TECHNOLOGICAL EDUCATIONAL INSTITUTE OF CRETE

SCHOOL OF ENGINEERING

DEPARTMENT OF INFORMATICS ENGINEERING

THESIS

ONTOLOGY-BASED SEARCH ENGINE WITH SPEECH RECOGNITION

ALEXANDROS KYRIAKAKIS

SUPERVISOR
PROFESSOR MANOLIS TSIKNAKIS

HERAKLION
FEBRUARY 2019

Copyright

Abstract

Searching for data throughout the internet is not always an easy task. The reason for that is that the internet is flooded with data and the most common searching techniques often fail to provide the best possible results for the users' needs. Therefore, researchers have for years been trying to find alternative ways of searching. Through that research, searching techniques based on semantic technologies have emerged. Over the years, search engines based on semantic technologies have gained a lot of popularity for their efficiency, but they are still under development.

The end goal of this thesis is to examine the use of semantic technologies for bioinformatics data annotation by employing domain specific ontologies and also to make use of natural language processing methods in order to aid the non-bioinformatics expert users to successfully search for bioinformatics related tools using natural language through either text or a speech recognition system. In addition, we make use of more searching techniques to compare their efficiency and also to ensure that the framework that was created throughout this thesis, is be able to provide the user with the best possible results for his needs.

For our experiments, a variety of questions of users searching for bioinformatics related tools from the SEQanswers.com forum were collected, and after having irrelevant words from those questions removed from two expert bioinformaticians, those questions were used as queries in the proposed framework. For the evaluation of the results, they were compared with the proposed tools that were given by forum users as answers for the collected questions. Our results indicate that in the majority, the framework's results are relevant to the queries, with each searching technique having different precision and response times.

Περίληψη

Η αναζήτηση δεδομένων στο διαδίκτυο δεν είναι πάντα εύκολη. Αυτό συμβαίνει επειδή το διαδίκτυο έχει πάρα πολλά δεδομένα και οι πιο διαδεδομένες τεχνικές αναζήτησης συχνά αποτυγχάνουν στο να παρέχουν τα καλύτερα αποτελέσματα για τις ανάγκες των χρηστών. Για αυτό το λόγο, ερευνητές προσπαθούν εδώ και χρόνια να βρουν εναλλακτικούς τρόπους αναζήτησης. Από αυτές τις έρευνες, ανακαλύφθηκαν τεχνικές αναζήτησης βασισμένες σε σημασιολογικές τεχνολογίες. Με τα χρόνια, μηχανές αναζήτησης βασισμένες σε σημασιολογικές τεχνολογίες έχουν απόκτηση πολύ φήμη για την αποτελεσματικότητά τους, όμως είναι ακόμα υπό ανάπτυξη.

Ο τελικός στόχος αυτής της πτυχιακής εργασίας είναι να ερευνηθεί η χρησιμότητα σημασιολογικών τεχνολογιών για τον σχολιασμό δεδομένων βιοπληροφορικής χρησιμοποιώντας οντολογίες συγκεκριμένου τομέα και επίσης να χρησιμοποιηθούν μέθοδοι επεξεργασίας φυσικής γλώσσας με στόχο να βοηθηθούν χρήστες που δεν ειδικεύονται στην βιοπληροφορική να κάνουν επιτυχής αναζήτηση για προγράμματα που σχετίζονται με την βιοπληροφορική χρησιμοποιώντας φυσική γλώσσα με κείμενο ή ομιλία. Επιπλέον, χρησιμοποιούμε περισσότερες τεχνικές αναζήτησης για να συγκρίνουμε την αποτελεσματικότητά τους και επίσης για να διασφαλίσουμε ότι το πρόγραμμα που αναπτύχθηκε σε αυτή την πτυχιακή εργασία, παρέχει τα καλύτερα δυνατά αποτελέσματα για τις ανάγκες του χρήστη.

Για τον αξιολόγηση του συστήματος, συλλέξαμε μία ποικιλία ερωτήσεων από χρήστες που έψαχναν για προγράμματα που σχετίζονται με την βιοπληροφορική από το φόρουμ SEQanswers.com, και αφού δύο ειδικοί βιοπληροφορικοί αφαίρεσαν μη-σχετιζόμενες λέξεις από τις ερωτήσεις, αυτές οι ερωτήσεις χρησιμοποιήθηκαν σαν ερωτήσεις αναζήτησης στο σύστημά μας. Για την αξιολόγηση των αποτελεσμάτων, τα αποτελέσματα συγκρίθηκαν με τα προγράαμματα που προτάθηκαν από χρήστες του φόρουμ σαν απάντηση για τις ερωτήσεις που συλλέξαμε. Τα αποτελέσματά μας δείχνουν ότι στην πλειοψηφία, τα αποτελέσματα του συστήματος σχετίζονται με τις ερωτήσεις αναζήτησης, ενώ κάθε τεχνική αναζήτησης έχει διαφορετική ακρίβεια και χρόνο ανταπόκρισής.

Acknowledgements

I take this opportunity to express my sincere gratitude to my supervisor Prof. Manolis Tsiknakis for his support throughout the development of this thesis. I would also like to thank Mr. Lefteris Koumakis, Mr. Giorgos Potamias and Mr. Alexandros Kanterakis for their continuous guidance, motivation and encouragement extended to me. I am extremely thankful for their valuable comments and feedback that lead me to widen my research and evolve this thesis while expanding my knowledge.

Table of Contents

Table of Figures

Table of Tables

1. Introduction

1.1. Speech Recognition

In the past few years, speech recognition has been an active research area. It has for years been considered a good bridge of communication between human and machine since speech is the primary mean of human communication, but only recently technology grew enough for automatic speech recognition systems to develop in a usable level. It has been the most exciting area of signal processing and it continues to develop rapidly. Many applications use it nowadays to change the way we live and work for the better. Those applications include: Voice Search, Digital assistance, Gaming, Living room and in-vehicle informative systems. Though, speech recognition systems are still far from perfect. Even though the error rates have reduced significantly, with the most accurate system having approximately 91% word accuracy, they error rates are still high enough to make those systems usable for specific purposes only [1]. Older human-machine communication means such as keyboard and mouse are significantly more accurate and efficient which have kept speech recognition systems in the dark. But the desire for humans to communicate with machines in their most common mean of communication, speech, have kept people into trying to improve those systems more and more. Converting sound signal to text and creating a pattern out of free speech has been the main problem of the attempts to translate natural human speech into machine language over the years.

1.2. Ontology-based Search

Nowadays there is an enormous amount of data in the web. Therefore, searching for something in the web is not an easy task. To make it easier for the users to challenge with the massive data that is massively growing in the web and receive the desirable result, it is necessary to change the traditional search methods. When searching for something specific, a regular search engine retrieves data based on keywords. The user has to spend a decent amount of time to screen and filter [2]. That problem can be partly resolved by using Ontology-based search engines [3]. An ontology is a vocabulary that represents a specific domain or subject matter [4]. Each ontology contains a set of terms. Each term can be weighed differently in the searching process which changes the results for a given query. The weighing of the terms is made to suit the domain of each ontology and improve the accuracy of the resulted data. The use of ontologies for searching is a semantic technology technique. Therefore, a search engine that is based on ontologies is also considered a semantic search engine.

1.3. Keyword-based Search vs Semantic Search

The web is flooded with keyword-based search engines. The most popular are Google, Bing and Yahoo with 80.65%, 8.76% and 7.77% volume of search ratios respectively

[5]. That means that a total of 97.18% of the searches are done in just 3 keyword-based search engines. However, semantic search engines are an alternative to these regular search engines. In a keyword-based search engine, the query is processed for preprocessing and text cleaning and afterwards the search is done. The retrieved documents are based on the amount of the matching words with the query. However, in semantic search engines the results are based on more factors. The frequency of the words, the syntactic structure of the natural language and other elements. In semantic search, the system understands the exact requirement of the user's search query and returns the corresponding documents which saves the user a lot of time discarding unrelative documents a regular search engine would have retrieved. Researchers have conducted a number of studies into evaluating the effectiveness of keyword-based search engines and some have proposed semantic technologies as a better alternative. Although the semantic search engines' popularity is increasing, it still is in very low percentages of the population's preferred search engines. Therefore, researchers have to improve such systems even more in order to retrieve more attention by the general public. Neither of the searching techniques is perfect, since a perfect search engine model would be the one that always returns the exact document the user was looking for. But the gap between present search engines and the 'perfect search engine' is constantly merging.

1.4. The purpose of the study

The end goal of this study is to

I. Investigate the use of semantics for the annotation of free text with domain specific ontologies.
II. Exploit open source systems in empowering the non-IT expert users to efficiently search domain specific content using natural language with the aid of specialized ontologies.
III. Create a search engine that takes advantage of both keyword based and ontology based techniques and compare their efficiency.

Our specific focus is to take advantage of existing research results and extend them in order to provide the users with the chance to represent their search queries in natural language (text or speech) and to dynamically find and retrieve suitable candidate resources, with the aid of information extraction algorithms guided by specific domain ontologies. For the proposed framework, we used the ontologies Embrace Data and Methods(EDAM) [6] and the Software Ontology(SWO) [7]. By taking advantage of Apache solr [8], we were able to translate the data from all the tools from the biotools registry [9] into ontology terms. By following the same process during the user's search, we translate the user's query into ontology terms, and by using a matching algorithm, we can determine the best tools from the biotools registry for the user's needs.

2. Background

2.1. Speech Recognition Background

Figure 1. Radio Rex: The first speech recognition machine. (Source: https://www.pinterest.com/pin/158963061822822448/)

Speech recognition has been a matter of discussion for almost a century. The first machine to recognize speech was manufactured in 1920. A toy, commercially named as Radio Rex that would move in the hearing of the word 'Rex' [10]. Further research on speech recognition began in 1936 with the result being a speech synthesis machine. Researching was abandoned due to a false conclusion that Artificial Intelligence technology [11] would be mandatory for further success.

In the 50s, the first attempts to create a system with automatic speech recognition were made. The main logic behind it was to identify the vowel part of each digit which could have a result of recognizing distinct syllables. The most successful achievement of that period was the vowel recognizer which was constructed at MIT Lincoln Labs [12].

During the 1960-1970 decade, special hardware for speech recognition were built since computers were too slow. Many new labs entered that area of research which helped solve the problem of identifying the time frames of each speech event. Being able to detect when speech starts and ends, resulted into significantly better recognition scores and less variety in the results. In the decade to come, many milestones would be achieved. Firstly, singular word recognition would become a usable technology based on several studies. There was a significant increase in the use of pattern recognition ideas in speech recognition. Other researches showed how ideas of linear predictive coding [13] and dynamic programming methods [14] could be applied for speech recognition. Also, the variety in the vocabulary recognized by the systems was extended greatly which was probably the most important milestone achieved that decade.

Another achievement of that decade was a system that would exclude a number of alternative results by using semantic information [15].

Now that isolated word recognition was in a desired level, in the next decade, the 1980s, the target problem to be solved was to recognize connected words. Which would as a result have a system able to recognize a string of words spoken one after the other. To achieve that, an algorithm to create a pattern by connecting individual words had to be used in various ways. Many matching procedures were used with each of them having its own implementation advantages. The one matching procedure that stood out that decade was the Hidden Markov Model [16] approach. That technique, with some improvements, is being used today on most of the practical speech recognition systems. Hidden Markov Model(HMM) is a doubly stochastic process that can be observed only through another stochastic process that produces a series of observations. Another technology introduced in the 80s was the idea of applying neural networks to solve problems in speech recognition systems. Although those networks were introduced in the 50s, they were not as useful initially because of the many practical problems they had at that time. In the 80s though, there was a deeper understanding of the strengths and limitations of neutrals networks which made them useful in improving speech recognition. Using the systems mentioned above, a large research program was introduced that aimed at achieving high word accuracy for a 1000 word continuous speech recognition, database management task. That program was called DARPA [17] and it was achieved with major research contributions from many significant laboratories.

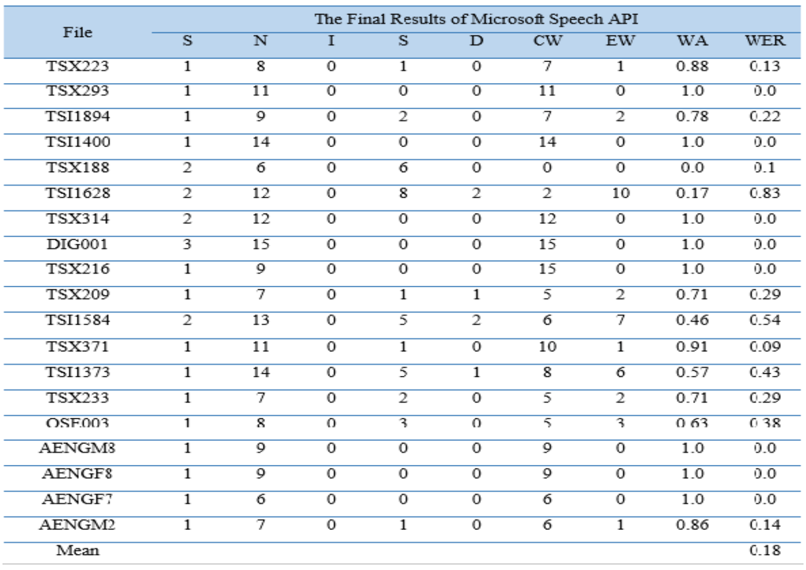
In the 90s, many innovations took place in the pattern recognition area. The problem of pattern recognition which required estimation of distributions for the data, changed into a problem involving minimization of the empirical recognition error. This fundamental change happened because it was accepted that the Bayes decision theory [18] could not be applied on speech recognition with the desired accuracy since it focused on providing the most fitting of a distribution function to the given dataset rather than helping to achieve the least recognition error, so that theory became inapplicable. This error reducing concept produced many techniques like discriminative training [19] and kernel based [20] methods. The Minimum Classification Error(MCE) [21] criterion together with a corresponding Generalized Probabilistic Descent(GPD) [22] training algorithm was an example of discriminative training and acted to approximate the error rate. One more example was the Maximum Mutual Information(MMI) [23] criterion. In this training, the common information between the acoustic observation and its correct lexical symbol averaged over a training set is maximized. Both MCE and MMI can lead to the best possible speech recognition performance. A key issue in those techniques, is choosing the most fitting speech material to train the recognition algorithm which actually means learning parameters of primitive speech patterns used to define basic speech units. The DARPA and HMM systems were also furtherly evolved in that decade.

During the next decade, the 2000s, some clustering techniques [24] were developed as well as a variational Bayesian estimation. This Bayesian approach was based on a posterior scattering of parameters. The problem of adaptive learning in speech recognition was solved by a technique that was developed and there was an introduction of the active learning algorithm [25] for automatic speech recognition. There were some improvements on Large Vocabulary Continuous Speech Recognition system [26] in 2005 to improve performance. In the next couple of years there was an analyzation of the difference in acoustic features between spontaneous and read speech using a big speech database. The application of Conditional Random Field [27] to combine local posterior estimates provided by multilayer perceptions was explored and they also tried to overcome time dependency problems by applying straight forward template matching methods. Around 2007, there was a proposal of using an alternative method that processes the group delay feature for processing the Fourier transform phase for extraction speech features [28].

2.2. Speech Recognition APIs

2.2.1. The Microsoft API

Microsoft's Speech API [29] has been developing since 1993. There is a series of increasingly improving speech recognition platforms released since that time. Speech API (SAPI) 5.3 [30] was very useful and worked on Windows Vista. As they announced "Windows Vista includes a new WinFX namespace, System.Speech. this allows developers to easily speech-enable Windows Forms applications and apps based on the Windows Presentation Framework". Microsoft's focus was to increase emphasis on SR systems and improve SAPI by using Hidden Markov Model(HMM). Recently Microsoft made the following announcement "History Achievement: Microsoft researchers reach human parity in conversational speech recognition".

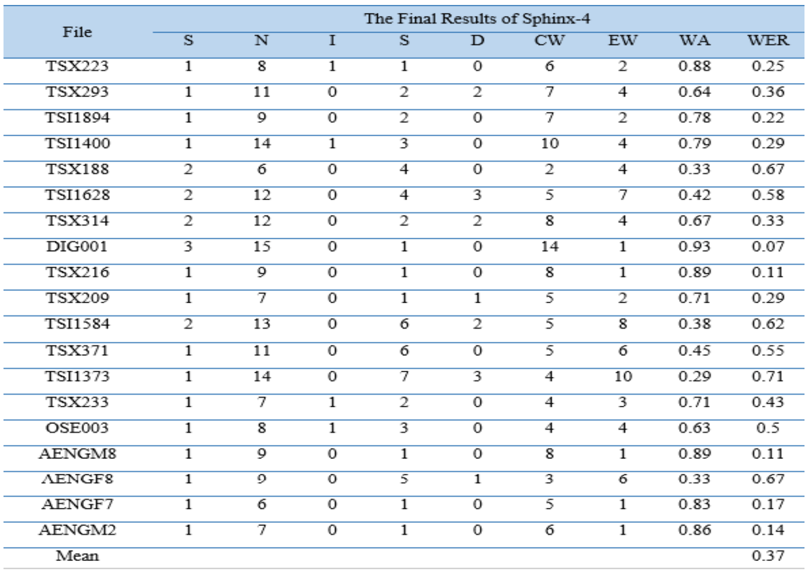


File	The Final Results of Microsoft Speech API								
	S	N	I	S	D	CW	EW	WA	WER
TSX223	1	8	0	1	0	7	1	0.88	0.13
TSX293	1	11	0	0	0	11	0	1.0	0.0
TSI1894	1	9	0	2	0	7	2	0.78	0.22
TSI1400	1	14	0	0	0	14	0	1.0	0.0
TSX188	2	6	0	6	0	0	0	0.0	0.1
TSI1628	2	12	0	8	2	2	10	0.17	0.83
TSX314	2	12	0	0	0	12	0	1.0	0.0
DIG001	3	15	0	0	0	15	0	1.0	0.0
TSX216	1	9	0	0	0	15	0	1.0	0.0
TSX209	1	7	0	1	1	5	2	0.71	0.29
TSI1584	2	13	0	5	2	6	7	0.46	0.54
TSX371	1	11	0	1	0	10	1	0.91	0.09
TSI1373	1	14	0	5	1	8	6	0.57	0.43
TSX233	1	7	0	2	0	5	2	0.71	0.29
OSE003	1	8	0	3	0	5	3	0.63	0.38
AENGM8	1	9	0	0	0	9	0	1.0	0.0
AENGF8	1	9	0	0	0	9	0	1.0	0.0
AENGF7	1	6	0	0	0	6	0	1.0	0.0
AENGM2	1	7	0	1	0	6	1	0.86	0.14
Mean									0.18

Figure 2. Experimental results of The Microsoft API on random audio files.
(Source: https://pdfs.semanticscholar.org/2e7e/bdd353c1de9e47fdd1cf0fce61bd33d87103.pdf)

2.2.2. The CMU SPHINX

The Sphinx system was developed at Carnegie Mellon University(CMU). It has a large vocabulary and it is an open source system. There are different versions for different tasks. Sphinx-4 [31] is written in Java and it supports finite grammar called Java Speech API and it doesn't use the same structure for all models. CMU Sphinx uses a strong acoustic model by using HHM model with training large vocabulary and the latest version of an HMM-based speech.

