A Framework for Building Lightweight Ontologies based on Semi-Structured Data for Semantic Annotation

A Dissertation

Presented to the

Graduate Faculty of the

University of Louisiana at Lafayette

In Partial Fulfillment of the

Requirements for the Degree

Doctor of Philosophy

Elshaimaa Elsayed Ali

Fall 2015

A Framework for Building Lightweight Ontologies based on Semi-Structured data for Semantic annotation

Elshaimaa Elsayed Ali

APPROVED:

DEDICATION

To

My husband, Abdelhamid Moursy,

Parents, Elsayed Ali & Sanaa Genena,

Son Hussein Moursy,

and

Daughters Sondos & Khadija Moursy

ACKNOWLEDGMENTS

All praise is due to Allah the almighty, the Most Gracious, and the Most Merciful on whom ultimately we depend for sustenance and guidance. I thank Allah for giving me the strength and ability to complete this work.

This work would not be accomplished without the effort of a lot of people. First and foremost, I would like to express my deep appreciation to my advisor, Dr. Vijay Raghavan for his help, support, and patience throughout my stay in the Center for Advanced Computer Studies (CACS). I would like to express my gratitude for the diligent effort he put to teach, guide and support his students not only as an advisor but also as a father. There have been days where I remember feeling stumped about the direction of my research, but, a meeting with Dr. Vijay would restore my motivation and raise my spirits. My respect and thanks also go to the members of my committee, Dr. Rasiah Loganantharaj, Dr. Anthony Maida and Dr. Ryan Benton, for their tremendous help, advice, and encouragement. I would like also to thank all the CACS students especially Ramish Sunkara, and Tejas Nomular, who worked with me to accomplish this work.

My great appreciation and deep love goes to my great father Elsayed Ali and loving mother Sanaa Genena who had always motivated, encouraged and supported me throughout my life journey. Their prayers, love and confidence provided me with the calming influence, and warm support best needed on hard days. All my love and appreciation go to my sister Esraa Ali and brothers, Ahmed and Mohamed Ali. Ahmed Ali selflessly gave all effort, support and time he can to help me and all my small family whenever needed.

I would not be more thankful to my family and friends in Egypt and the United States. It is an honor to dedicate this work to them as their support and prayer were a great help throughout this endeavor. Special thanks go to Abeer Emadah, Dalia Aboelfadl, and Safaa Shaban for all the support and help they gave to make this work a success. I would like to express my love to my sweet hearts, my son Hussein and daughters Sondos and Khadija. They had been the lights of my life. Thank you Hussein and Sondos for your patience, help and understanding, and Thank you Khadija, for the giggles and warm smiles you gave to this family especially on our stressful days.

Last but not least, I find it difficult to express my appreciation to my beloved husband, Abdelhamid Moursy, because it is so boundless. He is my most enthusiastic supporter, my best friend, and an amazing husband and father. His endearment and sacrifice to his family kept us safe and steady all times. I am grateful to my husband not only because he has given up so much to make my career a priority in our lives and to show confidence in my capabilities, but also because he shared every moment through the ups and downs of this entire amazing journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

API	Application Programming Interfaces
DL	Description Logic
EIF	European Interoperability Framework
FN	False Negative
FP	False Positive
HLA	Hyperspace Analogue to Language
IDF	Inverse Document Frequency
LOD	Linked Open Data
LSA	Latent Semantic Analysis
MAD	Median Absolute Deviation
MeSH	Medical Subject Heading
NLP	Natural Language Processing
OAI-PMH	Open Archives Initiative Protocol for Metadata Harvesting
OWL	Web Ontology Language
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
SKOS	Simple knowledge Organization System
SPARQL	Simple Protocol and RDF Query Language
TF	Term Frequency

TN	True Negative
TP	True Positive
TSCF-ISF	Term Synonym Concept Frequency-Inverse Sentence Frequency
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
XML	Extensible Markup Language

1 OVERVIEW

1.1 Introduction

"I have a dream for the Web (in which computers) become capable of analyzing all the data on the Web – the content, links, and transactions between people and computers. A 'Semantic Web', which should make this possible, has yet to emerge, but when it does, the day-to-day mechanisms of trade, bureaucracy and our daily lives will be handled by machines talking to machines. The 'intelligent agents' people have touted for ages will finally materialize."

Tim Berners-Lee, 1999

This was Tim Berners-Lee's (founder of the Semantic Web) famous quote in 1999. This quote described his vision of the Semantic Web. Semantic Web is the era of web technology known as Web 3.0. Web 1.0 is when the information age was born. It was the web of static documents, in fact Berners-Lee called it "The read only web". In other words, the early web allowed for users to search for information and read it and there was very little in the way of user interaction or content contribution. Web 2.0 focused on linking web pages, sharing data among websites and applications, the use of social networking, and community-based knowledge management. Web 2.0 is characterized as the web of user interaction. Web 3.0 is the "The Semantic Web". The Semantic Web as defined by the W3C [1] is "The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries.". Hence the Semantic Web is the web of shared data. There is a lot of information on the web but it is not part of the shared data. You can easily access the daily weather, your calendar of events, Google map to check locations of certain events, or review blogs to rate these locations. But, is it possible to know today's temperature together with the address of the location you are going to, and a rating of the venue, all in one place, for incense through checking your calendar? This is still not possible because we do not yet have a web of data. The Semantic Web would represent all kinds of content that can be easily understood by machines, independent of the underlining system, or programing technology. The ultimate goal is to satisfy the sophisticated needs of internet users. This is done via communicating with common semantic meanings.

The general vision of semantic web encompasses, many different views; a machine-readable data view, intelligent agents view, distributed database view, semantic annotation view, web application and improved searching view, and the web services view. The more important views among these that will influence all directions are the machine-readable data, and the semantic annotation views.

Semantics is the study of meaning which deals with the relationship between phrases, antonyms, synonyms, lexicology and what they denote. Semantic annotation is adding meta-data to describe data based on a semantic representation for the domain. Without a common machine understandable language for knowledge representation all other views are impossible. In order to deal with the uncertainty of information and the vastness of data, a standard is required that will provide a common mechanism and will allow web applications and services to exchange, store and retrieve data. The Semantic web is the web of meaningful data that can be captured, processed reused and communicated. This vision requires the existence of a knowledge base that will define concepts, and entities within a given domain in a language understandable, and communicated by machines, and web services. Ontology is the web 3.0 data representation for building the knowledge base, which is considered the backbone of the Semantic Web. The focus of this research is to build a framework for the development of domain ontologies –based on semi-structured data - to be used for semantic annotation of web documents.

Several definitions of ontologies exists in literature most common are:

- The philosophical definition is "Ontology is the theory of existence". In other words, ontology is the study of entities and their relations.
- The definition that is most cited in the computer science literature is, "An ontology is an explicit specification of a shared conceptualization" [2]. In this sense, ontologies provide a formal definition for concepts and their properties and relations, in a way that can be shared and processed.

The World Wide Web Consortium (W3C) has laid-out the foundations and the standards of the semantic web languages for developing ontologies. These standards promote common data formats and exchange protocols on the Web, most fundamentally the Resource Description Framework (RDF). In the next section we will briefly review the standards of the Semantic Web technology, the Semantic Web layer cake, and the semantic capabilities of each layer. In section 1.3 we will discuss the challenges facing the realization of the semantic web, then section 1.4 will highlight the motivation behind the research developed in this thesis. Section 1.5 will summarize the contributions proposed, and section 1.6 illustrates the structure and the organization of the remaining chapters.

1.2 Semantic Web Languages

The term semantic web often refers to the standards and technologies that enabled it [1]. Specifications of the Semantic Web languages represent a common infrastructure upon which Semantic Web applications are based. Semantic Web languages differ in their level of expressivity; they contain different structures suitable for expressing different sorts of information. Expressivity of the Semantic Web modeling languages is the ability of each language to describe certain aspects of the world. More expressive modeling language can express a wider variety of statements about the model.

In order to compare the expressivity of each language, we should give a brief discussion around the mathematical model behind them. Description Logics (DLs) is a subset of first order predicate calculus, so called because they focus on descriptions of concepts (classes) as the principal requirement for expressing logical propositions. A description logic system focuses on the use of classification and subsumption reasoning as its main rule of inference. DLs were designed as an extension to frames and semantic networks, which were initially not defined with formal logic-based semantics. DLs are the best candidate for specifying ontologies.

Realization of the Semantic Web is dependent on the existence of ontologies in various domains [3]. Knowledge representation systems that rely on description logics are based on two components: TBox (for terminological knowledge), and ABox (For assertions). This method of defining knowledge splits concepts, their properties and their relationships from individuals and their attributes and roles, which is expressed as assertions. TBox concept can be considered as the domain schema, while ABox is the instantiation of the domain schema in the form of assertions that relate to individuals. One can relate this to object-oriented modeling, where classes represent the TBox and objects represent the ABox.

Ontologies are the main model for knowledge representation in the Semantic Web. Ontologies describe the entities that consist of a set of types, properties, and relationships. Exactly what is provided around this varies, but these are the essentials of an ontology. Ontologies are expressed in the semantic web language defined by the W3C committee which is RDF (Resource description framework), RDFS (Resource Description Framework schema) and OWL (Web Ontology Language). They are all used to define ontologies with different level of semantics. Researchers in [4] have found that ontologies are the ideal knowledge model to formally describe web resources and its vocabulary and hence to make the underlying meaning of terms included in web pages explicit . Our focus here is not on the specification and the syntax of each language, but on the differences of the semantic capabilities of each relative to others, and its role in creating a semantic model for a given domain.

In the Semantic Web literature, the languages and the reasoning technologies has always been arranged as a stack called the layered cake. The latest one released by the W3C is presented in figure 2.1. It illustrates the hierarchy of the representation languages, where each layer exploits and uses the semantic capabilities of the one below, and each language is compatible with all the languages on top of it in the stack. Languages in the semantic layered cake are organized so that each language's expressive capabilities build on those of the ones below, i.e. as we go up in the layered cake, the level of expressivity increases.

We will review here the expressive capabilities of each layer, as explained in [5]. The first layer, URI and Unicode, follows the important features of the existing WWW. Unicode is a standard of encoding international character sets and it allows that all human languages can be used on the web using one standardized form. Uniform Resource Identifier (URI) is a string of a standardized form that allows to uniquely identify resources (e.g., documents). A subset of URI is Uniform Resource Locator (URL), which contains access mechanism and the (network) location of a document.

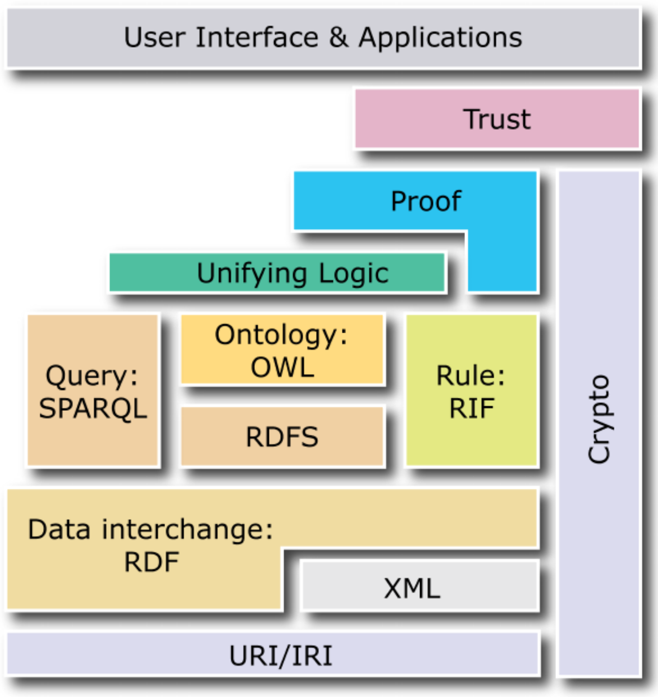


Figure 1.1 The Semantic web layered cake

Extensible Markup Language (XML) layer, together with XML namespace, and XML schema definitions, represent the common syntax used in the Semantic Web. XML is a general purpose markup language for documents containing structured information. Resource Description Framework (RDF) is the core syntax of semantic web languages; it addresses the issue of managing distributed data. All other semantic web standards build on the RDF specifications. RDF reuses many of the fundamental features of the web while extending them to suite the definition of distributed network of resources in ontologies. Its syntax is based on the Unicode XML syntax for exchanging data. The fundamental data structure to represent statements in RDF syntax is a triple. In the triple representation a statement is broken into three parts: Subject, Predicate, and Object. Figure 2.2 shows a graph model for an RDF statement.

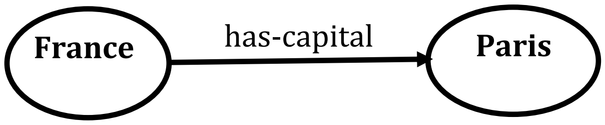


Figure 1.2 RDF statement

The Subject, the Predicate and the Object in the triples are all resources. In the Semantic Web there has to be a unique identifier for each resource. The same word might have different senses (ambiguity) or different resources can reference the same concept. Also, in some cases, ontology merging is needed; a node from one graph is merged with a node from another graph, if they reference the same resource. In RDF, this issue is resolved through the use of URIs. A URI provides a global identification for a resource that is common across the web.

All data in the Semantic Web use RDF as the primary representation language. The normative syntax for serializing RDF is XML, in the RDF/XML form.

To allow standardized description of taxonomies and other ontological constructs, an RDF Schema (RDFS) was created by extending the formal semantics within RDF. RDFS is a language with the expressivity to describe the basic elements of the ontologies. RDFS can be used to describe taxonomies of classes and properties and use them to create lightweight ontologies, which are ontologies in which concepts are connected by general associations, rather than strict formal connections. Chapter two will explain their expressive power in more details.

RDF is (roughly) limited to binary ground predicates, while RDFS is limited to a subclass hierarchy and a property hierarchy, with domain and range definitions of these properties. Stefen Staab summarizes the semantic limitations of RDF and RDFs [6] as follows:

- Local scope of properties: 'rdfs: range' defines the range of a property for all classes. Thus in RDF Schema we cannot declare range restrictions that apply to some classes only. For example, we cannot say that cows eat only plants, while other animals may eat meat, too.
- Disjointness of classes: Sometimes we wish to say that classes are disjoint. For example, male and female are disjoint. But in RDF Schema we can only state subclass relationships, e.g. female is a subclass of person.
- Boolean combinations of classes: Sometimes we wish to build new classes by combining other classes using union, intersection and complement. For example,

we may wish to define the class person to be the disjoint union of the classes male and female. RDF Schema does not allow such definitions.

- Cardinality restrictions: Sometimes we wish to place restrictions on how many distinct values a property may or must take. For example, we would like to say that a person has exactly two parents, and that a course is taught by at least one lecturer. Again such restrictions are impossible to express in RDF Schema.
- Special characteristics of properties: Sometimes it is useful to say that a property is transitive (like "greater than"), unique (like "is mother of"), or the inverse of another property (like "eats" and "is eaten by").

More sophisticated ontologies can be created with Ontology Web Language (OWL). As shown in figure 1.1, the OWL is a language derived from description logics, and offers more constructs over RDFS. It is syntactically embedded into RDF, so like RDFS, it provides additional standardized vocabulary. OWL is basically for defining and instantiating Web ontologies. An OWL ontology may include descriptions of classes, along with their related properties and instances. OWL is designed for use by applications that need to process the content of information with more reasoning, with deeper inferential power, and more. It facilitates greater machine interpretability of Web content than that supported by XML, RDF, and RDFS by providing additional vocabulary along with formal semantics. The three language versions of OWL in order of increasing expressiveness OWL are - OWL Lite for taxonomies and simple constrains, OWL DL for full description logic support, and OWL full for maximum expressiveness and syntactic freedom of RDF. In chapter 2 we will discuss the different levels of formalism and expressivity of semantics in information modeling.

RDFS and OWL have semantics defined and this semantics can be used for reasoning within ontologies and knowledge bases described using these languages. To provide rules beyond the constructs available from these languages, rule languages are being standardized for the Semantic Web as well. Two standards are emerging - RIF and SWRL. For querying RDF data as well as RDFS and OWL ontologies with knowledge bases, a Simple Protocol and RDF Query Language (SPARQL) is available. SPARQL is SQL-like language, but uses RDF triples and resources for both matching part of the query and for returning results of the query. Since both RDFS and OWL are built on RDF, SPARQL can be used for querying ontologies and knowledge bases directly as well. Note that SPARQL is not only a query language, it is also a protocol for accessing RDF data.

Formal proof together with the trusted inputs for the proof will mean that the results can be trusted, which is shown in the top layer of the figure 1.1. For reliable inputs, cryptography means are to be used, such as digital signatures for verification of the origin of the sources. The specifications and the implementation behind this layer is still undetermined. On top of these layers, applications with user interface can be built.

1.3 The Challenges of Realizing the Semantic Web

The need for intelligent retrieval, manipulation, and presentation of information underlies the importance of realizing the Semantic Web. The prime focus of the semantic web is sharing of semantically interoperable data, and the integration of heterogeneous data sources. The machine understandability of web documents requires the creation of meta-data (data about data) i.e. annotating documents. Accurate meta-data can only be derived from a well-modeled domain ontology based on deep understanding of the domain, which is a human-intensive effort, and is the major reason why the progress of the Semantic Web is slow. Thus, in this context, the challenges of the semantic web is summarized into:

- The problem of ontology repository. In the early days of the semantic web there were very few real domain ontologies available and a large share of ontologies published on the web were outdated, dead collections created in some academic research context as was discussed in [7]. As of 2015 with the urge for domain knowledge and the maturity of ontology standards and development tools, ontologies are being developed more rapidly and professionally, but highly dependent on domain expert, and most are developed for a specific usage. Some are very detailed and complex, in general, more complex ontologies tend to be more difficult for a human to comprehend, therefore more difficult to be maintained and reused. [8].
- The lack of the existence of common vocabulary for ontologies, and therefore for semantic annotation for a specific domain is the main cause of the interoperability, integration and reusability of ontologies.
- The ontology description changes overtime, which leads to the need of ontology maintenance and annotation regeneration. The manual effort of ontology maintenance and annotation regeneration is practically infeasible. One needs

domain ontologies that are collaboratively maintained and annotations that are automatically updated.

We propose a solution that focuses on solving the data integration and interoperability problem. In addition, to address the ontology development challenge. We develop a framework that is based on the Wikipedia knowledge. Besides creating the ability to automate some of the phases of ontology engineering, our goal is to simplify the construction of domain ontologies that is sufficient for the purpose of annotation.

1.4 Motivation

Let's consider this query presented to Google search engine "Free appointments with best 5 Neurologists within the range of 15 miles" Google retrieved about 6,190,000 results. The first 10 results have nothing to do with the geographic location. Assume that a web application is developed to be responsible for the task of appointment scheduling with physicians, figure 1.3 illustrates a high-level architecture of such an application.

Semantic applications involve multiple data sources, with different access points. Thus in order to retrieve the required information from each data source, there has to be a common communication layer. Thus data sharing between heterogeneous data sources is a primary challenge in semantic web.

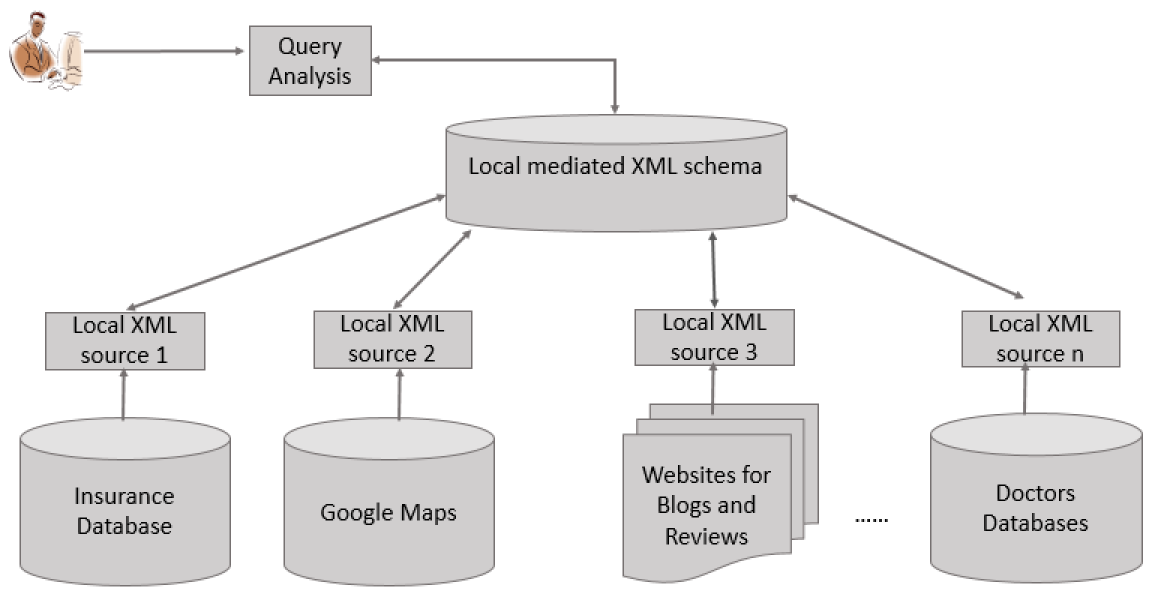


Figure 1.3 Data Integration problem for semantic applications

There are multiple types of heterogeneity of data resources [9], which can be classified to:

- Syntactic heterogeneity which is a result of the differences in representation format of data.
- Schematic or structural heterogeneity, and this refers to the native model or structure to store data which differ in data sources.
- Semantic heterogeneity which is due to the differences in interpretation of the 'meaning' of data.
- System Heterogeneity: use of different operating system, hardware platforms lead to system heterogeneity, which is out of the scope of this study.

An ontology-based solution for the problem of integration of heterogeneous data resources is illustrated in figure 1.3.

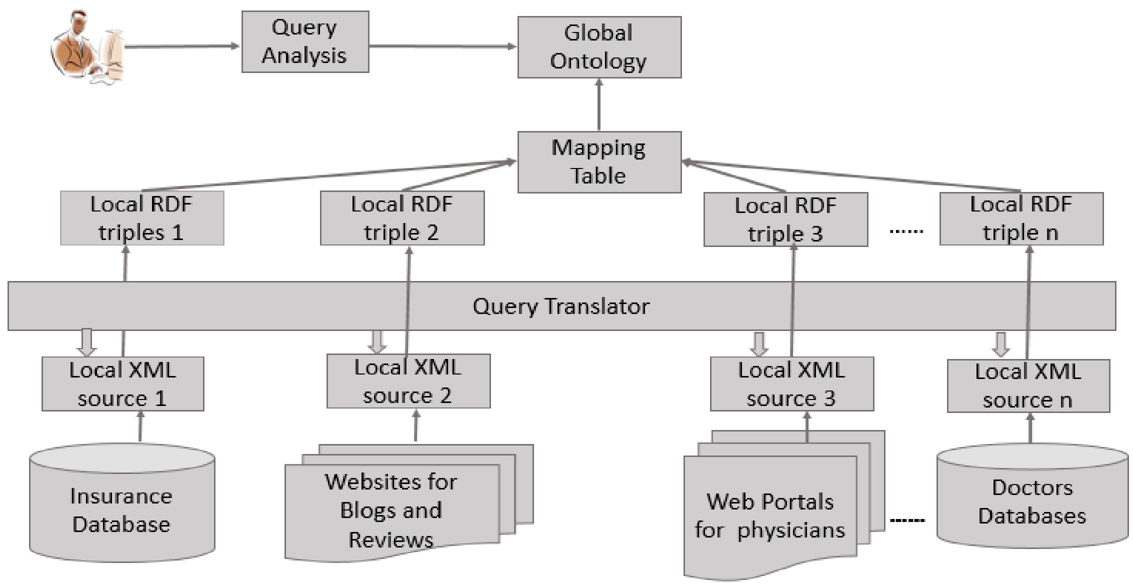


Figure 1.4 Ontology-based solution for heterogeneous data integration

Integration layer is developed using the Semantic Web standard formats, hence overcoming the syntactic, and the schematic heterogeneity.

The global ontology is designed on the conceptual level, then the query can be formulated from a conceptual perspective without considering the structure of the source with a global definition of entities, which in turn solves the semantic heterogeneity. "Ontologies are (meta) data schemas, providing a controlled vocabulary of concepts, each with an explicitly defined and machine-processable semantics. By defining shared and common domain theories, ontologies help both people and machines to communicate concisely, supporting the exchange of semantics and not only syntax" [10]. Hence the creation of ontologies to realize the Semantic Web demands the existence of a global semantic definition of concepts in a given domain, and at the same time interoperable with multiple domains. The creation of a global semantic definition of domain ontologies and the modeling of an upper schema that is sufficient to describe various domains is part of our contributions in this thesis.

1.4.1 Ontology Development

Formal domain knowledge in general and ontologies in particular are considered the building blocks of the Semantic Web. At the beginning the lack of domain ontologies and the lack of ontology repository was the main restriction for speedily realization of the Semantic Web. The development of a formal ontology with all its construction phases requires a domain expert with knowledge in ontology modeling, which makes it a very expensive, and slow task. Also fully automating this process is a problem that cannot be solved in the near future. On the other hand, the automation of some phases of this process has been the interest of ontology and linguistic researchers.

