system, Mimir[1], is used (Section 4.1). In the general public use case, this bespoke index is used alongside a Lucene[2] index (Section 4.2). On retrieval of documents at query time, ranking is applied to the appropriate indexing systems, according to the use case, and as described in Section 5. The report also describes, in Section 6, one instance of user interaction with the semantic indexes: the Semantic Search Interface created for the general public use case. Other Khresmoi uses of the semantic indexes are described in deliverables ([25, 14]). The report concludes with closing remarks in Section 7.

1.4 Availability of the prototype and software

The prototype is available via the K4E user interface, at:

- http://everyone.khresmoi.eu/hon-search/

[1] http://gate.ac.uk/mimir/
[2] http://lucene.apache.org/

2 Crawling

Before any document processing or indexing can take place in the Khresmoi architecture, documents for processing and indexing must first be collected. Since the bulk of documents indexed by Khresmoi are web documents, document collection is carried out primarily by web crawling. Two web crawlers are used in the Khresmoi architecture, to support the two use cases and software deployment patterns described in Section 1.2.

The web crawler that constitutes the K4E pipeline was described in [11]. In addition to the web pages gathered by this module, sources crawled specifically for the medical professional use case are parsed and pushed into the K4E pipeline to be indexed in the Lucene/Solr index, and to be annotated and indexed in Mimir, as explained in Section 1.2. Section 2.1 below gives details on this additional crawling process.

2.1 Crawling for the medical professional use case

2.1.1 Global properties of web crawlers

According to [17], there are two features that a web crawler must provide, and features that a web crawler should provide.The features that a web crawler must provide are robustness and politeness. Robustness is the ability to efficiently work with heterogeneous formats and structures; such heterogeneity is very present in the Khresmoi resources set. In particular, non-standard website development can mislead the crawler, so that it enters an infinite loop of crawled pages. This inadvertent side-effect is an example of a crawler trap. Politeness is related to the rate at which the crawler hits webpages. Specifically, the crawler must not perturb the performance of the server hosting the website, by overloading it. Explicit rules on crawling a specific website are usually defined by the Robots Exclusion Protocol: instructions given by the website owners are available in a robots.txt file which is placed in the root of the websites. These instructions usually include the hit-rate (such as one access every 30 seconds), forbidden directories (such as /admin/), or restricted time brackets (such as visiting-hours: 23:00EDT-05:00EDT).

The features a web crawler should provide are parallelization, scalability (to large websites), performance and efficiency, quality (the ability to focus on useful pages), freshness, and extensibility (to other data formats and protocols).

2.1.2 Design of the medical professionals crawling cystem

In the Khresmoi project, the HES-SO partner crawled web resources that were out of scope for the Khresmoi For Everyone pipeline. These web resources include medical practitioner resources, in particular websites that are gateways to large biomedical databases. Examples of such sites include: drugbank.ca, which contains thousands of entries for drugs, including drug data and comprehensive drug target information; clinicaltrial.gov, which is a registry of more than 150,000 clinical studies; and

PubMed which contains twenty million citations from biomedical journals. The very first version of the Khresmoi medical professional crawl system aimed at crawling a small number of tool websites for demonstration purposes. For this crawl, we investigated the use of open-source technologies, such as LucidWorks[3] and Apache Nutch[4]. These systems work well for crawling standard websites, and provide stylish and fast demonstration interfaces. Looking at the Manning features described above, we can say that they are quite robust, as that they were able to efficiently crawl most of the requested websites. They are also polite, as they support the robots.txt and sitemap files. Parallelisation and freshness are also strong points of these products.

Once we started to crawl the full set of resources requested for Khresmoi, however, we quickly reached the limits of these systems. First, in the Khresmoi workflow, the content crawled for the medical professional use case needs to be exported in a compliant format for loading in to the Khresmoi document repository, built with CouchDB. It was not obvious how such an export could be managed with LucidWorks, without refactoring of the source code. Second, some critical sources are not accessible via crawling. Specifically, for most of the large databases mentioned above, such as PubMed, the database website is only a gateway for manual search, and are not designed to be accessed by machines. Instead, the content is freely available for machine use via dedicated and standard protocols, such as services or FTP servers. For example, MEDLINE is not intended to be accessed via large numbers of HTTP requests, but is instead accessible via a dedicated Application Programming Interface, or via files available from the National Library of Medicine FTP servers. It therefore appeared that a fraction of the crawl had to be performed via specific pipelines which were out of the scope of the available crawling solutions.

It was therefore decided to design a dedicated crawling system for the Khresmoi project, using a robust crawler, GNU Wget. Wget retrieves content from a Web server, but in its recursive mode it can be used to mirror websites and can thus be viewed as a Web crawler. Considering the Manning features described above, Wget is very robust, as it was especially designed for completing crawls over slow or unstable network connections. Moreover, Wget is polite as it supports the Robots Exclusion Standard. Wget explores and downloads requested content from specified URLs, then saved files in a local directory structure as on the remote server. With regard to freshness, Wget can be instructed to inspect the timestamps of local and remote files, and download only the contents newer than the corresponding local ones. In terms of parallelization, Wget does not support duplicated processes for the same website. With regard to quality, Wget is not able to manage deduplication, i.e. a unique content was downloaded several times via different URLs. This may occur with local hyperlinks, or with different combinations of parameters encoded in the URL. Post-processing strategies were therefore added to the crawl process, in order to filter the downloaded URLs. This post-processing was carried out with a set of dedicated regular expressions. These regular expressions were manually designed for each resource, through an analysis of the website structure.

[3]http://www.lucidworks.com/
[4]http://nutch.apache.org/

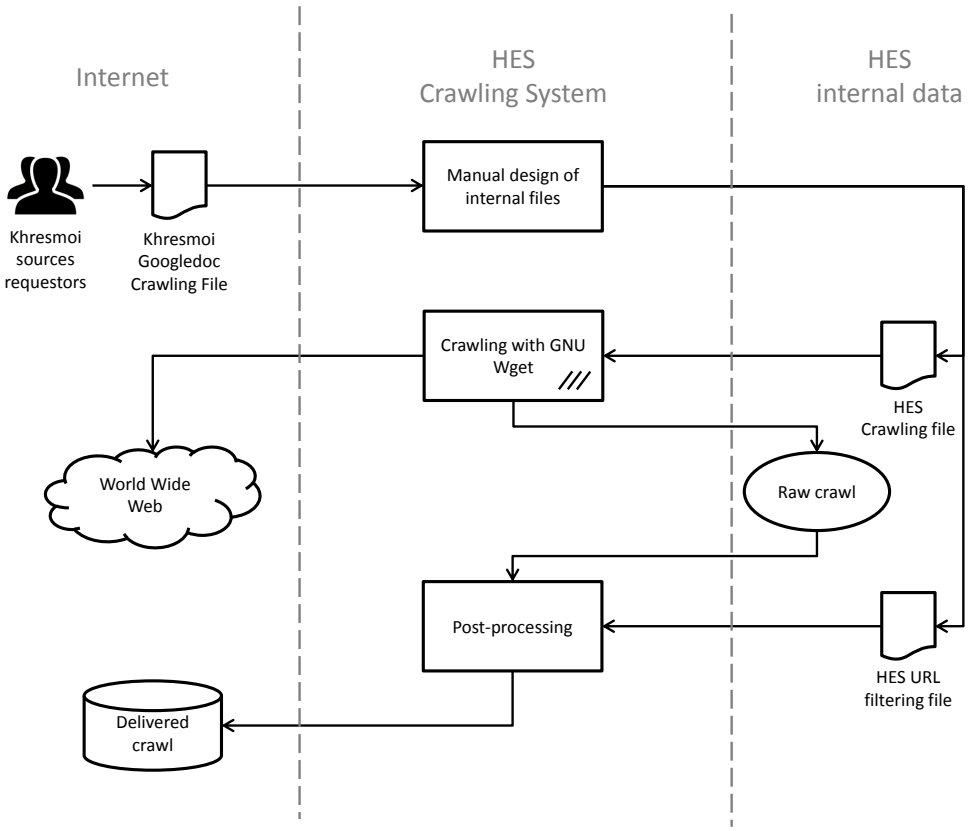


Figure 2: Medical professionals use case crawling workflow

2.1.3 Workflow of the medical professional crawling system

The final design of the workflow is illustrated in Figure 2. The workflow starts with a Khresmoi sources requestors, a person who requests sources that need to be included in the Khresmoi index.

Sources requestors fill in an online Khresmoi Googledoc Crawling file according to their needs. For each needed source, they specify a set of features in this document, including the home page URL, a short description of the source, and whether this resource is critical or not. Critical resources are processed as a priority. The requestors also fill in several metadata fields, such as the language of the site, and target users (for patients, for professionals or for both). Once completed, the Googledoc Crawling file is automatically accessed and used by the medical professional Crawling System. Prior to the crawl itself, several parameters are required to be set:

1. setting the seed set, i.e. the set of URLs with which Wget will start to explore the website graph;

2. setting the domain limits, in order to prevent Wget leaving the requested website and exploring the entire Web;

3. setting the maximum exploration depth, in order to avoid infinite loops;
4. setting the maximum number of retries for a download fail (default 5);
5. setting the time between two hits.