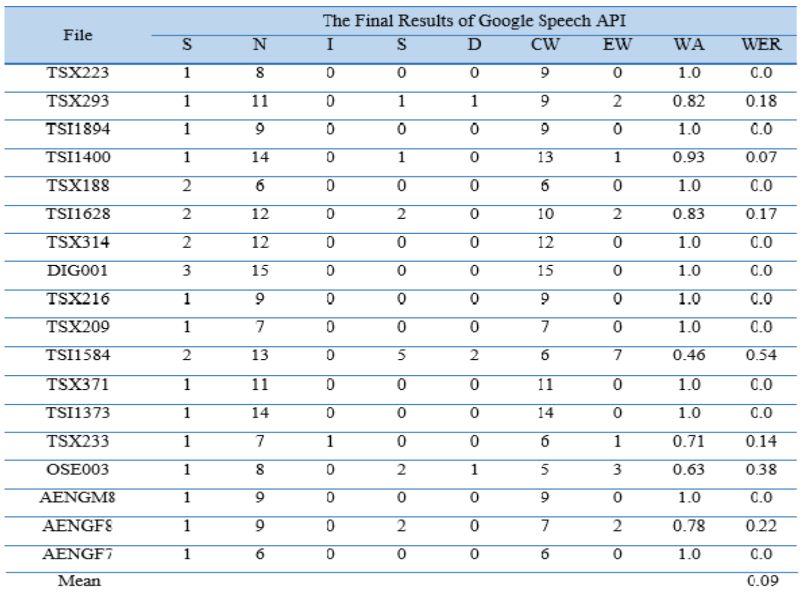


File	The Final Results of Sphinx-4								
	S	N	I	S	D	CW	EW	WA	WER
TSX223	1	8	1	1	0	6	2	0.88	0.25
TSX293	1	11	0	2	2	7	4	0.64	0.36
TSI1894	1	9	0	2	0	7	2	0.78	0.22
TSI1400	1	14	1	3	0	10	4	0.79	0.29
TSX188	2	6	0	4	0	2	4	0.33	0.67
TSI1628	2	12	0	4	3	5	7	0.42	0.58
TSX314	2	12	0	2	2	8	4	0.67	0.33
DIG001	3	15	0	1	0	14	1	0.93	0.07
TSX216	1	9	0	1	0	8	1	0.89	0.11
TSX209	1	7	0	1	1	5	2	0.71	0.29
TSI1584	2	13	0	6	2	5	8	0.38	0.62
TSX371	1	11	0	6	0	5	6	0.45	0.55
TSI1373	1	14	0	7	3	4	10	0.29	0.71
TSX233	1	7	1	2	0	4	3	0.71	0.43
OSE003	1	8	1	3	0	4	4	0.63	0.5
AENGM8	1	9	0	1	0	8	1	0.89	0.11
AENGF8	1	9	0	5	1	3	6	0.33	0.67
AENGF7	1	6	0	1	0	5	1	0.83	0.17
AENGM2	1	7	0	1	0	6	1	0.86	0.14
Mean									0.37

Figure 3. Experimental results of The CMU Sphinx system on random audio files. (Source: https://pdfs.semanticscholar.org/2e7e/bdd353c1de9e47fdd1cf0fce61bd33d87103.pdf)

2.2.3. Cloud Speech API

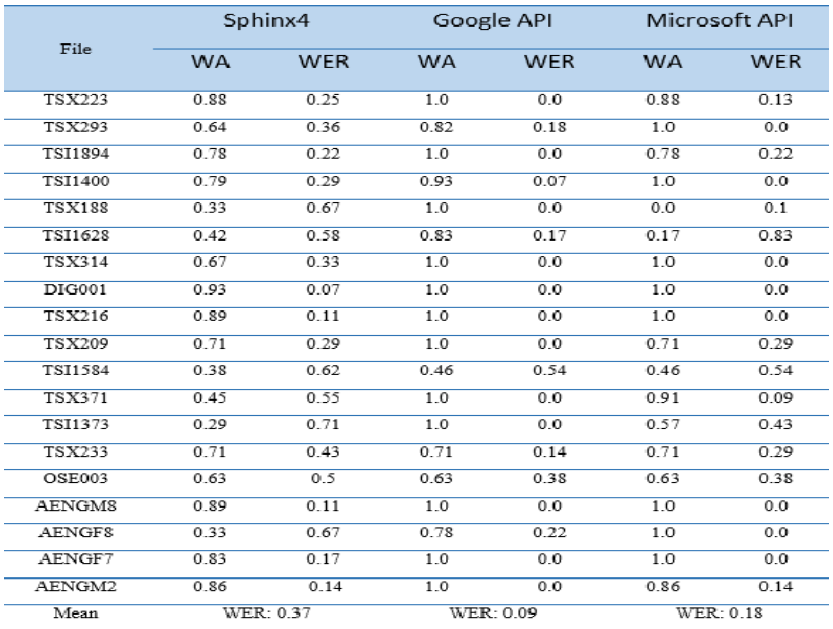
Cloud Speech [29] is an API created by google. Google has improved its speech recognition greatly by using a new technology that uses the deep learning neural networks. Google currently has by far the least percent of error rate with that being around 8%. Huge improvement in the past few years, in comparison with it being around 30 percent in 2013. According to Sundar Pichai, CEO of Google Inc. "We have the best investments in machine learning over the past many years. Indeed, Google has acquired several deep learning companies over the years, including DeepMind [32], DNNresearch [33] and JetPac [34]".



File	The Final Results of Google Speech API								
	S	N	I	S	D	CW	EW	WA	WER
TSX223	1	8	0	0	0	9	0	1.0	0.0
TSX293	1	11	0	1	1	9	2	0.82	0.18
TSI1894	1	9	0	0	0	9	0	1.0	0.0
TSI1400	1	14	0	1	0	13	1	0.93	0.07
TSX188	2	6	0	0	0	6	0	1.0	0.0
TSI1628	2	12	0	2	0	10	2	0.83	0.17
TSX314	2	12	0	0	0	12	0	1.0	0.0
DIG001	3	15	0	0	0	15	0	1.0	0.0
TSX216	1	9	0	0	0	9	0	1.0	0.0
TSX209	1	7	0	0	0	7	0	1.0	0.0
TSI1584	2	13	0	5	2	6	7	0.46	0.54
TSX371	1	11	0	0	0	11	0	1.0	0.0
TSI1373	1	14	0	0	0	14	0	1.0	0.0
TSX233	1	7	1	0	0	6	1	0.71	0.14
OSE003	1	8	0	2	1	5	3	0.63	0.38
AENGM8	1	9	0	0	0	9	0	1.0	0.0
AENGF8	1	9	0	2	0	7	2	0.78	0.22
AENGF7	1	6	0	0	0	6	0	1.0	0.0
Mean									0.09

Figure 4. Experimental results of Cloud Speech API on random audio files (Source: https://pdfs.semanticscholar.org/2e7e/bdd353c1de9e47fdd1cf0fce61bd33d87103.pdf)

2.2.4. Comparison Chart



File	Sphinx4		Google API		Microsoft API	
	WA	WER	WA	WER	WA	WER
TSX223	0.88	0.25	1.0	0.0	0.88	0.13
TSX293	0.64	0.36	0.82	0.18	1.0	0.0
TSI1894	0.78	0.22	1.0	0.0	0.78	0.22
TSI1400	0.79	0.29	0.93	0.07	1.0	0.0
TSX188	0.33	0.67	1.0	0.0	0.0	0.1
TSI1628	0.42	0.58	0.83	0.17	0.17	0.83
TSX314	0.67	0.33	1.0	0.0	1.0	0.0
DIG001	0.93	0.07	1.0	0.0	1.0	0.0
TSX216	0.89	0.11	1.0	0.0	1.0	0.0
TSX209	0.71	0.29	1.0	0.0	0.71	0.29
TSI1584	0.38	0.62	0.46	0.54	0.46	0.54
TSX371	0.45	0.55	1.0	0.0	0.91	0.09
TSI1373	0.29	0.71	1.0	0.0	0.57	0.43
TSX233	0.71	0.43	0.71	0.14	0.71	0.29
OSE003	0.63	0.5	0.63	0.38	0.63	0.38
AENGM8	0.89	0.11	1.0	0.0	1.0	0.0
AENGF8	0.33	0.67	0.78	0.22	1.0	0.0
AENGF7	0.83	0.17	1.0	0.0	1.0	0.0
AENGM2	0.86	0.14	1.0	0.0	0.86	0.14
Mean	WER: 0.37		WER: 0.09		WER: 0.18	

Figure 5. Comparison chart between Sphinx4, Google API and Microsoft API. Word Accuracy (WA), Word Error Rate (WER) (Source: https://pdfs.semanticscholar.org/2e7e/bdd353c1de9e47fdd1cf0fce61bd33d87103.pdf)

2.2.5. Speech APIs Conclusion

As shown on the chart above, Google's **Cloud Speech API** is by far superior than the other systems in terms of Word Accuracy(WA) which is approximately 91% and Word Error Rate(WER) which is by far the lowest with it being approximately 9%.

2.3. Systems taking advantage of Speech APIs

With the rapid growth of systems that use speech recognition for a variety of reasons, there is a growth of need for speech recognition APIs as well. As shown on the comparison above, the most efficient speech recognition API is by far Google's Cloud Speech API. There are many systems using this API as part of their functionality.

One of them is Google Assistant [35]. Google is using their own Speech Recognition API for their virtual assistant called Google Assistant. This system has been improved by using the Deep Neural Networks(DNN)[36] method which has the ability to highlight the main component of the dialogue systems and also by using several new deep learning architectures.

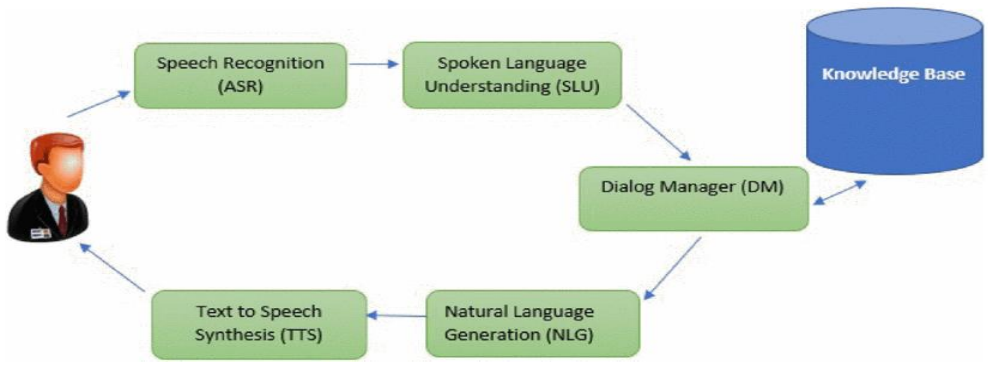

Speech Recognition (ASR)
Spoken Language Understanding (SLU)
Knowledge Base
Dialog Manager (DM)
Natural Language Generation (NLG)
Text to Speech Synthesis (TTS)

Figure 6. Google Assistant Structure.
(Source: https://ieeexplore.ieee.org/document/8301638)

2.4. Semantic Web Background

Semantic Web term originated by Tim Berners-Lee in 1999 in his book "Weaving the Web" where he also described the importance of metadata [37]. In 2000, Taalee company wrote a detailed description of the possible capabilities of the semantic search engine and how it could make use of the semantic web [38]. Later that year, the first international event about the Semantic Web took place describing already implemented semantic applications including semantic search [39]. During the next year, Tim Berners-Lee along with James Hendler and Ora Lassila wrote an article called 'The Semantic Web' [40] which is now the most cited article ever existed featuring the semantic web. The article gave emphasis on describing ontologies, knowledge representation, the role of agents and made some notes about future applications on the Semantic Web [41]. In the years to come, a lot of commercial applications were built using the technology that was built by Taalee which was merged with Voquette and finally merged to became Semagix [42].

In 2005, Peter Norvig wrote a blog called "Semantic Web Ontologies: What Works and What Doesn't" [43] that described problems that might occur by using semantic technologies such as lack of RDF [44] and OWL [45] files, multiple ontologies, CYC approach [46] to retrieving background knowledge, relying on the content supplier to provide metadata and that the most significant information is mostly in plain text form. A blog that was later argued to be mostly inaccurate on what would cause a problem and what not in using semantic technologies.

A few more years passed until big companies like Google and Bing started building large knowledge bases but eventually shied away from using Semantic Web standards. Microsoft acquired knowledge Powerset after realizing its importance. Apple created Siri which was using ontologies to capture knowledge. Google finally acquired Metaweb [47] which was consisted of 12 million facts initially and went along to build its own Knowledge Graph with automated and semiautomated tools.

Since then this area is being developed more and more and as a result there are now numerous systems using implemented ontology web search engines. They are based on a common set of ideas but in contrary to the regular search engines, ontology-based search engines retrieve information after a process that uses knowledge of the domain ontology. Domain ontology is a structured list of terms that describe a specific domain and can be used as a knowledge base.

2.5. Systems taking advantage of semantic technologies

2.5.1. Swoogle

Swoogle [48] was the first search engine for the semantic web and its main contributor was Li Ding. Swoogle is a crawler-based retrieval and indexing system for RDF and OWL documents. It consists of four main components: Semantic Web Document(SWD) discovery, data analysis, metadata creation and interface [49]. The SWD discovery component is responsible for searching and finding Semantic Web Documents throughout the web and keep up with the SWDs' updates. The second component, metadata creation, caches a snapshot of a semantic web document and it generates objective metadata about it in both syntax and semantic level. The data analysis component uses the result data of the previous two components to derive analytical reports. Lastly, the interface component's focus is to provide data to the Semantic Web community. Those four components of Swoogle can work independently and on different tasks.

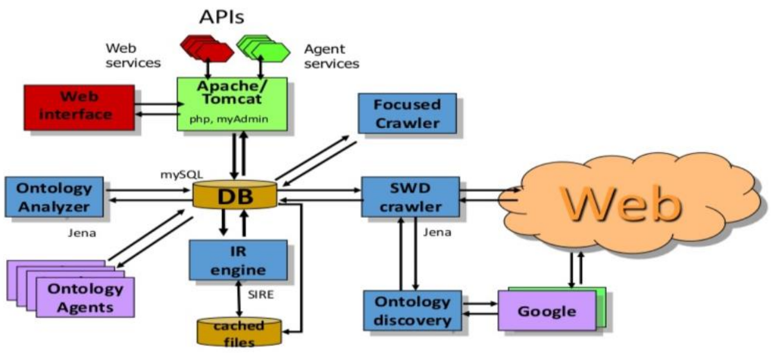


Figure 7. An overview of the swoogle architecture.
(Source: https://www.slideshare.net/suchinipriyangika/swoogle-69623597)

2.5.2. Watson Search Engine

Watson Search Engine [50] by IBM is called the gateway for the Semantic Web and it is an ontology-based semantic search engine that focuses on web ontologies. It searches ontologies and semantic documents by using keywords and restrictions that are related to the type of the entities. It retrieves metadata about the ontologies that characterizes each ontology such as the labels, size and comments. It also sometimes retrieves reviews of other users about the ontology. Watson can retrieve measures regarding ontologies and entities such us the depth of the hierarchy of an ontology. This allows applications to define selection criteria and filters. It also explores the content of each ontology through functions that include the possibility to ask for the subclasses of a RDF, DAMN or OWL class. Watson consists of three core activities and each one of them corresponds to a layer of its architecture. Those layers are the following:

- The ontology crawling and discovery layer: It explores ontology-based links and it collects the available semantic content they provide.
- The validation and analysis layer: It is the core of the Watson's architecture, it checks the quality and it makes sure that the data of the semantic information collected is computed, stored and indexed.
- The query and navigation layer: It allows access to the data indexed by the previous layer through various mechanisms that allow exploring its semantic features.

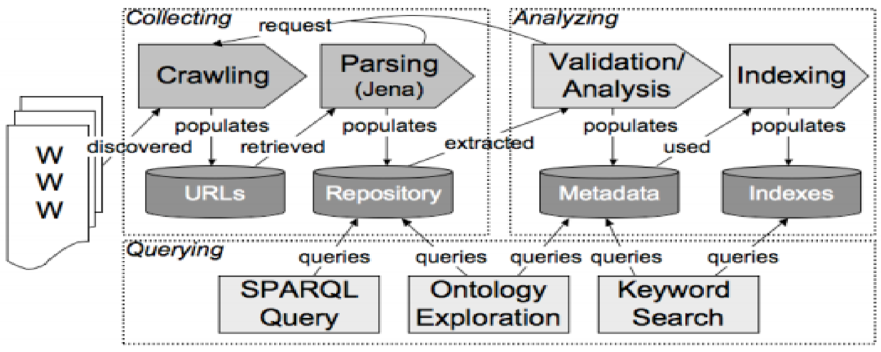


Figure 8. A functional overview of the main components of the Watson architecture. (Source: https://www.researchgate.net/figure/A-functional-overview-of-the-main-components-of-the-Watson-architecture_fig1_48989644)

2.5.3. IBRI-CASONTO

IBRI-CASONTO [51] was created as a ontology-based search engine for College of Applied Sciences(CAS), Sultanate of Oman. It supports English and Arabic and it is based on the RDF dataset and Ontological Graph. The IBRI-CASONTO system follows several steps to generate an efficient and scalable ontological graph.

- First step, the user determines the domain of the ontology.
- Second step, the user determines the language of the ontology as well as the editor.

- Third step, creation of the ontological graph.
- Fourth step, the definition of all the Classes and their sub and super-classes. Furthermore, in that step, the relationships between all classes are defined.
- Fifth step, the instances of each class are defined also called as individuals.
- Sixth step, the domain and the range of each property is defined as well as the relationships between the individuals.
- Seventh step, the data property is created, and the construct, range and domain are defined for each property.