1.5 Overview of the Proposed Contributions

Analyzing semi-structured data for information extraction has been a promising direction for generating robust knowledge systems throughout the past decade. The purpose of this research is to study the problems of creating semantic annotation, ontology development, interoperability and integration. We propose a solution that leverages a mature, well-organized and informative semi-structured data. Basically we use the structure of the Wikipedia for defining and modeling concepts that are required to be annotated. It is definitely impossible to create one model that will work with all domains, so we create a template structure for building an annotation model that is based on the Wikipedia structure and then, we build a Wikipedia based semantic schema in the form of an ontology that is an extension of the Simple knowledge Organization System (SKOS) [11] annotation model. We refer to the proposed semantic schema as Skos-Wiki. Hepp et al. discussed the idea of using the Wikipedia pages as URI for defining concepts in ontologies [12]. In our research, we consider this schema a Wikipedia-based standard for annotation. We will use it as the infrastructure for knowledge representation upon which we build our framework for generating semantic annotations. We propose a framework for building lightweight ontologies for semantic annotations. Lightweight ontology: is defined as the ontology that does not have to include all the components that could be expressed with formal languages, such as concept taxonomies, formal axioms, and disjoint and exhaustive decomposition of concepts [13].

The ontologies we consider in this research are all meant for the purpose of annotation. Lightweight ontologies are enough for annotating general domains, which is a basic foundation for semantic interpretation and disambiguation of domain entities and their intended context. However manual enhancement of the ontology through the addition of axioms, rules, disjoint sets, etc., is possible for future reasoning purposes.
A software framework is a universal, reusable software platform to develop software applications, products and solutions. Software frameworks include programs, code libraries, tool sets, and application programming interfaces (APIs) that bring together all the different components to enable development of a project or solution.

We define the word framework in this context as an abstraction in which we provide generic functionalities needed in the domain of semantic web. These functionalities can selectively be changed by additional user-written code, thus providing application-specific software. Figure 1.5 illustrates the main structure of the proposed framework. We are developing a framework that consists of a set of modules that contribute in automating the construction of lightweight ontologies. We base most of our knowledge on the structure of the Wikipedia, which represents the hierarchical links between categories and links between pages, in addition to the content.

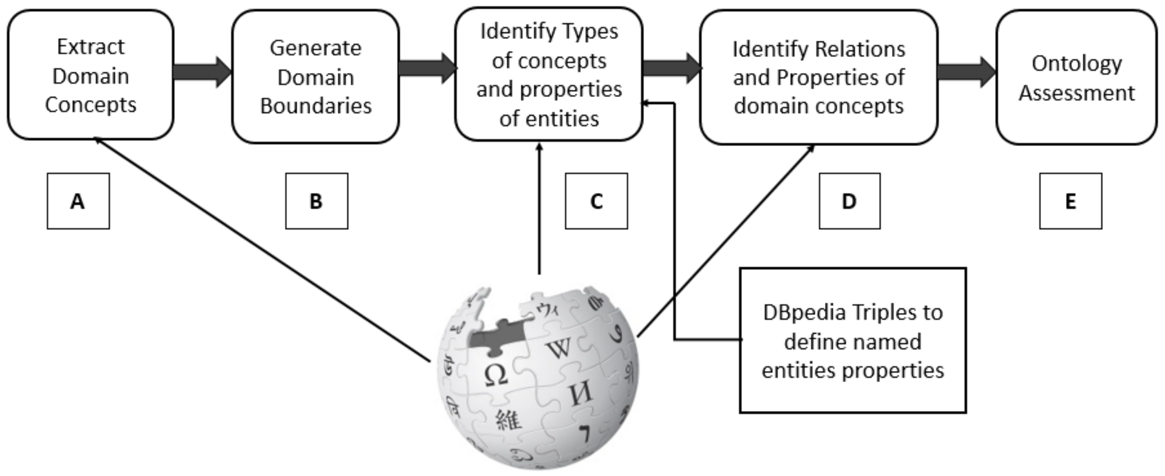

Extract Domain Concepts
Generate Domain Boundaries
Identify Types of concepts and properties of entities
Identify Relations and Properties of domain concepts
Ontology Assessment
A
B
C
D
E
DBpedia Triples to define named entities properties

Figure 1.5 Main modules for building lightweight ontologies.

The above figure represents the framework as a set of modules. In Module A; we start with a few selected concepts in the domain of interest, and consider them as the seed concepts for the ontology. Then we expand our concepts space by identifying related concepts based on exploring the Wikipedia link structure. In Module B, we generate the domain and subdomain boundaries by using hierarchical clustering of the concepts in the extended concept space. In Module C, we classify the generated terms and concepts into wiki concepts and named wiki entities according to the Skos-Wiki description. We use the DBpedia triples to perform named entity recognition. In module

D, we extract hierarchical concept relations by exploring the category graph for the Wikipedia. Finally in module E, we will evaluate the extracted ontology by comparing it to some of the mature existing topic maps that are created by a domain expert, and will conduct both manual and expert evaluations. The framework we proposed adopts a Wikipedia-based annotation schema to model the extracted concepts, which is then ready to be used to markup web documents.

1.5.1 Wikipedia Structure

The reason we adopted Wikipedia for building annotation model and extracting domain ontology is that it has good coverage. The articles have good quality, which makes it possible to apply natural language techniques reliably to extract concept relations. The English Wikipedia as for 2014, has over 4.5 million articles, which means it carries information for 4.5 million concepts, with linkages. The Wikipedia basically consists of text articles that are classified into different categories. The categories are arranged into a network structure with parent-child relationships. A category can have multiple parents and also multiple children.

The Wikipedia category structure can be viewed as a directed acyclic graph. An article is the primary construct of Wikipedia, containing details of the term to be described. The first paragraph usually provides a brief definition of the term, while the full article contains more details, history, relevant links to other information sources, etc. Article titles in Wikipedia contain mainly nouns, with only a few verbs and adjectives. Each Wikipedia article belongs to one or more categories and each article has in-links and out-links. The in-links represent the hyperlinks from other articles pointing to an article, and out-links represent the hyperlinks pointing from an article to other articles. Wikipedia has several advantages for the development of domain-specific ontologies, such as a wider coverage, very specific technical terms, large number of proper names (persons, places, books, products etc.) and more up-to-date content, than any other knowledge base [14]. Some Wikipedia articles include an Infobox section. The Infobox in a Wikipedia article is like a fact sheet that summarizes important facts about the article. Usually Infobox templates differ depending on the kind of topic the article covers. The information in an Infobox can be considered as a set of subject-attribute-value triples summarizing the key aspects of the article's subject [15].

1.5.2 Wikipedia information extraction

There have been several attempts to extract semantic information from the Wikipedia. Most of these were focused on generating gazetteers, classifying documents, word sense disambiguation, extracting relations between concepts, building semantic networks, extracting ontological relations, annotating documents with Wikipedia concepts, and lately semi-automatic development of frame nets. For the scope of this research, we will focus on the efforts made to use Wikipedia for finding domain entities, relationship extraction, and building ontologies for document annotation. The future scope of this research is extending the current schema to allow for building semantic lexical templates to define general semantic relations, and in turn the development of lexical frame nets. Vivaldi and Rodríguez developed a method to extract domain concepts from Wikipedia [16]. They used Wikipedia category corresponding to domain name, they retrieved all documents and categories under main concept forming two lists, and recursively iterate over pages, and categories assigning a score to each node based on the linkage between the first level, and the subsequent levels.

In 2007 Ruiz et al. made one of the first attempts to automate the extraction of certain types of relations from the Wikipedia [17].They were able to use the Wikipedia disambiguation pages and redirects to extract hyponymy (isA), hyperonomy (not-isA), holonymy (has part) and meronymy (part of). He also used several Natural Language Processing (NLP) techniques to extract certain lexical patterns from Wikipedia. Extracting ontological relations in the form of triplets was first introduced in [18] when they parsed articles' text together with the syntactic constructs of the Wikipedia to build the DBpedia. The DBpedia provides information about 1.8 million thing in the form of RDF triples. However the extraction method depends solely on parsing syntactic constructs. This method suffers from low article coverage, only 44% of the Wikipedia articles have info boxes, and the loss of important relations hidden in the structure and the text. Also there is not any method where domain knowledge together with its concepts and relations can be clearly identified. A different approach for extracting RDF triplets was developed in [19], where the authors used super-sub category pairs for triple extraction. Based on category structure in certain domains, they induced patterns of categorical relations that minimized the manual effort. Document annotation with Wikipedia concepts was presented in [14], where the author used Wikipedia page titles, and category names for annotating concepts in the text. The paper presented a technique for generating semantic summary for web documents using Wikipedia concepts. Wikipedia is used as a resource and the weight of each word is calculated using the Term Synonym Concept Frequency-Inverse Sentence Frequency (TSCF-ISF) measure. Sentences are ranked based on the scores they have been assigned and the summary is formed from the highest ranked sentences.

Our proposed model adopts the Wikipedia in several dimensions, as a semantic schema, as semantic network, with interlinks between domain concepts, and as a triple store for defining entity type and relations. Our contribution to the state of the art is:

- We design a Wikipedia-based unified annotation model that extends SKOS, and inherits DBpedia definition for named entities. Thus, we provide an interoperable multi-layered schema for the knowledge extracted from the Wikipedia. This schema acts as a global ontology for mapping different semantic structures.
- We adopt the Wikipedia link structure to extract domain concepts, and compute the relatedness between the concepts. Hence we rely on a widely approved knowledge base to build domain boundaries, which is a reliable method for building lightweight ontologies that reflect the intended context. In chapter two we discussed how this is enough for the purpose of annotation in most of the general domains.
- We implemented a hierarchical clustering-based method to extract subdomains in the extended concept space. Our method is based on internal measures, and Inconsistency of merging heights to automate the estimation of the optimal number of clusters, which in turn defines the number of subdomains, and the distribution of concepts.
- We used SPARQL queries over the DBpedia triple store for two things: named entity recognition, and the extraction of entity properties. We mapped the results of the

DBpedia to the Skos-Wiki semantic model. This approach is considered a method for ontology mapping between the definition of entities in DBpedia, which is considered inconsistent in lot of cases and context independent, to a global context-based ontology.

1.6 Thesis Outline

This thesis is organized in a way that every chapter focuses on a part of the above contributions. Chapter 2 discusses the topic of semantic annotation, and its challenges. The need for standard annotations and ontology repositories, to build a semantic layer, and the concept of annotation modeling will be explained, together with a brief survey of annotation schemas in literature. The benefits of extending existing annotation model versus developing new one will be highlighted, along with a description of some widely-accepted annotation schemas. Finally, we will present our proposed Wikipedia-based annotation model as an extension to the SKOS schema.

Chapter 3 will present approach for extracting domain concepts and the role of clustering in defining the domain boundaries. We will survey, in brief, the different clustering techniques, and the parameters upon which to define the suitable clustering technique. Different measures for semantic relatedness are presented. Then we will present the Wikipedia-based methodology for measuring semantic relatedness. Chapter 4 focusses on the problem of recognizing of named entities in the extracted concepts using Dbpedia ontology and Sparql queries to extract Dbpedia entities and their properties.

2 Ontology-based Semantic Annotation

2.1 Introduction

The term annotation in a general sense means adding metadata to describe data. Annotations can describe information resources (document, image etc.) or parts of an information resource, such as adding metadata to describe the content of different parts of a document, or tagging image objects. Semantic annotation is the process of adding formal metadata to web pages. This metadata links instances in a web page to defined concepts (classes) in an ontology. Ontologies are considered a formal semantic model that describes concepts in the documents to be annotated [14].

The process of semantic annotation involves three steps. First, an ontology creation process to obtain ontology that describes the domain of interest. Second, a data instance recognition process that discovers all instances of interest in the target web documents based on the defined ontology. Third, an annotation generation process that creates a semantic meaning disclosure file for each annotated document. Through the semantic meaning disclosure file, any ontology-aware machine agent can understand the target document [20]. Currently, the automation of entity recognition and annotation generation have been highly advanced and a survey for the state of art appears in [21] and [22]. However, the main challenge in the annotation process is the creation of domain ontologies.

Ding stated in [20] that "to ensure machine understandability web annotations have to be: explicit (makes an annotation publicly accessible), formal (makes an annotation publicly agreeable), and unambiguous (makes an annotation publicly identifiable)". According to Ding's statement the challenge of semantic annotation in a nutshell is due to the lack of standardized annotation model, and the inconsistencies and the differences in the definition of domain concepts and relations, in addition to the difficulty of the maintainability of domain knowledge (ontology). In short, poor and lack of annotations are the reasons for the deferment of the wide realization of the semantic web. Our scope, in this chapter, is focused on the development of semantic annotation model as a template for building a framework for semi-automated construction of domain ontologies that will be used in the annotation process. Formal annotation requires an annotation model that will define the specification of the annotation elements. The role of an annotation model is to ensure semantic interoperability. The Semantic interoperability, as defined in the European Interoperability Framework (EIF) [23], states that "Interoperability is concerned with ensuring that the precise meaning of the exchanged information is understandable by any other application that was not initially developed for this purpose. " Creating the definition of the knowledge base that will model the specifications of entities and concepts is a crucial step for information extraction, analysis and understanding semantics of the underlining domain. We are concerned here with the ontology modeling for semantic annotation. That is, a model to define the vocabulary that is semantically sufficient for the purpose of annotation for the semantic web is desired.

When developing an annotation model for semantic application we should address the following challenges:

- Interoperability with other annotation schemas: thus, we need to rely on an upper ontology that is generic enough for multiple domains.

- Lack of domain ontologies: thus, we need to design a semantic schema that can be instantiated in multiple domains in an automatic or semi-automatic fashion.
- Usability, and maintainability: thus, we should keep it simple.

In this research, we present an annotation schema that relies on the structure of the Wikipedia semantic network. The goal is to build a semantic schema that represents a template adapted from the Wikipedia structure to facilitate the learning of lightweight ontologies by extracting corresponding knowledge through the mining of semantics from the Wikipedia. Extending an upper ontology for building an annotation schema provides a controlled vocabulary and interoperable schema for a conceptual-level representation of domain knowledge, thus we represent the Wikipedia annotation schema as an extension of the SKOS – The Simple Knowledge Organization System- which allows for the definition of web concepts, taxonomies and thesaurus but cannot represent relations other than hierarchical relations. We extend the SKOS to accommodate different entity types and still be generic enough for wide range of domains. We named the new model Skos-Wiki.

The next section will analyze various dimensions of semantic modeling in order to define the characteristics of modeling lightweight ontologies for semantic annotations. Then, we will present different types of annotation schemas, especially the structure of the SKOS as an upper ontology. Then, the proposed annotation model will be presented, followed by a discussion on the impact of the modeled schema on building a semantic web framework for constructing interoperable domain ontologies.

2.2 Background

2.2.1 Dimensions of semantic modeling

In the light of the challenges discussed above, we will study various dimensions to consider when modeling information for building a domain ontology, in general, and annotation ontology specifically.

Level of expressivity

Level of expressivity of a model, is the amount of semantics we decide to embed in an ontology when modeling a given domain. Expressivity controls the level of semantic details to define for a concept, including properties, relations and logical assertions and axioms. We differentiate here between a term and a concept. A term is syntax (e.g., a string) that stands in for or is used to indicate the semantics. A concept is a unit of semantics (meaning), a node (entity) or a link (relation) in the mental or knowledge representation model [24]. When building an ontology, a concept is the primary knowledge construct, which is typically a class, relation, property, or attribute, and is generally associated with logical rules. Expressivity of an ontology has always been explained by the semantic spectrum. The first reference to the notion of semantic spectrum was at the National Conference on Artificial Intelligence the AAAI 1999. Since then, there have been several authors who adapted various aspects of this concept with different perspectives. The ontology spectrum is a type of semantic spectrum. Figure 2.1 demonstrates the different levels of semantic expressivity in ontology modeling. Expressivity level of ontologies ranges from an abstract taxonomy of concepts (with minimal parent-child relation) to a thesaurus of terms (with broader-than, narrower-than relation and association relation), to a conceptual model (concept hierarchy with generalized relations, properties and attributes) and, finally, to a logical theory. A logical theory is a semantic model focusing on real world semantics, extended with axioms and rules.

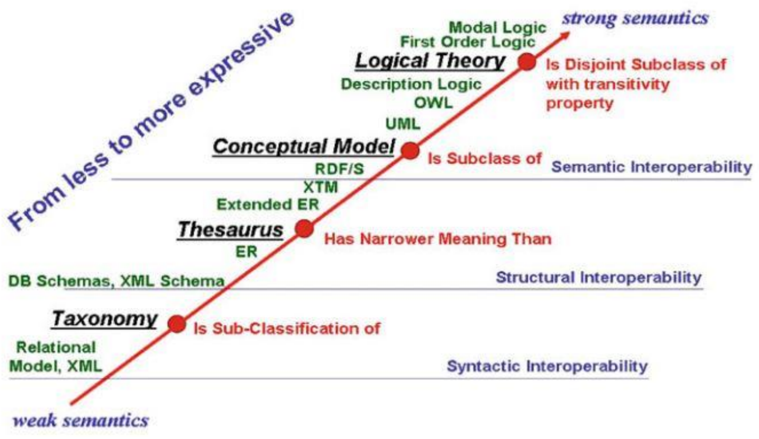


Figure 2.1The Ontology Spectrum [24]

Lightweight ontologies are more expressive than a conceptual model and less than logical theory. Lightweight ontologies would then typically consist of a hierarchy of concepts and a set of relations between those concepts [25]. Conversely, heavyweight ontologies add cardinality constraints, standalone axioms, reified statements (a statement about a statement) and more. In order for a knowledge representation system to be called an ontology it has to be defined with a logical language based on descriptive logics (RDF, RDFS, OWL). RDF and OWL are designed for systems in which data may be widely distributed (e.g., the Web). As such a system becomes larger, it becomes both impractical, and virtually impossible to know where all of the data in the system are located. Therefore, one cannot generally assume that data obtained from such a system are complete. Thus it is almost impossible to rely on the validity of reasoning and inferences in such an environment. This assumption, is known as the open world assumption [26]. On the other hand, a highly detailed schema (High expressivity) will raise the capabilities of the semantic inference and the usage of the ontology, but will definitely influence the computability of the inference and reasoning process negatively. Moreover, highly expressive ontology, with fine details, will be hard to maintain and the intense human effort involved in the construction of a highly expressive ontology, will slow down the development of domain ontologies as the definition of the semantic details of heavyweight ontologies cannot be automated.

Level of granularity

Semantic granularity addresses the different levels of specifications of an entity in the real world [27]. In general, the ontology should not contain all possible information about the domain. We should not specialize (or generalize) more than we need for the application under consideration. Defining the scope of the ontology and the usage helps in specifying the proper level of granularity. Major granularity levels, as presented in figure 2.2, are upper level ontology, mid-level ontology and domain ontology. Upper and mid-level ontologies are sometimes referred to as the foundational ontology. Upper ontology is a generic ontology describing general knowledge with concepts that are shared with most domains, such as time and space. The main usage for upper ontologies is to provide semantic interoperability and guide the development of domain ontologies.

A domain ontology, describes a domain, such as the medical domain or the electrical engineering domain, or narrower domains, such as the gene ontology. Upper ontologies are used to define foundational concepts in domain ontologies. Ontology mapping becomes easier if domain ontologies are derived from the same upper ontology.

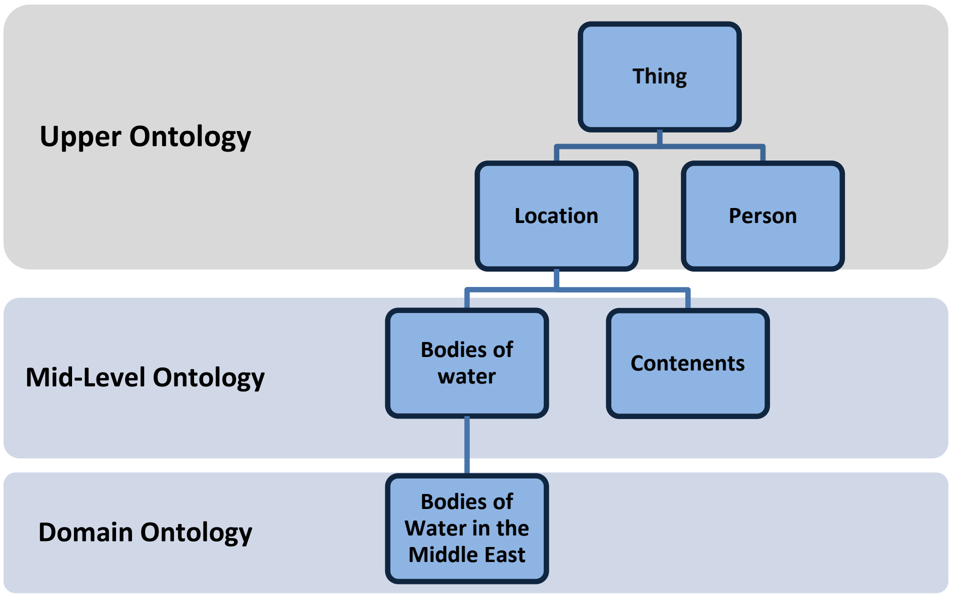


Figure 2.2 Major granularity levels

Semantic Interoperability

Semantic Interoperability is the ability of systems to exchange data, with other, new applications, that was not initially developed for use by such applications, while ensuring that the intended meaning is conveyed unambiguously and consistently. Lack of interoperability is due to heterogeneity conflicts. The community of distributed databases has given this subject a special attention and defined several taxonomies for defining conflicts, as in [28], [9] and [29]. We are concerned here with semantic heterogeneity, which considers the content and the meaning of an information item.

Park and Ram (Park and Ram 2004) summarized the types of semantic conflicts and they differentiated between semantic conflicts on the data level and the schema level. The resolution of these conflicts requires the understanding of the domain; this is where ontologies come in the picture. A multi-level ontology standardizes the definition of datatypes, on the upper level, named-entity properties and concept properties on the mid-level, and domain specification on the lower level.

Ontology Learning and Maintenance

When building an ontology schema, especially for the purpose of annotation, we have to keep in mind that to ensure the effectiveness of this model, we should be able to instantiate it in various domains to cover the diversity of knowledge especially on the web. Two important points have to be considered: How to simplify and accelerate domain ontology learning using data mining techniques, and How to ensure that the induced ontology accurately reflects the updates in the knowledge and facts, i.e. they are not obsolete.

2.2.2 Semantic Annotations

Annotation Schemas are designed to provide common vocabulary for annotating. We can classify annotation schemas into two types: metadata annotation schemas, and ontology-based annotation schemas. There are several annotation schemas in the literature for both types. However, we will focus on those that had played a part in the evolution of web semantics and are widely used.

2.2.2.1 Metadata Annotation Scheme

These are lists of vocabulary terms to annotate resources. The main goal of these vocabularies is to enhance the retrieval of resources on the web.

Dublin Core

The idea of core metadata for simple and generic resource descriptions was first discussed early in Dublin Core workshops. The fifteen-element "Dublin Core" achieved wide propagation as part of the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH). It has been approved as IETF RFC 5013, ANSI/NISO Standard Z39.85-2007, and ISO Standard 15836:2009. Dublin Core conforms to the W3C XML Schema 1.0. The Dublin Core standard originally includes two levels: Simple and Qualified. Simple Dublin Core comprised 15 elements; Qualified Dublin Core included three additional elements (Audience, Provenance and RightsHolder), as well as a group of element refinements (also called qualifiers) that could refine the semantics of the elements in ways that are useful in resource discovery. Since 2012 the two have been incorporated into the DCMI Metadata Terms as a single set of terms using the Resource Description Framework (RDF).

Schema.org

Schema.org is a joint initiative of the search engines Google, Bing, Yahoo and Yandex aimed at making it easier to index web pages in a way that facilitates the enhancement of search services. Schema.org provides a collection of shared vocabularies that webmasters can use to mark up their pages in ways that can be understood by the major search engines. Schema.org development is very pragmatic and terms are created as needed. The terms cannot be used for domain-specific classification, like in the biology domain or the genes domain. Also, it is not meant for extracting any semantic relations or inferences. Terms and properties can be added, but the schema cannot be extended.

Schema.org and Dublin core made a significant shift in the area of annotation, which, in turn, allowed for advanced search services and enhanced the capabilities of search engines in resource finding. Although they were the standard for annotation for the past decade, they lack the uniqueness and granularity properties, which make them insufficient for use within a specific community or a specific domain. Without the above standards, search applications as Google snippets and Yahoo Monkey would not have been possible. These annotations make search engines look smarter. The simplicity of these schemas made them popular and widely used. However, sophisticated semantic applications such as context extraction, information analysis, or extracting semantic relatedness are not possible with the metadata-based annotation.

Extending the structure of the vocabulary lists of Dublin core and Schema.org has to follow their general theme and adhere to the main purpose they were created for. Adding any semantic structures or ontological relations will contradict the original purpose, which is to enable simplicity of annotations and usages. These standards were made for document annotation, but not for concept annotation, and one can just extend the vocabulary list but not the semantic schema.

Ontology-based Annotation Scheme

Most ontology-based annotation efforts were domain-specific in the sense that they rely on an expert developed ontology to be used for annotating domain concepts. Some of these ontologies were instance of a generic upper ontology, such as SKOS; but, most were not related to any common structure or upper design. We are concerned with generic ontologies that can be extended and instantiated in various domains.