Once these parameters are set, crawling itself takes place using Wget. Wget crawls websites under the Khresmoibot name. Wget is instructed to inspect timestamps, in order to download only content that is updated relative to the local version. Wget also is instructed not to consider several file extensions which are not relevant for the index and potentially heavy to download, such as images, ZIP or CSV files. Pdf and MS Office files are also not crawled, as their treatment is not supported in the Khresmoi workflow. As mentioned above, Wget does not support parallelization for the same website. It is, however, possible to parallelize the crawling between different sources. In the Khresmoi project, forty sources are crawled at the same time. Once Wget has completed the crawling, a Perl post-processing script uses the Wget log file and local mirror, in order to:

1. Filter URLs, according to manually designed regular expressions. These are used as positive and negative patterns for each resource. An example of a positive pattern is `/www.aktive-diabetiker.at/index.php?article_id=\d+$/;`
2. Detect new content, updated content (usiing page timestamps), and deleted content, i.e. locally held content that is no longer available online.
3. Generate the delivered crawl file for loading in to the Khresmoi CouchDB document repository.

As of January 2014, 193 sources had been crawled specifically for the medical professional use case.

2.1.4 Format of the crawl files

Crawled resources are provided in zip files. For each resource, monthly update files contain all crawled contents, including created, updated, and deleted pages. The files contain downloaded content in the following format: the URL, then a Created/Updated/Deleted tag [C—U—D], then the raw html. The following rules are applied:

1. fields are terminated by `'!#|#!'`
2. lines start with `'!#$#!'`

2.1.5 Special crawling

PubMed PubMed is not crawlable via HTTP technologies. However, the Khresmoi partner HES-SO have a local and regularly updated copy of MEDLINE. PubMed is therefore provided as "fake" HTML pages, built with all PubMed content (Title, Abstract, PubMed unique identifier (PMID), Journal, Publication Year etc.) and the URL. PubMed citations to use are chosen according to a list of 600 journals that were selected by Khresmoi requestors. More than 2M abstracts were provided to the Khresmooi project as of January 2014.

English Wikipedia As for PubMed, Wikipedia is not crawlable, and once again a local and updated copy is used to create "fake" HTML pages. The choice of wikipedia pages that provide a medical information subset was not obvious. The wikipedia ontology is not usable for this task, as subcategories of the top ontology category "Health" contain a lot of noise. These categories include persons, movies, or books. For example, the category "Breast Cancer" subsumes the subcategory "Breast cancer survivors" which includes the page "Brigitte Bardot". It was therefore decided to manually create a set of 1200 categories by extending categories in the Wikipedia Category tree viewer[5], starting from the "Health" Category. Pages belonging to these categories are exported in the delivered crawl. This represents 32,000 Wikipedia pages.

German Wikipedia For German, the set of categories designed for English was also used. For each category, the HTML page was scraped, and a German translation of the category searched for. When a translation was available (245 times out of 1200), this German category was used to select the Khresmoi subset of the German Wikipedia. This represents 9,400 pages.

[5]http://en.wikipedia.org/wiki/Special:CategoryTree

3 Annotation

Once documents have been crawled, they are processed in order to extract section information, annotations linked to the Khresmoi knowledge base, and metadata. Section 3.1 describes the categorisation of documents in to sections, and the measurement of the relevancy of these sections to the health domain, for the general public use case. Section 3.2 describes semantic annotation, for the medical practitioner and general public use cases (3.2.1), and for processing German radiology reports for the imaging use case (3.2.4).

3.1 Document Section Categorisation and Calculation of Relevancy to Health

Document content extraction and classification was described in Section 3.1.3 in [11]. In this section we give an update of this process, including its motivation.

Disease Section Extractor As has been described in [11] the Disease Section Extractor is a rule based method for extracting well known sections from selected web sites. This extraction has been extended to include the French language. The currently supported web sites in French are Santé Pratique[6] and Santé Médecine [7].The same set of sections is extracted as from English web pages namely: Définition, Causes, Symptôme, Diagnostic and Traitement.

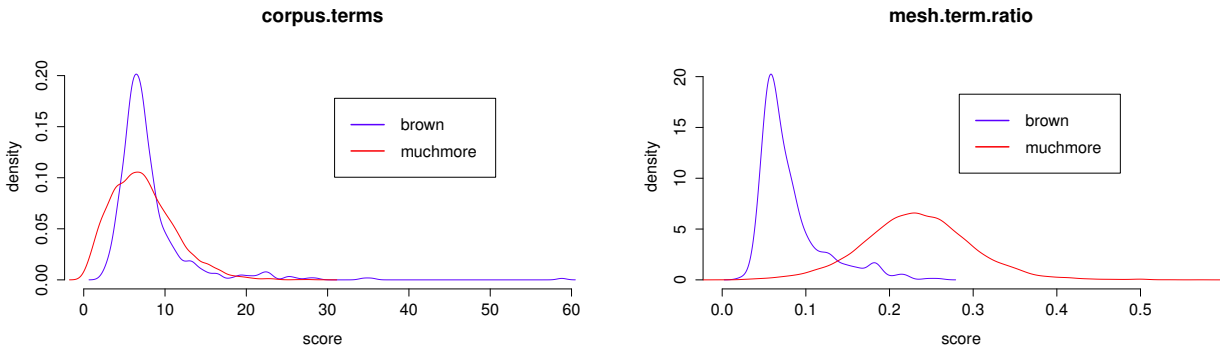
Relevancy Score The search results returned by the K4E search system are boosted based on their relevancy score. The goal of this score is to make a distinction between health related sources, and those not related to health. As described in [16] and [11], various algorithms were tested for this purpose. However, the score calculation retained has not been shown to be robust enough for sources to be rejected as non relevant, from the point of view of a health related search engine. Based on user feedback, the current relevancy score has proven not to be efficient enough. For the purpose of determining the systems capability to distinguish relevant from non relevant pages, HON has created the following testing setup:

Test collection has been created containing 501 non medical documents from the Brown corpus [8] and 7811 medical abstracts from the Springer MuchMore [9] collection.

Tested algorithms - all previously tested algorithms have shown an inability to distinguish the relevancy. Thus, a new algorithm the "MeSH terms ratio"(MTR) was created and tested.

[6] http://www.santepratique.fr
[7] http://sante-medecine.commentcamarche.net
[8] http://icame.uib.no/brown/bcm.html
[9] http://muchmore.dfki.de/resources1.htm

Results - Figure 3 shows the distribution of the results calculated by a) the current system (corpus.terms) and b) the MTR algorithm. The x-axis gives the health relevancy score calculated by the algorithm, while the density of documents for each score value is given on the y-axis. As can be seen from these diagrams, unlike the corpus.terms algorithm used currently, MTR has proved to be robust enough to be used as a threshold for keeping or discarding documents. Thus, the currently used algorithm is being replaced by MTR. Additionally, this new relevancy score calculation is supported for these languages: English, Czech, French, German, Spanish, Italian, Lithuanian, Dutch, Polish, Portuguese, Finish, Swedish and Croatian. However, due to the smaller size of the MeSH dictionary in languages other than English, the determined threshold is not applicable to all languages.



(a) Corpus terms(current) (b)MeSH terms ratio(MTR)

Figure 3: Relevancy score test results

3.2 Semantic Annotation

3.2.1 Semantic annotation for the medical practitioner use case

Semantic annotation is performed using a modular GATE application pipeline [4, 5] which consists of several sub-pipelines. Each sub-pipeline in turn consists of several processing steps (GATE "Processing Resources, PRs). Some processing resources use domain-specific data which is prepared using a separate process in the preparation phase. The pipeline itself performs semantic annotation on a large number of diverse documents in the annotation phase: crawled web documents in several languages, Wikipedia documents in several languages and Pubmed abstracts. The pipeline contains conditional processing resources which adapt the processing steps carried out for a particular document to the specific language and type of document. The same pipeline is also used to process corpora from earlier stages of the project and corpora which are not retrieved from CouchDB but have been created by other means. In addition the pipeline has been used for experimentation with machine learning approaches to improve the disambiguation step.

The complete annotation application, with all the GATE pipelines, processing resources and the prepared data is available to project partners in the internal project Subversion repository.

Annotation Pipeline

The pipeline consists of the following sub-pipelines:

1. Add and amend metadata (`metadata`): This sub-pipeline prepares all the metadata required for the Mimir index. This includes accessing metadata that originates from CouchDB metadata fields and adding additional metadata which is not included or missing from the CouchDB metadata fields. Depending on the later use of the metadata, it is added as either a document feature, annotations to existing document parts, or the document is edited and additional document text added and annotated. This is done in different ways for documents crawled from the web, Wikipedia documents and Pubmed abstracts. Metadata is added in the form of document features, annotations of document text that already exists or gets added in this sub-pipeline, or zero-length annotations with features that contain the necessary metadata. A detailed list of all metadata which is used by Mimir for either searching, retrieving or both is available in section 4.1 and in Tables A.1 and A.2.