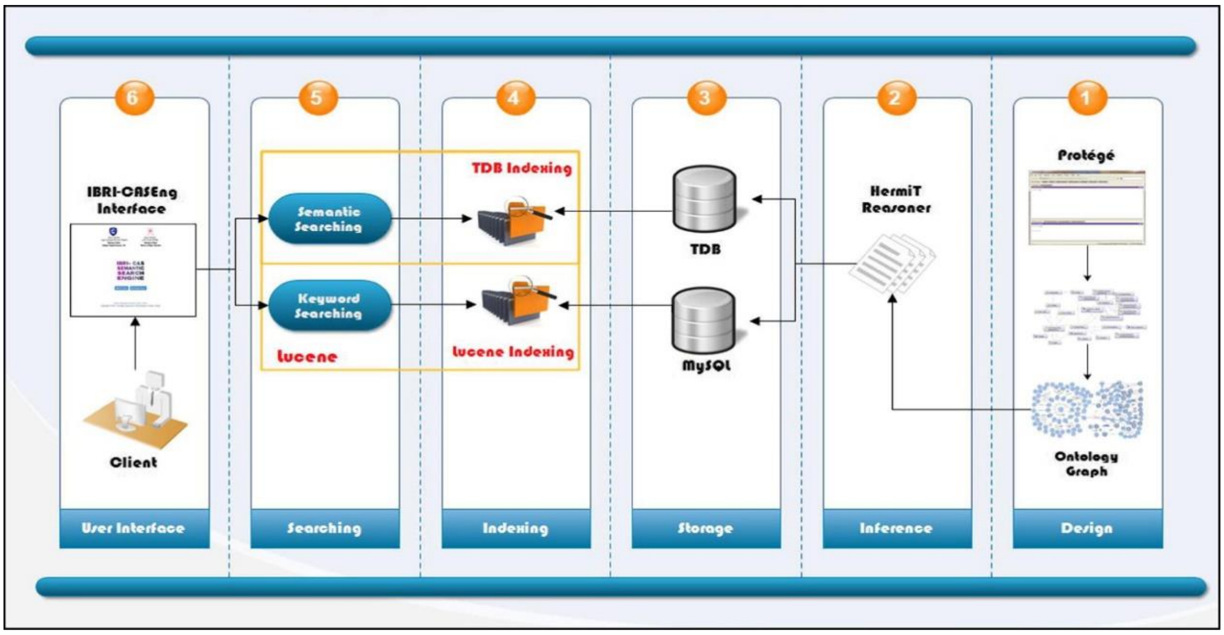


Figure 9. IBRI-CASONTO Structure.
(Source: https://www.researchgate.net/figure/IBRI-CASONTO-Structure_fig2_312517116)

2.5.4. PASS

Personalized Abstract Search Services(PASS) [52] by the IEEE Neural Network Council(NNC) is a web-based system used for searching abstracts of research papers. It uses fuzzy ontology [53] of term associations to support this feature. This system uses a user’s query and in response it displays results based on a keyword-based retrieval and it provides a set of terms lists for query refinement. Pass will also provide suggestions to the most relevant documents that are related to the user’s query and a list of related terms that are based on a collaborative filtering. This is a technique that recommends the papers that could be related to the user’s interests based on his own interests or on other users with similar interests. PASS also provides shopping cart like facilities and personalized folder features.

2.5.5. GOseek

GOseek [54] is a biological search engine. Its main function is after given a set of genes to return the most important genes that are semantically related to the genes given. The resulted genes are annotated to one of the Lowest Common Ancestors of the Gene Ontology(GO) [55] terms annotating the set of genes that are given. Afterwards it encodes the contribution of those terms into a numeric format. GOseek returns a numerical value which corresponds to the similarity of the GO terms with the given set of genes. It supports keyword-based queries with a specific form and as a result it returns a set of genes that is semantically related to the given set of genes.

2.5.6. NCBO Resource Index

Resource Index [56] is an ontology based index created by the National Center for Biomedical Ontology(NCBO). It uses 22 data resources and a set of over 200 ontologies in various domains. It also offers its users a user interface that allows them to search in multiple resources simultaneously using domain terms. The Resource Index processed metadata describing data elements to create semantic annotations of those metadata. It makes use of BioPortal [57] by using a concept recognition tool that identifies terms from BioPortal’s ontologies and finds the corresponding element with the identified terms.

BioPortal is a huge library that contains 245 biomedical ontologies and it provides access to the public. The users can browse, visualize and comment on ontologies either through a web interface or programmatically using web services. A number of its ontologies is contributed by the Open Biomedical Ontologies Foundry(OBC) [58], an effort to collect a collection of ontologies for biomedical use. Apart from OBC’s ontologies, BioPortal also uses available terms from the Unified Medical Language System(UMLS) [59]. BioPortal contains ontologies available in multiple formats such us OWL, OBO, RDF(S) and RRF.

To access BioPortal, the Resource Index uses BioPortal REST services. It creates a dictionary of terms that is meant to be used for direct annotations of data elements in biomedical resources. This dictionary contains 6,835,997 terms extracted out of 3,349,338 concepts from 206 ontologies, numbers that are still growing. Each concept is derived by either a unique URI contained in the ontologies or provided by the NCBO.

Apart from the ontology terms, another big source of information for the Resource Index is the data elements from the biomedical resources. The Resource Index currently has 22% of the publicly available biomedical resources indexed. Those biomedical resources are mostly formatted in XML [60] and most of them offer access through web services. The Resource Index used a custom wrapper to access each resources’ information which extracts the fields describing the data elements. After accessing the data elements describing each biomedical resource, the Resource Index follows 4 steps to create annotations:

I. Creation of direct annotations with ontology terms.
II. Semantic expansion of annotations.
III. Aggregation.
IV. Scoring of annotations.

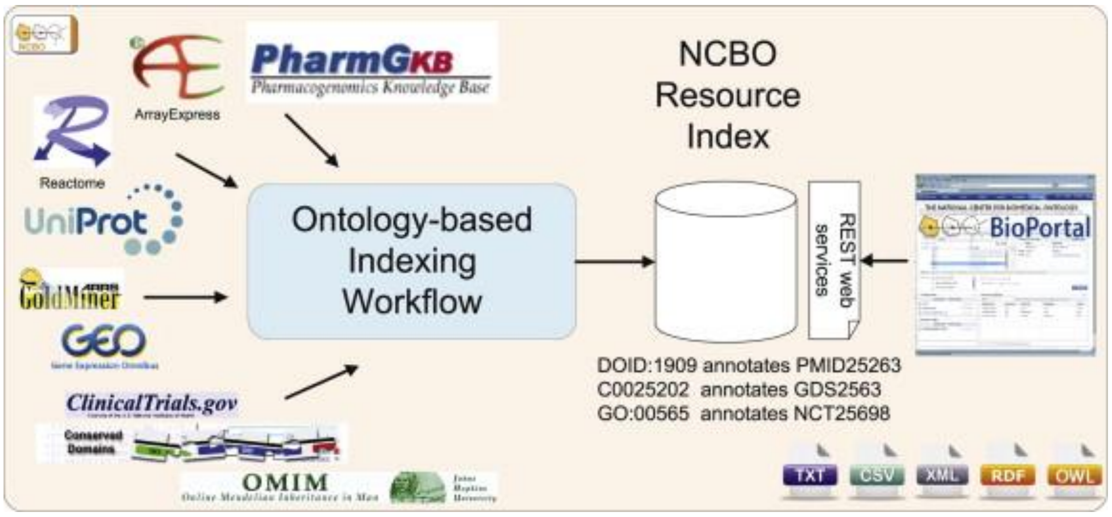

NCBO Resource Index
ArrayExpress
PharmGKB
Pharmacogenomics Knowledge Base
Reactome
UniProt
GoldMiner
GEO
ClinicalTrials.gov
Conserved Domains
OMIM
Online Mendelian Inheritance in Man
Ontology-based Indexing Workflow
REST web services
BioPortal
DOID:1909 annotates PMID25263
C0025202 annotates GDS2563
GO:00565 annotates NCT25698
TXT
CSV
XML
RDF
OWL

Figure 10. NCBO Resource Index Overview.
(Source: https://www.sciencedirect.com/science/article/pii/S1570826811000485)

2.5.7. GoWeb

GoWeb [61] is a semantic search engine that combines classic keyword-based search technology with text-mining and ontology background to achieve fast and efficient question answering. It has the ability to filter big result sets according to the Medical Subject Headings(MeSH) [62] and the GeneOntology(GO) to discard data that will not have any use for the user. Also, GoWeb's users can use GoPubMed [63] features along with the wide range of information sources that are publicly available in the web. It also offers highlighting and semi-automatic question answering mechanisms. The interface of GoWeb is organized specifically to make it easy to use for the users. It is structured in three panels. The first panel contains the background knowledge and possible derived metadata. The second panel contains the search bar where the users can structure their search queries. Finally, the third panel contains the area the results are presented after searching.

GoWeb uses a keyword based search to retrieve its results. It uses Yahoo! Search BOSS service. The results retrieved by the BOSS service is a usually big list of snippets. Afterwards, GoWeb uses its ontology background knowledge to map concepts to the snippets. It then uses GoPubMed algorithms to identify ontological concepts in the text lists. In order to identify gene names and protein, GoWeb uses the approach by Hakenberg. For names and places identification GoWeb makes use of the OpenCalais service. Then, GoWeb's filter mechanisms take place which uses the part-of and is-a relations from GO and the tree structure of MeSH. The most important concepts are selected by the occurrence frequency, the hierarchy level and, if it exists, a global frequency from a corpus that is pre-analyzed.

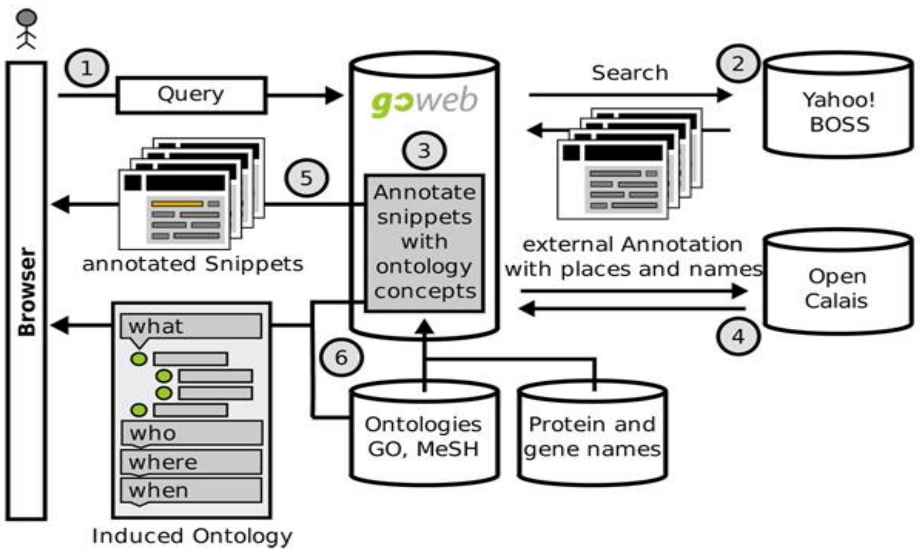


Figure 11. GoWeb workflow representation.
(Source: https://www.researchgate.net/figure/GoWeb-workflow-General-workflow-for-GoWeb-showing-the-main-components-and-the_fig1_26862850)

2.6. Apache Lucene

Before describing Apache Solr, the search platform the proposed framework makes use of, it is best to start with its core technology, Apache Lucene [64]. Lucene was developed in 2000 by Doug Cutting and it is actively been evolving since then. It is now the most used search technology with a strong community. Lucene is neither a server nor a web crawler, it is a code library. To use Lucene you have to write your own search code by using its API and you supply it with index documents. For Lucene, a document is a collection of fields. Each field consists of a name and a value which is either a text or a number. Lucene gives the user the ability to tokenize a field's string into a series of words called tokens, change words with their synonyms or perform other processes. After processing, the resulted indexed tokens are the terms. After that process, Lucene indexes every document in a disk. A mapping of a field's terms to associated documents is stored by the index together with the starting word position from the original text and then, the user can search for documents by providing a query. Lucene parses that query according to its syntax and after scoring each matching document it returns the top scoring documents.

Lucene offers many features. The most significant are:

- An inverted index that retrieves documents efficiently by using indexed terms.
- Text analysis components that turn a string into a series of words called terms.
- A query syntax with a parser and a plethora of query types.
- A flexible scoring algorithm based on sound Information Retrieval(IR) used to score matching documents.
- Many search enhancing features such as highlighting, query spellchecker, suggesters for query completion and various analysis components.

2.7. Apache Solr

Even though Apache Solr [8] is not a semantic search tool, it is one of the most popular open source enterprise search platforms. Solr is written in java and it offers search and indexing technology as well as highlighting, spellchecking and advanced tokenization capabilities. It is open source and it is integrated in the Apache Lucene project. The ideology behind Solr’s usage is simple. You feed it tons of information and afterwards you can retrieve data by asking questions. Providing Solr with information is called feeding and asking a question is called querying. Field analysis tells Solr how to deal with incoming data when building an index. Analysis is used both during indexing and searching. An analyzer generates token streams by examining the text of fields. Analyzers may consist of a number of tokenizers and filters. Tokenizers decompose field data into tokens and then filters examine those tokens and they either change them, discard them or create new ones. Even though analysis is used during indexing and querying, not the same process is used for both. During indexing, analysis often normalizes words to increase query-time precision. During querying the simple analyzer’s normalization is converting the query terms to lowercase. By using a simple <analyzer> for a field type your query must be very precise. Searching in Solr is very flexible. This process is handled by a request handler. A request handler is the logic that is used during searching. They can be customized in a variety of ways for ones needs. Solr gives a number of results after the query is handled by the request handler. The results are sorted by score and the result with the highest score is more likely to be what the user is looking for.

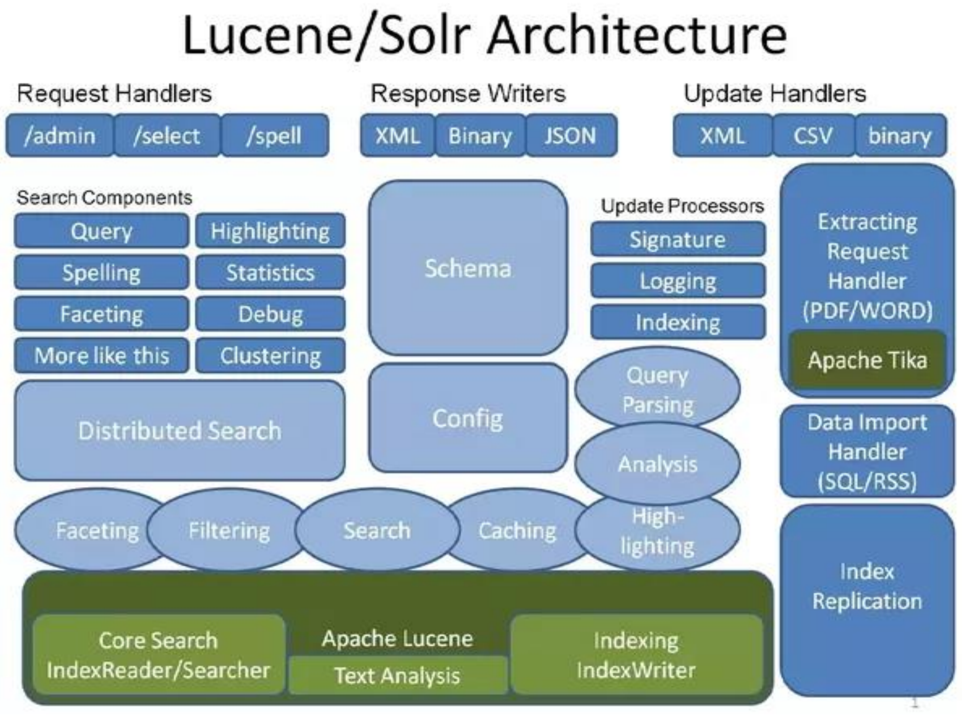


Figure 12. Apache solr architecture.
(Source: https://slideplayer.com/slide/2293397/)

2.7.1. Apache Stanbol

Apache Stanbol [65] extends traditional content management systems by adding semantic services to them. Integrating Stanbol into your system is an easy task since all its features are accessible via RESTful web services [66]. Apache Stanbol's Usage Scenarios are:

- Content Enhancement: Analyzation and enhancement of content.
- Using Custom Variables: Create and use locally defined entities.
- Using Multiple Languages: Enhancing content written in multiple languages.
- Semantic Search in Portals: Index enhancements and content items.
- Refactoring Enhancements: Refactor the property names and ontology types of the enhancement result.
- Transforming CMS [67] repository structures into ontologies.
- Provide repository structures as domain ontology.

Apache Stanbol consists of a set of components. Each of them is independent and is accessible via its own RESTful web interface. All components are implemented as OSGi bundles [68].

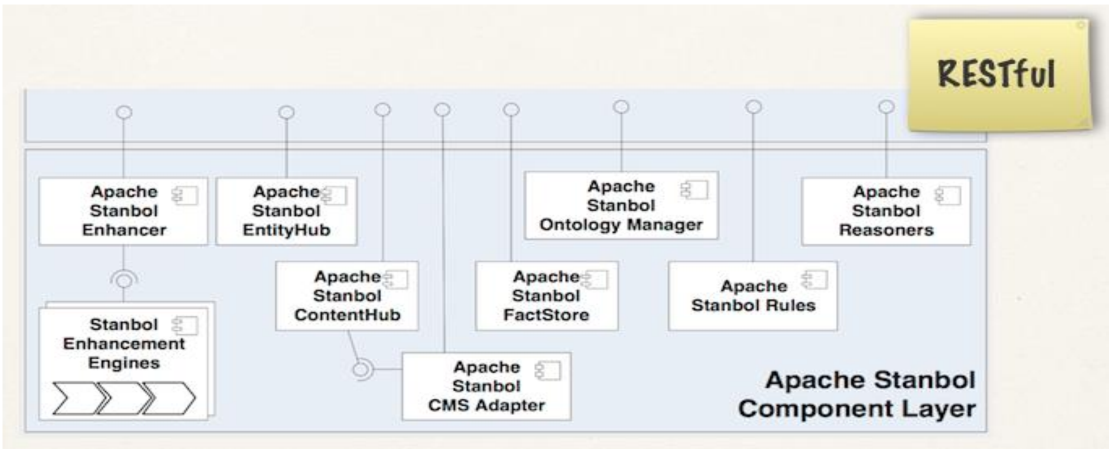

RESTful
Apache Stanbol Enhancer
Apache Stanbol EntityHub
Apache Stanbol Ontology Manager
Apache Stanbol Reasoners
Stanbol Enhancement Engines
Apache Stanbol ContentHub
Apache Stanbol FactStore
Apache Stanbol Rules
Apache Stanbol CMS Adapter
Apache Stanbol Component Layer

Figure 13. The Apache Stanbol Components.
(Source: https://stanbol.apache.org/docs/trunk/components/)

2.7.2. Redlink Solr Plugin

The Redlink Solr plugin [69] adds semantic capabilities to an existing Apache Solr build. Redlink uses Apache's Analysis API and it extracts named entities of documents during Solr updates. Integrating Redlink into an existing Solr build is simple and can be done within a few minutes.

2.7.3. Open Semantic Search

The Open Semantic Search [70] is a software that can be integrated on one's search engine. It contains research tools for easier monitoring, analytics, searching, discovery and text mining of large document sets and can be used for free on your own server. By entering a search query, you can navigate through the results found in multiple data sources. Interactive filters can be used to filter out unwanted data results. Your documents can be tagged with keywords, names, categories or text notes to make searching easier and more efficient in the future. Open Semantic Search can also be used for data visualization by using charts and graph views. It supports a large variety of formats so no matter how your documents are structured you can still use this software for searching. The supported formats are: txt, PDF, E-mail, JPG, TIFF [71], CSV [72], doc Word documents, Open documents(ODF) [73] and all video formats. Text can be automatically recognized from images or scanned documents with the Optical Character Recognition (OCR) system [74].

2.7.4. Solr and OLS

Ontology Lookup Service(OLS) [75] is a repository that contains biomedical ontologies. It aims to provide access to the latest version of every biomedical ontology. Browsing those ontologies can be done either within their website or by using the OLS API. Ontology Lookup Service uses Apache Solr in order to index ontologies for search. Two Solr cores are being used by OLS. The one is used to add an autosuggest feature and the other for storing documents for ontology and term meta-data.