SKOS – Simple Knowledge Organization System

SKOS is an OWL ontology to model knowledge organization systems such as thesauri and classification schemes. An individual knowledge organization system can be used in many different information systems. By defining the SKOS data model as an OWL ontology, the Semantic Web can then be used as a medium for publishing, exchanging, sharing and linking data involving these knowledge organization systems. SKOS allows for web definition of terms, taxonomies and thesauri for referencing and reuse on the semantic web. The history of the development of SKOS from 1990 to 2009 can be referenced in [30]. The SKOS is a lightweight upper ontology that can be instantiated in various domains. It follows the principle of making minimal ontological commitment as stated by Thomas Gruber [31] : "An ontology should require the minimal ontological commitment sufficient to support the intended knowledge sharing activities. An ontology should make as few claims as possible about the world being modeled, allowing the parties committed to the ontology freedom to specialize and instantiate the ontology as needed," which asserts that the simpler the ontology specification the easier it is to extend and instantiate it. SKOS is developed as an OWL ontology. The SKOS data model and the design choices made in modeling those components is explained in details in [30].

The main entities in this schema are, “Concept”, “ConceptScheme”, “Collection” and “OrderedCollection”. Relations and properties are designed to satisfy the specification of taxonomies and thesauri. Relations are either semantic hierarchical relations, such as “broader”,”narrower” and”related,” or mapping relations to define synonyms an equivalent concepts, such as “broadMatch”, “narrowMatch” and “closeMatch”. Properties are either labeling properties or documentation properties. Concepts can be identified with URIs, labeled with lexical strings assigned notations, documented with various notes, linked to other concept schemes, grouped and labeled into a collection, and mapped to concepts in other schemes.

We are proposing a Wikipedia-based semantic annotation model, as the mid-level schema that instantiates the SKOS and extends the semantic capabilities far more than thesauri and taxonomies.

DBpedia

DBpedia is a community effort to extract knowledge from the Wikipedia and make it available in the form of RDF triples [18]. DBpedia represents an RDF repository for the info-boxes in Wikipedia pages. Info-boxes are like fact sheets for named-entities. A DBpedia ontology was then constructed to detect inconsistencies in the definitions of similar named-entities [32]. In this chapter, we will be linking named-entities in the Wikipedia annotation model to their corresponding definitions in DBpedia. We will be modeling the DBpedia in an abstract way, but more details about classes and concepts will discussed in chapter four.

2.3 Skos-Wiki: Wikipedia-based Semantic Annotation Model

If we can extract concepts and relations within a given domain, then we can build an ontology. If we can represent this ontology with a controlled vocabulary, then it can be used for annotating documents that belong to the same domain. We propose an annotation model, named Skos-Wiki, and it is based on studying the structure of the Wikipedia knowledge and deriving a template that we use for extracting lightweight ontologies for most of the general domains.

Analyzing semi-structured data for information extraction has been a promising direction for generating robust knowledge systems throughout the past decade. In this research, we will benefit from the structure of the Wikipedia in developing a mid-level annotation schema that extends the upper level ontology SKOS and acts as a template for constructing domain ontologies that will be constructed using the knowledge in the Wikipedia. By extending SKOS, we follow a top-down approach for ontology development, which it is believed will speed up the learning of the domain ontologies and enhance their quality and interoperability.

The annotation elements defined by these models are able to annotate documents, to facilitate the location of resources and to relate one document to another. Basically, we use the structure of the Wikipedia network and the template within the Wikipedia pages to define different types of concepts for building an annotation model. This model will be used as the interoperable infrastructure for the knowledge representation upon which we build our framework for generating semantic annotations.

2.3.1 Wikipedia-based Annotation Model

As mentioned in the previous section, Wikipedia articles are the main constructs and each article provides the description for one of the Wikipedia concepts. This schema models concepts that are represented in the Wikipedia categories or articles. The model was designed in OWL. A schematic description of the main entities is shown in figure 2.3.

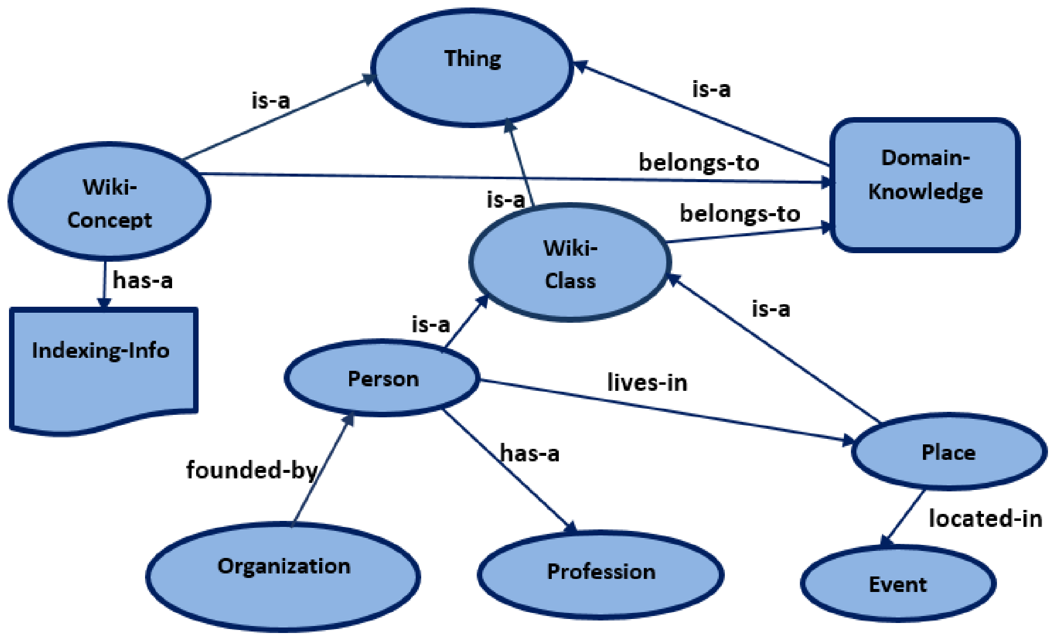


Figure 2.3 Wikipedia based annotation model

In the SKOS-Wiki model, we divide the concepts that are covered in Wikipedia articles as either a Wiki-Concept or Wiki-Class. A Wiki-Concept is an abstract concept that cannot be instantiated or enumerated but can be used for categorization such as the concept “Engineering”, “politics”, “computer science”. A Wiki-Class is a type that can be instantiated with an Is-A relation e.g. person, place organization and event, or any class that can be listed or enumerated, e.g. “tropical storms”, “lung diseases”. This distinction made it easier to associate specific relations and properties depending on the type of entity.

We consider the Domain Knowledge as the major category that describes the concepts of the annotation model. A group of wiki-concepts and entities Belongs-to Domain Knowledge. A Wiki-Concept or a Wiki-Class can belong to multiple domain knowledge depending on the context. For example, “Bill Gates” as an entity might belong to a knowledge domain as one of the ten wealthiest people in the world, and belong to another knowledge domain as the founder of Microsoft corp. This aspect allows for faceted browsing of concepts. Any relation can be defined between two wiki concepts, two wiki classes, or a concept and a class or an instance of a class. In OWL there are primitive built-in relations, such as the owl:subclass relation, and rdf:typeOf relation [6].

In OWL, in order to define any relation we need to define an object property. In this model, we define some basic relations that will fit for all domains as Lives-in between a person and place, Has-location between event and a place, etc. More relations are mentioned in the OWL definition of the model. The entity “IndexingInfo” is an entity that contains indexing information that best defines a Wiki-Concept. The Wiki model also includes a set of descriptors that are designed to define Wiki-classes and concepts, which are Location-descriptor, Person-descriptor, Event-descriptor, Organization-descriptor and Index-descriptor. The Index-descriptor includes important information such as synonyms which are terms that have the same meaning, keywords that represent common search terms for this concept, and ambiguous terms or homonym terms. Index-descriptor also includes highly related terms, which are terms that are frequently mentioned together but the relation cannot be easily extracted such as “Global warming” and “Greenhouse effect”. A descriptor acts as an annotation template that includes properties /slots that can define a “Wiki-Class”, or a “Wiki-Concept” based on the Wikipedia definition for this entity. Descriptors are inherited from the DBpedia ontology. The DBpedia ontology is a community-provides mappings that help normalize the variations in the properties and classes defined in the Wikipedia as Info-boxes [32].

2.3.2 Extending SKOS: The Multi-level Wikipedia Semantic model

In order to build a multi-level semantic model, we extend the SKOS ontology. A multi-level ontology enhances interoperability by resolving heterogeneity. The previously defined Wikipedia schema acts as the mediator between SKOS as an upper ontology, Wikipedia definition of concepts and DBpedia definition of named-entities. Figure 2.4 illustrates the multi-layered definition of the proposed semantic model. classes according to the Wikipedia structure, SKOS schema and DBpedia definitions of named entities.

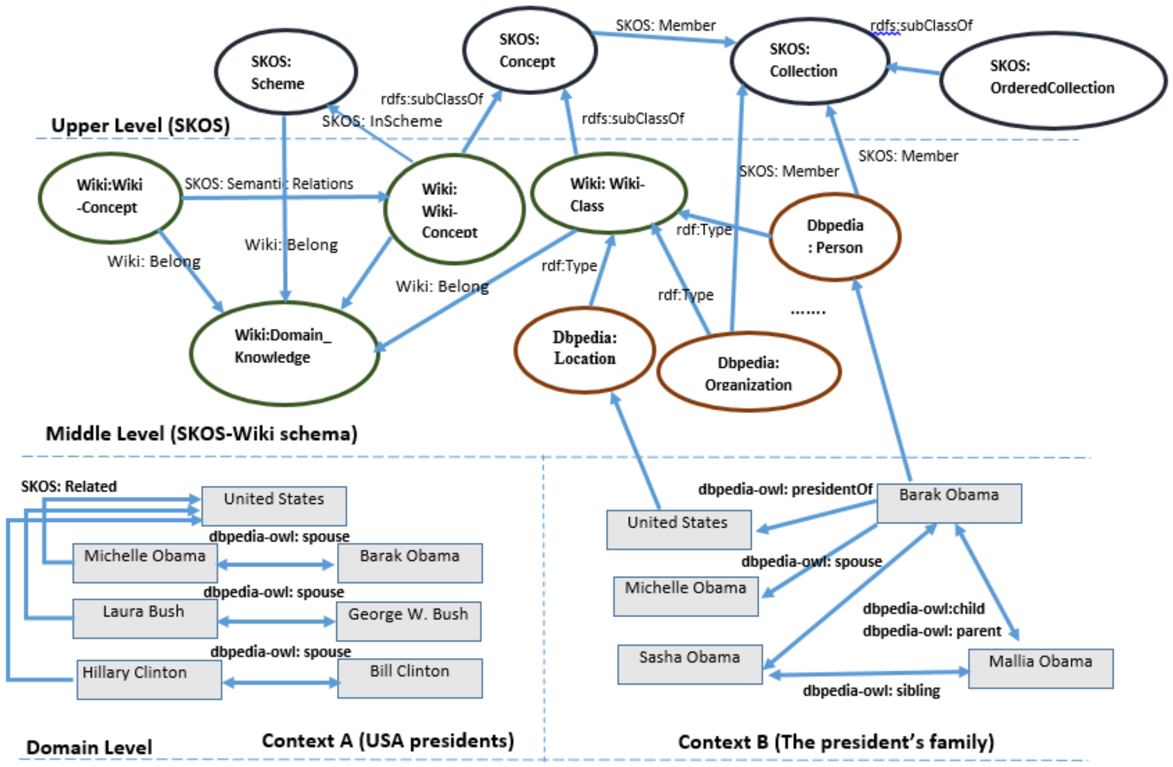


Figure 2.4The general structure of the multi-layered annotation model Skos-Wiki

As mentioned earlier, SKOS is a standard way to describe thesauri and other sets of terms for the semantic web. We extend the SKOS definition of a "SKOS:Concept" to include "Wiki-Concept" and "Wiki-Class", i.e. Wiki_Class and Wiki_Concepts are subclass of Concept. The multi-layered model enables different level of abstraction for domain annotations. The middle level SKOS-Wiki schema maps the definition of concepts and This model represents the backbone information structure of a framework for Wikipedia-based, semi-automatic information extraction and construction of a domain ontology. The domain level layer presents a sample of extracted domain ontologies, where the same entities are shared between both ontologies. For example, the entity

Barak Obama is mentioned in both ontologies with different contexts, in context A as one of the US presidents and in context B to show the president's family.

2.4 Conclusion

When modeling an ontology for building annotations, the expert point of view will not help in the context of meta-data annotations. A highly constrained and detailed schema will raise the complexity of the system, which, in turn, will limit the usage of the schema.

Our target is to build an annotation schema that focuses on the following goals:

- Accelerating the construction and learning of domain ontology.
- Establishing a common vocabulary to ensure interoperability and ease the process of ontology mapping. And
- Ensuring maintainability through self-maintenance or collaboratively maintained design.

In view of these goals, we developed a unified annotation schema that will be generic enough to accommodate various domains. In addition, we were able to:

- Define a Wikipedia–based annotation model based on the Wikipedia structure, the largest online semi-structured knowledge. Ontology-based information extraction from Wikipedia [33] has been a promising direction for semi-automated ontology construction. The proposed schema is an ontology template that will accelerate the development of domain ontologies.

- Extend the SKOS schema to build a Wikipedia-based annotation model, which enhances interoperability of the Wikipedia extracted knowledge and facilitates our ongoing research to build a semantic framework. SKOS has played an important role to enhance ontology engineering interoperability for linked data applications.
- Inherit the definition of DBpedia named-entities. By this, we were able to fill the gap between the Wikipedia extracted information and the RDF triples extracted from the Wikipedia info-boxes.
- Semantic models should also resolve problems like different senses (or contexts) for a concept, with an emphasis on a common or a most popular sense. We resolve the context problem by defining the "Wiki:domain_knowledge", which is an instance of "SKOS:Schema".

2.5 Future Work

We need to define extracted relations as an instance of semantic relation of the SKOS schema. We will model an entity that accommodate relations that are extracted with text mining, and natural language processing tools through the process of ontology construction. For example, in figure 2.4, relations as presidentOf or Ex_presidentOf is not defined in any of the three levels, we do not have any vocabulary that can be instantiated for these relations other than SKOS:related, or SKOS:Semantic_realtions and both were developed for the hierarchical type of relations. Hence, we need to extend this work to handle extracted relations within a specific domain.

We need to extend the proposed model for more specific domains. This model was developed to work best for creating lightweight annotation ontologies. We need to research the idea of extending the model for a more specific domain, like gene domain, or protein domain. We need to answer questions like is it feasible to extend a lightweight model to a heavyweight model. This is an ontology engineering question that worth answering.

3 Extracting Domain Concepts for Ontology Generation

3.1 Introduction

Ontology learning as defined in [34] is the process of automatic or semi-automatic creation of ontologies from a knowledge source. The process consists of the acquisition of the relevant terminology, identification of synonym terms / linguistic variants, the formation of concepts, hierarchical organization of the concepts (concept hierarchy), learning of relations, properties or attributes, and the appropriate domain and range, hierarchical organization of the relations (relation hierarchy), the instantiation of axiom schemata, definition of arbitrary axioms, and finally the ontology evaluation.

Since we are looking for a lightweight ontology (instance of SKOS-Wiki defined in the previous chapter), we only consider a subset of these tasks, which are the acquisition of domain terminologies, generating concept hierarchies, learning relations and properties, and ontology evaluation. Specifically, identifying relevant domain concept candidates for ontology learning, and identifying a semantic structure in the form of clustered concepts is the focus of this chapter.

In chapter one we presented the general structure of the proposed framework for constructing lightweight ontologies. When developing the framework modules, we rely on our understanding and distilling of the structure of the Wikipedia. Figure 1.5 shows the different modules for the proposed work. The current chapter is concerned with module A and B. In module A, we start with a selected set of domain concepts and consider them as the seed concepts for the ontology. Then we expand our concepts space using the Wikipedia link structure. In module B, we generate the domain and subdomain boundaries by using hierarchical clustering.

In the following section, we will lay out the background knowledge, and related work behind the proposed model. Then we will mention some related research in the area of concepts extraction, and Wikipedia-based information extraction and semantic relatedness. In section 3.3, we will illustrate the flowchart of the proposed approach for extracting domain concepts, and the generation of subdomains. The proposed model for extracting domain concepts, and computing relatedness between concepts is presented in section 3.4 (Module A in figure 1.5). The cluster-based approach for creating subdomains of related concepts will be presented in section 3.5 (Module B in figure 1.5). We will explain the experiments, data used, and results in section 3.6. Conclusion and future work are in section 3.7 and 3.8, respectively.

3.2 Background

The domain concept extraction phase is done in two steps. First we use the structure of the Wikipedia semantic network to build the domain concept space, and to measure the semantic relatedness of the concepts. This is implemented in module A. Then we use hierarchical clustering to generate subdomains of related concepts, and this is implemented in module B. In order to set the stage for the proposed approach, we will briefly review some foundational knowledge and related work in the area of clustering, in general, hierarchical clustering in particular, and related work for semantic relatedness measure and Wikipedia-based knowledge extraction. We will give some attention to proximity measures, and semantic similarity, considering their application on domain concepts. Section 3.2.1 will give a general and brief review for clustering analysis. Section 3.2.2, and section 3.2.3 will discuss the related work and background knowledge related to the generation of extended concept space, and semantic relatedness (module A). Section 3.2.4 focuses on hierarchical clustering, related to the generation of subdomains of related concepts (Module B).

3.2.1 Cluster Analysis

We “Cluster analysis is the organization of a collection of patterns (usually represented as a vector of measurements, or a point in a multidimensional space) into clusters based on similarity” [35]. Detecting groups (clusters) of closely related objects is an important problem in information extraction and data mining in general. There are many clustering methods that exist in the literature [36] . All clustering algorithms involve basic steps which are [35], pattern representation, pattern proximity, clustering, data abstraction, and assessment of clusters.

Pattern representation (optionally including feature extraction and/or selection) Pattern Representation includes the decision about which characteristics will represent each object (concepts in our case). Pattern representation is a well explored topic in statistical and pattern recognition [37]. Patterns are usually represented as multidimensional vectors, where each dimension is a single feature. The selected features for pattern representation, has to be the most descriptive and discriminatory features in the input set. Feature selection technique identify a subset of the existing features for subsequent use, while feature extraction techniques compute new features from the original set.

The selection of features to represent patterns of domain concepts has to reflect semantic information about the concept. Concepts or terms in documents are identified with the context in which they are mentioned. This is normally measured by the co-occurrence of the concepts with specific terms that define a context, or their term frequency (TF) and Inverse Document Frequency (IDF) in a given domain corpus. Clustering of domain concepts is based on the comparison of two concept patterns. This comparison expresses how related they are. Section 3.2.3 will discuss different methods in the literature for representing concepts and computing relatedness measures.

Pattern Proximity

Proximity measures define different ways to measure relationship between objects patterns. The definition of a pattern proximity measure should be appropriate to the data domain. There are two types of methods that estimate the relation between two patterns, distance measures and similarity measure [38].

Distance measures are numerical measures of how different two data objects are. They can be evaluated between two numeric vectors *x, y*, as d (*x, y*), where d is the distance function. A valid distance measure should be symmetric and obtains its minimum value (usually zero) in case of identical vectors. The most popular metric for continuous features is Euclidean distance. Manhattan distance is also used quite often, which is a special case of the Minkowski distance. Similarity measures are numerical measure of how alike two data objects are. The similarity function *S(x,y),* which compares the two vectors x and y should be symmetrical i.e. s(x,y) = s(y,x) and have a large value when x and y are "similar" and constitute the largest value for identical vectors. Common similarity measures include Cosine similarity (when the angle between two vectors is a meaningful measure) and Pearson Correlation. One benefit of using Pearson Correlation is that the accuracy of the score increases when data is not normalized. Hence this metric can be used when quantities (i.e. scores) vary within a wide range.

Clustering Algorithms

The clustering (grouping) step can be done in a number of different ways. The output clusters can be hard, where each pattern belong to exactly one cluster, or fuzzy (where each pattern has a variable degree of membership in each of the output clusters). A proximity measure and a clustering criterion mainly characterize a clustering algorithm. Farley and Raftery [39] suggest dividing the clustering methods into two main groups: hierarchical and partitioning methods. Han and Kamber [40] suggested adding two more categories, density based methods, model based methods all shown in figure 3.1.

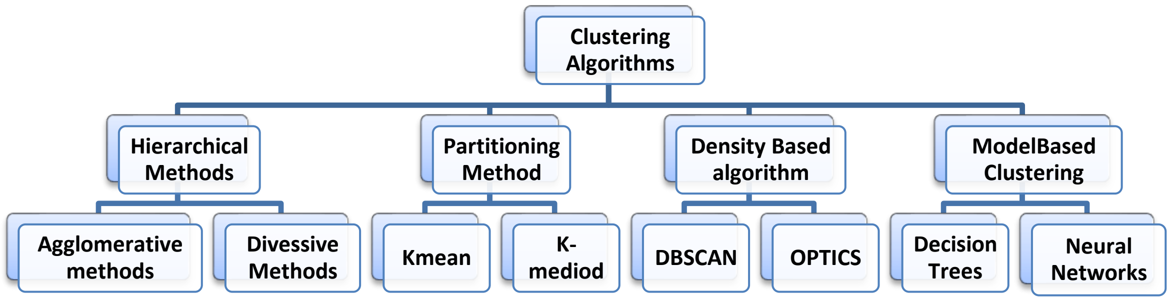


Figure 3.1 A classification of clustering methods

Partitioning clustering algorithms obtain a single partition of the data instead of a clustering structure with multiple levels. Partitioning methods start with an initial partition. Then they relocate instances by moving them from one cluster to another. Thus such methods require that the number of clusters will be pre-set by the user. To achieve global optimality in partitioned-based clustering, an exhaustive enumeration process of all possible partition is required. Two common example of partitioning methods are the K-mean clustering algorithm [41], and the K-mediod [40].

Density-based clustering algorithms assume that the points that belong to each cluster are drawn from a specific probability distribution. The idea is to continue growing the given cluster as long as the density (number of objects or data points) in the neighborhood exceeds some threshold, i.e the neighborhood of a given radius has to contain a least number of objects. The DBSCAN algorithm (density-based spatial clustering application) [42] discovers clusters of arbitrary shapes and is efficient for large spatial databases.

The Model-based clustering methods attempt to optimize the fit between the given data and some mathematical models. Unlike conventional clustering methods, which identifies groups of objects, model-based clustering methods also find characteristic descriptions for each group, where each group represents a concept or a class. The most frequently used model-based methods are decision trees and neural networks. For more information on decision trees-based clustering refer to [43], and for neural network-based clustering refer to [44].

We will give special attention to the hierarchical clustering algorithms, in section 3.2.4 as this is the clustering method most suitable solution for concept clustering, as will be discussed later in section 3.5.

Data Abstraction

In many applications that involve decision making, the resulting clusters have to be represented or described in a compact form to achieve data abstraction. Use of the centroid to represent a cluster is the most popular scheme. It works well when the clusters are compact or uniform. However, when the clusters are elongated or non-uniform, then this scheme fails to represent them properly. In such a case, the use of a collection of boundary points in a cluster captures its shape well[35]. Data abstraction is useful in decision making because it gives a simple and intuitive description of clusters which is easy for human comprehension.

Clustering Assessment

Cluster assessment methods only give an indication of the quality of the resulting clusters. Thus they can only be considered as a tool for the experts in order to evaluate the clustering results, but expert decision is the most important validation. There are two criteria proposed for clustering evaluation in [45] which are: compactness, where the members of each cluster should be as close to each other as possible, and separation, where clusters themselves should be widely spaced. Theodoritis classified the validation techniques into three categories [46]. The first is based on external criteria. This implies that we evaluate the results of a clustering algorithm based on a pre-specified structure. The second approach is based on internal criteria, in which we may evaluate the results of a clustering algorithm in terms of quantities that involve the vectors of the data set themselves (e.g. proximity matrix). The third approach of clustering validation is based on relative criteria. Here the basic idea is the evaluation of a clustering structure by comparing it to other clustering schemes, resulting by the same algorithm but with different parameter values using internal validation indices.

External Validation

Given a predefined clustered dataset, we can use different measures to assess the quality of the clustered instances, most common are Rand index, Jaccard coefficient. For our experimental validation we will compute the Rand Index:

$$\text{Rand Index} = \frac{TP+TN}{TP+FP+FN+TN} \quad \textit{Eq. 3.1}$$

These measures address the concepts True Positive (TP) (The decision that assigns two similar objects to the same cluster), and True negative (TN) (The decision that assigns two dissimilar objects to different clusters) to measure the percentage of decisions that are correct. Also F-measure is often used, to balance the contribution of false negatives by weighting recall through a parameter ß ≥ 0. Let precision and recall be defined as follows:

$$P = \frac{TP}{TP+FP} \qquad \textit{Eq. 3.2} \qquad\qquad R = \frac{TP}{TP+TN} \qquad \textit{Eq. 3.3}$$

Where P is the precision rate and R is the recall rate. We can calculate the F-measure by using the following formula:

$$F_{ß} = \frac{(ß^2+1)\,.P.R}{ß^2\;\; P+R} \qquad \text{Eq. 3.4}$$

Notice that when ß = 0, F_0 = P, i.e. recall has no impact on the F-measure and increasing ß allocates an increasing amount of weight to recall in the final F-measure. The RAND Index is used to measure the global quality of a solution. When a clustering is not correct, the measures of precision and recall are necessary to determine the reasons. In fact, there is a relation between recall and precision:

- A high value of the recall and a low value of the precision indicate that the resulting clustering has too few clusters: most objects that belongs to the same ideal cluster are grouped together, but are also grouped with objects of other ideal clusters.