   This sub-pipeline consists of the following main processing resources:

   - `LongestEntryLookup`: this processing ressource is used to store the prepared HON directory data and to lookup the data for the longest known prefix of a URL. The preparation of the metadata stored with this processing resource is described in Section 3.2.2.
   - `basicMetadata`: this step tries to determine the kind of document being processed based on various clues from the existing metadata, document name, and document content. It also processes the following metadata which is identical for all kinds of documents retrieved from CouchDB: `url`, `id`, `language`, `domain`, `dateCrawled`
   - `loadHonCodeData`: this step loads the prepared HON code data and stores it with the `LongestEntryLookup` processing resource
   - `createDefaultMeta`: this step processes the metadata for all crawled documents, but not Wikipedia articles or Pubmed abstracts.
   - `createPubmedMeta`: this step processes the metadata for Pubmed abstracts
   - `createWikipediaMeta`: this step processes the metadata for Wikipedia articles.
   - `createAllMeta2`: adds the `domainClass` metadata to all documents. Also determines the ISO 31661-1 two letter country code from the country name derived from the HON Code data, if available, or from the top level domain name of the URL.

2. Detect publication dates for selected pages from selected hosts. In this sub-pipeline the URL of the processed page is checked against a known list of domains and regular expressions for URL path names. If there is a match, the following steps are carried out:

   - dates in a variety of formats, including dates which only consist of a month and year, or just a year alone, are detected on the page in language-dependent way (seperately for the languages English, German, Czech, Spanish and French).
   - known publication date contexts are identified by nearby identifying text (e.g. "last updated:") and by identifying parts of the HTML DOM tree.
   - dates within a publication date context are parsed and converted to a standardized date representation. This date is stored as metadata and also the information that a publication date has been found is stored as metadata for this page.

3. Basic linguistic annotations (`basic-annots`): This sub-pipeline performs basic tokenisation and sentence splitting for all languages and in addition runs a POS tagger and morphological analyser which provides lemmata for English language documents.

4. Czech basic linguistic annotations (`czech_nlp`): In this sub-pipeline basic linguistic processing of Czech documents is performed. It consists of sentence splitting, tokenisation, POS tagging and lemmatisation with all steps tailored to Czech. While sentence splitting is provided by a Czech splitter from the Treex[10] framework [21], the rest is ensured by the MorphoDiTa[11] tool [24].

5. Identify indexable content and boilerplate (`findContent`): this sub-pipeline annotates the parts of the documents which are likely to contain content (as opposed to boilerplate text like navigation menus, company logos and the like). This is done in different ways depending on the kind of document: for Wikipedia documents, the whole document is marked as content. For Pubmed abstracts, the title and abstract are marked as content. For all other documents, the GATE BoilerPipe plugin is used to identify content and boilerplate parts of the document.

6. Identify stop words (`annotate-stopwords`): this sub-pipeline is executed for English language documents and uses gazetteer lists to mark tokens where the surface string or the lemma matches a stop word.

7. Semantic annotation of UMLS concepts (`umls-extgaz`): this sub-pipeline annotates occurrences of labels of UMLS concepts based on a set of gazetteer lists and then post-processes those annotations to create the annotation features needed later and remove obvious duplicates. The main steps of this sub-pipeline are:

[10] `http://ufal.mff.cuni.cz/treex`
[11] `http://ufal.mff.cuni.cz/morphodita`

- `setWordFlags`: this step marks each token as a possible candidate for starting, ending or being part of a multi-word UMLS term, based on the POS tag of the label.
- `ExtGaz:abbrvs`: annotate possible abbreviations that may denote UMLS concepts
- `ExtGaz:standard-root`: annotate potential UMLS concepts by matching a gazetteer lists of lemmata against the the lemmata of the tokens in the document
- `ExtGaz:standard-string`: annotate potential UMLS concepts by matching a gazetteer lists of UMLS labels against the string of the tokens in the document.
- `FeatureGaz:umls-types`: add information about the UMLS-types of potential matched concepts to the annotations.
- `RemoveDupes`: this removes various kinds of duplicate annotations, for example duplicates for identical spans and CUIs which originate from identical matches from the lemmata and string gazetteers, or annotations of different length but for the same CUI where several labels exist in UMLS and one label is a substring of the other.
- there are additional steps where, among other things, known spurious matches are removed and the textual name of the UMLS class of a concept is added to the annotation.

8. Czech semantic annotation of UMLS concepts (`czech_umls-tag`): this sub-pipeline is a Czech counterpart of `umls-extgaz`, sharing multiple processing resources with it as well as introducing the new ones, specific for Czech:
   - `ExtGaz:cs-string-case`: annotate potential UMLS concepts by case-sensitive matching gazetteer lists of UMLS labels against the string of the tokens in the document.
   - `ExtGaz:cs-string`: case-insensitive version of `ExtGaz:cs-string-case`
   - `ExtGaz:cs-root`: annotate potential UMLS concepts by matching gazetteer lists of lemmatised UMLS labels against the string of lemmata in the document.
   - `ExtGaz:cs-root-perm`: extends `ExtGaz:cs-root`. For UMLS labels shorter than 4 words, all permutations are contained in the gazetteer lists. It targets the issue of free word order in Czech, especially for noun–adjective combinations.
9. Semantic annotation of DrugBank concepts (`drugbank-extgaz`): This annotates occurrences of terms that match DrugBank labels from a gazetteer list.
10. Heuristic disambiguation between multiple possible concepts (`umls-disambiguate`): This sub-pipeline uses heuristic rules to choose

between several possible semantic annotations which overlap at some position in the document. The heuristic rules use information about the source of the match (UMLS concepts, DrugBank labels), the kind of match (abbreviations, lemmatised matches, original string matches), the semantic type (Drug, Anatomy, Disease or Intervention), the kind of overlap (shorter, equal span, arbitrary overlap) as well as the concrete CUI for UMLS concepts to disambiguate between several potential matches. The following main heuristic rules are applied:

- DrugBank matches have priority over UMLS matches
- longer matches have priority over shorter matches
- overlapping matches that start earlier in the document have priority over those that start within another annotation
- CUIs that denote a more specific concept have priority over more general ones

11. ML-based disambiguation between multiple possible concepts (`mlNN-features`, `mlNN-apply`, `mlNN-disambig`): This is done by three separate sub-pipelines which are responsible for creating the features used by the machine learning model, applying the machine learning model and creating the classification features, and using the classification features together with the results from the heuristic disambiguation to carry out the actual disambiguation. There are various versions of each of these sub-pipelines, from the various iterations in the manual annotation process and from different experimental approaches of how to use machine learning for the disambiguation. Details of this will be given in the updated version of deliverable D1.4.2 [22].

12. Preparation for Mimir Indexing (`mimirIndexing`): This sub-pipeline prepares annotation set `Meta` which is used by Mimir for indexing. In addition to this, Mimir uses the `Original markups` annotation set to re-construct the original HTML markups of the document. We modify this set and filter most of the original HTML markups to make it easier to view a cached copy of the document. The main steps of this sub-pipeline are:

    - `transferDefaultToMeta`, `copyTokensToMeta`, `removeSomeTokens`: move annotations for Sentences, Tokens, and SpaceTokens to the `Meta` output set, but only for those parts of the document which have been identified to be content previously.
    - `cleanupMarkups`, `addNonContentSpans`: remove most of the original HTML markups and insert instead styled `<br/>` markups to keep the line spacings which were originally caused by markup like `<h1>`, `<li>` or `<p>`. After this, styled `<span>` markups are inserted to identify those parts of the document which have been classified as content or non-content.

- `DocumentReset`, `MimirClient`: once all the annotations needed by Mimir have been created in the `Meta` set and the final `Original markups` set has been created, all other annotations are removed to keep the amount of data to be sent to the Mimir server as small as possible. The `MimirClient` PR sends the final document to the Mimir server for indexing.

3.2.2 Metadata preparation

The annotation pipeline adds metadata to the documents which is not provided as part of the document when it gets fetched from the CouchDB repository of crawled documents. The metadata is added to documents based on the URI prefix of the document. In order to provide a fast and efficient way to add the correct metadata to the document, we prepare optimized datastructures from the original data from various sources (see below) that contain all the necessary information.

The following resources are created:

- HONCodeDirectory labels: a JSON data structure that maps URL-prefixes to HON codes and their textual labels, and the city and country names. This information is extracted from the complete HONCodeDirectory information retrieved from the HONCodeDirectory server and used for documents where any of this metadata is not available from the CouchDB. In addition to the JSON data structure a gazetteer list of URL prefixes is also created which is used to perform longest-substring matching on the full URL of a document.

- Categories: this information is created from the internal specification document "HON Labels for the HON Search UI". A JSON map is created from this document which maps HON codes to (English) category names.

- Domain classes: this information is created from internal specification document "Khresmoi Crawl File". A CSV file and a lookup-file for URL-prefixes is generated from this document.