3. Methods

3.1. System architecture

For the **ontology-based search** of OntoSearch, during setup, we used Apache solr as a semantic annotator to extract ontology terms for each tool from the biotools registry. The results were saved in a database along with a score provided by Apache solr for each extracted term.

During runtime, the user can use either text or speech as input that is translated to text by the framework. The framework then uses Apache solr to extract ontology terms for the user’s query from the same ontologies that were used during setup. With a matching algorithm and by using the data saved in the database during setup, the framework finds the best possible results for the user’s query and returns them to the user.

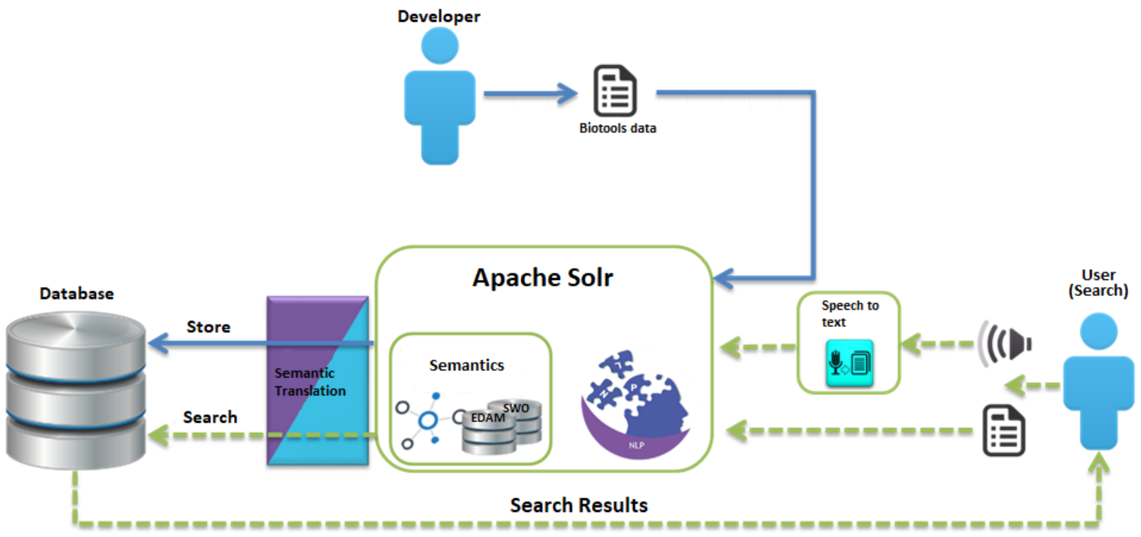


Figure 14. OntoSearch’s ontology-based search Architecture.

For the **keyword-based search** of OntoSearch, during setup, we fed Apache solr with all the tools from the biotools registry.

During runtime, the user can use either text or speech as input that is translated to text by the framework. The framework then takes advantage of Apache solr and with the query given by the user finds the best possible results for the user’s query and returns them to him.

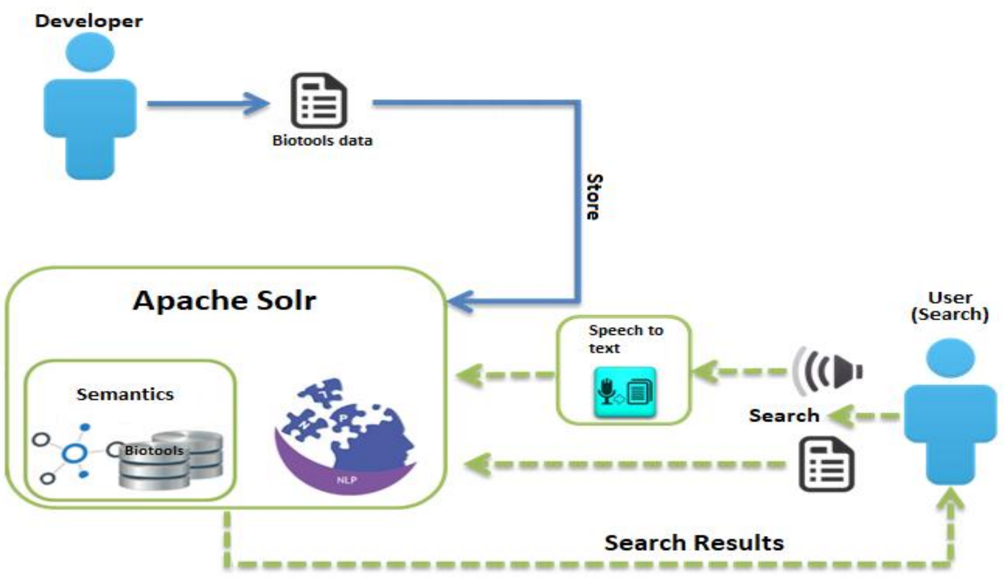


Figure 15. OntoSearch's keyword-based search architecture.

For the Hybrid search of Ontosearch, during setup, we fed Apache solr with the EDAM and the Software ontologies. Then, we used all tools from the biotools registry as queries and retrieved the 5 terms with the highest score assigned by solr from each ontology for each tool. We then created a new JSON file ,added the biotools registry in it and then added the top 5 terms from each ontology for each tool as free text in new fields. Lastly, we created a new Apache solr core and fed it with the new JSON file that we created.

During runtime, the user can use either text or speech as input that is translated to text by the framework. The framework then takes advantage of Apache solr and the core that was created during setup for the hybrid search. Then, with the query given by the user, it finds the best possible results for the user’s query and returns them to him.

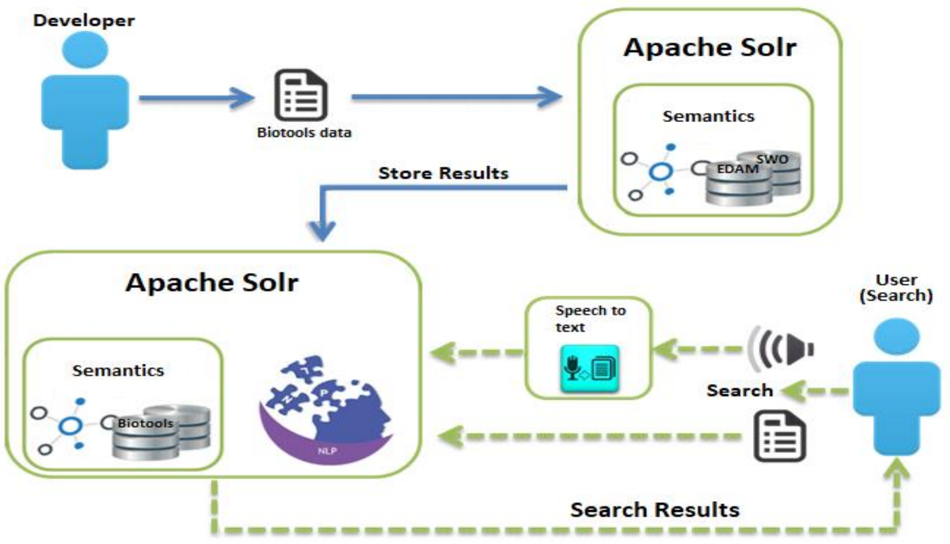


Figure 16. Ontosearch's hybrid search architecture.

Ontosearch, combines the three searching techniques and retrieves the best possible results from each one to the user.

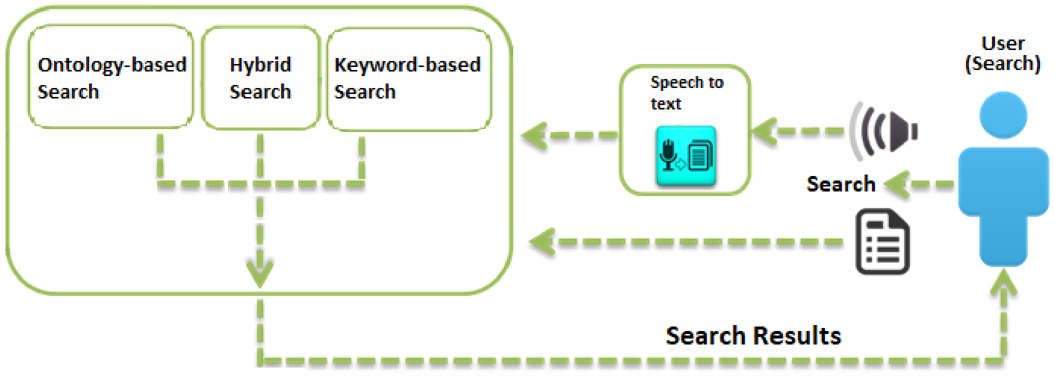


Figure 17. OntoSearch Architecture.

3.2. Speech to text solution

Speech to text is a process that consists of two main parts. Recognizing speech and transforming it into signal and turning that signal into text. This concept has concerned the biggest companies over the years. After comparing many speech-to-text APIs, the API that was used for the proposed framework was Web Speech API [76].

The Web Speech API accesses Google's speech recognition web service to transform speech into text. It is designed purely in JavaScript and it provides speech analysis and speech synthesis. The API is still in experimental phase and the reason for that is that its introduction has remained mostly speculative. A significant feature of the API is that the actual speech recognition happens through a web service. The developer doesn't directly interact with it but communicates with the user agent(browser) through certain events. It is the browser's responsibility to communicate with the web service. The Web Speech API is developed by the World Wide Web Consortium (W3C) Community and the initiative is mostly driven by Google and Openstream.

Out of the six most popular browsers, Google Chrome, Mozilla Firefox, Microsoft Internet Explorer, Microsoft Edge, Safari and Opera [77], only Google Chrome (version 25+) currently supports the Web Speech API. That's mostly because of the API's experimental nature but it is expected that more browsers will support the API in the near future.

The Web Speech API allows users to record audio from their microphone, after their permission is granted. That audio is sent through a HTTPS POST request to the web service and the results are processed within the limits of the browser.

The chosen API is used to form the users' questions in the search bar while the users can see their speech transformed into text in real time to identify any possible errors made by the API. Given the sound waves from the users, the API will encode those waves into wave files and finally into FLAC files. The proposed framework will retrieve responses from the API and the final text will be formed ready to be used and translated into a query by the NLP [78].

For the evaluation, 2 children, 12 females and 11 males spoke several sentences to evaluate the API. On the table below, the speakers are identified by a two-letter combination with the first letter indicating whether the speaker is a male (M), female (F) or child (C). the second column indicates the number of the sentences spoken by each speaker. The third column shows the percentages of the sentences that were recognized correctly. The fourth column shows the total number of words spoken in all sentences for each speaker. Finally, the last column provides the percentages of the words that were recognized correctly.

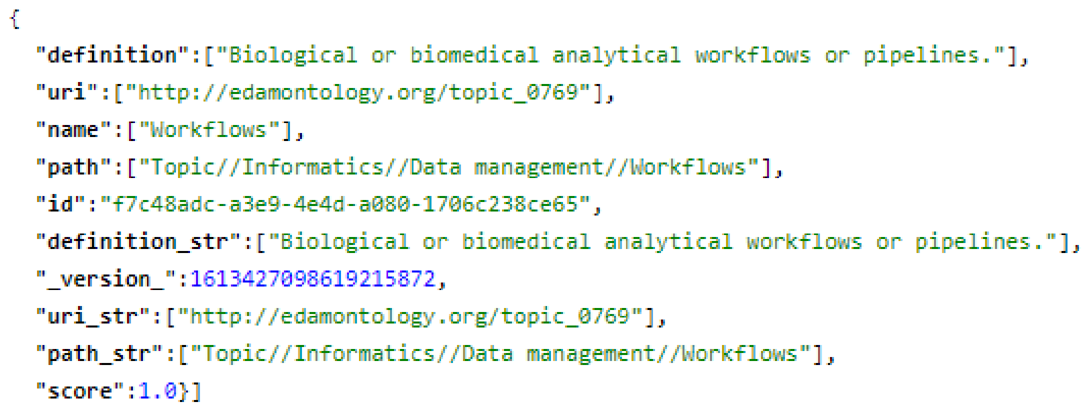
Table 1. Results of the Web Speech API evaluation grouped by the 24 speakers. (Source: http://www.juliusadorf.com/pub/web-speech-api.pdf)

Speaker	Sentences	Correct	Words	Word Accuracy
CA	60	33%	466	83%
CB	6	17%	49	75%
FA	60	23%	466	77%
FB	60	8%	467	65%
FC	60	20%	475	74%
FD	60	18%	478	74%
FE	60	24%	461	76%
FF	60	23%	484	77%
FG	60	17%	456	62%
FH	60	15%	481	61%
FI	60	20%	495	72%
FJ	60	35%	497	84%
FK	60	15%	503	65%
FL	60	27%	480	73%
MA	60	22%	466	72%
MB	60	28%	467	78%
MC	60	15%	475	71%
MD	60	37%	478	84%
MF	60	13%	484	68%
MG	60	22%	456	75%
MH	60	25%	481	77%
MI	60	23%	495	74%
MJ	60	10%	497	62%
MK	60	12%	503	72%
ML	60	23%	480	76%

3.3. Concept recognizer (ontology terms) from NLP

The concept recognizer is the core of the system. It is used by most of the components in the framework. The main objective of the concept recognizer is to retrieve the most suitable result for a given query created by the user in free text form. For all three searching techniques of the framework, ontology based, keyword based and hybrid, Apache solr full text search server is used but in a different fashion and for a different purpose.

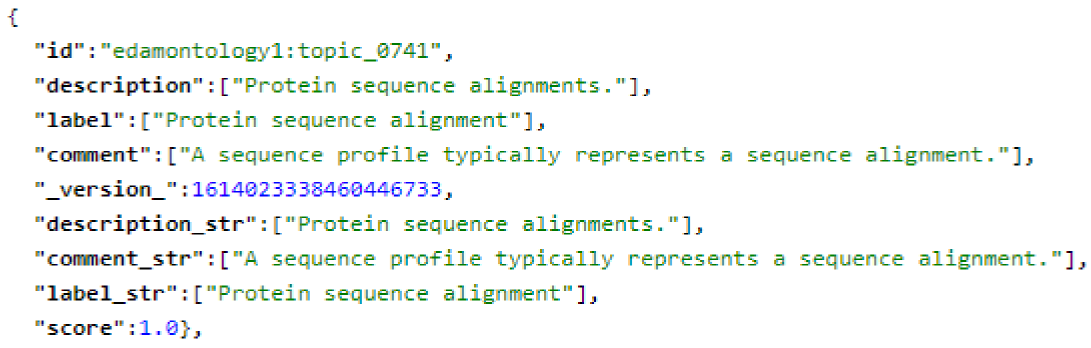
- For the **ontology based search**, two ontologies were used, Embrace Data and Methods(EDAM) and the Software Ontology(SWO). The two ontologies are used separately and have different results but use the concept recognizer in the same way. For each ontology, a JSON-format [79] file was created with specific fields that can be weighed differently during search. Those JSON files were imported into separate cores in Apache solr. The two ontologies have different fields and have their own custom made weight formula.
  - For **EDAM**, the weight formula that was used is biased to the uri, the name and the definition of each term. That means that if the searched text matches with any of those fields, then the ontology term is assigned a score which is directly bound to the weights assigned to the fields. The custom made weight for the EDAM ontology is : ***(uri * 1) + (name * 2) + (definition * 5)***.



```
{
  "definition":["Biological or biomedical analytical workflows or pipelines."],
  "uri":["http://edamontology.org/topic_0769"],
  "name":["Workflows"],
  "path":["Topic//Informatics//Data management//Workflows"],
  "id":"f7c48adc-a3e9-4e4d-a080-1706c238ce65",
  "definition_str":["Biological or biomedical analytical workflows or pipelines."],
  "_version_":1613427098619215872,
  "uri_str":["http://edamontology.org/topic_0769"],
  "path_str":["Topic//Informatics//Data management//Workflows"],
  "score":1.0}]
```

Figure 18. Example of a random EDAM term retrieved from Apache solr.

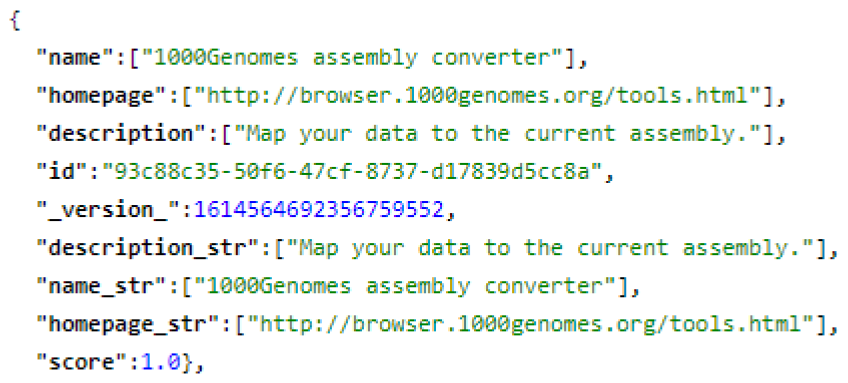
  - For **SWO**, the weight formula that was used is biased to the id, the description, the label and the comment of each term. That means that if the searched text matches with any of those fields, then the ontology term is assigned a score which is directly bound to the weights assigned to the fields. The custom made weight for SWO is : ***(id * 1) + (description * 10) + (label * 4) + (comment *4)***.



```
{
  "id":"edamontology1:topic_0741",
  "description":["Protein sequence alignments."],
  "label":["Protein sequence alignment"],
  "comment":["A sequence profile typically represents a sequence alignment."],
  "_version_":1614023338460446733,
  "description_str":["Protein sequence alignments."],
  "comment_str":["A sequence profile typically represents a sequence alignment."],
  "label_str":["Protein sequence alignment"],
  "score":1.0},
```

Figure 19. Example of a random SWO term retrieved from Apache solr.