- A low value of the recall and a high value of the precision indicate that the resulting clustering has too many clusters: most objects that are grouped together belongs to the same ideal cluster, but all the objects of the same ideal cluster are not grouped together.

Internal Validation

The validation here is based on the clustering data itself. These methods usually assign the best score to the algorithm that produces clusters with high similarity within a cluster, and low similarity between clusters. One drawback of using internal criteria in cluster evaluation is that high scores on an internal measure do not necessarily result in a good clustering scheme, because some indices favors large clusters, such as Davis Bouldin index [47] while others favors small clusters such as Silhouette measure [48].

3.2.2 Wikipedia-Based Ontology Learning

Concepts extractions is a subtask of ontology learning. The automated and semi-automated efforts for domain concept extraction in general, and for ontology learning in particular, rely mainly on the type of data to be used, and the techniques of the developed approaches. Data is either structured, unstructured or semi-structured. The techniques are either statistical, natural language processing or both. Statistical methods usually depend on, Poisson statistics, the Maximum Likelihood Estimation and Inverse Document Frequency between the frequency of words in a given corpus, and the commonality of the term in the language. A comprehensive study of methods and tools in this direction can be found in [49].

We will focus here on similar efforts, to the proposed one in this research that used Wikipedia, as a semi-structured knowledge source. Methods were developed to parse HTML tags and links, to extract ontology concepts, and map HTML structure to hierarchical ontological relations as in [50].

Also Hepp [12] viewed the Wikipedia articles as ontology elements, for which the URIs of Wikipedia entries serve as reliable identifiers. The disambiguation pages and redirects of the Wikipedia was used for word sense disambiguation. One of the most widespread ontologies is YAGO2 [51], which exploits a set of relation-specific heuristics to extract knowledge from Wikipedia, Geo-Names and WordNet. However, this ontology considers only about 100 different semantic relations between its concepts. Nastase and Strube [52] presented an automatically created concept network called WikiNet. WikiNet differs from YAGO2 in that it uses more general heuristics. This resource has around 500 different semantic relations, however, it heavily relies on Wikipedia categories and Info-boxes. As a consequence, WikiNet does not easily scale with different relations.

Recently Wei et al. [53] developed a method to extract domain-specific facet (DF-Miner), to discover DFs (Domain-specific facets) based on the hyperlink structure within the Wikipedia article. These efforts do not rely on a semantic schema, and lack the definition of domain boundaries. Attempts to use Wikipedia always aim to dump as many concepts and relations as possible, with no consideration of a specific domain or a predefined semantic schema.

3.2.3 Semantic Relatedness Measures

Before we propose our approach for extracting domain concepts and computing semantic relatedness, we find it important to highlight the different ways to compare domain concepts, and compute relatedness. Pederson et al. [54] emphasized the difference between semantic similarity, and semantic relatedness. Semantically similar concepts are related on the basis of their likeness. Semantic similarity measures is usually based on is-a relations that link concepts (directly or indirectly) found in a hierarchy, or any form of ontological knowledge. These measures rely on the path length between concepts. Semantic relatedness on the other hand, is a more general notion of relatedness, which can include other types of relations or can also be based on co-occurrence statistics from corpora.

The research in [55] has shown that humans use the context of words and concepts to build a mental semantic representation of concepts. Over time humans encounter similar context for different concepts, consequently they tend to agree on the semantic relatedness of concepts. The evaluation of a concept relatedness measure is mostly based on correlation with human judgment.

Harispe et al. had presented a comprehensive study of semantic components and similarity measure in [56]. Semantic relatedness measures are divided in the Harispe's literature review into two main categories: distributional based measures, and structure based relatedness measures (Knowledge based).

Distributional-based methods

Distributional based measures are sometimes denoted as corpus-based measures [57]. Distributional measures rely on the distributional hypothesis which considers that words occurring in similar contexts tend to be semantically close [58]. Several distributional measures have been proposed. The main methods are Spatial methods, which evaluate the relative positions of the two words in the semantic space defined by the context vectors, and probabilistic methods which express the semantic relatedness of words in term of probability of co-occurrence, i.e. regarding both, the contexts in which compared words appear and the contexts in which the two words co-occur.

Statistical analysis can be used to distinguish valuable patterns in order to highlight deeper co-occurrences between words. These patterns, can be identified using several techniques as Latent Semantic Analysis (LSA) [59] and Hyperspace Analogue to Language (HLA) [60]. Distributional models are unsupervised, they can be used to compare the relatedness of words expressed in corpora without prior knowledge regarding their meaning, or usage. A strong limitation of the distributional methods is that similar words may not co-occur [61]. For example synonyms, like, the words "road" and "street" almost never co-occurs.

Structure-based relatedness measure

The Structural approach relies on a knowledge representation for the domain under study. This representation can be in the form of a hierarchical taxonomy, ontology or a semantic graph. Measures of semantic similarity in this case are mostly path finding measures. Most of the mature attempts in this direction was developed using medical knowledge base. Rada et al. [62] used the Medical Subject Heading (MeSH) ontology to develop a method based on the path length between concepts that relied on "Broader than relations". This method links successively, less or more specific concepts as you travel from one concept to the other in the ontology. This measure was applied in the ranking of the documents retrieved from MEDLINE. Cimino [63] developed a measure called CDist which finds the shortest path between two concepts in the UMLS (consisting of MeSH, ICD-9-CM and SNOMED-CT), which although a relatively simple approach but had reliable results. Wu and Palmer [64] present a measure, that relies on finding the most specific concept, that subsumes both of the concepts being measured. The path length from this shared concept to the root of the hierarchy, or the ontology, is scaled by the sum of the distances of the concepts to the subsuming concept.

Structural approaches for computing similarity measures can be used to compare all types of elements defined in a Knowledge representation, i.e., terms/classes, or instances. Thus, these measures can be used to compare elements, which cannot be compared using text analysis. On the other hand, structure based methods require a knowledge representation describing the elements to compare. Moreover, the computational complexity is very high in case of using large knowledge bases.

Wikipedia-based semantic relatedness

The structure of the Wikipedia (as a knowledge base) has been used for computing semantic relatedness in various ways. Strube and Ponzetto [65] proposed

WikiRelate that exploits the Wikipedia link structure to compute the relatedness between Wikipedia concepts. WikiRelate counts the edges between two concepts and considers the depth of a concept in the Wikipedia category structure. This method requires generating the Wikipedia graph each time the Wikipedia changes, which is a very heavy computation. KORE [51] eliminates this issue by computing the relatedness scores between the contexts of two entities using the probability of co-occurrence of neighbor concepts.

3.2.4 Hierarchical Clustering

Hierarchical clustering can be either agglomerative or divisive. An agglomerative clustering algorithm begins with each pattern in a distinct cluster (singleton), and successively merges clusters together until they all form one cluster. A divisive algorithm begins with all patterns in a single cluster and perform splitting until a stopping criterion is met [35]. Hierarchical clustering algorithms produce a nested series of partitions based on a criterion for merging or splitting clusters depending on similarity. A hierarchical algorithm yields a nested tree (dendrogram) representing the nested grouping of patterns and similarity levels at which groupings change, as the one shown in figure 3.2.

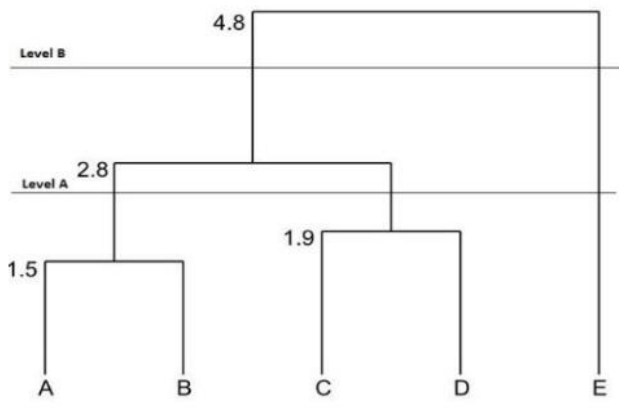


Figure 3.2 Hierarchical Clustering Dendrogram

Both agglomerative and Divisive hierarchical clustering, result in the same clustering scheme when the dendrogram is cut at the same height. We will focus on the agglomerative hierarchical clustering, as it is simpler, and used more often.

The pseudocode for the agglomerative hierarchical clustering is:

1. Compute the proximity matrix containing the distance between each pair of patterns. Treat each pattern as a cluster.
2. Find the most similar pair of clusters using the proximity matrix. Merge these two clusters into one cluster. Update the proximity matrix to reflect this merge operation. Merging of clusters depends on the linkage method used.
3. If all patterns are in one cluster, stop. Otherwise, go to step 2.

Cutting-off the dendrogram at a given height will give a partitioning clustering at a selected precision. The decision about the cut-off heights, defines the number of clusters resulted. In figure 3.2 the cut-off at level A will yield into 3 clusters, while the cut-off at level B will yield into 2 clusters.

Linkage methods

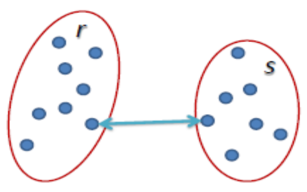
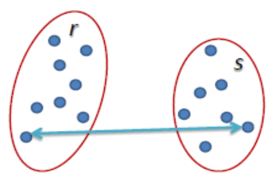
Linkage methods differ in the way they characterize the similarity between a pair of clusters. Popular linkage methods are: single linkage, complete linkage, average linkage, centroid linkage and ward linkage [35].

In Single Linkage method, the distance between two clusters is the minimum of the distances between all pairs of patterns drawn from the two clusters (one pattern from the first cluster, the other from the second), as shown in figure 3.3 (a). The single-link algorithm, by contrast, suffers from a chaining effect. It has a tendency to produce elongated clusters.

In Complete Linkage method, the distance between two clusters is the maximum of all pairwise distances between patterns in the two clusters, as shown in figure 3.3 (b). The complete-linkage algorithm produces tightly bound or compact clusters, as shown in figure 3.3 (b).

In Average Linkage method, the distance between two clusters is said to be equal to the average distance from any member of one cluster to any member of the other cluster. This is shown in figure 3.3 (c).

Centroid linkage method, computes the distance between the group centroids (cluster average), illustrated in figure 3.3 (d).

Single Linkage	**Complete Linkage**
 $L(r,s)=\min(D(x_{ri},x_{sj}))$	 $L(r,s)=\max(D(x_{ri},x_{sj}))$

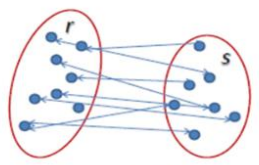
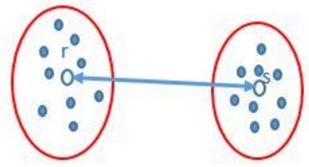
(a)	(b)
Average Linkage  $L(r,s)=\frac{1}{n_r n_s}\sum_{i=1}^{n_r}\sum_{j=1}^{n_s} D(x_{ri},x_{sj})$ (c)	**Centroid Linkage**  $L(r,s) = \lvert X_r - X_s \rvert$ (d)

Figure 3.3 Linkage methods

In all linkage methods, similarity is monotonically decreasing from iteration to iteration, except in case of centroid linkage, where centroid clustering is not monotonic. An inversions can occur in the dendrogram, when similarity suddenly increases in one of the iterations.

Another linkage method is the ward method which minimizes the variance between the merged clusters.

Determining the optimal number of clusters

Determining the optimal number of clusters is a non-trivial task. In particular the huge volume of data and the potentially high dimensionality of patterns increase the difficulty of achieving a measure for optimal clustering. Gordon [66] divided the methods to estimate the number of clusters into two categories: global methods and local methods. The Global methods intend to optimize the quality of the clusters produced by measuring a specific gain criteria for different number of clusters, and choose the number of cluster with optimal gain value. In contrast, the local methods are intended to test the hypothesis that a pair of clusters should be combined or not. Local methods mainly use the validation indices, which rely on maximizing the intra-cluster distance or minimizing the inter-cluster distance or both.

The decision gets more complicated in hierarchical clustering. The number of clusters is determined based on a cut-off height which is one of the merging heights. A certain height needs to be defined to produce the best quality clusters. It's even harder when defining a constant cut-off height will fail to recover all the significant clusters. In this case we need to look for several cut-off levels. Most of the dendrogram-based indices that depend on the merging heights favor large clusters. In our research we are concerned with clustering concepts, where the data set might vary in its generality. In some cases, we will be looking for small clusters for more specific subdomains, and in other cases we will be looking for larger clusters for more general subdomains.

The hierarchical cluster tree may naturally divide the data into distinct, well-separated clusters. This can be clear in a dendrogram diagram created from data, where groups of objects are densely packed in certain areas and not in others. We will be using the Inconsistency coefficient to determine candidate list of cut-off heights. This will be explained in details in section 3.5.3.

3.3 A Wikipedia-based Methodology for Extracting Domain Concepts

Recall from chapter 1, in figure 1.5 we introduced the general model for constructing lightweight ontologies. We will focus here on presenting our methodology for extracting domain concepts, and generating domain boundaries relying on the Wikipedia. Figure 3.4 shows the flow of the proposed methodology to extract domain concepts, and generate subdomain boundaries. The section where each block is explained is shown beside it. We rely in the first three steps on the Wikipedia link structure. Then we use hierarchical clustering to extract highly related domain concepts.

Section 3.4 and 3.5 explain the proposed methodology for extracting domain concepts and defining the subdomain boundaries using hierarchical clustering. They represent the implementation approach for module A and B, respectively (refer to figure 1.5). Section 3.6 presents the experimental work and the results.

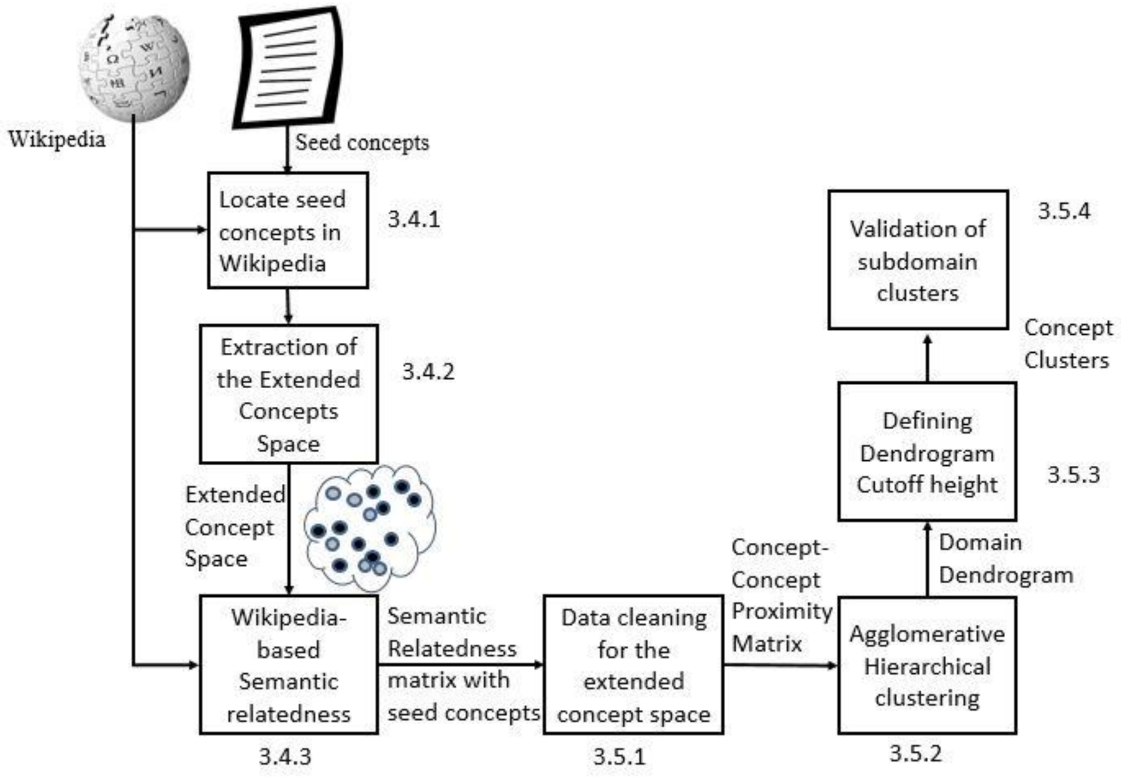

Wikipedia
Seed concepts
Locate seed concepts in Wikipedia
3.4.1
Extraction of the Extended Concepts Space
3.4.2
Extended Concept Space
Wikipedia-based Semantic relatedness
3.4.3
Semantic Relatedness matrix with seed concepts
Data cleaning for the extended concept space
3.5.1
Concept-Concept Proximity Matrix
Agglomerative Hierarchical clustering
3.5.2
Domain Dendrogram
Defining Dendrogram Cutoff height
3.5.3
Concept Clusters
Validation of subdomain clusters
3.5.4

Figure 3.4 The process for extracting domain concepts

It is important to map the terminologies used in our methodology to data clustering terminology:

- A Concept, is a pattern (A feature vector or an observation).
- A link weight, is an individual scalar component of the concept (a scalar component of the feature vector) that represent the relatedness weight between a concept and one of the seed concepts.
- A concept is mapped to the Wikipedia structure, either to a Wikipedia page or a Wikipedia category.
- The symbol “cat(p)” in the equations represents the set of all Wikipedia categories where the Wikipedia Page (p) belongs.
- The symbol “outlinks(p)” , represents the set of all pages where P has an out-link to.

- The symbol “child(c)” represents the set of children categories of category c.
- The symbol “parent (c)” represents the set of parent categories of category c.
- The symbol “page(c)” represent the set of pages having category c as one of their categories.

3.4 Generation of Extended Concept Space

We start with the seed concepts selected from the domain of interest. Then we use the Wikipedia link structure to generate the extended concept space.

3.4.1 The Selection of seed concepts

Seed concepts have to be selected putting in consideration boundaries and the diversity of the domain. The selection of highly related small number of seed nodes, will lead to a more specific domain, with tight relations than a bigger number with loose relations. Seed nodes can be extracted with one of the following ways:

- Document mapping: By Carefully selecting representative documents from a domain corpus, and mapping it to Wikipedia pages (using document- document similarity).
- Based on expert suggestion, where an expert would choose samples of domain concepts which are thought to be core domain concepts.
- Using an existing ontology or a high level topic map.

After the careful selection of seed concepts, we locate them in the Wikipedia. For each seed node, we extract the relevant Wikipedia node (page, or category) that represent each seed concept.

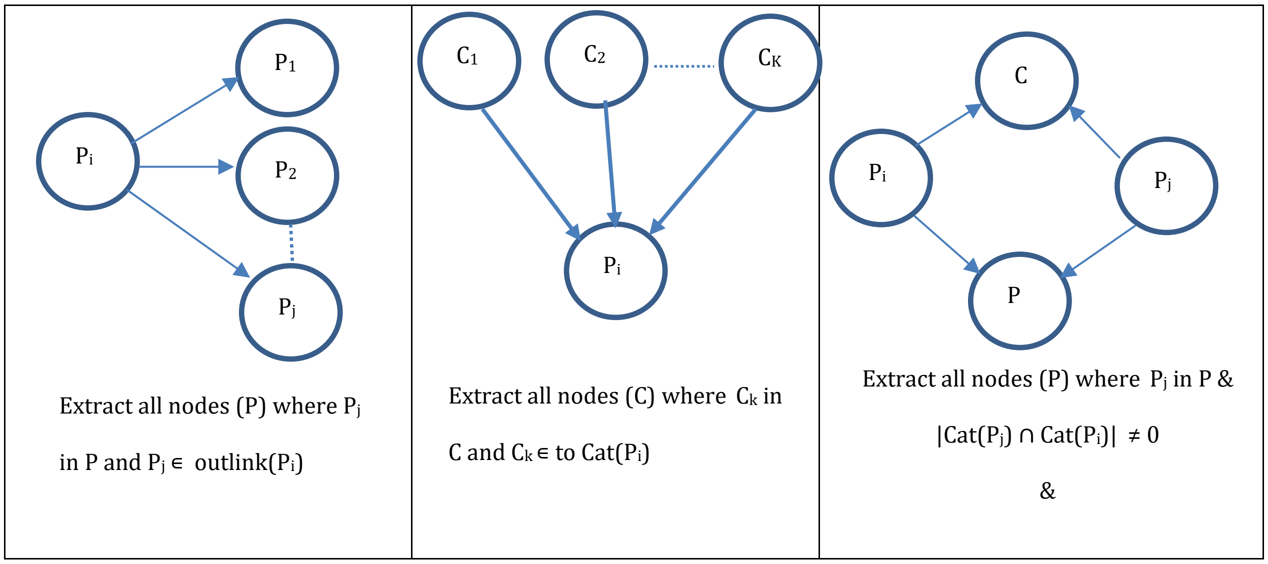
3.4.2 Generating the extended concept space

After locating all seed nodes in the Wikipedia semantic network, we build the extended concept space. This is a pool of concepts that vary from high domain relevance to weak relevance, and some can be considered completely irrelevant noise concepts.

For each seed concepts S_i belongs to S, where S is the set of all seed concepts:

If S_i is mapped to a Wikipedia page P_i :

- Extract the set of all pages P such that, for each Page (P_j) belongs to P there exist an out-link from P_i to P_j (as shown in figure 3.5-A). All extracted nodes are added to the set of concepts C_p .
- Extract the set of all categories C such that, for each concept c_k, belongs to C, c_k is a category of P_i.(as shown in figure 3.5-B) . All extracted nodes are mapped to the set of concepts C_t.



P_1
P_i
P_2
P_j
Extract all nodes (P) where P_j in P and $P_j \in$ outlink(P_i)
C_1
C_2
C_K
P_i
Extract all nodes (C) where C_k in C and $C_k \in$ to Cat(P_i)
C
P_i
P_j
P
Extract all nodes (P) where P_j in P &
$|Cat(P_j) \cap Cat(P_i)| \neq 0$
&

Figure 3.5 Generating the extended concept space in case the seed concept maps to a Wikipedia page

Extract the set of all pages P such that, for each page (P_j) belongs to P, P_j shares at least one category, and one out-link with P_i (as shown in figure 3.5-c). All extracted nodes are mapped to the set of concepts C_m.

If S_i is mapped to a Wikipedia category C_i:

- Extract the set of all categories C such that, for each concept c_j, belongs to C, c_i is a child category of C as shown in figure 3.6-A. All extracted nodes are mapped to the set of concepts C_n.

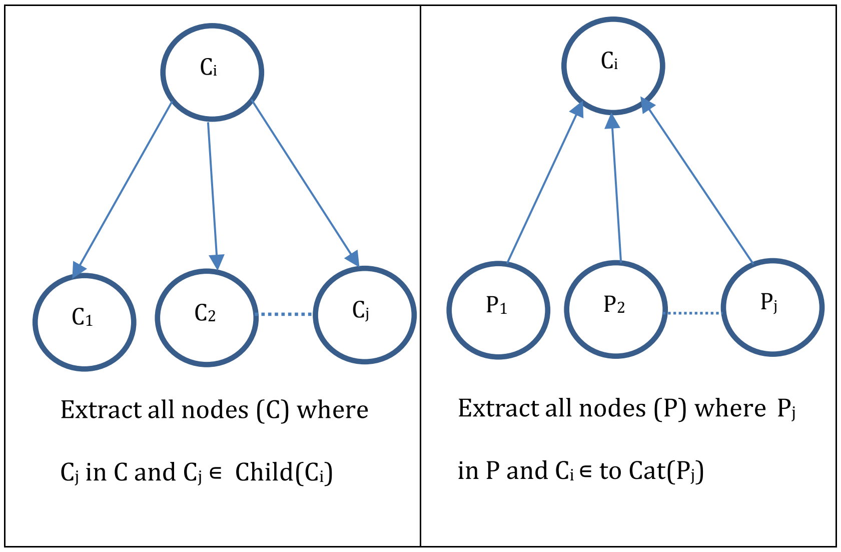

C_i
C_1
C_2
C_j
Extract all nodes (C) where
C_j in C and $C_j \in$ Child(C_i)
C_i
P_1
P_2
P_j
Extract all nodes (P) where P_j
in P and $C_i \in$ to Cat(P_j)

Figure 3.6 Generating the extended concept space in case the seed concept maps to a Wikipedia Category

- Extract the set of all Pages P such that, for each page (P_j) belongs to P, C_i belongs to the set of categories of P_j as shown in figure 3.6-B. All extracted pages are mapped to the set of concepts C_k.

  Extended concept space $C_{extend} = C_p + C_t + C_m + C_n + C_k$

  The level of expansion for each seed can be specified during the selection of each seed node. Which in turn allows for controlling the extended concept space according to the user opinion. We defined three levels of expansion for each seed concept:

- Level one: We expand the seed concept to include out-links if the concept is mapped to a Wikipedia article, or children categories if the concept is mapped to a Wikipedia category, or both in case the concept name is mapped to both, an article, and category.

- Level two: We expand a seed concept to include all concepts as in level one, in addition to the categories the concept belongs to, in case it is an article, and if the concept is a category we extract pages that belong to this category.
- Level three: We expand a seed node to include all concepts as in level one and two, in addition to the parent categories in case the concept is a category.