- Mapping from HONCodeDirectory country names to ISO 31661-1 alpha2 codes[12]. The list of all names for countries which occur in the HONCodeDirectory data is created and semi-automatically mapped to the ISO 31661-1 country codes. Mappings for country names which are not standard English names or which are not country names have been added manually.

3.2.3 Gazetteer files preparation

The details of how the gazetteer files for English-language annotation of UMLS terms are prepared are available in Section 3.1 in Deliverable D1.4.2 [22].

For the English-language annotation of DrugBank terms, the gazetteer files are directly prepared from the `http://linkedlifedata.com/resource/drugbank/drug/name` property of DrugBank concepts.

[12] `http://en.wikipedia.org/wiki/ISO_3166-1_alpha-2`

Gazetteer files for Czech annotation of UMLS terms were extracted from a list of UMLS terms and concepts retrieved from the Khresmoi knowledge base. They relate to their English counterparts via a unique concept identifier (CUI) and an identifier of a semantic type (TUI). In the same way as for English gazetteers, separate lists for preferred and alternative labels of concepts were extracted.

Before using the list of UMLS labels as a gazetteer, we carried out simple filtering on it. As a first step, we filtered out TUIs not relevant to Khresmoi. Subsequently, morphological analysis with MorphoDiTa dictionary [10] was applied on every single-word label to remove the items that can appear as function words, e.g. prepositions, conjunctions. In this way, only 7 labels are discarded from the list, though fixing many false positives in the text.

Since Czech is a morphologically rich language, the list of original UMLS labels (used in processing resources `ExtGaz:cs-string-case` and `ExtGaz:cs-string`) has only limited coverage. Thus, we constructed a gazetteer list with every word within the UMLS term lemmatised, which is applied on lemmatised documents (in `ExtGaz:cs-root`). Moreover, word ordering in Czech is more flexible than in English. This issue was addressed by another list, in which every lemmatised item shorter than 4 words was replaced by all permutations of its words (in `ExtGaz:cs-root-perm`).

The final number of labels contained in the gazetteer lists is as follows:

- `ExtGaz:cs-string-case`, `ExtGaz:cs-string` and `ExtGaz:cs-root`: 77,735 preferred, 67,427 alternative labels
- `ExtGaz:cs-root-perm`: 188,233 preferred, 175,857 alternative labels

Having annotated two sample medical documents from Czech Wikipedia, we conducted inspection of the proposed approach. It showed that the vast majority of annotated expressions really belonged to the medical domain. On the other hand, still many medical terms remained unlabeled, mainly due to the following reasons:

- the term is contained in UMLS, but in the part not relevant to Khresmoi
- the term is contained in UMLS within a larger, usually more specific, term

3.2.4 Named Entity Recognition and Relation Extraction from German radiology reports

We have implemented software to extract information about relevant named entities and relationships between named entities from German radiology reports. The software processes a corpus of text files which contain the radiology reports and creates for each input file an XML file that contains information about the named entities and relationships found. The identification of named entities (relevant anatomy and pathology terms) and relations (which pathology is observed or negated for which anatomy) is performed by an embedded modular GATE pipeline, which consists of several sub-pipelines. The embedded GATE pipeline processes each text document and creates annotations which are used to create the final XML output file.

The complete software for processing text files and creating the result XML files and the GATE pipeline and sub-pipelines, as well as all the processing resources, plugins and the software for the preparation of the gazetteer files is available to project partners in the internal project Subversion repository.

To run the the extraction on an input directory `[indir]` of text files and create the XML result files in an output directory `[outdir]` the following script invocation can be used: `./bin/runExtraction.sh indir outdir`. This uses the main pipeline `main.xgapp` internally to perform the semantic annotation.

The annotation pipeline `main.xgapp` consists of the following sub-pipelines:

1. `tokenisation`: tokenisation of the input text and post-processing of special tokens, like numbers and symbols

2. `sentencesplits`: this uses domain-specific heuristic rules and gazetteer lists to identify certain and likely sentence splits and rule out locations as sentence splits that contain dots in the context of common domain-specific abbreviations.

3. `german`: create German POS tags and find German noun phrases.

4. `radiologyNER`: identify anatomy and pathology terms that can either be linked to Radlex IDs, to UMLS, or to manually created lists of anatomy or pathology concepts.

   The main steps of this sub-pipeline are:

   - Several named entity gazetteer annotation steps, using the following gazetteer lists: Radlex preferred labels in German, for anatomy, pathology, modality, and procedure concepts. Radlex synonym labels which have been guessed to be German, for anatomy and pathology concepts. Manually created lists for anatomy, pathology and modality preferred and synonym terms.
   - Additional gazetteer annotation steps for known incorrect matches
   - Gazetteer annotation steps for matching preferred and alternate German labels from UMLS anatomy concepts
   - Feature gazetteer steps for enriching existing concept annotations with the German preferred term and for enriching certain pathology terms with their implied anatomy.
   - Post-processing steps for disambiguating between several matches at identical or overlapping spans and for identifying if matches have occurred at word boundaries, if they are likely to be part of a composite word or if they are part of a morphologically modified term.

5. `sections`: uses heuristic rules and gazetteers of common section headings to identify the various sections that sometimes can be identified in reports. This is done so it is possible to later filter out named entities which relate to

previous findings or the suspected reasons for the radiological inspections and limit the extraction to actual findings of the radiological inspection.

6. `negation`: identify sentences or parts of sentences where a negated statement about a pathology is made or the absence of a pathology or an observation is stated.

7. `radiologyRel`: identify various kinds of relations between pathology and anatomy concepts and add information about whether the relationship occurs in a negated context.

   The main steps of this sub-pipeline are:

   - `setLookupFlags`: this step adds various features to the annotations for anatomy and pathology named entities. These features contain information about whether the named entity occurs in negated context, within the context of a "Befund" (findings) section, and if the named entity is already part of a relationship.
   - `JAPE:relations01`, `Groovy:relations10`: these steps try to find actual relationships between anatomy and pathology mentions in the text. There are several kinds of relationships – named entities are assigned to relationships starting with the highest priority, highest-confidence kind, then remaining named entities are assigned to the remaining less confident relationships. For each relationship an annotation `RadiologyRel` is created which contains the details of the involved anatomy and pathology concepts (Radlex id, UMLS CUI if available, preferred term, matched string, complete string/suffix/prefix of the match, annotation ids of the anatomy and pathology named entity annotations, the number of the sentence in which the relationship occurs, and others)

   The relationship kinds currently implemented are, in decreasing order of priority:

   - `impliedAnatomy`: This relationship is implied by a single pathology term, e.g. "Steatosis" which expresses both the pathology and the location (in this case, the Liver).
   - `compoundAP`: A relationship which is expressed by a single composite term, e.g. "Nierenzyste".
   - `direct AP`: A relationship where the pathology immediately follows the anatomy term, separated only by white-space or a hyphen.
   - `normalA`: A relationship which expresses that a specific anatomy is stated to be without a pathology.
   - `unchangedA`: A relationship which expresses that the pathology of an anatomy is unchanged in comparison to a previous investigation.
   - `PinA`: A relationship where the presence of some pathology in an anatomy has been explicitly detected through a matching phrase pattern.

- `unknown`: for all pairs of anatomy and pathology terms that occur within a sentence, and where both the anatomy and pathology are not part of any other relationship, a relationship of this kind is created.

For all relationship kinds that contain an actual pathology, the relationship also contains a feature which identifies if the pathology has been mentioned in a negated context or not.

Resources Preparation

There are two kinds of resources which are used in the German radiology pipeline:

- Gazetteer lists: these lists are used to annotate possible occurrences of certain relevant concepts in the document and also to enrich existing annotations with additional information, based on the value of a feature of that annotation. Gazetteer lists are created in three different ways:
  - by querying the Khresmoi knowledge base and processing the query results and converting them to gazetteer lists
  - by manually editing files which are converted into gazetteer lists
  - by analysis of a corpus of reports and extraction of frequent interesting terms. The list of frequent terms is then converted semi-automatically to a gazetteer list.
- Heuristic rules: some heuristic rules have been created purely manually, based on domain knowledge and discussion with domain experts. In addition, a number of heuristic rules have been created based on the result of the analysis of frequent patterns that occur in the corpus.