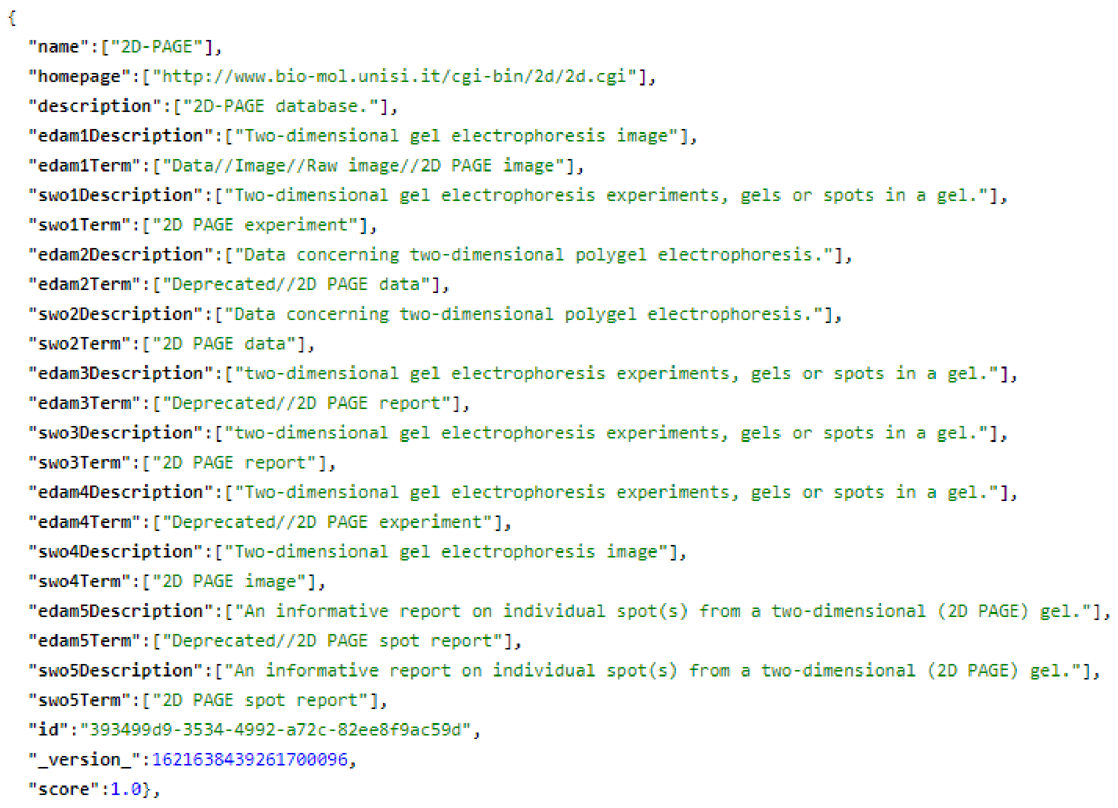
- For the **keyword based search** one Apache solr core is used. A JSON-format file was created that contains the ids, homepages, names and descriptions of all the tools from the biotools registry. This JSON file is imported in Apache solr and during search the fields are weighed differently. Each tool that was matched through either their description or their name or both, is assigned with a score from Apache solr that is directly bound with the weights of the fields. After running the search query, the tool that was assigned with the highest score represents the most suitable tool in the biotools registry for the user's needs. The custom made weight for the keyword based search is : ***(name * 1) + (description * 2)***.



```
{
  "name":["1000Genomes assembly converter"],
  "homepage":["http://browser.1000genomes.org/tools.html"],
  "description":["Map your data to the current assembly."],
  "id":"93c88c35-50f6-47cf-8737-d17839d5cc8a",
  "_version_":1614564692356759552,
  "description_str":["Map your data to the current assembly."],
  "name_str":["1000Genomes assembly converter"],
  "homepage_str":["http://browser.1000genomes.org/tools.html"],
  "score":1.0},
```

Figure 20. Example of a random tool retrieved from Apache solr.

- For the **hybrid search** Apache solr was fed with a JSON file that contained the EDAM ontology and the Software ontology. Then, all descriptions from the tools in the biotools registry were used as queries. The 5 terms with the highest score assigned by solr were saved for each tool. We then created a new JSON file that contained the ids, homepages, names and the 5 EDAM and SWO ontology terms retrieved previously for all the tools from the biotools registry. This JSON file was imported in Apache solr in a new core and during search the fields are weighed differently. Each tool that was matched through any of its fields, is assigned with a score from Apache solr that is directly bound with the field weights. After running the search query, the tool that was assigned with the highest score represents the most suitable tool in the biotools registry for the user's needs. The custom made weight for the hybrid search is: ***(name * 25) + (description * 50) + (edam1Description * 10) + (edam2Description * 10) + (edam3Description * 10) + (edam4Description * 10) + (edam5Description * 10) + (swo1Description * 10) + (swo2Description * 10) + (swo3Description * 10) + (swo4Description * 10) + (swo5Description * 10) + (edam1Term * 7) + (edam2Term * 7) + (edam3Term * 7) + (edam4Term * 7) + (edam5Term * 7) + (swo1Term * 7) + (swo2Term * 7) + (swo3Term * 7) + (swo4Term * 7) + (swo5Term * 7).***



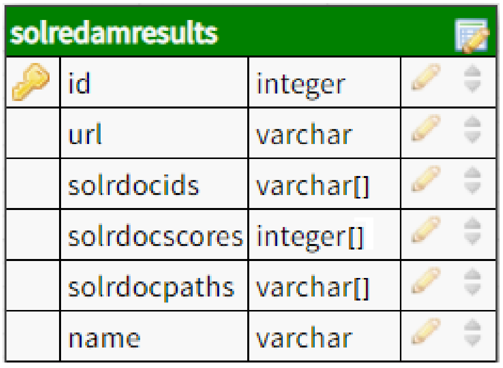
```
{
  "name":["2D-PAGE"],
  "homepage":["http://www.bio-mol.unisi.it/cgi-bin/2d/2d.cgi"],
  "description":["2D-PAGE database."],
  "edam1Description":["Two-dimensional gel electrophoresis image"],
  "edam1Term":["Data//Image//Raw image//2D PAGE image"],
  "swo1Description":["Two-dimensional gel electrophoresis experiments, gels or spots in a gel."],
  "swo1Term":["2D PAGE experiment"],
  "edam2Description":["Data concerning two-dimensional polygel electrophoresis."],
  "edam2Term":["Deprecated//2D PAGE data"],
  "swo2Description":["Data concerning two-dimensional polygel electrophoresis."],
  "swo2Term":["2D PAGE data"],
  "edam3Description":["two-dimensional gel electrophoresis experiments, gels or spots in a gel."],
  "edam3Term":["Deprecated//2D PAGE report"],
  "swo3Description":["two-dimensional gel electrophoresis experiments, gels or spots in a gel."],
  "swo3Term":["2D PAGE report"],
  "edam4Description":["Two-dimensional gel electrophoresis experiments, gels or spots in a gel."],
  "edam4Term":["Deprecated//2D PAGE experiment"],
  "swo4Description":["Two-dimensional gel electrophoresis image"],
  "swo4Term":["2D PAGE image"],
  "edam5Description":["An informative report on individual spot(s) from a two-dimensional (2D PAGE) gel."],
  "edam5Term":["Deprecated//2D PAGE spot report"],
  "swo5Description":["An informative report on individual spot(s) from a two-dimensional (2D PAGE) gel."],
  "swo5Term":["2D PAGE spot report"],
  "id":"393499d9-3534-4992-a72c-82ee8f9ac59d",
  "_version_":1621638439261700096,
  "score":1.0},
```

Figure 21. Example of a random tool retrieved from apache solr from the hybrid core.

3.4. Data storage and retrieval

The data storage and retrieval phase is needed only for the **ontology based search**. After importing the two ontologies, EDAM and SWO, in Apache solr, a database table is created for each ontology. All data for each tool from the biotools registry exists in a JSON formatted file. The data that is needed for our framework is each tool's id, name, homepage and description. All descriptions from all the tools were used as queries in Apache solr. The two database tables created contain the results from those query runs. Apache solr has two cores created for the ontology based search, the one has the EDAM ontology imported and the other has the SWO ontology imported.

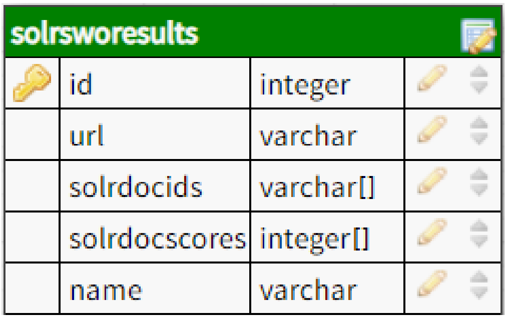
- For the core that has the **EDAM ontology** imported, the descriptions of all the tools were used as queries. The ids of the terms that were retrieved from that query execution, as well as their paths and scores assigned by Apache solr, were saved in the solrresultsedam table. Each table record contains the id, url and name of the tool as well as the ids, scores and paths of each term that was a result from the query execution in Apache solr with a maximum number of 500 terms.



solredamresults		
🔑	id	integer
	url	varchar
	solrdocids	varchar[]
	solrdocscores	integer[]
	solrdocpaths	varchar[]
	name	varchar

Figure 22. Database table used to store and retrieve solr results using EDAM.

- For the core that has the **Software ontology (SWO)** imported, the descriptions of all the tools were used as queries. The ids of the terms that were retrieved from that query execution, as well as their scores assigned by Apache solr, were saved in the solrresultsswo table. Each table record, contains the id, url and name of the tool as well as the ids, scores and paths of each term that was a result from the query execution in Apache solr with a maximum number of 500 terms.



solrsworesults		
🔑	id	integer
	url	varchar
	solrdocids	varchar[]
	solrdocscores	integer[]
	name	varchar

Figure 23. Database table used to store and retrieve solr results using SWO.

The data from the two tables are used during search, and with a matching algorithm the most suitable tools for the user's needs are found during run time.

3.5. Query Mechanism

The query mechanism works differently for the keyword based search, the ontology based searches and the hybrid search. The moment the user runs a search, the first step is to clear the query from any possibly 'harmful' characters. Apache solr uses some characters as special characters for a more complex query run. The list of Apache solr's special characters is: (\, +, -, &&, ||, !, (,), [,], {, }, ", ^, ~, *, ?, :, /, AND, OR, NOT). To prevent Apache solr from using those characters or words in a special way, the character '\' is added before each one of them before the query run. After that process, the query is ready to run.

- For the **ontology based** searches, the query runs in both Apache solr cores that have EDAM and SWO ontologies imported respectively. For each ontology, Apache solr returns a maximum of 22 ontology terms that matched with the user’s query with each one having a score that represents how well each term matches with the query. Then, with a matching algorithm and with all the data of the terms and the scores that are matched with each tool from the biotools registry that are stored in our database, for each tool, the average score of the terms that match with both the tool and the user’s query is saved. The tools that collect the highest scores from that process are the result for each ontology respectively.

- For the **keyword based** search, the query runs in the Apache solr core that has all the biotools imported in. The results that are retrieved are ordered descendingly by the score that Apache solr assigned them with. The higher the score, the most matching is the tool for the user’s needs. The result the user receives from the keyword based search is the top scoring tools retrieved by the solr search.
- For the **hybrid** search, the query runs in the Apache solr core that has all the biotools imported in and the top 5 terms from the EDAM ontology for each tool. The results that are retrieved are ordered descendingly by the score that Apache solr assigned them with. The higher the score, the most matching is the tool for the user’s needs. The result the user receives from the hybrid search is the top scoring tools retrieved by the solr search.

3.6. Implementation of the framework

The developed framework, named OntoSearch, gives you the option to either type or record the search query. After having the search query, the user can press the search button to retrieve the results from OntoSearch. The framework will run the query and if found, retrieve the most suitable tools for the user’s query, categorized by the results retrieved from each search technique.

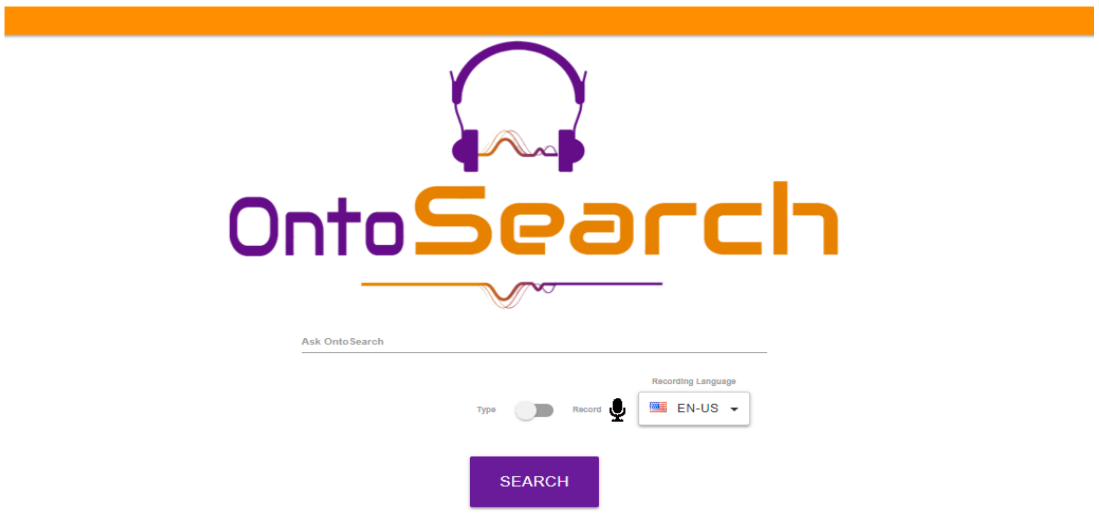
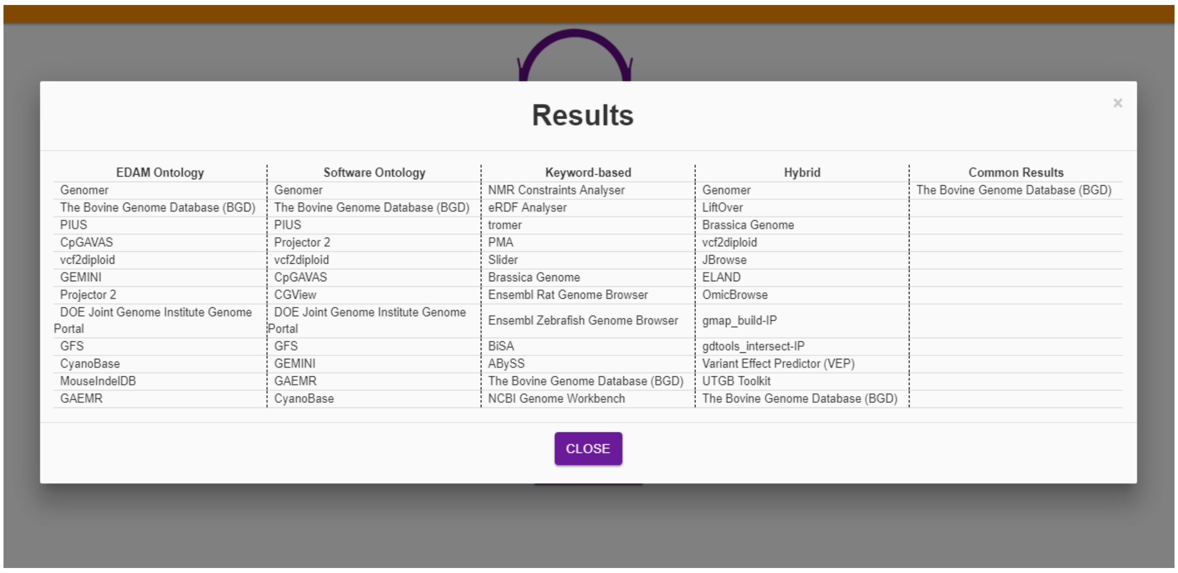


Figure 24. OntoSearch interface. (Screenshot taken from the developed framework)

After running an example query search using the query 'Genome sequence analyser', OntoSearch retrieved a list with the most suitable tools for that purpose that exist in the biotools registry. The first column indicates the results that were retrieved from the EDAM ontology-based search. The second column indicates the results that were retrieved from the SWO ontology-based search. The third column indicates the results that were retrieved from the keyword-based search and the fourth column indicates the results that were retrieved from the hybrid search. All four columns are ordered descendingly with the first tool from each column being the most suitable for the user according to each searching technique. The fifth column contains all the tools that all four searches retrieved as a result and it is ordered descendingly by the average position each tool was ranked in all four searches. Each result is a link for each tool's web page.



EDAM Ontology	Software Ontology	Keyword-based	Hybrid	Common Results
Genomer	Genomer	NMR Constraints Analyser	Genomer	The Bovine Genome Database (BGD)
The Bovine Genome Database (BGD)	The Bovine Genome Database (BGD)	eRDF Analyser	LiftOver	
PIUS	PIUS	tromer	Brassica Genome	
CpGAVAS	Projector 2	PMA	vcf2diploid	
vcf2diploid	vcf2diploid	Slider	JBrowse	
GEMINI	CpGAVAS	Brassica Genome	ELAND	
Projector 2	CGView	Ensembl Rat Genome Browser	OmicBrowse	
DOE Joint Genome Institute Genome Portal	DOE Joint Genome Institute Genome Portal	Ensembl Zebrafish Genome Browser	gmap_build-IP	
GFS	GFS	BiSA	gdtools_intersect-IP	
CyanoBase	GEMINI	ABySS	Variant Effect Predictor (VEP)	
MouseIndelDB	GAEMR	The Bovine Genome Database (BGD)	UTGB Toolkit	
GAEMR	CyanoBase	NCBI Genome Workbench	The Bovine Genome Database (BGD)	

Figure 25. OntoSearch results modal. (Screenshot taken from the developed framework)

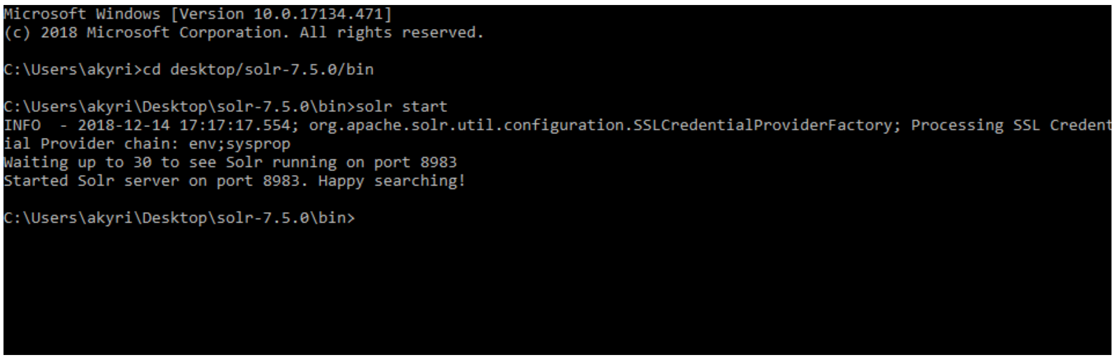
4. Deployment

4.1. Setup

For the construction and compilation of this framework, NetBeans IDE 8.2 [80] was used and it was compiled using jdk1.8.0_191. JavaServer Pages(JSP) [81] technology was used for its creation as a web application which runs on Apache Tomcat [82] server v8.0.27.0 and takes advantage of the Apache Maven software tool [83]. The server-side part, runs on the Java [84] language while client-side is based on web languages: JavaScript [85], PHP [86], CSS [87] and HTML [88]. The SQL [89] language was used to connect with the database, retrieve and save data. PostgreSQL [90] v9.5 was used through the XAMPP [91] control panel v3.2.2 and pgAdmin3 software v1.22.2 was used to design the tables. The framework, during search, takes advantage of Apache solr [8] v7.5.0.