So we noticed that in case of general concepts, if we explored all levels to include all related concepts, then the amount of irrelevant concepts, or weakly related, increases. One should choose a higher level (level one), in case of a general concept, and lower level (two, or three) for more specific concepts. The reason for this modification, is to discover the effect of the choice of the seed concepts, and the expansion level on the precision of the extended concept space and consequently on the clusters of subdomain. In the experimental work, we will perform two experiments. In first one we just selected the seed nodes without the definition of the expansion level. In the second one we added a value for the expansion level for each seed concepts.

This process for building the extended concept space should minimize the capture of noise concepts, because every concept added to the concept space is selected to have a certain relation with the seeds. Nevertheless, extended concept space still contains a considerable amount of noise that cannot be neglected. The reason for this is that this process is highly dependable on the selection of seed concepts. Some of the seed concepts might belong to different domains in the same time, which in turn will capture concepts from different domains that will have a very low relevance to the original domain.

3.4.3 Wikipedia-based relatedness measure for pattern representation

We assume that all concepts in the extended concept space are related to the domain of interest, and each concept is related to any other concept with a certain weight. Here, we will present a proposed method for measuring semantic relatedness between any two concepts, given that the two concepts are mapped to a Wikipedia article or category. This method will be used for pattern representation matrix. The rows of the pattern matrix represent concepts in the extended concept space, and the columns are seed concepts. Each entry in the matrix *W(x,y)* represents a weight for the relation between the concept '*x*' and a seed concept '*y*', i.e. the relatedness with the seed concepts is considered the features to describe each concepts.

We classified the Wikipedia nodes that represent each concept in the extended concept space into two types, article node, and category node. The relatedness matrix is an undirected matrix so we consider three links: "Category – Category" link (link between two nodes both of type category), "Article – Article" link (link between 2 nodes both of type article), "Category – Article" link (link between 2 nodes one of type category and the other of type article).

Article – Article link :

The weight of the link between article p_i and article p_j, where p_i *and* p_j are Wikipedia pages corresponding to concepts c_i, c_j is

$$Wp_i, p_j \;=\; max\{W(P_i, P_j), W(P_j, P_i)\} + \frac{cat(p_i) \cap cat(p_j)}{|cat(p_i)|}$$

Where $W(P_i, P_j)$ is the weight of the link from P_i to P_j if exists. $W(P_j, P_i)$ is the weight of the link from P_j to P_j if exists .

- $W(P_i, P_j) = \frac{|(\text{outlinks(Pi)} \cap \text{outlinks(Pj)}|}{|\text{outlinks(Pi)}|}$
- $W(P_j, P_i) = \frac{|\text{outlinks(Pi)} \cap \text{outlinks(Pj)}|;}{|\text{outlinks(Pj)}|}$

Weight is boasted by 0.2 if the link exists in the introduction section. The values of $W_{Pi,Pj}$ ranges from 0 to 2.4.

Category – Category link:

The weight of the link between category C_i *and* C_j, where C_i *and* C_j are Wikipedia categories corresponding to concepts c_i, c_j is

$$W(C_1,C_2) = \max\{W_p(C_1,C_2), W_p(C_2,C_1)\}, + \max\{W_c(C_1,C_2), W_c(C_2,C_1)\} + \max(W_a(C_1,C_2), W_a(C_2,C_1))$$

- $W_p(C_1, C_2) = \frac{|Parent(C1) \cap Parent(C2)|}{|Parent(C1)|}$ *Percentage of Common Parents to total parents of* C_1
- $W_c(C_1, C_2) = \frac{|Child(C1) \cap Child\ (C2)|}{|child(C1)|}$ *Percentage of Common Children to total children of* C_1
- $W_a(C_1, C_2) = \frac{|Page(C1) \cap Page(C2)|}{|Page(C1)|}$ *Percentage of Common Articles to all articles of* C_1

The weight is boasted by 0.2 if there is a parent child link between the 2 categories the value of $W(C_1,C_2)$ ranges from 0 to 3.

Article – Category link

The weight of the link between article P and category C is :

- W(P,C) = $\frac{|X|}{|outlinks(P)}$ *where X = {x: x ∊ outlinks(P), & cat(x) = C}*

- W (P, C) is the percentage of out-links of P sharing category C.

- W (P, C) = W (P, C) + 0.2 if P ∊ Page (c) i.e. the weight is boasted by 0.2 if p belongs to the set of articles belonging to c. The value $W_{p,c}$ ranges from 0 to 1.2.

All types of weights are of equal importance, and each type is normalized differently according to its range.

3.5 Clustering Analysis of the Extended Concept Space

3.5.1 Data Cleaning

We mentioned earlier, that during the process of building the extended domain space, there were some irrelevant concepts that were captured. In general determining whether or not an observation is an outlier is ultimately a subjective exercise [67] . Outliers as defined in [68] "An outlying observation, or "outlier," is one that appears to deviate markedly from other members of the sample in which it occurs". In our case we do not have a highly deviated observations, but we have very low values for relatedness measures that reflect either unrelated, or very weakly related concepts to the main domain of interest. Removing these concepts will dramatically influence the vision, and the interpretation of the resulting clusters.

In order to detect the relevance of each concept with respect to the extended concept space, we calculated an importance factor, which is the total relatedness of the concept with the seed concepts. Irrelevant concepts are concepts with extremely low importance factor, while concepts with high importance factor are the most valuable concepts.

In most cases the distribution of the importance values is not a uniform distribution (as will be shown in the experimental results). Thus we will not be able to use the regular z-score values to eliminate outliers as defined in [68]. Mean, and standard deviation are very sensitive to outliers. Hampel [69] rediscovered the usage of median, and Median Absolute Deviation (MAD) for outlier detection. Median is like mean, a measure of central tendency but insensitive to the presence of outliers.

MAD is defined as the median of the absolute deviation from the median, given by:

$\text{MAD} = \text{Median}_i\,|(X_i - \text{Median}_j(X_j))|$

X_i in this case is the list of the importance values, and Median_j is its median. MAD is the median of the list of the absolute values of the deviation from the median.

Then we must define the elimination criterion, which remains a subjective aspect, which depends on the tolerance we accept for weak relatedness. Concepts to be omitted has to satisfy a threshold, where $X_i < |\text{Median} - T^*\text{MAD}|$. It was suggested by Miller [70] to use T=3 for a highly conservative, T=2.5 for a moderately conservative and T = 2 for a low conservative decision.

3.5.2 Data Clustering

After constructing the pattern representation matrix for the extracted domain concepts, we use it to construct a dissimilarity matrix (proximity matrix), using Euclidean distance. We chose to use hierarchical algorithms because they are more flexible than partitioning algorithms. Hierarchical algorithms works well on data sets containing non-uniform clusters including well-separated, chain-like, and concentric clusters, whereas a typical partitioning algorithms such as the k-means works well only on data sets having uniform clusters.[35].

Partitioning Algorithms especially k-means are sensitive to outlier, and in our case outliers are very common in the extended domain concepts. Moreover, when using partitioning algorithms the number of clusters has to be defined in advance, while in hierarchical clustering the cut-off threshold is defined based on the dendrogram structure. In some cases where the data set is not large, the merging levels of hierarchical structure of the dendrogram can be interpreted as hierarchical relations between the concepts. We used agglomerative hierarchical clustering to cluster the extended concept space. In Hierarchical clustering we need to decide the linkage method and the cut-off height.

Linkage method

To choose the linkage method that yields the optimal clustering; we computed the Cophenetic correlation between the original dissimilarity matrix and the clustering matrix with each linkage method. This method was stated in [71]. The highest correlation indicated the best linkage method.

3.5.3 Defining the Cut-off in the domain dendrogram

One way to determine the natural cluster divisions in a data set, is to compare the height of each link in a cluster tree with the heights of neighboring links below it in the tree, using the inconsistency measure. A link that is approximately the same height as the links below, indicates that there is no distinct division between the objects joined at this level of the hierarchy. These links are said to exhibit a high level of consistency, because the distance between the objects being joined is approximately the same as the distances between the objects they contain. On the other hand, a link whose height differs noticeably from the heights of the links below it, indicates that the objects joined at this level in the cluster tree are much farther apart from each other than their components. This link is said to be inconsistent with the links below it. In cluster analysis, inconsistent links can indicate a candidate for a natural division. Inconsistency coefficient [72] of the links in the cluster tree identifies abrupt change in the similarities between objects.

The inconsistency function compares each link in the cluster hierarchy with adjacent links that are less than two levels below it in the cluster hierarchy. This is called the depth of the comparison. The objects at the bottom of the cluster tree are called leaf nodes. They have no further objects below them, and they have an inconsistency coefficient of zero. Clusters that join two leaves also have a zero inconsistency coefficient.

To compute the inconsistency coefficients for the merging heights in a dendrogram consider figure 3.7. The inconsistency coefficient of link x at height H_x, with the links two levels below it, y, z at heights H_y, and H_z respectively is defined by I_x.

$$I_x = \frac{Hx - \text{mean}(Hx, Hy, Hz)}{\text{Std}(Hx, Hy, Hz)}$$

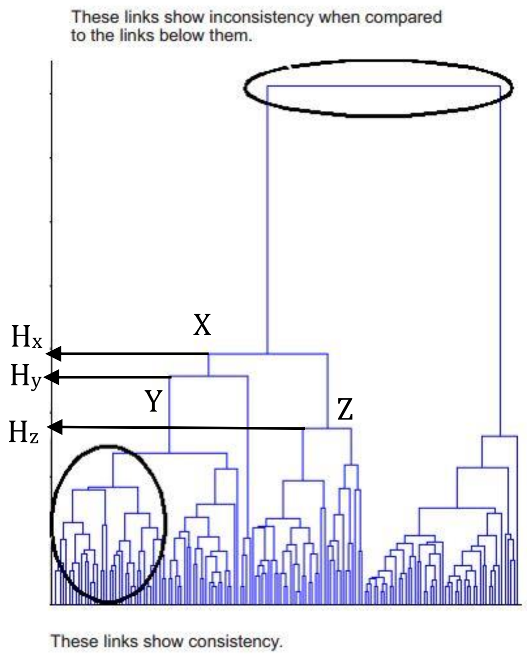

These links show inconsistency when compared to the links below them.
X
Hx
Hy
Y
Z
Hz
These links show consistency.

Figure 3.7 Inconsistency in the links of a dendrogram

In order to define the number of clusters follow these steps:

• We compute the inconsistency coefficient to choose the range of cut-off that will represent a natural division.

We choose candidate heights with high inconsistency coefficients, each will yield a different clustering scheme.

• Using different validation indices to decide the best clustering scheme will yield different decisions, which indicates that there is no unanimous choice regarding the optimal number of clusters.

• Using the R package for statistical computing NBClust, we compute several validation indices for the candidate heights, and consider the decision of the majority.

• For each cluster at the optimal static cut-off we compute the average intra-cluster distance, and the number of concepts per cluster. Clusters with high intra-cluster distance, and few number of concepts are considered outliers.

Clusters with low intra-cluster (compared to other clusters), and high number of concepts (exceeding certain threshold) are re-clustered.

3.5.4 Validation methodology

Validation of clustering experiments has always been a challenge. Clustering is an unsupervised learning method, and for that reason it has no a priori labeling information. As mentioned earlier, the validation of the clustering experiments is either by external validation, based on previous knowledge about data, or by internal validation, based on the information intrinsic to the data alone. External validation, especially in the case of concept clustering, requires the existence of a pre-specified structure in the same domain of interest, which is not always easy to find. We, therefore, constructed the validation of the extended concept space on multiple phases.

The domain within which we performed our experiments is the domain of information retrieval. For external validation, we used the indices appearing in two books, titled, "Introduction to Information Retrieval" [73], and "Search Engines : Information Retrieval in Practice" [74]. For each book, we used the book index to extract all the concepts, and we created two concept lists for the validation of the extended concept space. Then for the latter book only, we were able to manually cluster the extracted concepts based on the context in which they were mentioned. We used those concepts to validate the clusters representing subdomains that were derived via a hierarchical clustering method. For internal validation we used four internal indices which are due to Duda [75], Dunn [76], Marriott [77], and Hartigan [78] indices to determine the optimal number of clusters, and we also used them to determine the quality of the clusters produced.

3.6 Experimental Work and Results

We developed our experiments on the domain of "Information retrieval science". We generated the seed concepts by exploring the main topics of this domain. Illustrated in figure 3.8 are the top concepts in the hierarchy of this domain.

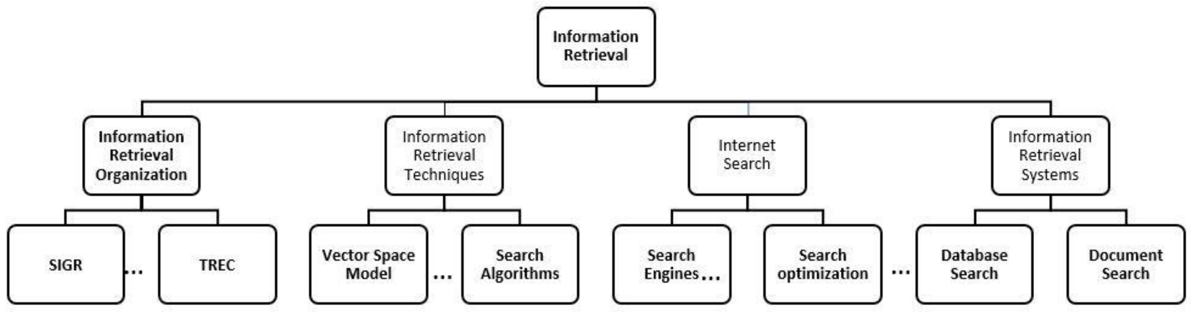


Figure 3.8 Main topics in the chosen domain knowledge (Information retrieval)

We processed this domain twice. In the first experiment, we chose a set of seed concepts randomly from the domain, independent of their hierarchical level, and independent of the knowledge that will be used for external validation. In the second experiment, we chose the concepts taking in consideration the perspective of the knowledge that will be used for external validation. Also we considered the level of expansion for each seed.

3.6.1 The first Experiment

In the first experiment we had seven seed nodes, which are: 'information retrieval', 'web search engine', 'vector space model', 'subject indexing', 'text retrieval conference', 'special interest group on information retrieval', and 'latent semantic indexing'. To generate the extended concept space, we applied the method in section 3.4.2. The extended concept space then contained 457 concepts.

Total Number of concepts in the extended concept space (TP + FP)	457
Total number of concepts in the book index (TP + FN)	666
Total number of concepts in the extended concept space not in the book index (FP)	191
Total number of concepts in the extended concept space and in the book index (TP)	266

Table 3.1 Validation of the extended concept space with the book "Introduction to Information Retrieval"

We compared these concepts (to check reliability of the extended concept space) against the index of the book "Introduction to information retrieval" [73], mentioned earlier. Table 3.1 shows the results of the comparison.

Precision = $\frac{TP}{TP+FP}$ = 58.2% Recall = $\frac{TP}{TP+FN}$ = 40 %

Almost 41 % of the retrieved concepts are not part of the book index. Some of these concepts are weekly related to the domain, and others represent companies, locations or person's names, which are related to the domain but will not appear in an academic book. We also validated the extended concept space against the book "Search engines: Information retrieval in practice" and the results are shown in table 3.2

Total Number of concepts in the extended concept space (TP + FP)	457
Total number of concepts in the book index (TP + FN)	364

Total number of concepts in the extended concept space not in the book index (FP)	273
Total number of concepts in the extended concept space and in the book index (TP)	184

Table 3.2 Validation of the extended concept space with the book "Search Engines: Information Retrieval in Practice"

Precision = 40.26%   Recall = 50.55%

We eliminated most of the weakly related concepts using the Median Absolute Deviation (MAD) method explained in section 3.5.1. Then, we were left with 325 concepts to cluster. That is, 23% of the concepts in the expanded concept space were weakly related concepts.

On revalidating the extended concept space after noise elimination with the first book the results were as shown in table 3.3.

Total Number of concepts in the extended concept space (TP + FP)	325
Total number of concepts in the book index (TP + FN)	666
Total number of concepts in the extended concept space not in the book index (FP)	149
Total number of concepts in the extended concept space and in the book index (TP)	176

Table 3.3 Validation of the extended concept space after noise elimination for with the book "Introduction to Information Retrieval"

Precision (after noise elimination) = 54.15% , Recall (after noise elimination) = 26.4%

The corresponding results for the second book are shown in table 3.4.

Total Number of concepts in the extended concept space (TP + FP)	325
Total number of concepts in the book index (TP + FN)	364
Total number of concepts in the extended concept space not in the book index (FP)	175
Total number of concepts in the extended concept space and in the book index (TP)	150

Table 3.4 Validation of the extended concept space after noise elimination for experiment one with the book "Search Engine: Information Retrieval in Practice"

Precision (after noise elimination) = 46.15%

Recall (after noise elimination) = 41.2 %, Recall decreases with the decrease of the TP concepts.

We constructed the feature matrix by measuring the relatedness of each concept with the seed nodes. The distance between each pair of concept vectors is calculated via Euclidean distance. Then we clustered the distance matrix with hierarchical clustering using the Ward linkage method. The choice of the linkage method was based on the highest Cophenetic correlation between the clustering matrix and the distance matrix. The dendrogram of the clustered concepts is shown in figure 3.9. In order to define a set of candidate cut-off levels, we compute the inconsistency coefficient for all merging heights of the dendrogram.

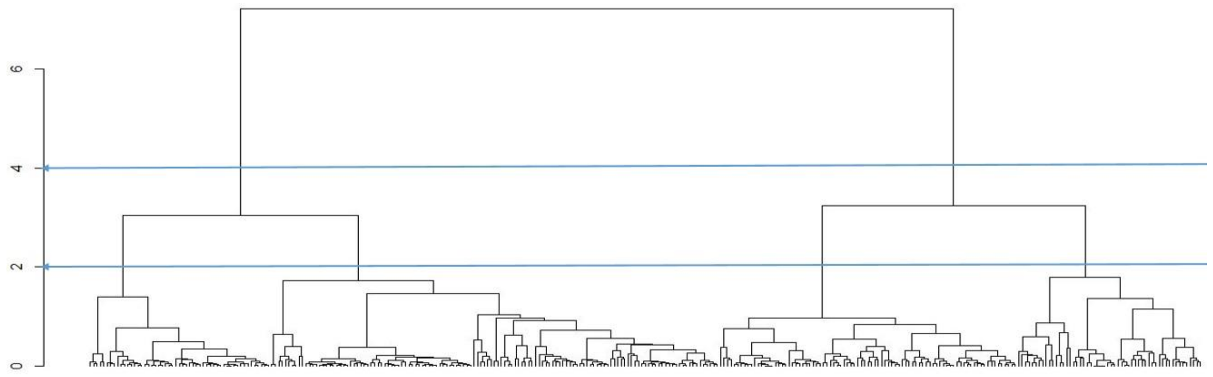


Figure 3.9 The dendrogram for the hierarchical clustering of experiment one

Figure 3.10 shows a plot for the inconsistency coefficient for all merging heights. We define a threshold for minimum, and the maximum height for cut-off. For the maximum cut-off height, we will only consider merging heights below two, as a cut-off level at the height higher than two will yield less than seven clusters. In fact, smaller clusters with more specific subdomains are always preferable. In these experiments, we make the assumption that the minimum possible number of clusters should be equal to the number of seed nodes.

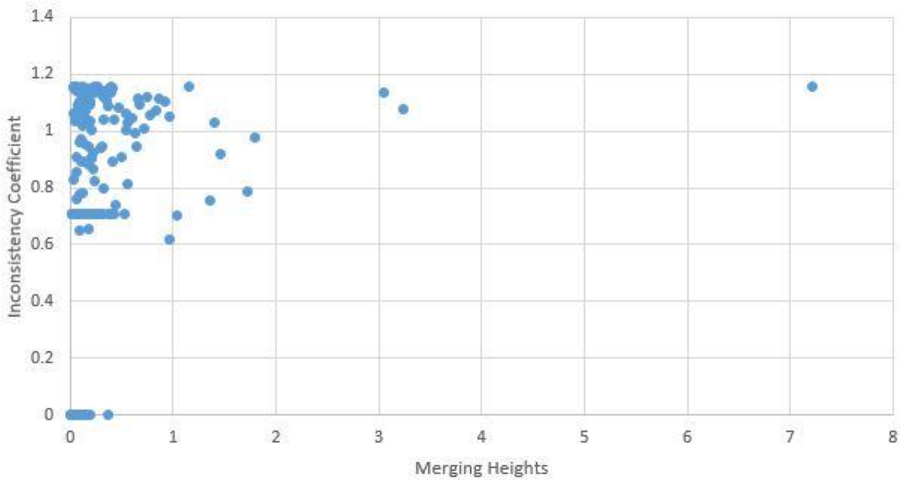


Figure 3.10 Inconsistency values for all merging heights

On the other hand, the minimum cut-off height is the smallest height that will not result in a singleton cluster (a cluster with one concept), we made the assumption that the cut-off height should not yield a cluster with one element. Figure 3.11 shows the previous graph for points within the height threshold boundaries.

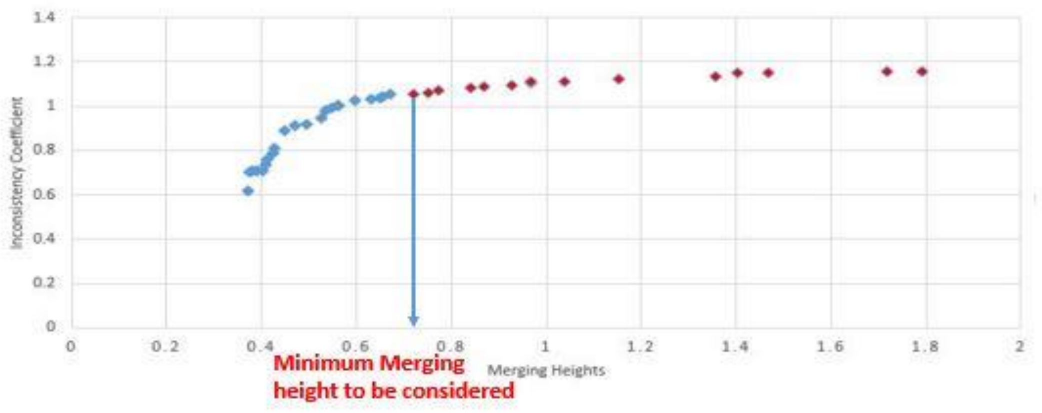


Figure 3.11 Inconsistency coefficients within the threshold boundaries

We will consider the heights corresponding to the top twenty values for the inconsistency coefficient as the candidate points to consider for cut-off heights, which represent the red dots in figure 3.11. These points will be further analyzed by computing the following validation indices, using Nbclust R package [79]:

- The Duda index [75] relies on the ratio of the sum of squared errors within clusters when the data are partitioned on a certain level, versus the sum of squared errors within clusters right before partitioning. The optimal value is maximized.
- Dunn index's [76] goal is to identify sets of clusters that are compact, with a small variance between members of the cluster, and well separated such that the means of different clusters are sufficiently far apart, as compared to the within cluster variance. The optimal value is maximized.
- Hartigan index [78] relies on the trace of the within-group dispersion matrix. The maximum difference between hierarchy levels is taken as an indication of the correct number of clusters in the data.

- Marriot index [77] is the same as the Hatigan index, but computes the determinant of the within-group dispersion matrix instead.

All of the above indices rely on the intra-cluster, and the inter-cluster dispersion, but they use different methodology to define the optimal cluster, this way we have computed the optimal cut-off from several different points of view.

In table 3.5 we show the results of computing the above mentioned internal validation indices for the candidate number of clusters. The highlighted cells represent the partition with the optimal index value.

Number of clusters	**Duda**	**Dunn**	**Hartigan**	**Marriot**
9	0.4731	0.0357	24.5565	141.6254
11	**0.8068**	0.0357	**19.5151**	116.8609
13	0.7670	0.0357	20.1820	**44.3987**
14	0.6555	0.0357	18.7003	28.2685
15	0.7378	0.0432	18.5418	24.5465
16	0.564	0.0432	16.4964	20.7977
17	0.5363	0.0432	16.2972	18.8785
19	0.2847	0.0528	14.2840	10.3920
20	0.6015	0.0528	14.2311	7.3696
23	0.5535	0.0528	13.3783	3.6994
27	0.4903	0.0602	12.4934	1.4536
30	0.5247	0.0741	10.6268	1.0649
33	0.6748	0.0745	9.4790	0.7393
35	0.2770	0.0745	9.3376	0.4579
37	0.611	0.0745	9.0212	0.3652
38	0.4331	**0.0838**	9.2314	0.3059
39	0.3624	0.0838	8.9555	0.2616
41	0.6414	0.0838	9.1223	0.2012
43	0.5566	0.0444	8.5563	0.1851
44	0.5663	0.0444	7.7729	0.1422

Table 3.5 Values for the internal validation for candidate cluster numbers

From the above results we concluded that the optimal partitioning is biased towards a small number of clusters with a large number of concepts. The two partitioning schemes with 11 and 13 clusters have a very similar distribution of concepts. So, we validated each cluster -only for the partitioning scheme with 13 clusters- against the clusters previously prepared from the book "Search Engines: Information Retrieval in Practice" (The book that got better Recall for experiment one in both cases, before and after noise elimination). The results are shown in table 3.6. For cluster validation, we will only consider concepts that are included in the book index, and we compare it with the clusters constructed by our algorithm. There is a total of 150 concepts. The data in the column "Label", represents the label of the majority cluster (compared to the book index), while the data in the total column represents the number of concepts in the cluster that belong to the index.