The following result sets are retrieved from the Khresmoi knowledge base using SPARQL queries:

- German preferred labels for anatomy concepts (4978 concepts, of which 674 have a mapping to a UMLS CUI)
- German preferred labels for pathology concepts (1673 concepts, of which 619 have a mapping to a UMLS CUI)
- German preferred labels for modality concepts (56 concepts, of which 18 have a mapping to a UMLS CUI)
- German preferred labels for procedure concepts (314 concepts, of which 64 have a mapping to a UMLS CUI)
- Mapping of Radlex IDs to UMLS CUIs (20713 pairs): this is used to find UMLS CUIs for Radlex terms where the German label has been added manually.
- Radlex synonym terms for anatomy concepts (19700 rows, of which 4860 are classified to be German or Latin)

- Radlex synonym terms for pathology concepts (2311 rows, of which 1589 are classified to be German or Latin)
- UMLS German preferred labels of anatomy concepts which do have a mapping to Radlex (414 rows)
- UMLS German alternate labels of anatomy concepts which do have a mapping to Radlex (178 rows)
- UMLS German preferred labels of anatomy concepts which do not have a mapping to Radlex (944 rows)
- UMLS German alternate labels of anatomy concepts which do not have a mapping to Radlex (216 rows)

In addition the following manually curated lists are used:

- Additional preferred anatomy terms (37 entries)
- Additional preferred pathology terms (53 entries)
- Additional preferred modality terms (5 entries)
- Additional anatomy synonyms (12 entries)
- Additional pathology synonyms (21 entries)
- Contexts and composites in which to ignore certain anatomy or pathology terms (46 entries)
- Term expansions for generating additional gazetteer entries from existing ones (e.g. abbreviations, frequent Latin/German mappings, frequently used alternate wordings, 128 entries)
- Terms, term suffixes and term prefixes for avoiding matching anatomy or pathology terms (35 entries)

The final gazetteer files are automatically created from these input files.

4 Indexing

The information crawled (Section 2), extracted (Section 3.1) or annotated (Section 3.2) are indexed into the two indexing systems used within *Khresmoi*. This reflects the two text use cases and different software deployment scenarios as described in Section 1.2. Section4.1 gives details of the indexing within the semantic indexing system Mimir, while Section 4.2 describes the process of creating the Lucene/Solr index used by K4E search engine.

4.1 Indexing in Mimir

Documents annotated for the medical practitioner use case are indexed in to the Mimir search engine. Mimir is a multi-paradigm index based on OWLIM[13] and MG4J[14], which allows indexing and retrieval of full text alongside annotations and ontology queries (using SPARQL). Mimir is fully described in [15]. In this section, we first describe the place of Mimir within Khremsoi indexing and search, and then describe the index schemas used for the use cases.

4.1.1 Use of Mimir for indexing and search

Documents crawled as described in Section 2 are retrieved and annotated, using the pipelines described in Section 3. Each document is pushed to an open Mimir index. Once all documents are indexed, this index is closed and made available for search. Retrieval, annotation and pushing to an index is co-ordinated as a workflow, using the Talend data integration tool, and as described in [19]. This allows for parallelisation of indexing. The current architecture uses around 20 indexing threads, but this may be further scaled.

Large indexing jobs are split in to several smaller parts, each comprising 5-10 million documents, which are run in series. This is to guard against overnight failures destroying an entire index. Once all portions are complete, they are federated together in to a single Mimir index.

Two live indexes are maintained: a test index, and a production index. Typically, a new index run will result in a test index, and once this is complete, it will be switched with the existing production index, to give a new production index.

4.1.2 Mimir schema

The Mimir indexes store the text of documents, together with annotations created by the GATE pipelines described in Section 3. The indexed annotations can be broken down in to three groups, as summarised below, and as illustrated in Figure 4. The precise schema is fully described in Tables A.1 and A.2.

- Annotations representing the text of the document. These include Token and Sentence annotations. Token annotations represent the full text of the

[13] https://www.ontotext.com/owlim
[14] http://mg4j.dsi.unimi.it/

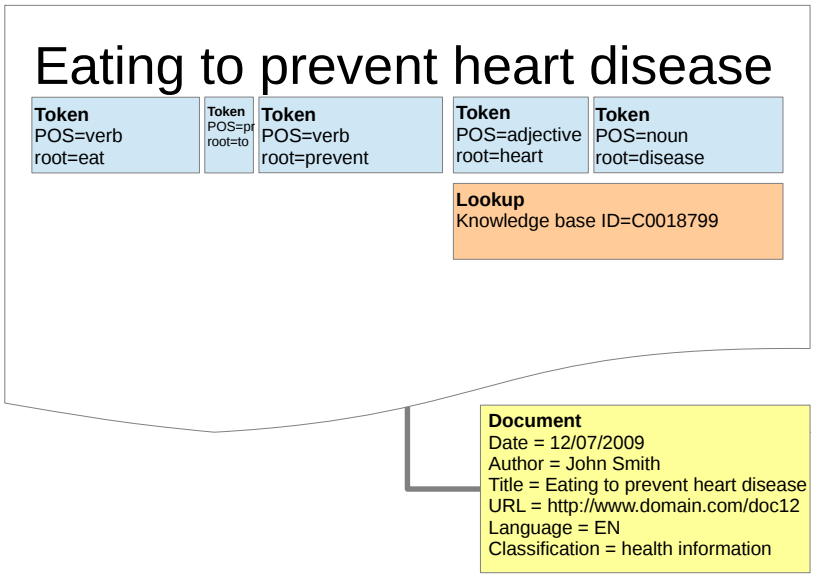


Figure 4: Annotations indexed by Mimir: a partial view. The diagram gives an indication of some of the annotations indexed by Mimir, on a partial document, for illustration. Blue areas illustrate Token annotations, with root and POS (part of speech) features; orange shows the Lookup semantic annotation, with a linkage to the knowledge base; yellow shows a sample of document metadata.

document, stemmed words, the morphological root of words, and their part of speech. The token annotations may be used to perform full text search over Mimir.

- Annotations representing the semantics of terms in the document. The main semantic annotation is called Lookup. This is added to texts by the GATE annotation pipeline. Each annotation contains information linking it to character offsets in the text, and a link to the Khresmoi Knowledge Base. The annotation therefore describes how the referenced text is defined in the knowledge base. The Lookup annotation may be used to perform semantic search, i.e. retrieving those documents that contain a particular concept from the knowledge base.

- Annotations representing metadata. Metadata represented in this way includes dates, authors, and titles. Metadata annotations may be used to constrain, or narrow, the result set returned by a query. For example, a metadata constraint could be defined to describe documents published after a certain date. This constraint could be added to any query, in order to restrict the query's result set to just those documents published after that date. The main metadata annotation is Publication, an annotation that spans the document, and which contains information on document date, authorship, title, and classification.

In addition to using metadata to constrain query result sets as described above, a Mimir client may also retrieve any metadata for any given document, This is done using a *documentMetadata* API call. This is typically used to retrieve metadata, such as document dates, categorisation, etc., for clients to use when rendering documents. The stored metadata is described in Table A.3.

4.2 Indexing in Lucene

In order to be used by the K4E search engine, content extracted from crawled web pages is stored in CouchDB, as described in Section 3.1.1 in [11] and indexed in a Lucene/Solr index. The index contains two type of documents (field "docType" in the TableA.5). "HTML" documents contain the information about the web page such as title, body text etc., while "Image" type document contains textual information related to the image extracted from the web page. Two types of information are indexed, for both HTML and Image documents.
First, basic textual information such as title, text etc. for HTML, and title, alt etc. for Image, are indexed into the language related fields (e.g. title_en or alt_en in case of the web page in English). Complete list of index fields is provided in Table A.5, Table A.4, TableA.6 and Table A.7. The language of the crawled web page is detected by the service described in [11]. This enables usage of language specific analyzers both at indexing and query time, thus augmeting the systems recall. These analyzers perform stop word removal, stemming as well as decompounding [3] for languages such as German or Finnish. A full list of searchable fields for both

HTML and Images is given in the Table A.4.

Second, a large amount of metadata is extracted, stored and indexed (Table A.5). This metadata is used at query time either for display within the search results list (e.g. url, snippet) or to perform grouping of the results returned by the system (e.g. contentMD5 - grouping pages having the same content, domain - grouping pages from the same host). Additionally, each page is enriched with the set of facets listed in Table A.6 as well as sets of facets related to trustability, readability [12] and HON labels [20], as given in Table A.7. The information described above are extracted from the crawled web pages in order to meet the general public use case requirements described in [2]. This enrichment of crawled web pages enables the creation of various filters [8], which are proposed to users within the K4E search engine advanced mode.

5 Ranking

Documents processed as described in Section 3, and indexed as described in Section 4, are queried by Khresmoi user interface clients. This section describes the result set ranking made available by these search engines. Ranking for the Mimir index (4.1 above), is described in Section 5.1. Ranking for the Lucene index (4.2 above), is described in Section 5.2.

5.1 Ranking in Mimir

In the user-centred evaluations with physicians as reported in [1], the ranking of the results was not transparent to the users and did not meet all of their expectations. To this end, we have developed a scorer for Mimir that exploits the rich document metadata created by the document preprocessing, annotation and indexing pipeline described in the previous sections, as well as information from external information sources, to provide a better result ranking. A scorer in Mimir is a module that post processes the search results that are per-se in no particular order by assigning a numerical score to each document in the result set. The documents are ranked from the highest to the lowest score when the search result is accessed. The goal of the new scorer is to surpass the quality of the BM25 scoring that has been used up to now.

One of the major issues identified in the course of the evaluations were that documents from generally "popular" sources, i.e. Web sites with high traffic, did not show up in the top search results. Furthermore, language and locality play an important role in the users' expectations. It was pointed out that documents in the user's own language as well as documents from sources in their own country should be preferably ranked higher.