4.2. Installation Guide

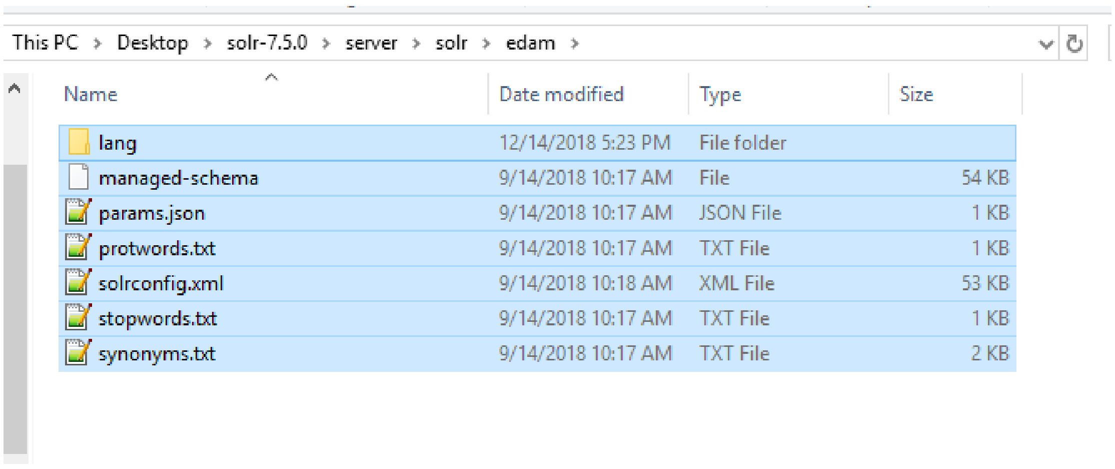
Firstly, download Apache solr v7.5.0 binary release and unzip it. Then through the command prompt, navigate inside solr-7.5.0/bin folder as shown in the picture below by using your own path to that folder. After that, write the command 'solr start' and Apache solr should start on port 8983.



```
Microsoft Windows [Version 10.0.17134.471]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\akyri>cd desktop/solr-7.5.0/bin

C:\Users\akyri\Desktop\solr-7.5.0\bin>solr start
INFO  - 2018-12-14 17:17:17.554; org.apache.solr.util.configuration.SSLCredentialProviderFactory; Processing SSL Credent
ial Provider chain: env;sysprop
Waiting up to 30 to see Solr running on port 8983
Started Solr server on port 8983. Happy searching!

C:\Users\akyri\Desktop\solr-7.5.0\bin>
```

Figure 26. Starting apache solr from cmd.

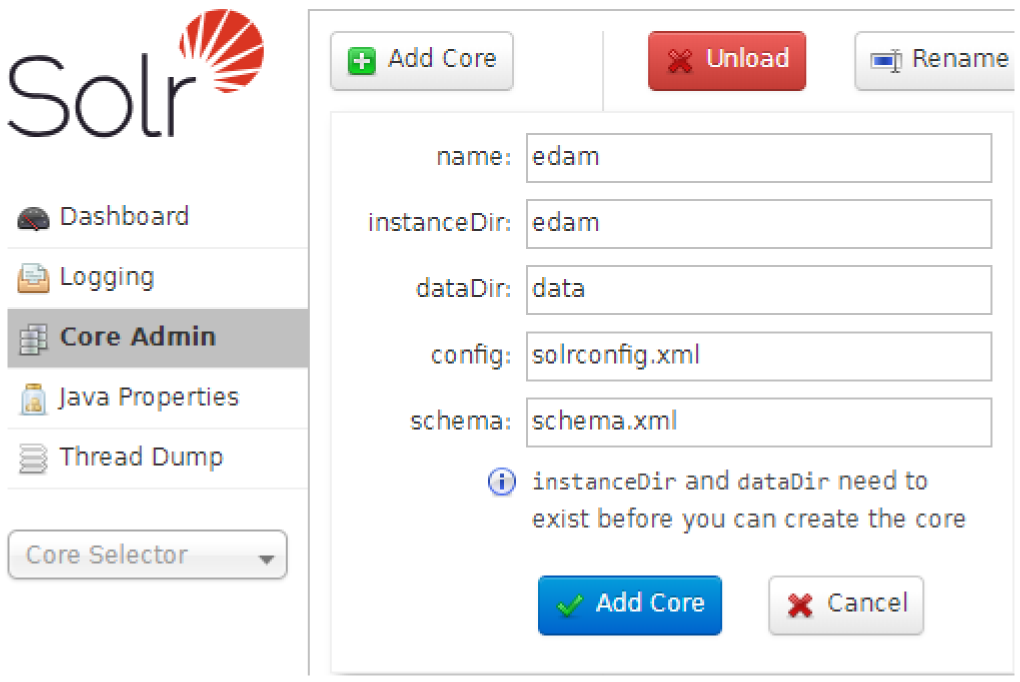
Then, create four folders named edam, swo, biotools and hybrid under solr-7.5.0/server/solr directory. You will need to copy and paste all the files and folders that exist under solr-7.5.0\server\solr\configsets_default\conf directory to all four folders created previously as shown to the example figure below. Do that for edam, swo, biotools and hybrid folders.


This PC > Desktop > solr-7.5.0 > server > solr > edam >

Name	Date modified	Type	Size
lang	12/14/2018 5:23 PM	File folder	
managed-schema	9/14/2018 10:17 AM	File	54 KB
params.json	9/14/2018 10:17 AM	JSON File	1 KB
protwords.txt	9/14/2018 10:17 AM	TXT File	1 KB
solrconfig.xml	9/14/2018 10:18 AM	XML File	53 KB
stopwords.txt	9/14/2018 10:17 AM	TXT File	1 KB
synonyms.txt	9/14/2018 10:17 AM	TXT File	2 KB

Figure 27. Example of the EDAM folder.

After completing that step, navigate to localhost:8983 through a browser. You will need to create four new cores with the names and instanceDirs edam, swo, biotools and hybrid respectively as shown in the example figure below.

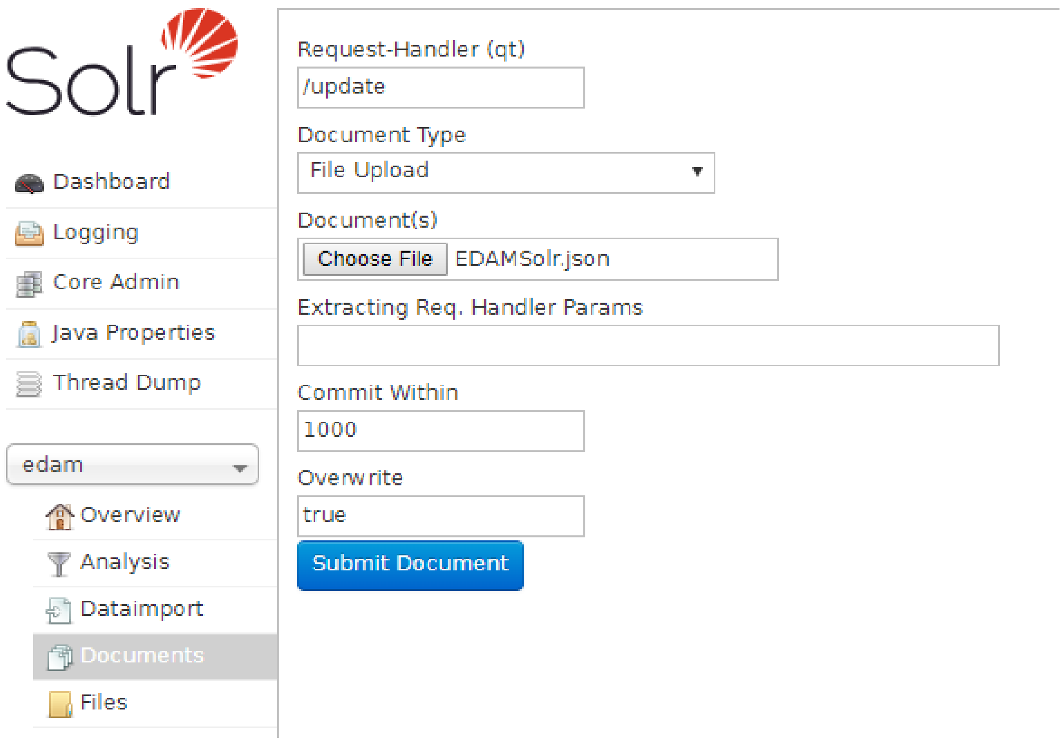


name:	edam
instanceDir:	edam
dataDir:	data
config:	solrconfig.xml
schema:	schema.xml

instanceDir and dataDir need to exist before you can create the core

Figure 28. Example of creating EDAM core in apache solr.

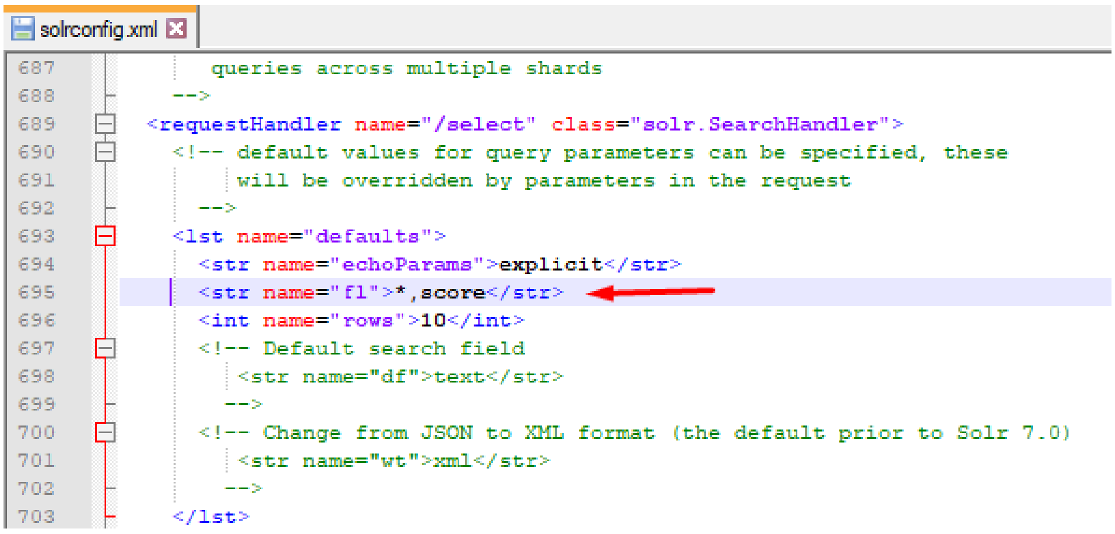
For all three cores, you will need to upload the given json files, EDAMSolr.json, swoSolr.json, biotoolsSolr.json and hybridSolr.json for each core respectively as shown in the example figure.



Request-Handler (qt): /update
Document Type: File Upload
Document(s): EDAMSolr.json
Extracting Req. Handler Params:
Commit Within: 1000
Overwrite: true

Figure 29. figure of uploading the edam json file in apache solr's EDAM core.

The last step to finish with Apache solr configuration, is to open solrconfig.xml file for all four cores that is located under solr-7.5.0\server\solr\edam, solr-7.5.0\server\solr\swo, solr-7.5.0\server\solr\biotools and solr-7.5.0\server\solr\hybrid respectively. You will need to add the line '<str name="fl">*,score</str>' in the /select requestHandler as shown in the figure below.



```
      queries across multiple shards
    -->
  <requestHandler name="/select" class="solr.SearchHandler">
    <!-- default values for query parameters can be specified, these
         will be overridden by parameters in the request
      -->
    <lst name="defaults">
      <str name="echoParams">explicit</str>
      <str name="fl">*,score</str>
      <int name="rows">10</int>
      <!-- Default search field
         <str name="df">text</str>
        -->
      <!-- Change from JSON to XML format (the default prior to Solr 7.0)
         <str name="wt">xml</str>
        -->
    </lst>
```

Figure 30. Apache solr cores' configuration.

After completing those steps, Apache solr is set and ready. The next step is to download and install pgAdmin software. After installing it, create a new database named ontosearch and run the sql script given in the file OntoSearchDB.sql. After running that script, you should have two tables created in the ontosearch database with the names solrresultsedam and solrresultsswo like shown in the figure below.

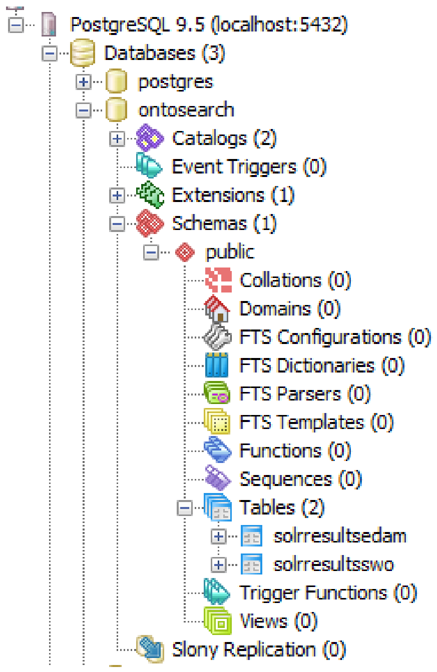
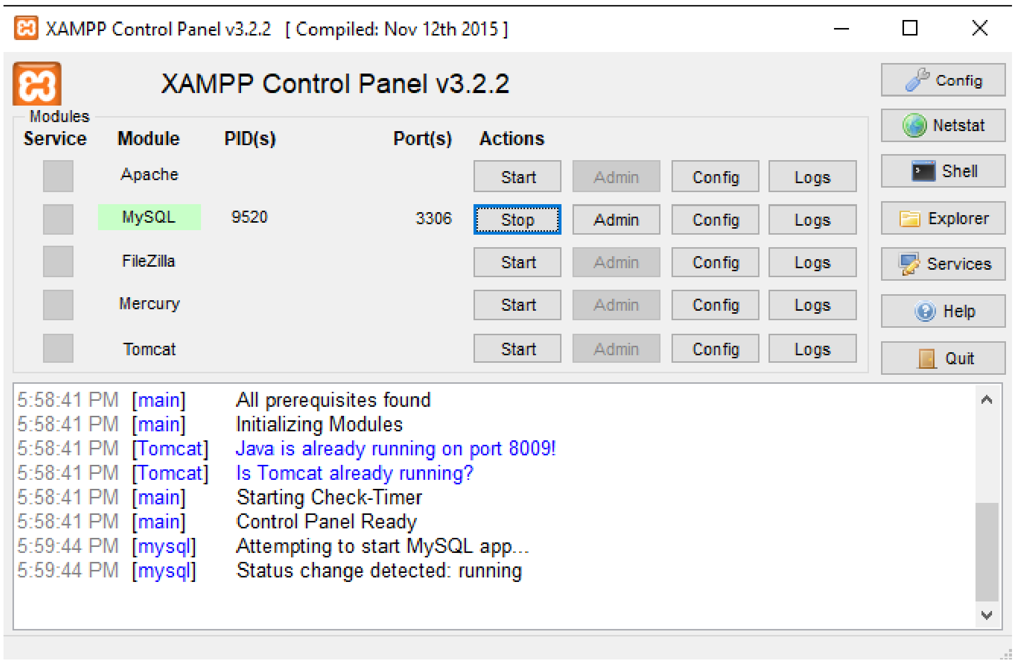


Figure 31. PGadmin3 ontosearch database.

After the database is set, download and install XAMPP Control Panel. Download MySQL service through XAMPP, if you haven't downloaded it already during installation, and start it as shown in the figure.

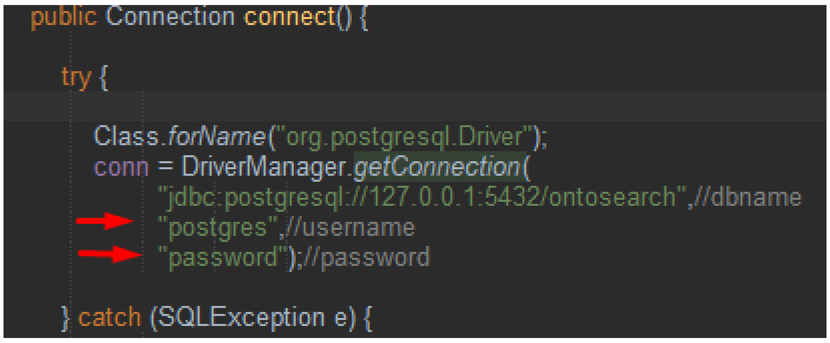

XAMPP Control Panel v3.2.2 [Compiled: Nov 12th 2015]
XAMPP Control Panel v3.2.2
Config
Netstat
Shell
Explorer
Services
Help
Quit
Modules

Service	Module	PID(s)	Port(s)	Actions			
	Apache			Start	Admin	Config	Logs
	MySQL	9520	3306	Stop	Admin	Config	Logs
	FileZilla			Start	Admin	Config	Logs
	Mercury			Start	Admin	Config	Logs
	Tomcat			Start	Admin	Config	Logs

```
5:58:41 PM  [main]     All prerequisites found
5:58:41 PM  [main]     Initializing Modules
5:58:41 PM  [Tomcat]   Java is already running on port 8009!
5:58:41 PM  [Tomcat]   Is Tomcat already running?
5:58:41 PM  [main]     Starting Check-Timer
5:58:41 PM  [main]     Control Panel Ready
5:59:44 PM  [mysql]    Attempting to start MySQL app...
5:59:44 PM  [mysql]    Status change detected: running
```

Figure 32. XAMPP Control Panel.

Next step, is to download and install NetBeans IDE v8.2 and open the given NetBeans project 'OntoSearch.zip'. In the DBController class, you will need to change the username and password to your own database username and password as shown in the figure below.



```
public Connection connect() {

    try {

        Class.forName("org.postgresql.Driver");
        conn = DriverManager.getConnection(
                "jdbc:postgresql://127.0.0.1:5432/ontosearch",//dbname
                "postgres",//username
                "password");//password

    } catch (SQLException e) {
```

Figure 33. OntoSearch DBController class.

The last step, is to build and run the project under Tomcat server in NetBeans. The web application should start under http://localhost:8084/OntoSearch/ .

5. Experiment (Storing and searching tools from the biotools registry)

5.1. Biotools

Biotools [9] is supported by the European Research Infrastructure for life science information(ELIXIR). It is a registry that contains tools that associate with bioinformatics and it is meant to aid bioinformaticians and scientists find, understand, and select resources and also use and connect them in workflows. Each tool has well-defined data processing functions. Biotools contain software applications available either for immediate use as online services or in a form an individual can download, install, configure and run. Biotools registry is based on 6 main objectives:

I. Build and maintain a complete registry of high quality software descriptions/metadata.
II. Provide a web portal enabling registration, editing, search and discovery for the registry content.
III. Support a community for the continuous maintenance of the registry content and development of the web portal features.
IV. Expose results of tool performance benchmarking, online service monitoring and other metrics of software and service quality.
V. Integrate the registry with common workbench environments in order to improve resource interoperability.
VI. Support registry stakeholders including tool providers and end-users.

In an attempt to highlight the domain independent nature of our solution we will evaluate the system using tools from the biotools repository as resources to be searched and the EDAM and Software Ontologies in comparison to a keyword based and a hybrid search.

5.2. Embrace Data and Methods (EDAM) Ontology

The EDAM ontology [6] has been actively developed since 2009 releasing multiple versions per year, resulting in the current version 1.21. The EDAM ontology was developed to provide a comprehensive means of classifying bioinformatics operations, types of identifiers and data, data formats and topics suitable for large scale semantic annotations. EDAM Ontology's main design principles are relevance to its target applications, simple usability for the users and efficient maintainability for its developers.

To ensure the first principle, relevance, EDAM has to fully cover the concepts of common bioinformatics. In order to achieve this, an ontology named myGrid [92] that had a similar purpose had to be used as a starting point. A number of resources were analyzed and used as sources of concepts. Those resources included many collections of tools, as well as the EMBOSS [93] suite and the BioMoby Service Ontology [94]. The Nucleic Acids Research's database [95] and Web server catalogues were used as sources of topics.