Cluster#	**Label**	**Total**	**Misplaced concepts**
1	Crawls and Feeds	57- concepts 14 concepts (in index)	9 misplaced concepts Search Engine (2) Information Retrieval models (3) Text Processing (3) Queries and Interface (1)
2	Text Processing	56 – Concepts 30 concepts (in index)	15 misplaced concepts Search Engine (4) Information Retrieval models(4) Crawls and Feeds (1) Social Search (1) Information Science (5)
3	Query and Interfaces	19 Concepts 16 concept (in index)	10 misplaced concepts Search Engine (5) Information Retrieval models (2) Social Search (2) Information Science (1)
4	Crawls and feeds	45 concepts 14 (in index)	9 misplaced concepts Search Engines (3) Information Retrieval models (1) Query and Interfaces (3) Social Search (2)

5	Search Engines	87 concepts 34 (in index)	22 misplaced concepts Information Retrieval Models(6) Text Processing (5) Queries and Interfaces (4) Crawls and Feeds (2) Social search (5)
6	Information Science	10 concepts 8 (in index)	4 misplaced concepts Information retrieval Models (1) Text Processing (2) Queries and Interface (1)
7	Information Retrieval Models	6 concepts 6 (in index)	2 misplaced concepts Search Engines (1) Information Science (1)
8	Information Retrieval Models	13 concepts 13 (in index)	6 misplaced concepts Search Engines (4) Queries and Interfaces (2)
9	Search Engines	5 concepts 4 (in index)	1 misplaced concept Queries and Interfaces (1)
10	Queries and Interfaces	13 concepts 6 (in index)	3 misplaced concepts Search Engines (1) Information Retrieval Models(1) Information Science (1)
11	Social search	7 concepts 1 (in index)	0 misplaced
12	NA	2 concepts none in index	0 misplaced
13	Search Engines	5 concepts 4 (in index)	1 misplaced Information Retrieval Models (1)

Table 3.6 Analysis of the Clustering Scheme (with 13 cluster).

The above data is distributed over 7 clusters labeled “Search Engines”, “Information Retrieval Models”, “Text Processing”, “Query and Interface”, “Crawl and Feeds”, “Social Search”, and “Information Science”. These topics are mapped respectively to rows x_1- x_7 in the contingency table in table 3.7. We construct a contingency table, in order to compute Rand Index, Precision and Recall.

	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}	y_{11}	y_{13}	**A**
x_1	2	4	5	3	12	0	1	4	3	1	0	3	38
x_2	3	4	2	1	6	1	4	7	0	1	0	1	30
x_3	3	15	0	0	5	2	0	0	0	0	0	0	25
x_4	1	0	6	3	4	1	0	2	1	3	0	0	21
x_5	5	1	0	5	2	0	0	0	0	0	0	0	13
x_6	0	1	2	2	5	0	0	0	0	0	1	0	11
x_7	0	5	1	0	0	4	1	0	0	1	0	0	12
b	14	30	16	14	34	8	6	13	4	6	1	4	

Table 3.7 The contingency table for the clusters of experiment one.

The rows represent ground truth clusters, and the columns represent the clusters to be validated. For validation we will compute the Rand Index, Precision.

Rand Index $=\frac{TP+TN}{TP+FP+FN+TN}$, Precision $= \frac{TP}{TP+FP}$

Total number of concepts in the extended concepts space n= 150, and all pairs of concepts N= n (n-1)/2 = 11175.

TP + FP = all positives in all clusters.

$$TP + FP = \binom{14}{2} + \binom{30}{2} + \binom{16}{2} + \binom{14}{2} + \binom{34}{2} + \binom{8}{2} + \binom{6}{2} + \binom{13}{2} + \binom{4}{2} + \binom{6}{2} + \binom{4}{2} = 1446.$$

TP represents the pairs that are of the same type.

$$TP = 7\binom{2}{2} + 7\binom{3}{2} + 6\binom{4}{2} + 6\binom{5}{2} + 2\binom{6}{2} + \binom{7}{2} + \binom{12}{2} + \binom{15}{2} = 346$$

FN (False negatives) represents pairs that should be grouped together, but they are not.

TN + FN + TP + FP = N.

TN + FN =9729.

Table 3.8 shows the values for TP, TN, FP, FN.

	Same Cluster	**Different Cluster**
Same Class	TP = 346	FN = 1550
Different Class	FP = 1100	TN = 8179

Table 3.8 Analysis of the contingency table.

Rand Index = 76.2%

Precision = 24%

In this experiment we had a low precision, and recall for the extended domain space, and for the extracted clusters, when compared to an external book index, this is due to the following observations:

- Low precision of the cluster is not an indication for a lousy clustering. For example in cluster number 1, it contained 57 concepts mostly about crawls and feed, but we only considered the 14 concepts belonging the book index (the part of the cluster that belongs to the book index). The rest mostly represent named entities and company names.
- The two selected partitioning schemes contain clusters that contain related concepts at different generality levels. Some very specific level concepts like search engine names as "Yahoo, Yandex…etc." were clustered with general concepts as "internet search indexing", which will later make it hard to explore specific relations between concepts.
- Clusters for very general concepts, such as "Information Science," are too big and include concepts from several different sub-domains.

The above results are due to the fact that the selected seed concepts were chosen without taking in consideration the perspective of the book with which we compare (The ground truth), i.e. we chose the seed nodes randomly from the domain of information retrieval. Moreover, there were no parameters to control the expansion level of each seed. These two factors were considered in experiment two.

3.6.2 The second experiment

In this experiment we tried to overcome the above mentioned problems. We selected seed nodes from the book "Search Engine: Information retrieval in Practice" in [74] , which we will be using as a ground truth for external validation. Also we added a semantic level for each seed node to indicate the level of expansion needed for each.

Seed Concept	Semantic Level
Information retrieval	2
Relevance	3
IR evaluation	3
Vector space model	3
Information retrieval applications	2
Internet search algorithms	3
Natural Language Processing	1
Document retrieval	3
Web Indexing	3

Table 3.9 Seed nodes and semantic level for experiment two

In table 3.9 we show each seed node with the corresponding chosen semantic level. The semantic level is defined such that level 3 is deeper than level 2, level 2 is deeper than level 1, etc. Semantic levels allow for a controlled expansion of the domain concepts. This way we hope to minimize the grapping of unrelated or weakly related concepts.

Once again, to generate the extended concept space, we applied the method in section 3.4.2. The extended concept space contained 357 concepts. We compared these concepts against the index of the book and the results are listed in Table 3.10.

Total Number of concepts in the extended concept space (TP + FP)	357
Total number of concepts in the book index (TP + FN)	364
Total number of concepts in the extended concept space not in the book index (FP)	152

Total number of concepts in the extended concept space and in the book index (TP)	205

Table 3.10 Analysis of the extended concept space for experiment two

Precision = 57.4%   Recall = 56.3 %

The extended concept space usually contains named entities, and weakly related concepts. The noise elimination technique in section 3.5.1 was applied and, after the noise elimination, we compared the extended concept space with the same book, and the results are in table 3.11.

Total number of concepts after noise elimination (TP + FP)	309
Total number of concepts in the book index (TP + FN)	364
Total number of concepts in the extended concept space not in the book index (FP)	117
Total number of concepts in the extended concept space and in the book index (TP)	192

Table 3.11 Analysis of the extended concept space for experiment two after noise elimination

Eliminated noise concepts = 18.6 % of the extended concept space.

Precision (after noise elimination) = 62.7 % Recall = 52.7 % (The Recall decreased because the TP concepts decreased after noise elimination).

The increase in the precision is due to selecting the seed concepts from the book index, and the addition of semantic level to control the expansion of each seed as needed.

We measured the relatedness of each concept with the seed nodes, to construct the relatedness matrix and the distance matrix. According to the value of the Cophenetic correlation between the clustering matrix, and the distance matrix, the average linkage method was found to be the best option. The dendrogram of the clustered concepts is shown in figure 3.12. In order to define a set of candidate cut-off levels, we computed the inconsistency coefficient for all merging heights of the dendrogram.

As shown in figure 3.12, we consider merging heights between 0.53 and 0.89. All heights less than 0.53 will produce some singleton clusters, and heights higher than 0.89 will produce total number of clusters less than nine, which is the number of seed nodes. According to the inconsistency analysis of these heights, the various cut-off heights that should be considered are at five different levels. We computed the same internal validation indices, as mentioned in experiment one, to choose the optimal number of clusters.

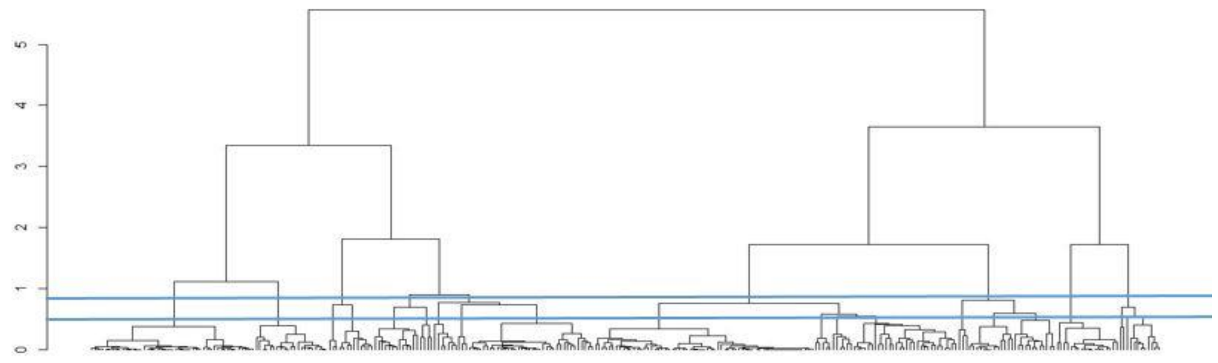


Figure 3.12 The dendrogram for the hierarchical clustering of experiment two.

The results are listed in table 3.12. Two indices, Duda, and Dunn favor a size of 10 clusters, and the other two Marriot, and Hartigan favor a size of 14 clusters.

Number of clusters	Duda	Dunn	Hartigan	Marriot
10	0.8934	0.0912	12.9554	3849.0370
11	0.7911	0.0912	23.4431	2560.8300

12	0.5982	0.0816	55.6266	1969.7542
14	0.0006	0.0816	44.8425	711.2770
15	0.2683	0.0816	2.0062	464.3295

Table 3.12 Values for the internal validation for candidate cluster numbers for experiment two

We inspected the two clustering schemes (10 clusters, and 14 clusters), we found that the scheme with 14 clusters has a better distribution of concepts than the scheme with 10 clusters. In table 3.13 we present a detailed analysis, one-by-one, for each cluster of the selected scheme. The analysis in the table represents the cluster label, total number of concepts per cluster, and the number of misplaced concepts together with their correct cluster label. We retrieved this ground truth information through the external validation with the predefined classes from the book index.

Cluster#	**Label**	**Total**	**Misplaced concepts**
1	Information Science	All Concepts =37 In index = 22	11 misplaced concepts Search Engine (2) Information Retrieval models and evaluation (4) Text processing (4) Ranking and Indexing (1)
2	Information Retrieval Models and Evaluation	All Concepts = 41 In Index = 32	16 Misplaced Search Engine (8) Text processing (5) Ranking and Indexing (2) Information Science (1)
3	Information Retrieval Models and Evaluation	All Concepts =26 In index= 21	9 Misplaced Search Engine (7) Ranking and Indexing (2)
4	Information Retrieval Models and	All Concepts = 3 In index = 3	None
5	Information Science	All Concepts = 47 In index = 18	None
6	Information Science	All Concepts = 21 In index = 14	3 Misplaced Text Processing (1) Search Engines (2)
7	Search Engines	All Concepts = 17 In index = 15	5 Misplaced Text Processing (3) Information Science (2)

8	Search Engines	All Concepts = 63 In index = 25	10 Misplaced Information Retrieval Models and Evaluation (2) Text Processing (4) Ranking and Indexing (1) Information Science (3)
9	Text Processing	All Concepts = 7 In index = 5	None
10	Information Retrieval Models and Evaluation	All Concepts = 3 In index = 3	None
11	Search Engines	All Concepts = 13 In index = 11	1 Misplaced Text Processing (1)
12	Ranking and Indexing	All Concepts = 10 In index = 6	1 Misplaced Information Science (1)
13	Text Processing	All Concepts = 3 In index = 3	None
14	Search Engines	All Concepts = 18 In index = 14	None

Table 3.13 Analysis of the Clustering Scheme (with 14 cluster)

The above data is distributed over five classes of topics, which are “Search Engines”, “Information Retrieval models and evaluation”, ”Text Processing”, “Ranking and Indexing”, and “Information Science”. These topics are mapped respectively to rows x_1-x_5 in the contingency table in table 3.14.

	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}	y_{11}	y_{12}	y_{13}	y_{14}	**a**
x_1	2	8	7	0	0	2	10	15	0	0	10	0	0	14	68
x_2	4	16	12	3	0	0	0	2	0	3	0	0	0	0	40
x_3	4	5	0	0	0	1	3	4	5	0	1	0	3	0	26
x_4	1	2	2	0	0	0	0	1	0	0	0	5	0	0	11
x_5	11	1	0	0	18	11	2	3	0	0	0	1	0	0	47
b	22	32	21	3	18	14	15	25	5	3	11	6	3	14	

Table 3.14 The contingency table to calculate the Rand Index

We follow the same procedure as in experiment 1, to find Rand Index, Precision.

n = 197, where n is the size of the extended concepts space.

N = 18336

TP + FP = 1766, then TP = 757

TN + FN + TP + FP = N

TN + FN = 16570, then FN = 1725

	Same Cluster	Different Cluster
Same Class	TP = 757	FN = 1725
Different Class	FP = 1009	TN = 14845

Table 3.15 Analysis of the contingency table for experiment two.

According to table 3.15, Rand index = 85.1%, Precision = 42.8 %.

The results of experiment two can be interpreted as follows:

- The increase of the value of the Rand Index indicates the resemblance of the partitioning scheme to the prepared clusters from the book index.
- Although the precision value is not high, with an expert inspection of each cluster, these values could be much higher. This is because, in our validation method, we only compared derived clusters against the portion of the concepts that belong to the book index. In reality, most of the concepts in each cluster are highly related (i.e. the book index is not the best choice to be used as the ground truth).
- We might need to repeat the process of selecting the seed nodes, and creating the extended concept space, multiple times (module A in figure 1.5) until we reach an acceptable precision and recall compared with the selected external validation index. For example in the first experiment, the concept "search engines" was one of the seed concepts. When generating the extended concept space, a large number of the concepts, were representing named entities related to commercial search

engines (company names, CEO names,.etc.) almost 22%, which might be of interest in a different context.

- In the second experiment, when this concept was replaced with the concept "Internet search algorithms", the named entities in the extended concept space were very few (just around 3%).

3.7 Conclusion

This chapter proposed a semi-automatic method for the acquisition of the relevant domain concepts, based on the Wikipedia link structure. Domain term acquisition is an important step in the process of ontology learning. The method includes:

- A method for computing semantic relatedness between terms.
- Clustering of concepts via hierarchical clustering.
- Defining the optimal number of clusters, by analyzing multiple cut-off levels in the dendrogram, and the computation of internal clustering indices.

We tested our model with two experiments in the domain of "information retrieval" and concluded the following:

- The quality of this process depends on the careful selection of seed nodes.
- Defining an appropriate semantic level for expansion helps in decreasing the number of weakly related concept.
- The generated extended concept space is always affected by the chosen perspective of the domain. It is worth considering the perspective, and the context of the documents to be annotated before extracting the domain knowledge concepts.

3.8 Future Work

The proposed work was developed as part of a semantic framework to build semantic knowledge needed for semantic web page annotation. The applicability of this framework depends on the existence of a visual interface that will allow the user to visualize how each seed node is expanded, the relatedness between those concepts and the concept clusters generated.

A visual interface will allow for the expert to control the generated clusters by:

- Merging or re-clustering the output clusters, if needed, and
- Inspecting and filtering out weakly related, or misplaced concepts in each cluster.

Moreover, we need to be able to compute and visualize the semantic level (general vs specific) with respect to the Wikipedia structure.

4 Skos_Wiki Ontology Learning

4.1 Introduction

“The Semantic Web is a Web of Data, however, to make the Web of Data a reality, it is important to have the huge amount of data on the Web available in a standard format, reachable and manageable by Semantic Web tools. Furthermore, not only does the Semantic Web need access to data, but relationships among data should be made available, too, to create a Web of Data. This collection of interrelated datasets on the Web can also be referred to as Linked Data”[80].

Linked Data is in the core of the Semantic Web vision, which aims for large scale integration and reasoning of data on the Web. Almost all semantic applications are based on the accessibility and the integration of Linked Data at various semantic levels. A typical case of a large Linked Dataset is DBPedia, which, makes the content of Wikipedia available in RDF. The importance of DBPedia is not only that it includes Wikipedia data, but also it incorporates links to other datasets on the Web, e.g., to GeoNames (A geographical database that covers all countries and contains over eight million place names). By providing those extra links (in terms of RDF triples) applications may exploit the extra (and possibly more precise) knowledge from other datasets when developing an application. Our main goal is to provide a lightweight ontology for specific domains that is consistent and reliable based on a semantic schema extended from an upper layer ontology SKOS and a dataset that represents the core of the Linked Data (DBpedia).

In chapter one, we mentioned the stages of the process of creating lightweight ontologies, and figure 1.5 illustrated the main modules. In this chapter, we propose our methodology for implementing module D. This Module involves the extraction of relations and properties for the clustered concepts. At this point we will rely mainly on the information already extracted in the DBpedia. Hence we will take a closer look on the strength and the weakness of DBpedia. Moreover, we will explain in more detailed fashion the classes and the properties of the SKOS, and the Skos-Wiki ontologies, and focus on mapping the definition of entities in the DBpedia to the Skos-Wiki ontology.

This Chapter is organized as follows: in section 4.2 we give a detailed background about the SKOS ontology, and strengths and weakness of the DBpedia. In section 4.3 we revisit the Skos-Wiki in more depth. In Section 4.4 we propose a methodology for creating domain specific instances for the Skos-Wiki. In section 4.5 we illustrate the experimental work, and some samples of the generated instances. Lastly, sections 4.6 and 4.7 will give our conclusions, and suggested future work to pursue in the direction of ontology learning and extraction of semantic relations.

4.2 Background

4.2.1 SKOS Revisited

The SKOS is a common data model for knowledge organization systems such as thesauri, classification schemes, subject heading systems and taxonomies. By using SKOS, a knowledge organization system can be expressed as machine-readable data. It can then be exchanged between computer applications and published in a machine-readable format on the Web (Miles 2009). We discuss the definition of the SKOS ontology in some detail, as explained in the W3c recommendations [11].

SKOS has three main classes; skos:Concept, skos:ConceptScheme, skos:Collection, all of type owl: class. Owl W3C specifications are in [81]. The skos:ConceptScheme is an aggregation of instances of skos:Concept, together with the semantic relations between these concepts. The skos:inScheme is an owl:ObjectProperty, and skos:hasTopConcept defines one or more concepts in the scheme as being at the top level in the hierarchy. The Class skos:Collection is mainly for defining list-like knowledge organizations, like Gazetteers.

The SKOS defines 3 main types of properties:

- Lexical Labels: These are skos:prefLabel, skos:altLabel, and skos:hiddenLabel. All are instances of owl:AnnotationProperty. These properties define classes of different types of labeling. All are pairwise disjoint properties, and a resource has no more than one value of skos:prefLabel. The domain of these properties is the class of all resources (rdfs:Resource), and the range is the RDF plain literals.
- Notations: Although the property, skos:notation, is used for unique identification of a concept, the SKOS ontology does not define any constraints for that. Thus, the SKOS allows a concept to have multiple notations and a notation can be given to multiple concepts. The domain of these properties is the class of all resources (rdfs:Resource), and the range is the RDF plain literals.

- Documentation Properties: Also known as note properties, they are used to provide information relating to SKOS concepts. There is no restriction on the format or the nature of this information, e.g., it could be plain text, hypertext, or an image. It could be a definition, information about the scope of a concept, editorial information, or any other type of information.
- Semantic Relations: SKOS defines two types of semantic relations: associative and hierarchical:
    - Associative relations: The only associative relation is skos:related, which is used as a link between two SKOS concepts.
    - Hierarchical relations: hierarchical link between two concepts indicates that one is in some way more general than the other. SKOS hierarchical properties are: skos:broader and skos:narrower. These are used to assert a direct hierarchical link between two SKOS concepts (it is not a transitive relation). A triple "<A> skos:broader <B>" asserts that 'B', the object of the triple, is a broader concept than 'A', the subject of the triple. Similarly, a triple "<C> skos:narrower <D>" asserts that 'D', is a narrower concept 'C'.
- Transitive hierarchical relations: SKOS defines the properties skos:broaderTransitive and skos:narrowerTransitive for transitive relations. A triple "<A> skos:broaderTransitive <B>" represents a direct or indirect hierarchical link, where 'B' is a broader "ancestor" of 'A'. Similarly a triple "<C> skos:narrowerTransitive <D>" represents a direct or indirect hierarchical link,

where ‘D’ is a narrower "descendant" of ‘C’. These properties are used to infer the transitive closure of the hierarchical links, which can then be used to access direct or indirect hierarchical links between concepts.

Some facts about the SKOS ontology are:

- skos:Concept, skos:ConceptScheme are two disjoint classes.
- skos:Concept can take part in zero, one or more skos:ConceptScheme.
- There is no way to close the boundary of a concept scheme. While it is possible, using skos:inScheme, to say that SKOS concepts B, C and D take part in concept scheme A, there is no way to say that only B, C and D take part in A.
- Two instances of skos:Concept can be related, without belonging in the same skos:ConceptScheme.
- The owl:ObjectProperty skos:inScheme has no domain, i.e. the domain is the class of all resources (rdf: resource), this gives the flexibility to define anything as a part of a scheme.

4.2.2 DBpedia Revisited

The DBpedia is a community project that extracts structured, multilingual knowledge from Wikipedia and makes it freely available using the Semantic Web and Linked Data standards. This project started in 2006. The goal of the DBpedia was to parse the information in different Wikipedia Infobox templates, into RDF triples. However, DBpedia ontology schema was not developed until late 2010. The ontology organizes the knowledge on Wikipedia in 320 classes, which form a subsumption hierarchy, and are described by 1,650 different properties.

The DBpedia Ontology is a shallow, cross-domain ontology, which has been collaboratively and manually created based on the most commonly used Infoboxes. Unfortunately, there were a lot of inconsistencies in the DBpedia triples. The inconsistencies arise from the case that DBpedia ontology has no domain and range restriction for a number of properties and no class disjointness axioms included. Moreover, the properties names, and values are based on parsing the Wikipedia templates, which are sometimes different for the same entity type.

The inconsistencies in the DBpedia are either on the modeling level, or on the data level. Modeling inconsistencies are the inconsistencies in the DBpedia schema. For example, the class "Person" in DBpedia has a property that denotes the date of birth. This property it sometimes referred to as "DBpedia:birthDate", "DBpedia:dateofbirth" or "DBpedia:birthday".

Also an instance of the class "DBpedia:Person", is sometimes defined as an instance of the class "DBpedia: person", or "DBpedia:Natural_person" or "DBpedia:Agent". Another example, which we faced when trying to import the DBpedia definition of entities, is that DBpedia can have multiple classes with the same name in different semantic levels in the hierarchy. For example the class "Organization" is defined as a subclass of "Agent", and as a main class (Subclass of Thing). Since OWL does not allow this, the DBpedia defines it by using two different spelling, "Organization" and "Organisation", each with different subclasses. The data inconsistencies refer to wrong or incomplete data values for class properties, which result during the parsing of the Wikipedia template.

On the other hand, The DBpedia includes 1.8 billion facts , linked to more than 30 other data sets in the Linked Open Data (LOD) cloud [82], which makes it the most important data set in the Linked Data. Hence, we rely on the DBpedia for defining the Wiki_class (named entities).

In the next section, we explain how we inherit the definition of the named entities in the DBpedia, and at the same time provide a consistent ontology definition

4.3 SKOS-based Annotation Schema: Skos-Wiki

SKOS, a W3C recommendation, was designed as a general schema for knowledge organization systems with highly interoperable and simple semantic structures. It lacks specialized features to represent some of the details needed for defining a lightweight ontology for a specific domain. The Skos-Wiki is a semantic annotation model, which is an extension to the SKOS schema to accommodate the definition of lightweight ontologies, not just thesaurus and taxonomies. Also it inherits the DBpedia definition of named entities. This section presents the detailed version of the SKOS-Wiki schema. An overview of the schema definition was illustrated in figure 2.4.

4.3.1 Skos-Wiki Classes

The classes that are a part of the Skos-Wiki are skos-wiki: Wiki-concept, and the skos-wiki: Wiki-class, skos-wiki: Domain-Knowledge, skos-wiki:Wiki-Person, skos-wiki:Wiki-Organization, and skos-wiki: Wiki-Place. All are illustrated in Figure 4.1, as an extension of the SKOS. The description of each class is as follows:

- **skos-wiki:Wiki-class**: is a subclass of skos:concept. This class represent all entities that can be enumerated, (named entities), as places, people, organization, .etc.
- We defined these three essential subclasses of skos-wiki:Wiki-class because they exist in most domains, which are: skos-wiki:Person, skos-wiki:Place, skos-wiki:Organization. These concepts are a subset of the named entities defined in the DBpedia ontology. They were defined as owl: equivalentclass. This is a built-in property that links a class description to another class description. Hence we link the descriptions of all subsets of skos-wiki:Wiki-class to their corresponding ones in the DBpedia Ontology. The reason we created a separate entity skos-wiki:Wiki-class, when we want to import the named entities in DBpedia, is to be able to create axioms and consistency rules over specific relations to overcome the inconsistency problem in the DBpedia schema.