In Mimir, the scoring of the results takes place after the matching documents have been identified by the core semantic retrieval engine. Mimir provides a plugin-like mechanism that allows the development of custom scorers by implementing the `gate.mimir.search.score.MimirScorer` interface. An index can be configured to use a particular scorer for ranking.

To address the points raised regarding ranking in the user-centred evaluations, we have developed a scorer that can take into account document metadata fields as listed in Table A.3, as well as external information, namely the Alexa Rank as provided by the Alexa Web Information Company, to address the user feedback about popular sites not being present in the top results.

5.1.1 Alexa Rank

Alexa[15], is a long-established website analytics company providing traffic information for more than 30 milion websites worldwide. The best-known metric is the Alexa Traffic Rank, an indicator for the popularity of a web site measured by means of toolbars and browser plugins that are used by millions of people. It combines an

[15] `http://www.alexa.com`

estimated number of daily unique visitors with an estimated number of page views over a sliding window of three months.

To provide a better understanding of the Alexa Rank in the context of Khresmoi, some statistics and samples are given from a test index containing 255,276 documents (`test20131106`). Table A.11 shows the Alexa Ranks of some of the domains present in the document URLs. Specifically, samples from the top-, middle- and bottom-ranked domains are shown. Of the 1,535 domains, 291 (18.9%) do not have an Alexa Rank. The top- ranked site is – not surprisingly – `google.com`, Wikipedia comes third and the first dedicated medical site is `nih.gov` at rank 331 closely followed by `webmd.com` at rank 345. The mean of 3,282,168.66 and the median of 1,109,668.50 show that the rank distribution is asymmetric with a positive, non-parametric skewness of 0.44.

Being a subsidiary of Amazon.com, the website statistics are commercially available for automated processing (pay per request) through the Alexa Web Information Service[16], which is part of the Amazon Web Services offering. To associate a document with the according Alexa Rank of its website, the following processing steps are performed: For each document,

1. extract the domain name from the document's URL,
2. retrieve the Alexa Rank from the Alexa Web Information Service (if not already retrieved), and
3. link the document ID to the Alexa Rank.

The domain name extraction step is not straight-forward, as some websites are hosted by large platforms like Blogspot or Wordpress, but have different Alexa Ranks, e.g. `annromick.wordpress.com` (rank: n/a), `mileon.wordpress.com` (rank: 1,775,658), or `rafalafena.wordpress.com` (rank 3,860,272). We have used a list of *effective* top-level domains provided by the Mozilla Foundation[17]. In addition to the official top-level domains administered by the Internet Assigned Numbers Authority (IANA), this list also includes domains like `blogspot.com`, that have developed an unofficial, de-facto top-level domain status. The list still had to be extended as, for example, `about.com` was not covered.

The Alexa score factor of a domain *dom* is calculated as

$$score_{alexa}(dom) = 1 - \frac{rank_{alexa}(dom) - rankMin}{rankMax - rankMin} \quad (1)$$

where a rank exists, with *rankMin* and *rankMax* are the minimum and maximum rank of all domains in the document set, respectively. If no Alexa Rank is available for a domain, the score factor is set to 0.

[16] `http://aws.amazon.com/en/awis`

[17] `http://mxr.mozilla.org/mozilla-central/source/netwerk/dns/effective_tld_names.dat?raw=1`

5.1.2 Document metadata as score factors

The document metadata fields that can be used to influence scoring can be categorized as follows:

- **Numerical:** Metadata such as the trustability or readability scores can be directly used (with appropriate value normalization, if necessary) for score calculation.
- **Simple textual:** Fields like the document language, for example, can be used by comparing the query language (or a language preference set by the user) and setting the soring factor accordingly to zero or one, depending on whether the languages match, or not.
- **Complex textual:** If documents, where query terms appear in the title or in fields generated by the annotation process (e.g. `health_topic_terms`), should get a higher score, a numerical metric has to be defined reflecting the similarity.

5.1.3 Combining the score factors

To calculate the final score, we use a linear combination of the different factors. It has to be noted that the metadata-derived scoring factors do not replace information retrieval scores like *tf-idf* or *BM25*, but rather are used to boost certain document properties with respect to the query. Hence, a score like *BM25* can be used as the base scoring factor in such a linear combination with a rather high weight and lower weights can be attributed to the other factors.

The weights of the factors are defined in a configuration file that is set as initialization parameter in the Mimir configuration where the scorers available to the indexes are defined. In the future it would also be possible to set the weights dynamically based on user preferences.

More formally, given a set of scoring factors f_1 to f_n with weights w_1 to w_n, the final score of a document d is calculated as follows:

$$score(d) = \sum_{i=1}^{n} w_i * f_i(d) \tag{2}$$

with the sum of the weights being 1.

With this scoring mechanism in place, it is possible to address the issues raised by the professional users in course of the user-centered evaluations.

5.2 Ranking in Lucene

After the query search terms are entered into the K4E search engine, they are processed by the ExtendedDisMax parser [7]. This enables search for the query terms in different fields with different weights(boost). Scoring between query and document is a combination of the Boolean model and Vector Space model of information retrieval [6].The following interface language (e.g. English) related Solr index fields are used as query fields:

- title_en
- h1_en
- h2_en
- h3_en
- text_en

The filed "text_en" contains the body text extracted from the web page for the HTML type documents. Documents having the query matched in the title have a boost of 2 compared to other search fields. Once the list of matching documents has been identified using the search terms, documents matching the whole query (phrase), bigrams or trigrams in close proximity within "title" or "text" fields are boosted in the results list. For example, in the case of the query "breast cancer treatment" documents containing "breast cancer", "cancer treatment", "breast cancer treatment" separated with less that 3 terms will be given higher importance.

Additionally, documents are also boosted based on their medical content relevancy score [11], 3.1, thus giving more importance to health related documents.

Apart from the query entered by the user, synonyms are also taken into account during the process of query formulation. HON has opted for query time synonym expansion for the following reasons:

- the index stays the same size
- synonyms can be changed at any time, with no need to re-index
- original terms keep their occurrence information

However, according to the Solr documentation [9] this is not a good approach since the query time expansion results in the following problems:

1. Multi-word synonyms will not work as phrase queries.
2. The IDF of rare synonyms will be boosted, causing unintuitive results.
3. Multi-word synonyms will not be matched in queries.

To deal with these problems, a custom synonym parser was created as a Solr plugin. This parser also enables application of a custom weight for the original query terms and their synonyms. For K4E, MeSH terms are used for query expansion. Taking for example the query "breast cancer", MeSH proposes these synonyms:

- breast neoplasm
- breast neoplasms
- breast tumor
- breast tumors

- cancer of breast
- cancer of the breast

Thus the query “breast cancer” will be parsed as:

+((”breast cancer”) ^W_o ((”breast neoplasm”) (”breast tumor”) (”cancer breast”) (”cancer breast”))∧W_s)

where W_o and W_s are the original query and synonym weights. In the K4E they have been set to 10 and 0.1, respectfully.

6 Semantic Search Interface

The semantic search interface (SearchPro) within K4E search as described in [20] is available in all European languages, however the search is available only in English. In order to perform the search user can either choose the next query term from the proposed list (Figure5 a)) or type in their own query terms (Figure5 b)).

(a) Choose from the list

(b)Type query

Figure 5: Semantic search interface

The results returned by the query " *'Disease or Syndrome''treated by''Amifostine'* " are given in Figure6. Each result consists of title, snippet and link to the page itself.

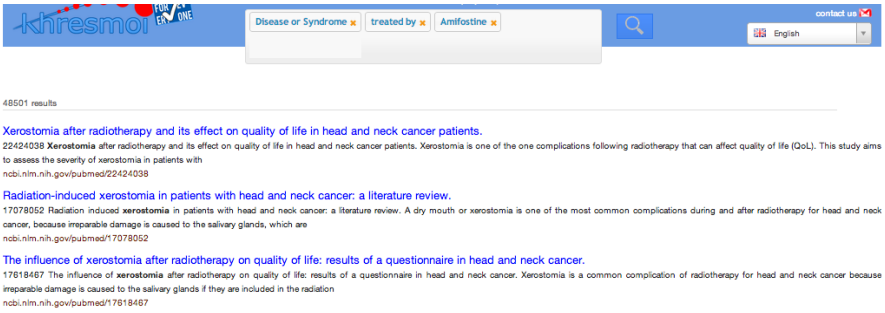


Figure 6: Semantic search results

7 Conclusion and Future Work

This report has presented the end-to-end collection and processing of documents for the Khresmoi project, and their delivery to the end user via ranked indexes. As such, it has presented the technical heart of Khresmoi. We have described how documents are crawled and stored prior to processing. We have described how this processing enriches Khresmoi documents, by adding health section information and scoring documents as to their relevancy for health related search; by adding semantic links to the Khresmoi Knowledge Base, and by adding document metadata. The enriched documents are stored in Lucene and Mimir indexes, and ranked results made available to clients. A specific instance of a client has been presented, in which the user interacts with the knowledge base in order to formulate semantic queries.