To ensure convenient usability by humans, EDAM ontology does not have excessively broad or deep branches. The ontology is oriented around a small number of sub ontologies with each one having a specific meaning. The main sub ontologies rooted in the top level of EDAM's hierarchy are: Operation, Data, Topic and Format.

Lastly, to keep EDAM ontology maintainable, the software development methods that are used are agile and dynamic. That way, changes can be delivered with good response time using limited resources.

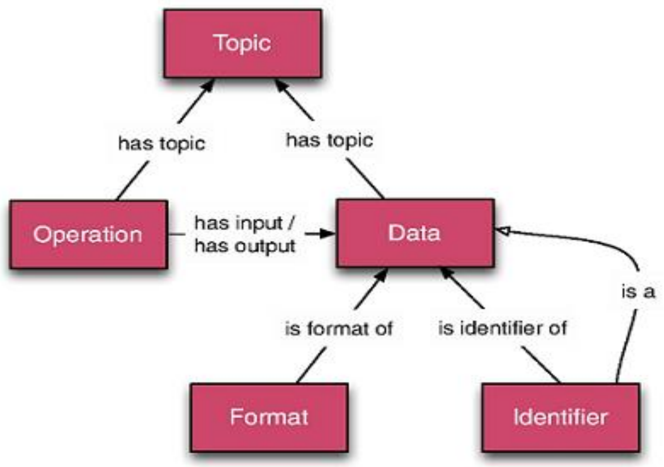


Figure 34. Embrace data and methods (EDAM) ontology schema.
(Source: https://academic.oup.com/bioinformatics/article/29/10/1325/255660)

5.3. The Software Ontology (SWO)

The Software Ontology (SWO) [7] is an ontology that describes the software used within computational biology. This includes bioinformatics resources as well as any software tools that were used during the preparation and maintenance of data. SWO is being actively developed due to the interest in the recording and reproducibility of biomedical investigations. It provides the required vocabulary and identifiers for the software part of describing the provenance of computations automatically.

Describing software and its input and output data is also important for software searching and application development as well as workflows construction. An ontology used to describe softwares, such as SWO, can help all the areas stated above. The scope of the ontology though, is not limited only in covering bioinformatics. It needs to cover any tools used during the analysis, management and presentation of biological data. For each software, SWO needs to develop to include the software's range, the descriptions of its objectives, the input and output data, its version as well as some aspects of its project details. To date though, neither SWO nor any other ontology that describes softwares cover all that information.

The Software Ontology project is based on the following agile principles:

I. The SWO's users, domain experts and ontology engineers are all active contributors on this project.

II. Tight engagement with the users in order to gather requirements and to have ontology modelling sessions as iterative activities with a new increment being a result of each iteration.

III. Acknowledgement that requirements and priorities my alter during the development phase.

IV. Encouragement of organized and cross functional groups of developers.

V. Testing is mandatory and it has to occur frequently.

VI. Provision of frequent interactions with the associates for discussion, testing and refinement.

In order for these principles to apply and succeed on creating an agile ontology, a certain engineering method has to be followed. That method can be summarized in the following steps:

I. Requirements gathering from stakeholders.

II. Requirements prioritization, the complexity of implementing each requirement has to be estimated and requirements have to be ranked by the participants.

III. Implementation of the top ranked requirements that resulted from the previous step.

IV. Evaluation of the product, the results of questions act as queries within the ontology are evaluated.

Apart from SWO, other promising efforts that have been made to develop an ontology that describes softwares broadly are: Description Of A Project(DOAP) [96], OWL-S [97], Data mining tools ontologies, the Ontology of Biomedical Investigations(OBI) [98], EDAM, the Bioinformatics Resource Ontology(BRO) [99] and myGrid Ontology.

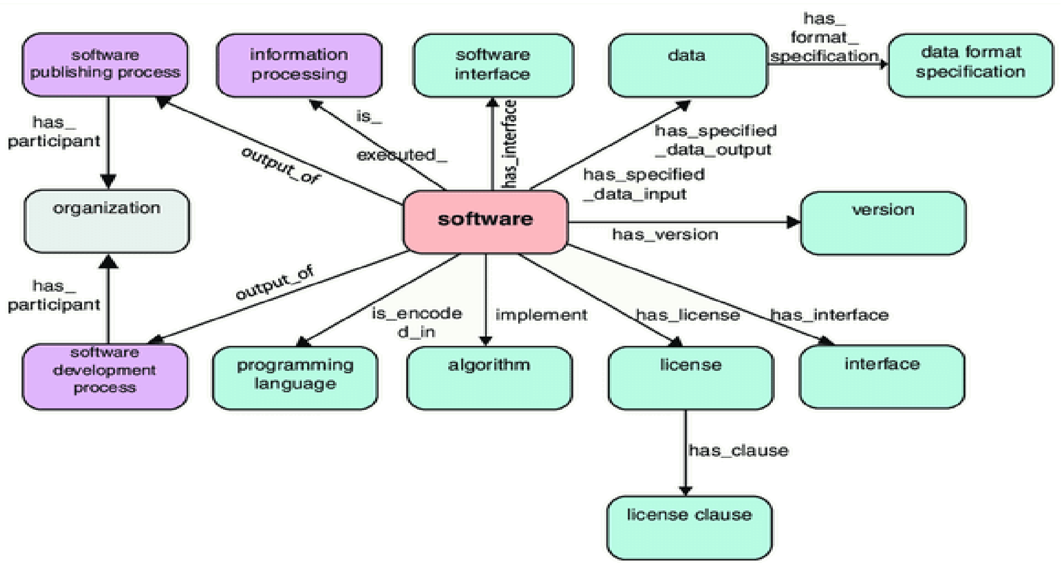


Figure 35. Software ontology (SWO) schema. (Source:https://www.semanticscholar.org/paper/The-Software-Ontology-(SWO)%3A-a-resource-for-in-data-Malone-Brown/a585548fccdfe3e3bf89c78f7126ccf332617e77/figure/1)

5.4. Test case

To test the efficiency of the framework, we collected 13 questions from SEQanswers [100] from users that were searching for a bioinformatics related tool for a specific purpose. For each question, we collected the tools that were proposed as answers from the users to check if each proposed tool is contained in the results of the framework. Things to be taken into consideration by using this method of case testing are that the users might haven't proposed the most suitable tools for the users' questions, the users' requirements might not be completely senseful and that the most suitable tools for the user might have been released later than the question was asked. The original questions of the users were :

1. "Hi everyone,I am doing exome sequencing, paired end, but not used to analyze reads.After alignement, how do i proceed the BAM file?Thank you"
2. "Hi,I am trying to quantify the number of reads mapping to genes in a highly redundant bacterial genome (i.e., often several exact gene copies in the genome).Ive mapped reads to genes with bbmap, and quantified using featurecounts. My problem is that reads are only counted for one of the copies of redundant genes, and not all of them.I just want to count the number of expressed genes (I want to count all of the copies since I cant distinguish them). Ive tried playing around with multi-mapping parameters in featurecounts, but nothing has changed.Does anyone here know how to do this?Thanks!"
3. "Hello im trying to calculate the percentage of covered CpG sites in my RRBS library and compare it with total CpG sites in reference genome. i got splitting report from Bismark (see bellow)q1- could i say CpG sites in my RRBS library are equal to number of Total methylated Cs in CpG context + number of Total C to T conversions in CpG context (around 19 million) ? if No how i can find total CpG sites in RRBS library?q2- i downloaded pig CGI annotation and counted all CpG sites but the total was around 2 million. sound very low for me. how i can find the actual number of CpG sites in reference genome?q3- is there a way to determine CpG sites per chromosome and compare it with CpG sites in each chromosome of reference genome?Final Cytosine Methylation Report = Total number of Cs analysed- 141645338Total methylated Cs in CpG context- 7904886Total methylated Cs in CHG context- 50683Total methylated Cs in CHH context- 107717Total C to T conversions in CpG context: 12298571Total C to T conversions in CHG context: 35912924Total C to T conversions in CHH context: 85370557"
4. "Hi All,I have a fastq file which I would like to split into 2 files with every other read going into the 2 separate files. What would the Split function command line be for this? I am a new to computing, so it you are most explicit that would be helpful.Best,"
5. "Hello everyone, is it possible to calculate depth of coverage and read depth using Bismark ? if yes how and if No what is the best way to calculate depth of

coverage and read of depth from RRBS reads taking from illumina sequencing ?Best"

6. "Hi all,Do you think there is an efficient way of downsampling only a few regions of a bam files (in my case the regions with a too high coverage).The idea, would be too randomly remove reads in regions where the coverage is above a given coverage.Indeed, in my analyses, those regions cause some steps of the pipeline to become really slow.Thanks for all your suggestions"
7. "Hi, community,I am analyzing the taxonomic profiling of my shotgun data. Which are 100bp paired-end reads from Illumina Hiseq. Now I am using Metaphlan2 to do the metagenomics profiling. However, the profiling result is far away from Illumina 16S miseq results. Since I have also been using Illumina 16S Miseq to test the taxonomy of my samples for several years. I have two the control samples and treatment samples. In Metaphlan2 results, it gave me around 30 archaea and 70 bacteria for control samples, while Miseq 16S reads tell me that only around 15 archaea and 85 bacteria for control samples. For treatment, shotgun profiling told me 60 archaea and 40 bacteria, while Miseq gave me 20 archaea and 80 bacteria. For my experience, this kind of sample could not achieve that much archaea abundance than bacteria. Furthermore, some(not all) bacteria and archaea composition are different between Miseq result and Metaphlan2 result.Why is the result so different? Are there any suggestions why the two method result differs so much?I am confused. Looking forward to a help."
8. "Hello,Human RNA-seq dataset was generated from Illumina HiSeq 3000 with 2X100 cycles run.The first step is making alignment of the reads to the human genome. These are many aligner, such as: Bowtie, GASSST, PASS, SOAP, BOAT. Each aligners has different performs in different kinds of data.Which is the best suitable aligner for RNA-seq data?Thank you in advance for great help!Sincerely,Yue"
9. "Is there a tool that can tell me coverage per some uniform interval along a set of reference contigs? I know I could use bedtools coverage and build a bed file defining the intervals, but I was asked if there was not already a tool that could just do uniform intervals on its own without having to setup a bedfile to define the intervals."
10. "I have two fastq files from paired end sequencing. I got those two files after converting a bam file to fastq. I was doing a quality check on the files, when I saw the number of sequences option in FASTQC tool gave different number for both files.The number of sequences for read 1 was : 508168252The number of sequences for read 2 was : 512336921Shouldnt this be the same?"
11. "Hi, Im very new to the forum, if you have any suggestions, please share them.So im working with honey bee to identify QTL associated to a resistance trait (quantitative trait) using GWAS (by calling SNPs).Honey bee queens are diploid and males progeniture inherit 1/2 of their genes (so they are haploid).We

want to sequence the queens, but cant use them, so we sampled the males progeny. I made 4 pools of 6 males for each queen.My question is this: Is it better to do all my pipeline to SNP calling on each of the 4 pools individually before joining the results together, or should i joint the pools raw reads before starting my pipeline to enhance statistical power?My logic is that if I dont initially joint them, I can later compare allele frequency between each replicate to insure that rare variants are not eliminated.thanks"

12. “Hello everyone.I have a questions on gene expression profiling.1. Does the end product of gene expression profiling is in a DNA sequence? If so, does the sequence is in normal DNA sequence or mutated DNA sequence?2. What are the method use to develop the expression profile?3. Please share any related article of the method. Ive have search a lot on the method, but i cant understand since im not from bioinformatic background.4. If anyone familiar with BRCA1 gene expression profiling, can anyone share with me the development of the profile? Thank you for your time and reply."
13. “Dear users,Id like to ask you for your advise on a certain project that I am working on.My job is to retrospectively analyse NGS data from patients suffering from ALL and MM (IGH).However, I am not a bioinformatician. I just received the sequencing data in Excel (and .csv) files and Im supposed to analyse these data.Of course, Ive heard of IMGT/VQUEST and IgBlast and it is no problem for me to work with these programmes. But the issue is that I cant do this with tens of thousands of sequences.After further research, I came across the R tool tcR. It seems to be a well-done programme. Unfortunately, I always receive error messages when trying to integrate my files in it.I found out that it might be useful to convert my files into .txt files or to work with VDJ-tools, Immunoseq, mixcr or other tools. But I do not have access to these programmes or they require Linux (which I dont have; Windows).In addition, the sequencing data in my Excel files are not in fasta-format. But if I would change this by editing the sequences manually Id be busy for the next three months.My goal is to analyse my data for gene usage and to be able to search quickly for potential subclones.I look forward to hearing your opinion! :)Thanks in advance!Pablo"

Those questions were reviewed and made ‘search engine friendly’ by two expert bioinformaticians. Irrelevant words were removed from the questions and the result is only the keywords that would be used in a search engine. All those questions were used as queries in the framework and the purpose was to check if the tools that were proposed by the users were retrieved as a result from the search. The framework was set to give a maximum of 1000 results per query. The queries we used to run that test after the cleanup process were:

1. “BAM, analyze reads, exome sequencing, paired end”
2. “quantifying transcripts, including redundants, map using bbmap”
3. “annotate, bismark, chromosome, conversion, cover, cpg coverage, cytosine, methylate, rrbs, split, rrbs library, actual number in reference genome”

4. “command line, compute, fastq, read, split, partition the odd or even reads”
5. “calculate depth of coverage, read depth, Bismark, RRBS reads”
6. “downsampling, regions of a bam files, remove read above coverage”
7. “shotgun metagenomics, 16S amplicon, taxonomic profiles, Metaphlan, Miseq, differences”
8. “alignment, boat, bowtie, gassst, genome, hiseq, human, aligner, RNA-seq, HiSeq 3000”
9. “coverage per 10kb interval, reference contigs”
10. “reads, paired end fastq, number of sequences”
11. “allele, allele frequency, bee queen, call, diploid, frequence, gene, gwas, haploid, inhertit, join, male, pipeline, progeniture, progeny, qtl, rare, raw, read, replicate, resistance, sequence, snp, statistic, statistical power, trait, variant”
12. “article, bioinformatic, brca1, dna, expression, gene, method, normal, profile, sequence”
13. “analyse NGS, csv file, non fasta-format”

5.5. Results – first iteration

Initially we wanted to assess the power of pure semantic search against the common keyword-based search. For this experiment we used the keyword-based search, the EDAM search and the SWO search. The results for the 13 questions are shown in the following table.

Table 2. Test case results for each question.

No.	Proposed Tools	EDAM		SWO		Keyword-based	
		Yes/No	Rank	Yes/No	Rank	Yes/No	Rank
1	SAMtools	No	-	No	-	No	-
2	BBMap	Yes	462	No	-	Yes	1
	BWA	No	-	No	-	No	-
3	SeqMonk	No	-	No	-	No	-
	Bismark	Yes	807	Yes	267	Yes	5
	Methylkit	No	-	Yes	591	Yes	27
4	BBMap	Yes	569	Yes	170	Yes	284
5	SeqMonk	Yes	908	No	-	No	-
6	VariantBam	Yes	816	Yes	107	Yes	521
7	Kraken	Yes	16	Yes	60	Yes	305
8	STAR	Yes	246	Yes	225	Yes	4
	Hisat2	Yes	398	Yes	227	Yes	21
9	Mosdepth	No	-	No	-	No	-
	deepTools	No	-	No	-	No	-
10	BBMap	Yes	195	Yes	305	No	-
11	SAMtools	No	-	No	-	Yes	775

12	DESeq2	Yes	150	Yes	161	Yes	645
	edgeR	Yes	454	Yes	388	No	-
	LIMMA	Yes	820	Yes	904	No	-
13	Galaxy	No	-	Yes	593	No	-

From all 20 proposed tools by users, 15 of them were found by the search engine from at least one search. That makes the ensemble of predictors find rate being at 75%.

The following table contains the final results of each search.

Table 3. Test case final results.

	EDAM	**SWO**	**Keyword-based**	**Ensemble of predictors**
Tools Found	12	12	10	15
Find Rate(%)	60	60	50	75
Average Rank	486	333	257	359

The results retrieved from the two searching techniques differ significantly. The semantic searches have similar results with each other with the SWO-based being better than the EDAM-based in terms of average rank but both semantic searches gave remarkably different results than the keyword-based search. The ontology-based searches having a higher average rank may be due to the keyword-based search using not only the description but also the name of the tools to rank them. Another explanation can be that the EDAM ontology isn't as fitting for our use since the SWO ontology has only a slightly higher average rank than the keyword-based search.

5.6. Results – second iteration

The results from the first iteration indicate that the keyword-based search provide better results that the pure semantic search in terms of average rank. On the other hand the semantic search provided better results in terms of accuracy. Looking at the results from the first iteration, and other test experiments that we used, we can safely come into the conclusion that the ontology searches (EDAM and SWO) provide similar results but different than the keyword-based. An indicative example is shown in the screenshot of the UI at Figure 25. These findings indicate that a combination of semantic search and free text search could provide better results and that was our motivation to create the hybrid search even though it was not planned in the beginning.

As explained in section 3.3 the hybrid search is an Apache solr search over JSON files that contain:

- All names and descriptions from the tools in the biotools registry
- 5 EDAM terms and their (free text descriptions) with the highest score assigned by solr when searched using the description of the tool.

- 5 EDAM terms and their (free text labels) with the highest score assigned by solr when searched using the description of the tool.
- 5 SWO terms and their (free text descriptions) with the highest score assigned by solr when searched using the description of the tool.
- 5 SWO terms and their (free text labels) with the highest score assigned by solr when searched using the description of the tool.

The results of the three previous searches (EDAM, SWO, Keyword-based) and the hybrid search are shown in the following table.

Table 4. Test case results for each question with hybrid search.

No.	Proposed Tools	EDAM		SWO		Keyword-based		Hybrid	
		Yes/No	Rank	Yes/No	Rank	Yes/No	Rank	Yes/No	Rank
1	SAMtools	No	-	No	-	No	-	No	-
2	BBMap	No	-	No	-	Yes	1	Yes	30
	BWA	Yes	462	No	-	No	-	No	-
3	SeqMonk	No	-	No	-	No	-	No	-
	Bismark	No	-	No	-	Yes	5	Yes	25
	Methylkit	Yes	807	Yes	267	Yes	27	Yes	77
4	BBMap	No	-	Yes	591	Yes	284	Yes	488
5	SeqMonk	Yes	569	Yes	170	No	-	No	-
6	VariantBam	Yes	908	No	-	Yes	521	Yes	534
7	Kraken	Yes	816	Yes	107	Yes	305	Yes	211
8	STAR	Yes	16	Yes	60	Yes	4	No	-
	Hisat2	Yes	246	Yes	225	Yes	21	Yes	244
9	Mosdepth	Yes	398	Yes	227	No	-	Yes	367
	deepTools	No	-	No	-	No	-	No	-
10	BBMap	No	-	No	-	No	-	No	-
11	SAMtools	Yes	195	Yes	305	Yes	775	Yes	396
12	DESeq2	No	-	No	-	Yes	645	Yes	500
	edgeR	Yes	150	Yes	161	No	-	No	-
	LIMMA	Yes	454	Yes	388	No	-	Yes	600
13	Galaxy	Yes	820	Yes	904	No	-	No	-

After adding the Hybrid search, the ensemble of predictors find rate is 80% since 16 out of the 20 proposed tools were found from at least one search.