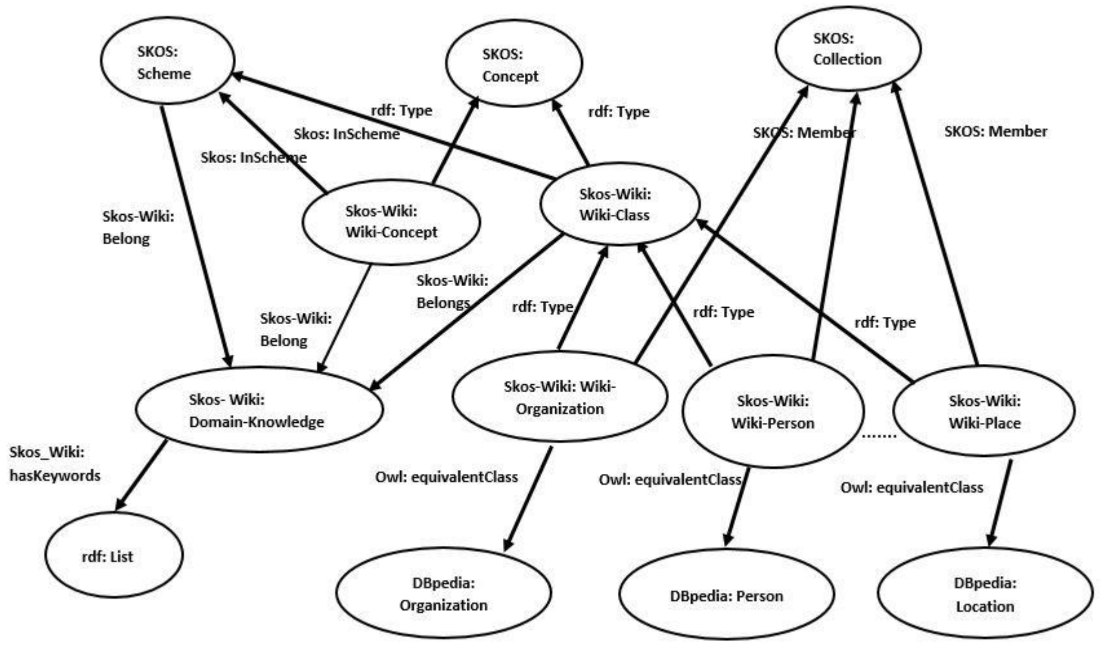


Figure 4.1 Detailed skos-wiki Schematic diagram.

- **skos-wiki:Wiki-Concept**: is a subclass of skos:Concept. This class represent all other domain concepts (that can not be enumerated).
- Both Wiki-class and Wiki-concept are modeled so one can distinguish between a named entity and a concept in a given domain.
- **skos-wiki:Domain-knowledge**: is a class to define the domain name, and concepts. The domain knowledge can consist of a set concepts, a set of schemes or both.

4.3.2 Skos-Wiki Properties

We present here all the properties used in the above schema, and their domain and range. Some of these properties are already defined in the SKOS schema, some are part of the OWL language, and some are exclusive to the Skos-

Wiki ontology. In order to ensure consistency, we define domain and range for all properties, including those inherited from SKOS ontology. The properties are defined are as listed in table 4.1

skos-wiki:belong , is an skos-wiki relation to define, which concepts belong to which domain. Also it defines the subdomains represented by skos:Conceptscheme within each domain.
domain : skos-wiki: wiki-class union skos-wiki: wiki-concept union skos:Conceptscheme
range : skos-wiki:Domain_knowledge
skos-wiki:hasKeywords, is a relations that associate a domain knowledge with a list of keywords, that can identify the domain. This relation is to make the distinguishing between two domains faster. So to check domain relevance one can just query the skos-wiki:hasKeywords and the skos:preflabel (will be defined shortly).
domain: skos-wiki:Domain_knowledge.
range: rdf:List.
skos: related, this is a sub-relation from the skos:SemanticRelation, we use it to define that two concepts are related.
domain: skos:Concept.
range: skos:Concept.
skos:member, we adopted this relation to be able to define a list of named entities (a gazateer).
domain: skos-wiki: Wiki-class.
range: skos:Collection.
skos:inScheme, we adopted this relation to define which concepts belong to which conceptscheme. A skos:Conceptscheme represents a subdomain in a domain knowledge (skos-wiki:Domain-knowledge).
domain: skos:Concept including (skos-wiki: Wiki-concept, skos-wiki:Wiki-class).
range: skos:ConceptScheme.
skos:prefLabel, this is a labeling property for all objects of type skos:concepts, skos-wiki:Domain_knowledge.
domain: skos:concepts (including subconcepts) union skos-wiki: Domain_knowledge.
range: xsd:string.

Table 4.1 Skos-Wiki properties

Here are some additional facts to consider:

- **skos:semanticRelations** : is an SKOS relation between any two concepts, and is the parent of all semantic relations defined earlier in the SKOS schema. We extend a sub-relation, skos:broader and skos:narrower to be defined between knowledge domains. So we can say "<A> skos:broader <B>," where A, B are domain knowledge. This means that domain knowledge A is more general than domain knowledge B, or B is more specific than A.
- **Equivalence:** We link the named entity classes and the properties to DBpedia via the owl:equivalentClass axiom. As an example the class skos-wiki:Wiki-Place is defined as follows:

```
<EquivalentClasses>
    <Class IRI="http://dbpedia.org/ontology/Place"/>
    <Class IRI="#Wiki_Place"/>
  </EquivalentClasses>
```

  Equivalent classes refer two different entities, but with the same definition. We allow addition of more restrictions on skos-wiki:Wiki-Place, such as disjoint axioms with all other named entities.
- **Domain and range:** In most of the properties and relations, the SKOS ontology does not specify domain or range, or sometimes both. They do that to ensure flexibility. But since we are extending the SKOS to add more specific types, we find it crucial to specify the domain and the range for each concept.

- The development of any ontology, especially a multi-domain ontology as the Skos-Wiki requires an incremental effort. The specifications we provide at this time are sufficient to the experiments we are performing, but much more named entities should be added in future, as needed.

4.4 Creating Instances of Skos-Wiki

In chapter 3, we defined a Wikipedia-based method to extract domain concepts, and define subdomains via hierarchical clustering. Recall figure 1.5, in chapter 1, where the module D represents the phase of extracting concepts relations and properties. The focus of this section is on expanding on that module.

Entity recognition and relation extraction is a wide area, and extensive efforts based on statistical methods and/or NLP tools have been done in this direction. In this chapter, we will focus on extracting hierarchical relationships from the Wikipedia semantic network, and will make use of the existing relations and properties in the DBpedia triple store. Extracting relations from natural language text, is one of our goals that we will pursue as part of our future plans for this framework.

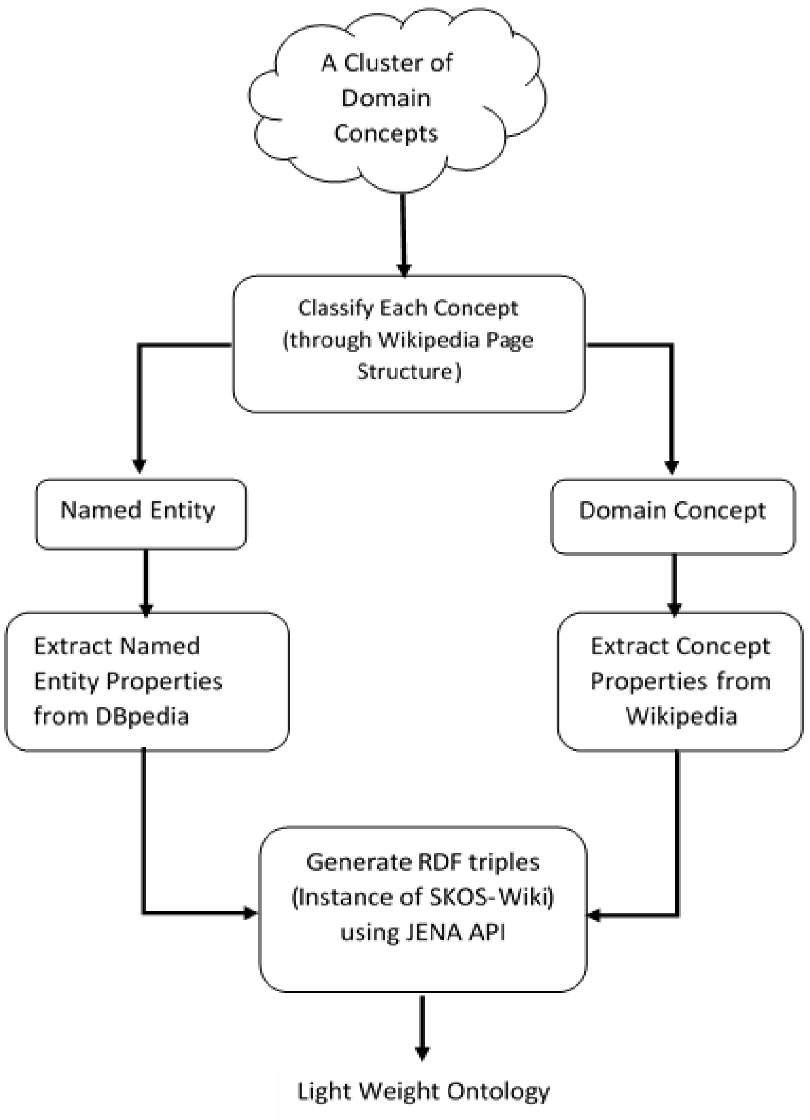

A Cluster of Domain Concepts
Classify Each Concept (through Wikipedia Page Structure)
Named Entity
Domain Concept
Extract Named Entity Properties from DBpedia
Extract Concept Properties from Wikipedia
Generate RDF triples (Instance of SKOS-Wiki) using JENA API
Light Weight Ontology

Figure 4.2 the process of creating instances of Skos-Wiki.

Figure 4.2 illustrates the flow of the proposed method. We will process one concept cluster at a time. We start by classifying the domain concepts according to the proposed semantic model into skos-wiki:wiki-concept or skos-wiki:wiki-class (named-entity). Then, for the concepts that are named entities, we identity the type for each (to be explained in section 4.4.1). Then for each named entity, we query the

Dbpedia to extract the URI of the concept definition, using a set of predefined SPARQL queries (explained in section 4.4.2). For each domain concept, we define the directly related concepts and their direct parents from the Wikipedia semantic network. Following that, we extract the definition of the concept, if available, (explained in section 4.4.3). In each of the previous two stages, the output will be presented as OWL instances of the Skos-Wiki ontology. The Jena API is what we used to generate those instances.

4.4.1 Classification of the Domain Concepts

Wikipedia pages, for named entities, has a fact sheet called Infobox. The template of an Infobox differs according to the type of entity. Figure 4.3 shows Infoboxes for two types of named entities (person, and place).

The Algorithm for the classification of concept X works as follows:

1. Parse the Wikipedia page representing a concept

   If the page does not include an infobox then:

   - it is of type Wiki-Concept
   - Generate an RDF triple : X rdf:type skos-wiki: Wiki-Concept

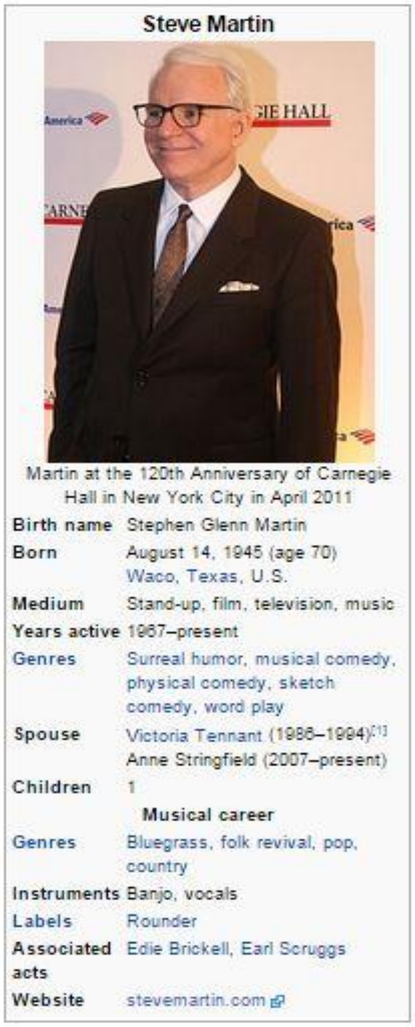
   Else

2. If the page includes an infobox then:

   - Parse the infobox title to extract the entity type.
   - Execute the SPARQL queries 4.1, 4.2, and 4.3 and compare the output with the entity type, to extract skos-wiki type and DBpedia type.

- Generate RDF triple “X rdf:type skos-wiki:Wiki-Person” or “X rdf:type skos-wiki:Wiki-Place” or “X rdf:type skos-wiki:Wiki-Organization”.

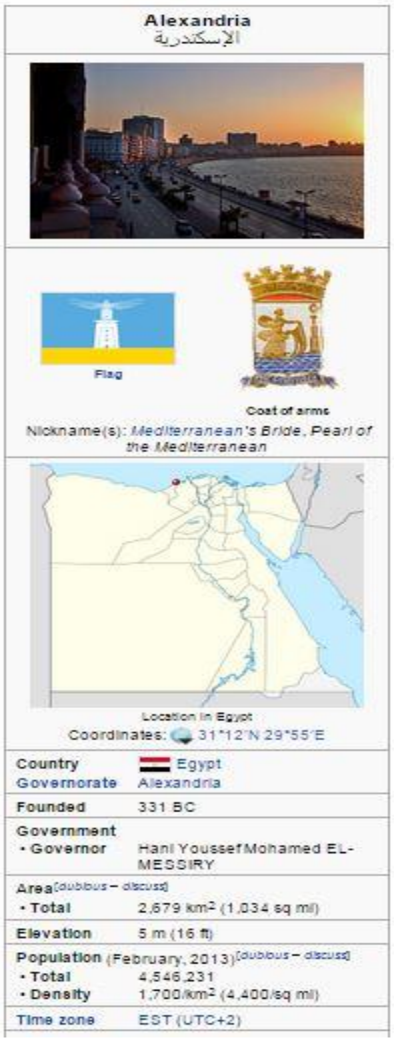
To extract all subclasses of dbpedia:Person, dbpedia:Place, dbpedia:Organization, we use the SPARQL queries in query 4.1, 4.2, 4.3 respectively.



Steve Martin

Martin at the 120th Anniversary of Carnegie Hall in New York City in April 2011

Birth name	Stephen Glenn Martin
Born	August 14, 1945 (age 70) Waco, Texas, U.S.
Medium	Stand-up, film, television, music
Years active	1967–present
Genres	Surreal humor, musical comedy, physical comedy, sketch comedy, word play
Spouse	Victoria Tennant (1986–1994)[1] Anne Stringfield (2007–present)
Children	1
Musical career	
Genres	Bluegrass, folk revival, pop, country
Instruments	Banjo, vocals
Labels	Rounder
Associated acts	Edie Brickell, Earl Scruggs
Website	stevemartin.com



Alexandria
الإسكندرية

Flag

Coat of arms

Nickname(s): *Mediterranean's Bride, Pearl of the Mediterranean*

Location in Egypt

Coordinates: 31°12′N 29°55′E

Country	Egypt
Governorate	Alexandria
Founded	331 BC
Government	
• Governor	Hani Youssef Mohamed EL-MESSIRY
Area[dubious – discuss]	
• Total	2,679 km^2 (1,034 sq mi)
Elevation	5 m (16 ft)
Population (February, 2013)[dubious – discuss]	
• Total	4,546,231
• Density	1,700/km^2 (4,400/sq mi)
Time zone	EST (UTC+2)

Figure 4.3 Wikipedia Infobox templates for entity type person and place

SELECT ?class ?label WHERE { 4.1

?class rdfs:subClassOf <http://dbpedia.org/ontology/Person>.

```
    ?class rdfs:label ?label.
    FILTER(lang(?label) = "en")
}
```

```
SELECT ?class ?label WHERE {                                          4.2
    ?class rdfs:subClassOf <http://dbpedia.org/ontology/Place>.
    ?class rdfs:label ?label.
    FILTER(lang(?label) = "en")
}
```

```
SELECT ?class ?label WHERE {                                          4.3
    ?class rdfs:subClassOf
<http://dbpedia.org/ontology/Organization>.
    ?class rdfs:label ?label.
    FILTER(lang(?label) = "en")
}
```

If the page does not include an infobox, we classify the concept as a skos-wiki:Wiki-concept. A cluster is defined as a skos:ConceptScheme and all concepts are related to the scheme with skos:inScheme relation. Samples will be used to illustrate this process in the experimental work section.