The report has presented a large and significant body of work from Khresmoi. The indexing and ranking of semantically enriched documents allows for novel and useful user interactions with the search system. To date, the software described has been used to succesfully perform two Khresmoi index production runs.

The remaining semantic indexing and annotation work within Khresmoi will be focused in two directions:

- Performing further production indexing runs, in order to provide indexes for user tests.
- Correcting problems and errors in response to feedback from these user tests.

8 References

[1] Frederic Baroz, Célia Boyer, Manfred Samia Chahlal, Manfred Gschwandtner, Lorraine Goeuriot, Jan Hajic, Allan Hanbury, Marlene Kritz, Jérémy Leixa, João Palotti, Natalia Pletneva, Rafael Ruiz de Castañeda, Alexander Sachs, Matthias Samwald, Priscille Schneller, Veronika Stefanov, and Zdenka Uresova. D10.1: Report on user tests with initial search system. *Khresmoi project public deliverable*, 2013.

[2] Célia Boyer, Manfred Gschwandtner, Allan Hanbury, Marlene Kritz, Natalia Pletneva, Matthias Samwald, and Alejandro Vargas. D8.2: Use case definition including concrete data requirements. *Khresmoi project public deliverable*, 2012.

[3] Martin Braschler and Bärbel Ripplinger. How effective is stemming and decompounding for german text retrieval? *Inf. Retr.*, 7(3-4):291–316, September 2004.

[4] Hamish Cunningham, Diana Maynard, Kalina Bontcheva, Valentin Tablan, NIraj Aswani, Ian Roberts, Genevieve Gorrell, Adam Funk, Angus Roberts, Danica Damljanovic, Thomas Heitz Mark A. Greenwood, Horacio Saggion, Johann Petrak, Yaoyong Li, and Wim Peters. *Text Processing with GATE (Version 6)*. University of Sheffield, 2011.

[5] Hamish Cunningham, Valentin Tablan, Angus Roberts, and Kalina Bontcheva. Getting more out of biomedical documents with gate's full lifecycle open source text analytics. *PLoS Computational Biology*, 9(2):e1002854, 02 2013.

[6] Apache Software Foundation. Class similarity. `http://lucene.apache.org/core/3_5_0/api/core/org/apache/lucene/search/Similarity.html`, 2014.

[7] Apache Software Foundation. Extendeddismax. `http://wiki.apache.org/solr/ExtendedDisMax`, 2014.

[8] Apache Software Foundation. Solrfacetingoverview. `http://wiki.apache.org/solr/SolrFacetingOverview`, 2014.

[9] Apache Software Foundation. Synonymfilterfactory. `http://wiki.apache.org/solr/AnalyzersTokenizersTokenFilters#solr.SynonymFilterFactory`, 2014.

[10] Jan Hajič. *Disambiguation of Rich Inflection (Computational Morphology of Czech)*. Nakladatelství Karolinum, 2004.

[11] Allan Hanbury, William Belle, Nolan Lawson, Ljiljana Dolamic, Natalia Pletneva, Matthias Samwald, and Célia Boyer. D8.3: Prototype of a first search system for intensive tests. *Khresmoi project public deliverable*, 2012.

[12] Allan Hanbury, Célia Boyer, Ljiljana Dolamic, and João Palotti. D1.6: Report on automatic document categorization, trustability and readability. *Khresmoi project public deliverable*, 2013.

[13] Allan Hanbury, Célia Boyer, Martin Gschwandtner, and Henning Müller. KHRESMOI: towards a multi-lingual search and access system for biomedical infromation. In *Proceedings of Med-e-Tel*, Luxembourg, 2011.

[14] HESSO. Deliverable 9.4.2 Updated prototype incorporating updated specifications and innovative user interface. *Khresmoi project deliverable*, 2012.

[15] HON. Deliverable 4.1.1 Indexed documents in the biomedical domain. *Khresmoi project deliverable*, 2011.

[16] Georg Langs, Andreas Burner, Joachim Ofner, René Donner, Henning Müller, Adrien Depeursinge, Dimitrios Markonis, Célia Boyer, Alexandre Masselot, and Nolan Lawson. D2.2: Report on and prototype for feature extraction and image description. *Khresmoi project public deliverable*, 2012.

[17] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[18] Konstantin Pentchev. Deliverable 5.2 Large Scale Biomedical Knowledge Server. *Khresmoi project deliverable*, 2012.

[19] Konstantin Pentchev. Deliverable 5.5 Sustainable biomedical knowledge infrastructure. *Khresmoi project deliverable*, 2014.

[20] Natalia Pletneva, Sascha Kriewel, and Marlene Kritz. D8.5.2: Prototype of a second search system based on feedback. *Khresmoi project public deliverable*, 2013.

[21] Martin Popel and Zdeněk Žabokrtský. TectoMT: Modular NLP framework. In *Proceedings of the 7th International Conference on Advances in Natural Language Processing (IceTAL 2010)*, pages 293–304, Reykjavik, Iceland, 2010.

[22] Angus Roberts, Johann Petrak, and Niraj Aswani. Deliverable 1.4.2: Report on Coupling Manual and Automatic Annotation. *Khresmoi project deliverable*, 2013.

[23] Ivan Martinez Rodriguez and Miguel Angel Tinte Garcia. D6.4.3: Prototype and evaluation of the Full Cloud Infrastructure. *Khresmoi project public deliverable*, 2013.

[24] Drahomíra Spoustová, Jan Hajič, Jan Raab, and Miroslav Spousta. Semi-supervised training for the averaged perceptron POS tagger. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 763–771, Athens, Greece, 2009.

[25] UDE. Deliverable 3.6 Final flexible user Interface framework and documentation. *Khresmoi project public deliverable*, 2014.

Appendices

A Tables

Table A.1: Annotations indexed in Mimir: lexico-syntactic and semantic

Group	Annotation	Feature	Description
Lexico-syntactic	Token	kind	Part of speech
		category	Number, punctuation, etc.
		root	Morphological root
		stem	From Porter stemmer
	Sentence		
	Original-Document		A single, tokenised, annotation spanning the original document content, excluding anything that has been added
Semantic	Lookup	class	URI of the class in the Khresmoi knowledge base
		inst	URI of the instance in the Khresmoi knowledge base
		semanticType	A textual representation of class, e.g. the label of the UMLS class
		khresmoiType	One of "Drug", "Anatomy", "Disease", "Investigation"
		source	One of "UMLS" or "DrugBank"

Table A.2: Annotations indexed in Mimir: metadata

Annotation	Feature	Description
Publication	date	The date if known, as a number of the form YYYYMMDD, feature missing otherwise
	corpus	Web.HON.Eng, Web.HON.Other, Web.HES-SO.Eng, Web.HES-SO.Other
	containsImages	true or false
	language	
	idType	"WebUrl" or "PubMed"
	locationCountry	country as registered in the HONCode directory for the site, or empty
	locationCity	city es registered in the HONCode directory for the site, or empty
	hesClassification	"professional" or "patient"
Publisher		tokenised journal name for PubMed, empty otherwise
Tag		non-tokenised annotations, with features "name" and "code"
Category		non-tokenised annotations, with feature "name"
Id		For Pubmed, the pubmed id, the URL for all other documents, always tokenised
Domain		The Domain in the form some.host.name for an URL http://www.some.host.name/path.html, tokenised. Has the feature "name" with the identical content
Author		For Pubmed, author names, tokenised and as they occur in Pubmed, otherwise missing
AuthorLastName		For Pubmed, the last name of the author as defined by pubmedi, otherwise missing
AuthorForeName		For Pubmed, the fore name of the author as defined by Pubmed, otherwise missing
img		with feature src, spanning every img tag in the original html document (possibly 0-length), src feature is the (relative URL) of the image itself
title		tokenised, spanning the title in the original html document or the title of the pubmed article by the processing pipeline (i.e. excluding generated metadata text)
DomainClass		with feature "class", giving a classification from the crawl sources document

Table A.3: Metadata retrievable from Mimir via API calls

Metadata item	Description	Notes
date	date published	
dateCrawled	date crawled	
title	title as given in html title tag	
authors	authors of document	pipe separated list
authorsLastNames	last names of authors of document	pipe separated list
publisher	publisher of document	
corpus	part of full index to which this document belongs, e.g. pubmed, wikipedia	
id	unique identifier of document	
idType	type of identifier, e.g. pubmed identifier	
domain	domain part of document URL	
url	URL of document	
locationCountry	location country of publisher	
locationCity	location city of publisher	
hesClassification	classification assigned by medical professionals crawl	
containsImages	does this document contain images?	
images	URLs of images in document	pipe separated list
language	language of document	
categories	HON categories assigned to document	pipe separated list
tags.names	Names of HON tags assigned to document	pipe separated list
tags.codes	Codes of HON tags assigned to document	pipe separated list
readability.difficult	readability of document, numeric	
trustability.Justifiability	justifiability of document, numeric	
trustability.Attribution	attribution of document, numeric	
health_topics	topics covered by document	pipe separated list
health_topic_terms	terms describing health topics	pipe separated list
domainClasses	classes assigned from the crawl sources document	comma separated list