The Hybrid search found 11 tools which is 55% accuracy and the average rank of the tools is 315. We have to state that the hybrid search is more complicated than the other searches in terms of combinations and weights of the variables that we are using. EDAM search has 3 variables, SWO search has 4 variables, and keyword-search two variables while hybrid search 22 variables. The high number of variables in the case of the hybrid search give us the opportunity to test various combinations of variables-weights for different domains and fine tune our search engine even more. This, along with deployment of other domains and ontologies could be a future work for this thesis.

The results can be found in the following table.

Table 5. Test case results with hybrid search.

	EDAM	**SWO**	**Keyword-based**	**Hybrid**	**Ensemble of predictors**
Tools Found	12	12	10	11	16
Find Rate(%)	60	60	50	55	80
Average Rank	486	333	257	315	348

As show, the Hybrid search has almost the same average rank as the SWO-based search. That enhances the possibility of the EDAM ontology being less fitting than the SWO ontology for our use and lessens the possibility of the average rank difference being a result of the usage of the name of the tools for ranking since both keyword-based and hybrid use it.

The implemented ontology search has been implemented with an architecture that takes advantage of the Apache Solr indexing system and relational databases. On the other hand the free text and hybrid search take advantage of the Apache Solr solely. The average response times for each search methodology can be found in the following table.

Table 6. Test case response times.

	EDAM	SWO	Keyword-based	Hybrid
Response Time (ms)	475	285	202	376

The response times are acceptable for a real time search engine (even with a corpus of 12000 tools).

We have also to mention that the results when we used as input the original question without the cleaning from the experts have a big differentiation on precision and average rank which was significantly lower. This is expected since the plain text contains a lot of "noise" in our case due to grammar and syntax. We do not expect to be an issue for our system since the end users in reality are used to search with keywords and not plain text, an indicative example is the well-known search engine Google.

6. Conclusions

The end goal of this thesis has been to examine the use of semantic technologies for bioinformatics data annotation by employing domain specific ontologies and also to make use of natural language processing methods in order to aid the non-bioinformatics expert users to successfully search for bioinformatics related tools using natural language through either text or a speech recognition system. By using different search methods, we were able to create a hybrid search engine that takes advantage of multiple searching techniques to ensure that the users receive the best results possible for their needs. We were also able to compare the efficiency of those techniques by experimenting on the system with different test case scenarios.

As part of this thesis, we developed a web based framework that is able to interact with the user through free speech or text and allows him to search for bioinformatics related tools in real time. The user describes the functionality or the name of the tool he is seeking for and the framework, using semantic and non-semantic technologies, if found, retrieves the tools that fit the user’s description. The results obtained by the framework are ordered descendingly, with the first tool from each list being the best possible solution for the user for each searching technique and they are also separated in different lists, with each one being a result of a different searching technique.

The results that were obtained by the experiment, show that the system returns relevant results to the users’ needs, with each searching technique having different results and response times.

Bibliography

[1] D. Yu, "Automatic Speech Recognition." [S.l.]: SPRINGER LONDON LTD, 2016.

[2] M. Gao, C. Liu, and F. Chen, "An Ontology Search Engine Based on Semantic Analysis," in *Third International Conference on Information Technology and Applications (ICITA'05)*, vol. 1, pp. 256–259.

[3] J. Ye, P. He, et al., "The design and implementation of the 1911 revolution ontology search engine," in *2012 9th International Conference on Fuzzy Systems and Knowledge Discovery*, 2012, pp. 1257–1260.

[4] B. Chandrasekaran, J. R. Josephson, and V. R. Benjamins, "What are ontologies, and why do we need them?," *IEEE Intell. Syst.*, vol. 14, no. 1, pp. 20–26, Jan. 1999.

[5] S. H. Alahmadi, "Information Retrieval of Distributed Databases A Case Study: Search Engines Systems," in *2018 1st International Conference on Computer Applications & Information Security (ICCAIS)*, 2018, pp. 1–5.

[6] J. Ison, M. Kalas, et al., "EDAM: an ontology of bioinformatics operations, types of data and identifiers, topics and formats," *Bioinformatics*, vol. 29, no. 10, pp. 1325–1332, May 2013.

[7] J. Malone, A. Brown, et al., "The Software Ontology (SWO): a resource for reproducibility in biomedical data analysis, curation and digital preservation," *J. Biomed. Semantics*, vol. 5, no. 1, p. 25, Jun. 2014.

[8] D. Smiley, *Apache Solr enterprise search server : enhance your searches with faceted navigation, result highlighting, relevancy-ranked sorting, and much more with this comprehensive guide to Apache Solr 4.* .

[9] J. Ison, K. Rapacki, et al., "Tools and data services registry: a community effort to document bioinformatics resources," *Nucleic Acids Res.*, vol. 44, no. D1, pp. D38–D47, Jan. 2016.

[10] S. J.Arora and R. Pal Singh, "Automatic Speech Recognition: A Review," *Int. J. Comput. Appl.*, vol. 60, no. 9, pp. 34–44, Dec. 2012.

[11] S. J. Russell and P. Norvig, "Artificial Intelligence : A Modern Approach," 2016.

[12] R. P. Lippmann, "Review of Neural Networks for Speech Recognition," *Neural Comput.*, vol. 1, no. 1, pp. 1–38, Mar. 1989.

[13] D. O'Shaughnessy, "Linear predictive coding," *IEEE Potentials*, vol. 7, no. 1, pp. 29–32, Feb. 1988.

[14] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Trans. Acoust.*, vol. 26, no. 1, pp. 43–49, Feb. 1978.

[15] T. Winograd, "Understanding natural language," *Cogn. Psychol.*, vol. 3, no. 1, pp. 1–191, Jan. 1972.

[16] T. N. Sainath, "Speech and Language Algorithms Group Improvements to Deep Neural Networks for Large Vocabulary Continuous Speech Recognition Tasks Speech and

Language Algorithms Group,” 2014.

[17] J. S. Garofolo, L. F. Lamel, et al., “DARPA TIMIT acoustic-phonetic continous speech corpus CD-ROM. NIST speech disc 1-1.1,” *Unknown*, vol. 93, 1993.

[18] J. M. B. and A. F. M. Smith, “Bayesian Theory,” *Meas. Sci. Technol.*, vol. 12, no. 2, pp. 221–222, Feb. 2001.

[19] P. C. Woodland and D. Povey, “Large scale discriminative training of hidden Markov models for speech recognition,” *Comput. Speech Lang.*, vol. 16, no. 1, pp. 25–47, Jan. 2002.

[20] D. Comaniciu, V. Ramesh, and P. Meer, “Kernel-based object tracking,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 5, pp. 564–577, May 2003.

[21] B.-H. Juang, W. Hou, and C.-H. Lee, “Minimum classification error rate methods for speech recognition,” *IEEE Trans. Speech Audio Process.*, vol. 5, no. 3, pp. 257–265, May 1997.

[22] S. Katagiri, C. Lee, B. Juang, “New discriminative training algorithms based on the generalized probabilistic descent method,” 1991.

[23] E. Ertin, J. W. Fisher, and L. C. Potter, “Maximum Mutual Information Principle for Dynamic Sensor Query Problems,” 2003, pp. 405–416.

[24] K.-F. Lee, S. Hayamizu, at al., “Allophone clustering for continuous speech recognition,” in *International Conference on Acoustics, Speech, and Signal Processing*, 1990, pp. 749–752.

[25] D. Hakkani-Tur, G. Riccardi, and A. Gorin, “Active learning for automatic speech recognition,” in *IEEE International Conference on Acoustics Speech and Signal Processing*, 2002, p. IV-3904-IV-3907.

[26] K.-F. Lee, “On large-vocabulary speaker-independent continuous speech recognition,” *Speech Commun.*, vol. 7, no. 4, pp. 375–379, Dec. 1988.

[27] H. Tseng, P. Chang, et al., “A Conditional Random Field Word Segmenter for Sighan Bakeoff,” 2005.

[28] M. A. Anusuya and S. K. Katti, “Speech Recognition by Machine: A Review,” *IJCSIS) Int. J. Comput. Sci. Inf. Secur.*, vol. 6, no. 3, 2009.

[29] R. Sukanesh, S. Veluchamy, and M. :. Karthikeyan, *International journal of engineering research.*, vol. 3, no. 2. IJERA, 2014.

[30] M. D. Network, “Microsoft Speech API (SAPI) 5.3“, Microsoft, [Online]. Available https://msdn.microsoft.com/en-us/library.[Accessed 17 July 2018].

[31] W. Walker, P. Lamere, et al., “Sphinx-4: A Flexible Open Source Framework for Speech Recognition,” 2004.

[32] C. Beattie, J. Z. Leibo, et al., “DeepMind Lab,” Dec. 2016.

[33] J. Li, L. Deng, et al., "Fundamentals of speech recognition," in *Robust Automatic Speech Recognition*, Elsevier, 2016, pp. 9–40.

[34] G. L. Manney and W. H. Daffer, "Jet and Tropopause Products for Analysis and Characterization (JETPAC)," Sep. 2012.

[35] V. Kepuska and G. Bohouta, "Next-generation of virtual personal assistants (Microsoft Cortana, Apple Siri, Amazon Alexa and Google Home)," in *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, 2018, pp. 99–103.

[36] G. Hinton, L. Deng, et al., "Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.

[37] T. Berners-Lee and M. Fischetti, *Weaving the Web : the original design and ultimate destiny of the World Wide Web by its inventor*. Harper Audio, 1999.

[38] A. Sheth, "Semantic Web and Information Brokering: Opportunities, Commercialization, and Challenges," *Kno.e.sis Publ.*, Sep. 2000.

[39] J. Townley, "The Streaming Search Engine That Reads Your Mind," 2000.

[40] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific American*, vol. 284. Scientific American, a division of Nature America, Inc., pp. 34–43, 2001.

[41] T. Berners-lee, J. Hendler, and O. Lassila, "The Semantic Web A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities.", 2001.

[42] W. Research and A. P. Sheth, "An Interview with Amit Sheth: The Information Systems Perspective on Semantic &quot;Semantic technology is here to stay...&quot;," *AIS SIGSEMIS Bull.*, vol. 1, no. 1, pp. 14–72, 2004.

[43] P. Norvig, "Semantic Web Ontologies: What Works and What Doesn't," *AlwaysOn-the Insid. Netw.*, 2005.

[44] M. D'Aquin and E. Motta, "Watson, more than a Semantic Web search engine," *Semant. Web*, 2011.

[45] Y. alSafadi, J.-F. Baget, et al., "OWL Web Ontology Language Overview.", 2004.

[46] T. Garfat and L. Fulcher, "Characteristics of a Relational Child and Youth Care Approach.", 2018.

[47] W.D. Hillis and B. Ferren, "Meta-Web," Dec. 2003.

[48] L. Ding, T. Finin, et al., "Swoogle," in *Proceedings of the Thirteenth ACM conference on Information and knowledge management - CIKM '04*, 2004, p. 652.

[49] O. Verhodubs, "Towards the Ontology Web Search Engine.", 2015.

[50] M. D 'aquin, C. Baldassarre, et al., "WATSON: SUPPORTING NEXT GENERATION SEMANTIC WEB APPLICATIONS 1.", 2008.

[51] A. Sayed and A. Al Muqrishi, "IBRI-CASONTO: Ontology-based semantic search engine," *Egypt. Informatics J.*, vol. 18, no. 3, pp. 181–192, Nov. 2017.

[52] D. H. Widyantoro and J. Yen, "A fuzzy ontology-based abstract search engine and its user studies," in *10th IEEE International Conference on Fuzzy Systems. (Cat. No.01CH37297)*, vol. 2, pp. 1291–1294.

[53] Q. T. Tho, S. C. Hui, A. C. M. Fong, and Tru Hoang Cao, "Automatic fuzzy ontology generation for semantic Web," *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 6, pp. 842–856, Jun. 2006.

[54] K. Taha, "GOseek: A gene ontology search engine using enhanced keywords," in *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2013, pp. 1502–1505.

[55] M. Ashburner, C. A. Ball, et al., "Gene Ontology: tool for the unification of biology," *Nat. Genet.*, vol. 25, no. 1, pp. 25–29, May 2000.

[56] C. Jonquet, P. LePendu, et al., "NCBO Resource Index: Ontology-based search and mining of biomedical resources," *Web Semant. Sci. Serv. Agents World Wide Web*, vol. 9, no. 3, pp. 316–324, Sep. 2011.

[57] N. F. Noy, N. H. Shah, et al., "BioPortal: ontologies and integrated data resources at the click of a mouse," *Nucleic Acids Res.*, vol. 37, no. Web Server, pp. W170–W173, Jul. 2009.

[58] B. Smith, M. Ashburner, et al., "The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration," *Nat. Biotechnol.*, vol. 25, no. 11, pp. 1251–1255, Nov. 2007.

[59] O. Bodenreider, "The Unified Medical Language System (UMLS): integrating biomedical terminology," *Nucleic Acids Res.*, vol. 32, no. 90001, p. 267D–270, Jan. 2004.

[60] F. Yergeau and J. Cowan, "Extensible Markup Language (XML) 1.1 (Second Edition).", 2006.

[61] H. Dietze and M. Schroeder, "GoWeb: a semantic search engine for the life science web.," *BMC Bioinformatics*, vol. 10 Suppl 10, no. Suppl 10, p. S7, Oct. 2009.

[62] C. E. Lipscomb, "Medical Subject Headings (MeSH).," *Bull. Med. Libr. Assoc.*, vol. 88, no. 3, pp. 265–6, Jul. 2000.

[63] A. Doms and M. Schroeder, "GoPubMed: exploring PubMed with the Gene Ontology," *Nucleic Acids Res.*, vol. 33, no. Web Server, pp. W783–W786, Jul. 2005.

[64] S. Langer and J. Beel, "Apache Lucene as Content-Based-Filtering Recommender System: 3 Lessons Learned.", 2017.

[65] The Apache Software Foundation, "Apache Stanbol - Apache Stanbol Documentation." 2010.

[66] L. Richardson and S. Ruby, "RESTful web services. ", 2007.

[67] M. Rouse, "content management system (CMS)," TechTarget,[Online]. Available http//searchsoa. techtarget. com/definition/content-managementsystem.[Accessed 17 August 2016], 2011.

[68] OSGi Alliance. and Osgi, *OSGi service platform : release 3, March 2003*. IOS Press, 2003.

[69] A. Volpini, "Adding Semantic Search to Apache Solr.", 2014.

[70] "Open Semantic Search: Your own search engine for documents, images, tables, files, intranet & news." 2018.

[71] R. H. Wiggins, H. C. Davidson, et al."Image File Formats: Past, Present, and Future," *RadioGraphics*, vol. 21, no. 3, pp. 789–798, May 2001.

[72] Y. Shafranovich, "Common Format and MIME Type for Comma-Separated Values (CSV) Files.", 2005.

[73] R. Weir, "OpenDocument Format: The Standard for Office Documents," *IEEE Internet Comput.*, vol. 13, no. 2, pp. 83–87, Mar. 2009.

[74] A. Pandey, V. Sharma, et al., "Optical Character Recognition (OCR)," *Int. J. Eng. Manag. Res.*, no. 7, 2017.

[75] R. Côté, P. Jones, R. Apweiler, and H. Hermjakob, "The Ontology Lookup Service, a lightweight cross-platform tool for controlled vocabulary queries," *BMC Bioinformatics*, vol. 7, no. 1, p. 97, Feb. 2006.

[76] J. Adorf, "Web Speech API," 2013.

[77] J. Oh, S. Lee, and S. Lee, "Advanced evidence collection and analysis of web browser activity," *Digit. Investig.*, vol. 8, pp. S62–S70, Aug. 2011.

[78] G. G. Chowdhury, "Natural language processing," *Annu. Rev. Inf. Sci. Technol.*, vol. 37, no. 1, pp. 51–89, Jan. 2005.

[79] T. Bray, Ed., "The JavaScript Object Notation (JSON) Data Interchange Format," Dec. 2017.

[80] I. D. E. NetBeans, "Version 8.2.", NetBeans [Online]. Available: https://netbeans.apache.org. [Accessed 5 November 2018].

[81] B. A. Burd and B. A., *JSP: JavaServer pages*. M & T Books, 2001.

[82] A. Vukotic and J. Goodwill, *Apache Tomcat 7*. Berkeley, CA: Apress, 2011.

[83] F. P. Miller, A. F. Vandome, and J. McBrewster, "Apache Maven," 2010.

[84] J. Gosling and J. Gosling, *The Java language specification*. Addison-Wesley, 2000.

[85] D. Flanagan, *JavaScript : the definitive guide*. O'Reilly, 2006.

[86] A. Tozawa, M. Tatsubori, et al., "Copy-on-write in the PHP language," *ACM SIGPLAN Not.*, vol. 44, no. 1, p. 200, Jan. 2009.

[87] B. Bos, T. Celik, I. Hickson and H. W. Lee, "Cascading Style Sheets, level 2 Revision 1 (CSS 2.1) Specification.",World Wide Web Consortium(W3C), 2011.

[88] C. Musciano and B. Kennedy, *HTML, the definitive guide*. O'Reilly, 1997.

[89] C. J. Date and H. Darwen, *A guide to the SQL Standard: a user's guide to the standard relational language SQL*. Addison-Wesley, 1989.

[90] K. Douglas and S. P. Douglas, *PostgreSQL : a comprehensive guide to building, programming, and administering PostgresSQL databases*. Sams, 2003.

[91] A. Friends, "XAMPP Apache+ MySQL+ PHP+ Perl," *Apache Friends*, 2014.

[92] K. Wolstencroft, P. Alper, et al., "Grid ontology: bioinformatics service discovery," *Int. J. Bioinform. Res. Appl.*, vol. 3, no. 3, p. 303, 2007.

[93] P. Rice, I. Longden, and A. Bleasby, "EMBOSS: the European Molecular Biology Open Software Suite.," *Trends Genet.*, vol. 16, no. 6, pp. 276–7, Jun. 2000.

[94] M. D. Wilkinson and M. Links, "BioMOBY: An open source biological web services proposal," *Brief. Bioinform.*, vol. 3, no. 4, pp. 331–341, Jan. 2002.

[95] M. Y. Galperin and G. R. Cochrane, "Nucleic Acids Research annual Database Issue and the NAR online Molecular Biology Database Collection in 2009," *Nucleic Acids Res.*, vol. 37, no. Database, pp. D1–D4, Jan. 2009.

[96] E. Dumbill, "DOAP: Description of a Project,", 2010.

[97] M. Burstein, J. Hobbs, et al., "OWL-S: Semantic Markup for Web Services," *W3C Memb. Submiss.*, 2004.

[98] B. Peters and T. O. Consortium, "Ontology for Biomedical Investigations," *Nat. Preced.*, Aug. 2009.

[99] J. D. Tenenbaum, P. L. Whetzel, et al., "The Biomedical Resource Ontology (BRO) to enable resource discovery in clinical and translational research," *J. Biomed. Inform.*, vol. 44, no. 1, pp. 137–145, Feb. 2011.

[100] J.-W. Li, R. Schmieder, et al., "SEQanswers: an open access community for collaboratively decoding genomes," *Bioinformatics*, vol. 28, no. 9, pp. 1272–1273, May 2012.