4.4.2 Creating Instances of skos-wiki:Wiki-class

In the previous section, we introduced a method to classify the DBpedia type by comparing the Infobox title to the DBpedia subclasses (Subclasses of dbpedia:Person, dbpedia:Place, dbpedia:Organization). Next, we need to link the Skos-Wiki concept generated in the previous section to its's URI definition in the DBpedia ontology. We extract the URI of the DBpedia instance using SPARQL query 4.4. Given X the entity name, and Y the DBpedia entity type, we extract the URI of the entity in the DBpedia, Z.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>                    4.4
 PREFIX dbo: <http://dbpedia.org/ontology/>
  SELECT ?x
  WHERE {
      ?x a dbo:Y .
      ?x rdfs:label ?name .
      FILTER(bif:contains(?name, X))
  }
```

4.4.3 Creating Instances of skos-wiki:Wiki-concept

The definitions of concepts in DBpedia are not complete and represent a mirror of the text in the Wikipedia page, .i.e. no semantic relations can be extracted. Hence, if a concept is classified as a skos-wiki:Wiki-concept, we expect it will be more appropriate to extract the properties from the Wikipedia semantic network. The introduction section to be mapped to the property skos:definition (a subproperty of skos:note).

We extract the following properties for each concept X:

- A parent category Y, which will be mapped to the property skos:braoder, such that (Y is broader than X).
- The Concept name, which will be mapped to the property skos:prefLabel.
- All the out-link concepts (only in the introduction section), to be mapped to the property skos:related.
- The introduction section to be mapped to the property skos:definition (a subproperty of skos:note).

For each cluster A:

- A cluster is mapped to a skos:ConceptScheme
- The cluster label is mapped to skos:prefLabel for the scheme.
- Each concept in a cluster is related to the corresponding scheme with a skos:inScheme relation.

4.5 Experimental Work

We carried out the experiments using the clustered concepts formed in the previous chapter. We selected two clusters (cluster 1 and cluster 3), from the output of the Experiment two in the previous chapter. We analyze them according to the process in figure 4.2. Named entities are not common in the domain of interest; hence, we cannot find a cluster with multiple named entity types, to demonstrate.

We first classify them according to the method in section 4.4.1. Table 4.1 shows the result of the classification. The symbol sw, stands for skos-wiki, and the symbol dbo stands for DBpedia ontology. To extract all subclasses of dbpedia:Person, dbpedia:Place, dbpedia:Organization, we use the SPARQL queries in query 4.1, 4.2, 4.3 respectively.

Cluster 1 "Information Science"	Entity Type	Cluster 3 "Information Retrieval and Evaluation"	Entity Type
Preservation_library_and_ archival_science	sw:Wiki-Concept	Discounted_cumulative_gain	sw :Wiki-Concept
Information_technology	sw:Wiki-Concept	Latent_semantic_analysis	sw :Wiki-Concept
Categorization	sw:Wiki-Concept	Temporal information_ retrieval	sw: Wiki-Concept
Ontology_information_science	sw:Wiki-Concept	Binary_Independence_Model	sw:Wiki-Concept
Language_model	sw:Wiki-Concept	Relevance_feedback	sw:Wiki-Concept
Philosophy_of_information	sw:Wiki-Concept	Full_text_search	sw:Wiki-Concept
Sensitivity_and_specificity	sw:Wiki-Concept	Fuzzy_retrieval	sw:Wiki-Concept
Information society	sw:Wiki-Concept	Extended_Boolean_model	sw:Wiki-Concept
Binary_classification	sw:Wiki-Concept	Probabilistic_relevance_ model	sw:Wiki-Concept
Robert_R_Korfhage	sw:Wiki-Class dbo: scientist	Information_needs	sw:Wiki-Concept
Nicholas_J_Belkin	sw:Wiki-Class dbo: scientist	Automatic_Content_ Extraction	sw:Wiki-Concept
As_We_May_Think	sw:Wiki-Class NA	Cluster_labeling	sw :Wiki-Concept
Harmonic_mean	sw:Wiki-Concept	EXCLAIM	sw:Wiki-Concept
Tony_Kent_Strix_award	sw:Wiki-Class NA	Poliqarp	sw:Wiki-Concept
Social_information_seeking	sw: Wiki-Concept	Faceted_search	sw:Wiki-Concept
Hypertext	sw:Concept	Information_Retrieval_ Specialist_Group	sw:Wiki-Concept
Collaborative_information_ seeking	sw:Wiki-Concept	Negative_search	sw:Wiki-Concept
Database	sw:Wiki-Concept	Compound_term_processing	sw:Wiki-Concept
Scalability	sw:Wiki-Concept	Query_expansion	sw:Wiki-Concept

Information_extraction	sw:Wiki-Concept	Cosine_similarity	sw:Wiki-Concept
Science_technology_and_society	sw:Wiki-Concept	Term_Discrimination	sw:Wiki-Concept
Accuracy_and_precision	sw:Wiki-Concept	Query_likelihood_model	sw:Wiki-Concept
C_J_van_Rijsbergen	sw:Wiki-Class dbo: scientist	SimRank	sw:Wiki-Concept
Latent_Dirichlet_allocation	sw:Wiki-Concept	Concept_search	sw :Wiki-Concept
Melvin_Earl_Maron	sw:Wiki-Class dbo: scientist	Rocchio_algorithm	sw :Wiki-Concept
Karen_Spärck_Jones	sw:Wiki-Class dbo: scientist		
Standard_Boolean_model	sw:Wiki-Concept		
Calvin_Mooers	sw:Wiki-Class dbo:Agent		
Information_management	sw:Wiki-Concept		
Vannevar_Bush	sw:Wiki-Class dbo: scientist		
Curse_of_dimensionality	sw:Wiki-Concept		
Ranking	sw:Wiki-Concept		
Lexical_database	sw:Wiki-Concept		
WordNet	sw:Wiki-Concept		
Preservation_library_and_ archival_science	sw:Wiki-Concept		

Table 4.2 Classification of concepts in cluster one and cluster 3 in experiment two.

Considering the results, in the above table we will highlight the following:

- The above results pertaining to concept classification distinguish between a class and a concept, and show the type of the class concepts, as defined in the DBpedia.
- The highlighted concept, “Information Retrieval Specialist Group” is the only misclassified concept. This concept refers to an organization representing an

information retrieval group in the UK. Thus it should be classified as belonging to the class of type organization. But the Wikipedia page does not have an organization Infobox template in this page, so neither our method nor the DBpedia was able to recognize it as a class of that type.

- The other two highlighted concepts “As_we_may_think” and “Tony_Kent_Strix_award”, were correctly classified as concepts of type class, but we were not able to pick the correct DBpedia classification, because so far, we only handle three types: Person, Place, and Organization. However these concepts are classified in the DBpedia as article, and award respectively.

Following the classification phase, is the creation of the instances as explained in section 4.4.2, and 4.4.3. In table 4.3, and table 4.4 we will present some samples of the generated triples, for the skos-wiki: wiki-concept, skos-wiki: wiki-class.

Accuracy_and_precision	<Accuracy_and_precision> <rdf: type> <skos-wiki: Wiki_concept> <Accuracy_and_precision> <skos:prefLabel> <” Accuracy_and_precision”> <Accuracy_and_precision> <skos:broader> <”IR Evaluation”> <Ranking> <skos:broader> <” ”> <Accuracy_and_precision> <skos:definition> <Accuracy and precision are defined in terms of systematic and random errors. The more common definition associates accuracy with systematic errors and precision with random errors. Another definition, advanced by ISO, associates trueness with systematic errors and precision with random errors, and defines accuracy as the combination of both trueness and precision.>
Ranking	<Ranking> <rdf: type> <skos-wiki: Wiki_concept> <Ranking> <skos:prefLabel> <” Ranking”> <Ranking> <skos:broader> <” ”> <Ranking> <skos:related> <” ”> <Ranking> <skos:related> <” ”> <Ranking> <skos:related> <” ”> <Ranking> <skos:definition> <A ranking is a relationship between a set of items such that, for any two items, the first is either 'ranked higher than', 'ranked lower than' or 'ranked equal to' the second.[1] In mathematics, this is known as a weak order or total preorder of objects. It is not necessarily a total order of objects because two different objects can have the same ranking. The rankings themselves are totally ordered. For example, materials are totally preordered by hardness, while degrees of hardness are totally ordered.

	By reducing detailed measures to a sequence of ordinal numbers, rankings make it possible to evaluate complex information according to certain criteria. Thus, for example, an Internet search engine may rank the pages it finds according to an estimation of their relevance, making it possible for the user quickly to select the pages they are likely to want to see. Analysis of data obtained by ranking commonly requires non-parametric statistics.>
Extended_Boolean_model	< Extended_Boolean_model > <rdf: type> <skos-wiki: Wiki_concept> < Extended_Boolean_model > <skos:prefLabel> <" Extended_Boolean_model"> < Extended_Boolean_model > <skos:broader> <" Information_retrieval_ techniques "> < Extended_Boolean_model > <skos:related> <"Standard Boolean model "> < Extended_Boolean_model > <skos:related> <"Vector Space Mode"> < Extended_Boolean_model > <skos:related> <"relevance feedback"> < Extended_Boolean_model > <skos:related> <" query expansion"> < Extended_Boolean_model > <skos:related> <" "> < Extended_Boolean_model > <skos:definition> <"The Extended Boolean model was described in a Communications of the ACM article appearing in 1983, by Gerard Salton, Edward A. Fox, and Harry Wu. The goal of the Extended Boolean model is to overcome the drawbacks of the Boolean model that has been used in information retrieval. The Boolean model does not consider term weights in queries, and the result set of a Boolean query is often either too small or too big. The idea of the extended model is to make use of partial matching and term weights as in the vector space model. It combines the characteristics of the Vector Space Model with the properties of Boolean algebra and ranks the similarity between queries and documents. This way a document may be somewhat relevant if it matches some of the queried terms and will be returned as a result, whereas in the Standard Boolean model it was not. Thus, the extended Boolean model can be considered as a generalization of both the Boolean and vector space models; those two are special cases if suitable settings and definitions are employed. Further, research has shown effectiveness improves relative to that for Boolean query processing. Other research has shown that relevance feedback and query expansion can be integrated with extended Boolean query processing">

Table 4.3 Samples of the instances of skos-wiki: Wiki-concept.

As shown in table 4.3, not all defined properties are extracted for each concept, and the reasons are:

- Since we enforced the definition of domain and range for each property, then if the concept in the range is not an instance of skos: concept, the relation will not be complete, resulting in inconsistent triples. This can be seen in two relations <skos: broader>, and <skos: related>.

- This usually occurs with concepts that are related, but not in the core of the domain. In the above samples, the first two concepts that have empty relations are part of the cluster "Information Science". While this is not the case with concepts in the core clusters, such as "Information retrieval".
- This problem can be temporarily avoided with the manual choice of the clusters of interest.

Robert_R_Korfhage	<Robert_R_Korfhage> <rdf: type> <skos-wiki:Wiki_Person> <Robert_R_Korfhage> <rdf: resource> < http://en.dbpedia.org/resource/ Robert_R_Korfhage >
Nicholas_J_Belkin	< Nicholas_J_Belkin > <rdf: type> <skos-wiki:Wiki_Person> < Nicholas_J_Belkin > <rdf: resource> < http://en.dbpedia.org/resource/ Nicholas_J_Belkin >
C_J_van_Rijsbergen	< C_J_van_Rijsbergen > <rdf: type> <skos-wiki:Wiki_Person> < C_J_van_Rijsbergen > <rdf: resource> < http://en.dbpedia.org/resource/ C_J_van_Rijsbergen >
Melvin_Earl_Maron	< Melvin_Earl_Maron > <rdf: type> <skos-wiki: Wiki_Person> < Melvin_Earl_Maron > <rdf: resource> < http://en.dbpedia.org/resource/ Melvin_Earl_Maron >
Karen_Spärck_Jones	< Karen_Spärck_Jones > <rdf: type> <skos-wiki:Wiki_class> < Karen_Spärck_Jones > <rdf: resource> < http://en.dbpedia.org/resource/ < Karen_Spärck_Jones >
Calvin_Mooers	<Calvin_Mooers > <rdf: type> <skos-wiki:Wiki_Person> < Calvin_Mooers > <rdf: resource> < http://en.dbpedia.org/resource/ Calvin_Mooers>
Vannevar_Bush	< Vannevar_Bush > <rdf: type> <skos-wiki: Wiki_Person> < Vannevar_Bush > <rdf: resource> < http://en.dbpedia.org/resource/ Vannevar_Bush >

Table 4.4 Samples of the instances of skos-wiki:wiki-class

The information in table 4.4 totally relies on the DBpedia definition of each named entity, which in turn, might include some errors and inconsistencies. For example, the entity named "Calvin_Mooers" was classified in the DBpedia as an

"Agent", which is an Equivalent class to the class Person, but the subclass, which is "scientist" was not defined.

4.6 Conclusion

This chapter focused on the details of the semantic schema Skos-Wiki, and the relations between all three schemas: SKOS, DBpedia, and Skos-Wiki. We presented a method for generating instances of the skos-wiki, including concepts, relations and properties. The proposed method was able to generate triples that reflect the semantic structure and relations of a domain.

The quality of this step, especially the step of creating instances of Skos-Wiki:Class, depends on the quality of the DBpedia instances, which in some cases suffers due to the presence of inconsistencies.

We discovered that adding axioms to restrict the domain and range of each relations, might result in an incomplete relation. As in the case of relations like skos: related, skos: broader. We think the resolution for this problem is either by removing the range restriction of the semantic relation, or by considering relations only between concepts in the boundaries of the extended concept space. This decision is equivalent to the choice between the generation of open domain boundaries or closed domain boundaries.

4.7 Future Work

Despite recent increase in the number of different techniques available for ontology learning for the Semantic Web, there is still a large gap between formal ontological definitions and lexical and NLP data. Thus this denotes that still there are much more semantics in the natural text, needs to be captured in the corresponding ontology. Analyzing NLP data to extract formal ontological relations, will convert the natural text into triples. This does not only require NLP, and statistical analysis of the text, but also the existence of a semantic model that will accommodate anonymous semantic relations extracted from natural text via NLP tools. We need to create templates of semantic relations, which will consider differences in lexical and morphological meaning of relations of the same type. This is a very challenging, but a promising direction to realize the vision of Linked Data and Semantic web and link actions to things.

5 Skos-Wiki Down the Road

5.1 Summary of the Proposed Contributions

The model proposed in this dissertation is an attempt to create a lightweight ontology, with domain-independent schema, while maintaining the domain boundaries. The interoperability of the generated ontology is an important requirement, so that concepts in one ontology can be linked to others of different ontologies in the Linked Data cloud. We assure interoperability by creating a multi-layer semantic schema, extending SKOS, a simple and upper level ontology, and by linking named entities to their corresponding definitions in DBpedia. We extended SKOS to build the Skos-Wiki, which yields a richer semantics and accommodates different types of concepts. Thus, we increased its semantic capabilities from defining thesaurus and taxonomies to lightweight ontologies.

The generation of ontology is based on the acquisition of domain concepts, and defining domain boundaries. We adopted the Wikipedia semantic network to generate the concept space, for a specific domain, and define semantic relatedness between concepts. Then we clustered the concept space, based on their relatedness, using hierarchical clustering. We implemented a procedure for defining the number of clusters based on the merging heights.

With the proposed approach we were able to attain clustering accuracy 85% in Rand index, and Precision of almost 43%. The phase of generating the Skos-Wiki instances, relied mostly on classifying the type of concept by parsing the Wikipedia page template. In case the concept is a named entity, we add a link to its definition in the DBpedia ontology; otherwise, we use the Wikipedia semantic network to extract instances of concepts, properties, and hierarchical relations.

5.2 Impact of the Proposed Model

The interest in publishing semantic data on the Web, and building web applications based on Semantic Web languages, has grown and matured in the last years. Mostly empowered by the Linked Data initiative. The goal of the Linked Data is to transform the Web into a global knowledge base. The term Linked Data is mostly referred to as the set of best practices for publishing and connecting structured data on the Web. In the past four years, data providers have been adopting these practices. Efforts for creating linked data are either domain-oriented, which focus on one domain, and create the appropriate semantic structure to define this domain, or generic, involving collaborations that focus on creating links between concepts and their definition in global data sets, like Wikipedia, DBpedia, WordNet…etc. The first lacks interoperability, and the second lacks the granularity.

Corcho, et al. discussed the challenges of ontology engineering in the era of Linked Data in [83]. One of the dimensions they studied was the types of ontologies that are being created, published and used. Linked Data does not impose any restrictions, on the type of ontology used to describe the data that is being published, and hence ontology modeling is being given insufficient attention in general. When thinking about data publication – sometimes we can even find cases where data refers to ontology terms that do not actually exist, or entities that are not instances of any ontology. Another possible drawback of the current Linked Data practices is that the Linked Data publishers are the ones who decide on the vocabularies to be used for annotation, and select concepts and properties from diverse ontologies, and repositories, without checking whether they are actually compatible. That is, it is unknown whether the original semantics of the reused terms are preserved in the ontology being developed.

The motivation behind the work proposed in this dissertation is to provide the methods to semi-automatically create a domain knowledge that fits in the large picture of the Linked Data. We believe that our proposed model provides an interoperable solution, while maintaining the nature, and the granularity of the domain. We designed the Skos-Wiki to extend the SKOS as an upper level ontology, without interfering in the domain vocabularies. As we illustrated throughout this study, the domain vocabulary relies on the knowledge domain we use to extract the instances from. We used Wikipedia as a community-accepted knowledge, but a sub-community can specify its own domain corpus, or a different semantic network.

On the other hand, Linked Data proponents had urged the need for more ontology engineers, who are able to generate vocabularies that can be used for creating linked data. These engineers may not have received enough training in knowledge modeling, resulting in badly designed combinations of ontologies, SKOS concept schemes and Schema.org terms or other annotation ontologies. Providing support to this type of ontology engineers and linked data producers will be one of the main challenges that the ontology engineering field will have to address in the upcoming era. We believe that frameworks that incorporate Linked Data technologies and specifications are the right way to go.

5.3 Future Work

The intension of the proposed work is to develop different interdependent modules for creating and publishing of Linked Data for the semantic web. We believe that these modules need to be framed using a Graphical user interface, that will allow for the integration of the developed modules, in addition to any new one. This tool development is a major, long-term project that will incorporate efforts in multiple disciplines. The tool should allow for:

- The visualization of ontology triples that correspond to the ontology under development.
- The iteration over the different phases of the ontology development, based on the desired quality.
- The ability to create resource links to the DBpedia. In addition to the feasibility of generating another ontology, there should be support for creating links between different domains.

Another discipline that is a part of this ongoing study is the extraction of semantic relations based on natural language processing. Thus we need to expand the Skos-Wiki model to accommodate semantic relations, not only hierarchical relations. The model should be able to decrease the large gap between formal ontological definitions, and the lexical representations and NLP data. The challenge of accommodating semantic relations extracted via statistical and/or NLP tools comes from the large diversity of these relations. We suggest the creation of template structure for different types of relations and the linking of each template to the different lexical representations in a given language. In fact, this solution should be language independent in the modeling sense, but language dependent in the phase of building lexical representation for each template. We have done some investigations in this direction, but we discovered that this is a wide area and needs to be explored as a separate research problem.

Bibliography

1. Koivunen M-R, Miller E. W3c semantic web activity. Semantic Web Kick-Off in Finland. 2001:27-44.
2. Gruber TR. A translation approach to portable ontology specifications. Knowledge acquisition. 1993;5(2):199-220.
3. Baader F. The description logic handbook: theory, implementation, and applications: Cambridge university press; 2003.
4. Studer R, Benjamins VR, Fensel D. Knowledge engineering: principles and methods. Data & knowledge engineering. 1998;25(1):161-97.
5. Allemang D, Hendler J. Semantic web for the working ontologist: effective modeling in RDFS and OWL: Elsevier; 2011.
6. Staab S, Studer R. Handbook on ontologies: Springer Science & Business Media; 2010.
7. Hepp M, Bachlechner, D., Siorpaes, K. Harvesting Wiki consensus: Using Wikipedia as vocabulary for knowledge management IEEE Internet Computing. 2007;11:54-65.
8. Zhang H, Li Y-F, Tan HBK. Measuring design complexity of semantic web ontologies. Journal of Systems and Software. 2010;83(5):803-14.
9. Sheth AP. Changing focus on interoperability in information systems: from system, syntax, structure to semantics. Interoperating geographic information systems: Springer; 1999. p. 5-29.
10. Alexander M, Steffen S. Ontology learning for the semantic web. IEEE Intelligent Systems. 2001;16(2):72-9.
11. Miles A, Bechhofer S. SKOS simple knowledge organization system reference. W3C recommendation. 2009;18:W3C.
12. Hepp M, Bachlechner, D., Siorpaes, K., editor OntoWiki: community-driven ontology engineering and ontology usage based on Wikis. international symposium on Wikis; 2006.
13. Corcho O. Ontology based document annotatiion: trends and open research problems. International Journal on Metadata, Semantics and Ontologies. 2006;1(1).
14. Schonhofen P. Annotating documents by Wikipedia concepts Web Intelligence and Intelligent Agent Technology; Sydney, Australia: IEEE; 2008.
15. Wu F, Weld, D. Automatically Refining the Wikipedia Infobox Ontology. WWW-Semantic / Data Web - Semantic Web III; Beijing, China2008.

16. Vivaldi J, Horacio, R., editor Finding Domain Terms using Wikipedia. LREC International Conference; 2010.

17. Ruiz-Casado M, Alfonseca E, Castells P. Automatising the learning of lexical patterns: An application to the enrichment of wordnet by extracting semantic relationships from wikipedia. Data & Knowledge Engineering. 2007;61(3):484-99.

18. Auer S, Bizer C, Kobilarov G, Lehmann J, Cyganiak R, Ives Z. Dbpedia: A nucleus for a web of open data: Springer; 2007.

19. Liu Q, Xu K, Zhang L, Wang H, Yu Y, Pan Y. Catriple: Extracting triples from wikipedia categories. The Semantic Web: Springer; 2008. p. 330-44.

20. Ding Y, Embley, D W . editor Using Data-Extraction Ontologies to Foster Automating Semantic Annotation. ICDE; 2006.

21. Oliveira P, Rocha J, editors. Semantic annotation tools survey. Computational Intelligence and Data Mining (CIDM), 2013 IEEE Symposium on; 2013: IEEE.

22. Andrews P, Zaihrayeu I, Pane J. A classification of semantic annotation systems. Semantic Web. 2012;3(3):223-48.

23. Chen D, Doumeingts G. European initiatives to develop interoperability of enterprise applications—basic concepts, framework and roadmap. Annual Reviews in Control. 2003;27(2):153-62.

24. Obrst L. Ontological architectures. Theory and Applications of Ontology: Computer Applications: Springer; 2010. p. 27-66.

25. Davies J. Lightweight ontologies. Theory and Applications of Ontology: Computer Applications: Springer; 2010. p. 197-229.

26. Drummond N, Shearer R, editors. The open world assumption. eSI Workshop: The Closed World of Databases meets the Open World of the Semantic Web; 2006.

27. Fonseca F, Egenhofer M, Davis C, Câmara G. Semantic granularity in ontology-driven geographic information systems. Annals of mathematics and artificial intelligence. 2002;36(1-2):121-51.

28. Park J, Ram S. Information systems interoperability: What lies beneath? ACM Transactions on Information Systems (TOIS). 2004;22(4):595-632.

29. Kim W, Seo J. Classifying schematic and data heterogeneity in multidatabase systems. Computer. 1991;24(12):12-8.

30. Baker T, Bechhofer S, Isaac A, Miles A, Schreiber G, Summers E. Key choices in the design of Simple Knowledge Organization System (SKOS). Web Semantics: Science, Services and Agents on the World Wide Web. 2013;20:35-49.

31. Gruber TR. Toward principles for the design of ontologies used for knowledge sharing? International journal of human-computer studies. 1995;43(5):907-28.

32. Töpper G, Knuth M, Sack H, editors. Dbpedia ontology enrichment for inconsistency detection. Proceedings of the 8th International Conference on Semantic Systems; 2012: ACM.

33. Wimalasuriya DC, Dou D. Ontology-based information extraction: An introduction and a survey of current approaches. Journal of Information Science. 2010.

34. Stelios K, Dimitris, A. Consensus Building in Collaborative Ontology Engineering Processes. Journal of Universal Knowledge Management. 2006;vol. 1 (no. 3):199-216.

35. Jain AK, Murty MN, Flynn PJ. Data clustering: a review. ACM computing surveys (CSUR). 1999;31(3):264-323.

36. L. Kaufman PR. Finding Groups in Data - An Introduction to Cluster Analysis New York: John Wiley & Sons, Inc.; 1990.

37. Webb AR. Statistical pattern recognition: John Wiley & Sons; 2003.

38. Diday E, Simon J. Clustering analysis. Digital Pattern Recognition: Springer; 1980. p. 47-94.

39. Fraley C, Raftery AE. How many clusters? Which clustering method? Answers via model-based cluster analysis. The computer journal. 1998;41(8):578-88.

40. Han J, Kamber M, Pei J. Data mining: concepts and techniques: Morgan kaufmann; 2006.

41. Hamerly G, Elkan C, editors. Alternatives to the k-means algorithm that find better clusterings. Proceedings of the eleventh international conference on Information and knowledge management; 2002: ACM.

42. Ester M, Kriegel H-P, Sander J, Xu X, editors. A density-based algorithm for discovering clusters in large spatial databases with noise. KDD; 1996.

43. Yook D. Decision tree based clustering. Intelligent Data Engineering and Automated Learning—IDEAL 2002: Springer; 2002. p. 487-92.

44. Xu R, Wunsch DC. Neural Network–Based Clustering. Clustering. 2009:111-62.

45. Berry MJ, Linoff G. Data mining techniques: for marketing, sales, and customer support: John Wiley & Sons, Inc.; 1997.

46. Theodoridis SaK, K. Pattern Recognition: Academic press; 1999.

47. Davies DL, Bouldin DW. A cluster separation measure. Pattern Analysis and Machine Intelligence, IEEE Transactions on. 1979;(2):224-7.

48. Rousseeuw PJ. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Journal of computational and applied mathematics. 1987;20:53-65.

49. Hazman M, El-Beltagy SR, Rafea A. A survey of ontology learning approaches. database. 2011;7:6.

50. Wang S, Zeng Y, Zhong N. Ontology extraction and integration from semi-structured data. Active Media Technology: Springer; 2011. p. 39-48.

51. Hoffart J, Seufert S, Nguyen DB, Theobald M, Weikum G, editors. KORE: keyphrase overlap relatedness for entity disambiguation. Proceedings of the 21st ACM international conference on Information and knowledge management; 2012: ACM.

52. Nastase V, Strube M. Transforming Wikipedia into a large scale multilingual concept network. Artificial Intelligence. 2013;194:62-85.

53. Wei B, Liu J, Zheng Q, Zhang W, Wang C, Wu B. DF-Miner: Domain-specific facet mining by leveraging the hyperlink structure of Wikipedia. Knowledge-Based Systems. 2015;77:80-91.

54. Pedersen T, Pakhomov SV, Patwardhan S, Chute CG. Measures of semantic similarity and relatedness in the biomedical domain. Journal of biomedical informatics. 2007;40(3):288-99.

55. McDonald S, Ramscar M, editors. Testing the distributional hypothesis: The influence of context on judgements of semantic similarity. Proceedings of the 23rd Annual Conference of the Cognitive Science Society; 2001.

56. Harispe S, Ranwez S, Janaqi S, Montmain J. Semantic Measures for the Comparison of Units of Language, Concepts or Entities from Text and Knowledge Base Analysis. arXiv preprint arXiv:13101285. 2013.

57. Panchenko A, Morozova O, editors. A study of hybrid similarity measures for semantic relation extraction. Proceedings of the Workshop on Innovative Hybrid Approaches to the Processing of Textual Data; 2012: Association for Computational Linguistics.

58. Harris ZS. Distributional structure. Papers in structural and transformational linguistics: Springer; 1970. p. 775-94.

59. Landauer TK, Dumais ST. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. Psychological review. 1997;104(2):211.

60. Lund K, Burgess C, Atchley RA, editors. Semantic and associative priming in high-dimensional semantic space. Proceedings of the 17th annual conference of the Cognitive Science Society; 1995.

61. Lemaire B, Denhiere G. Effects of high-order co-occurrences on word semantic similarities. arXiv preprint arXiv:08040143. 2008.

62. Rada R, Mili H, Bicknell E, Blettner M. Development and application of a metric on semantic nets. Systems, Man and Cybernetics, IEEE Transactions on. 1989;19(1):17-30.

63. Caviedes JE, Cimino JJ. Towards the development of a conceptual distance metric for the UMLS. Journal of biomedical informatics. 2004;37(2):77-85.

64. Wu Z, Palmer M, editors. Verbs semantics and lexical selection. Proceedings of the 32nd annual meeting on Association for Computational Linguistics; 1994: Association for Computational Linguistics.

65. Strube M, Ponzetto SP, editors. WikiRelate! Computing semantic relatedness using Wikipedia. AAAI; 2006.

66. Gordon AD. Null models in cluster validation. From data to knowledge: Springer; 1996. p. 32-44.

67. Hodge VJ, Austin J. A survey of outlier detection methodologies. Artificial Intelligence Review. 2004;22(2):85-126.

68. Grubbs FE. Procedures for detecting outlying observations in samples. Technometrics. 1969;11(1):1-21.

69. Hampel FR. The influence curve and its role in robust estimation. Journal of the American Statistical Association. 1974;69(346):383-93.

70. Miller J. Short report: Reaction time analysis with outlier exclusion: Bias varies with sample size. The quarterly journal of experimental psychology. 1991;43(4):907-12.

71. Saraçli S, Doğan N, Doğan İ. Comparison of hierarchical cluster analysis methods by cophenetic correlation. Journal of Inequalities and Applications. 2013;2013(1):1-8.

72. Martinez WL, Martinez AR. Computational statistics handbook with MATLAB: CRC press; 2001.

73. C. Manning PR, HSchütze, . Introduction to Information Retrieval. 2008.

74. Croft WB, Metzler D, Strohman T. Search engines: Information retrieval in practice: Addison-Wesley Reading; 2010.

75. Duda RO, Hart PE. Pattern classification and scene analysis. J Wiley and Sons. 1973.

76. Dunn† JC. Well-separated clusters and optimal fuzzy partitions. Journal of cybernetics. 1974;4(1):95-104.

77. Marriott F. Practical problems in a method of cluster analysis. Biometrics. 1971:501-14.

78. Hartigan JA. Clustering algorithms: John Wiley & Sons, Inc.; 1975.

79. Charrad M, Ghazzali N, Boiteau V, Niknafs A. NbClust: an R package for determining the relevant number of clusters in a data set. Journal of Statistical Software. 2014;61(6):1-36.

80. Linked Data - W3C 2015. Available from: http://www.w3.org/standards/semanticweb/data.

81. McGuinness DL, Harmelen Fv. "OWL Web Ontology Language Overview" MIT2004 [updated 12 November 2009]. Available from: http://www.w3.org/TR/owl-features/.

82. Lehmann J, Isele R, Jakob M, Jentzsch A, Kontokostas D, Mendes PN, et al. DBpedia-a large-scale, multilingual knowledge base extracted from wikipedia. Semantic Web Journal. 2014;5:1-29.

83. Corcho O, Poveda ‐ Villalón M, Gómez ‐ Pérez A. Ontology engineering in the era of linked data. Bulletin of the American Society for Information Science and Technology. 2015;41(4):13-7.

Elshaimaa Elsayed Ali. Bachelor of Computer Engineering , Arab Academy for Science and Technology, Spring 1998; Master of Science, University of Louisiana at Lafayette, Spring 2009; Doctor of Philosophy, University of Louisiana at Lafayette, Fall 2015
Major: Computer Science
Title of Dissertation: A Framework for Building Lightweight Ontologies based on Semi-Structured data for Semantic annotation
Dissertation Director: Dr. Vijay V. Raghavan
Pages in Dissertation: 148; Words in Abstract: 229

ABSTRACT

The Semantic Web vision emerged in 1999 by Tim Berners Lee. In 2001 The W3C declared the Semantic Web as the web of shared data. Hence the realization of this vision requires linking definition of concepts in web documents to a semantically integrated and interoperable structure to define a domain. This process is known as semantic annotation. However, the main challenge in the annotation process is the creation of domain ontologies.

This dissertation is proposing a framework for building lightweight domain ontologies, as instances of an interoperable semantic structure. We designed a multilayered interoperable semantic schema, to define lightweight domain ontologies. This is an extension of SKOS schema, and makes use of the Wikipedia structure for defining concepts. Also, we proposed a Wikipedia-based approach for extracting concepts for ontology learning.

We developed experimental work using a prepared set of clustered concepts in a domain of interest, which is the Information Retrieval domain. We attained almost 85% accuracy in terms of the Rand Index and a precision value of about 45%. Then we used SPARQL queries, to extract relevant concept properties from DBpedia, and map it to the definition of the modeled annotation schema. The produced triples demonstrated a

promising direction to explore for generating lightweight ontologies to be used for annotation. This is considered an important step towards the realization of the Linked Data vision of the Semantic Web.

Biographical Sketch

Elshaimaa Elsayed Ali was born on November 26th, 1975 in Alexandria, Egypt. She received a Bachelor of Science and a Master of Science in Computer Engineering from the Arab Academy for Science and Technology, Alexandria, Egypt. In 2009, she received a Master of Science in Computer Science from University of Louisiana at Lafayette. She defended her Doctor of Philosophy dissertation in Fall 2015. Her research interests include semantic web, ontology modeling, and text mining. She is a proud mother of three children.