Table A.4: Lucene Searchable Fields

HTML		Image	
Extracted	Index	Extracted	Index
h1	h1_lang	alt	alt_lang
h2	h2_lang	title	title_lang
h3	h3_lang	precedingText	followingText_lang
title	title_lang	largePrecedingText	largePrecedingText_lang
text	text_lang	largeFollowingText	largeFollowingText_lang
		pageTitle	pageTitle_lang

Table A.5: Lucene Meta Fields

Field name	Document type
id	HTML, Image
site	HTML, Image
url	HTML, Image
docType	HTML, Image
language	HTML, Image
contentMD5	HTML, Image
snippet	HTML
description	HTML
domain	HTML, Image
location	HTML
beginDate	HTML
endDate	HTML
imageType	Image
cleanedFilename	Image
height	Image
width	Image
parentUrl	Image
contentLength	Image

Table A.6: List of Facets

Facet	Value	Description
khresmoi_sections	List of extracted sections	Sections extracted from predifined set of hosts
health_topics	List of topics	Health topics to which the web page was classified [11]
health_topic_terms	List of keywords	List of keywords extracted from the web page [11]
is_forum	True/NULL	True if the web page is a forum
has_video	True/NULL	True if the web page containes a video
source_hon	True/NULL	True if the page was crawled by HON
source_hes	True/NULL	True if the page was crawled by HES-SO
hes_label_professional	True/NULL	Site labeled as professional by HES-SO
hes_label_patient	True/NULL	Site labeled as professional by HES-SO
hes_label_for_all	True/NULL	Site labeled as professional by HES-SO

Table A.7: Additional Facets

	Facet	Value
Trustability	trustability_Authoritative trustability_Complementarity trustability_Privacy trustability_Attribution trustability_Justifiability trustability_Transparency trustability_Financial trustability_Advertising trustability_Date	score $[0 \cdots 1]$
Readability	redability_easy redability_difficult	score $[0 \cdots 1]$
HON labels	hon_label_*Code*	TRUE/NULL

Table A.8: HON labels, part 1

Code	Label Name	Description
E011	Continuing medical education	Education and training for health professionals
E61	Various	Various
E41	Association	Other organizations
E121	University Hospital	Healthcare organization
E42	Public institution	Public authorities
E43	Regional Council	Public authorities
E46	Partner organization recognized by the government	Public authorities
E47	Learned medical society	Health professionals' organizations
E44	Health organization	Healthcare organization
E45	Public body	Public authorities
E48	Trade union	Various
E341	Health professional - medical & paramedical	For health professionals
E342	Other individual	For general public
E21	Equipment Supplier	Products&services
E22	Pharmaceutical Group	Products&services
E23	Service Provider	Products&services
E443	Health cooperation group	Other organizations
E442	Committee for the Protection of Persons	Other organizations
E441	Cancerology coordination centre	Healthcare organization
E446	Regional cancerology network	Healthcare organization
E445	Monitoring of drugs, medical devices and therapeutic innovation	Public authorities
E444	Regional Public Health group	Other organizations
E02	University	Education and training for health professionals
E01	Training centre	Education and training for health professionals
E461	Health agency and institute	Healthcare organization
E462	Order	Health professionals' organizations
E414	Corporate Foundation	Products&services
E415	Other association	Other organizations
E411	Registered association	Other organizations
E412	Association of public utility	Other organizations
E413	Foundation of public utility	Other organizations
E51	Social Security	Other organizations
E521	Insurance Company	Products&services
E522	Mutual insurance	Products&services
E431	Governmental department	Public authorities
E114	Other specialty	Healthcare organization
E111	General practice	Healthcare organization
E113	Plastic surgery	Healthcare organization
E112	Dentistry	Healthcare organization
E11	Private medical practice	Healthcare organization
E13	Cancer centre	Healthcare organization
E12	Hospital	Healthcare organization
E15	Clinic	Healthcare organization
E14	Reference Center	Healthcare organization
E17	Pharmacy	Healthcare organization
E16	Biological Laboratory	Healthcare organization
U06	General Population (For-patients)	For general public
U07	Health Professionals	For health professionals
AFF_TIT_LAB_04	Education & Prevention tools	Health and medical information

Table A.9: HON labels, part 2

Code	Label Name	Description
V70	Day to day living	Health and medical information
V21_1	Practice guidelines	Health and medical information
V29_1	CME for Professionals	Education and training for health professionals
V29_2	Patient Education	Health and medical information
V21_2	Publication of Clinical Trials	Research
V56	Medical Informatics	Health and medical information
V57	Education centers	Education and training for health professionals
V54	Professional Network	Health professionals' organizations
V55	Various	Various
V52	Public organizations	Public authorities
V50	Sites of private individual - Health professionals	Web sites of health professionals
V51	Sites of private individual - Non-health professional	Web sites of individuals
V58	Health portal	Health and medical information
V17	Patient support group(s)	Patients organizations
V33_1	Demographics & epidemiology	Health and medical information
V33_2	Prognosis	Health and medical information
V31_1	Differential diagnosis	Health and medical information
V23	Women's health	Health and medical information
V21	Scientific publication (Research paper)	Research
V20	Universités	Education and training for health professionals
V27	Medical records	Health and medical information
V25	Research	Research
V29	ContinuingEducation (CME)	Education and training for health professionals
V28	Case Studies	Health and medical information
V37_1	Procedure overview	Health and medical information
V37_2	Procedure pre-op and post-op	Health and medical information
V19_1	Societies recognized by the French Medical association	Other organizations
V08_1	Hospitals	Healthcare organization
V08_2	Clinics	Healthcare organization
V08_3	Medical Offices	Healthcare organization
V19_2	Other societies (not recognized)	Other organizations
V19_3	Other professional organizations	Health professionals' organizations
V05	Description of services or products	Products&services
V04	Dictionary (Definitions)	Health and medical information
V07	FAQs	Health and medical information
V06	Drug(s)	Health and medical information
V01	Alternative medicine	Health and medical information
V03	Clinical trial(s) (ongoing)	Research
V09	Insurance companies	Products&services
V08	Hospital, Clinic Medical center(From-Medical-Establishment)	Healthcare organization
V62	Health administration sites	Public authorities
V61	Editeur et/ou diffuseurs dédiés à l'information de santé	Health and medical information
V60	Other providers of health goods and services	Products&services
V60_1	Providers of medical devices	Products&services
V60_2	Providers of medical materials	Products&services
V41	NGO	Other organizations
V40	Genetics	Health and medical information
V43	e-Health (Telemedecine)	Health and medical information
V42	Patient's associations	Patients organizations
V45	Pharmaceutic group	Products&services
V44	Governmental websites	Public authorities
V47	Paper press	Other organizations
V46	Sounds	Other organizations
V49	Television	Other organizations
V48	Radio	Other organizations

Table A.10: HON labels, part 3

Code	Label Name	Description
V20_1	Universities	Education and training for health professionals
V30	Causes And Risk Factors	Health and medical information
V31	Signs And Symptoms	Health and medical information
V32	Treatment	Health and medical information
V33	Statistics	Health and medical information
V34	Prevention	Health and medical information
V35	NonHealth Medical Site	Various
V36	Tests	Health and medical information
V37	Procedures	Health and medical information
V38	Images	Health and medical information
V39	Videos	Video
V13	Medical or health information (DiseaseInformation)	Health and medical information
V36_1	Test overview	Health and medical information
V36_3	How to interpret results	Health and medical information
V36_2	How to prepare for test	Health and medical information
V06_1	Drug uses (indications and dosing)	Health and medical information
V06_2	Side effects	Health and medical information
V06_3	Interactions	Health and medical information
V06_4	Warning & recalls	Health and medical information
V19	Health associations, organisations, and professionals	Healthcare organization
V02_5	Reseaux communautaires	Other organizations
V16	Nutrition (Diet)	Health and medical information
V14	Men's health	Health and medical information
V15	News	Health and medical information

Table A.11: Sample domains with Alexa Traffic Rank

Domain	Alexa Rank
google.com	1
youtube.com	3
yahoo.com	4
wikipedia.org	6
microsoft.com	33
huffingtonpost.com	64
free.fr	286
nih.gov	331
webmd.com	345
...	...
addictionrecoveryguide.org	722,387
dh.org	725,911
spauldingrehab.org	731,729
chu-bordeaux.fr	736,387
womenshealthmatters.ca	737,128
nhs24.com	737,744
medtempus.com	739,177
thepharmaletter.com	745,769
...	...
appointmentseasy.com	22,944,197
diamip.org	23,425,688
angeliq-us.com	23,680,251
hudlegekontoret.no	23,819,541
vesalius.com	23,869,306
wileyonlinelibrary.com	23,994,266
amcvhs.com	24,012,454
isge.org	24,140,280