WIKITOLOGY:
A NOVEL HYBRID KNOWLEDGE BASE DERIVED FROM WIKIPEDIA

by
Zareen Saba Syed

Thesis submitted to the Faculty of the Graduate School
of the University of Maryland in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2010

ABSTRACT

Title of dissertation: WIKITOLOGY:
A NOVEL HYBRID KNOWLEDGE BASE
DERIVED FROM WIKIPEDIA

Zareen Saba Syed, Doctor of Philosophy, 2010

Dissertation directed by: Professor Timothy W. Finin
Department of Computer Science and
Electrical Engineering

World knowledge may be available in different forms such as relational databases, triple stores, link graphs, meta-data and free text. Human minds are capable of understanding and reasoning over knowledge represented in different ways and are influenced by different social, contextual and environmental factors. By following a similar model, we have integrated a variety of knowledge sources in a novel way to produce a single hybrid knowledge base i.e., Wikitology, enabling applications to better access and exploit knowledge hidden in different forms.

Wikipedia proves to be an invaluable resource for generating a hybrid knowledge base due to the availability and interlinking of structured, semi-structured and un-structured encyclopedic information. However, Wikipedia is designed in a way that facilitates human understanding and contribution by providing interlinking of articles and categories for better browsing and search of information, making the content easily understandable to humans but requiring intelligent approaches for being exploited by applications directly.

Research projects like Cyc [61] have resulted in the development of a complex broad coverage knowledge base however, relatively few applications have been built that really exploit it. In contrast, the design and development of Wikitology KB has been incremental and has been driven and guided by a variety of applications and approaches that exploit the knowledge available in Wikipedia in different ways. This evolution has resulted in the development of a hybrid knowledge base that not only incorporates and integrates a variety of knowledge resources but also a variety of data structures, and exposes the knowledge hidden in different forms to applications through a single integrated query interface.

We demonstrate the value of the derived knowledge base by developing problem specific intelligent approaches that exploit Wikitology for a diverse set of use cases, namely, document concept prediction, cross document co-reference resolution defined as a task in Automatic Content Extraction (ACE) [1], Entity Linking to KB entities defined as a part of Text Analysis Conference - Knowledge Base Population Track 2009 [65] and interpreting tables [94]. These use cases directly serve to evaluate the utility of the knowledge base for different applications and also demonstrate how the knowledge base could be exploited in different ways. Based on our work we have also developed a Wikitology API that applications can use to exploit this unique hybrid knowledge resource for solving real world problems.

The different use cases that exploit Wikitology for solving real world problems also contribute to enriching the knowledge base automatically. The document concept prediction approach can predict inter-article and category-links for new Wikipedia articles. Cross document co-reference resolution and entity linking pro-

vide a way for specifically linking entity mentions in Wikipedia articles or external articles to the entity articles in Wikipedia and also help in suggesting redirects. In addition to that we have also developed specific approaches aimed at automatically enriching the Wikitology KB by unsupervised discovery of ontology elements using the inter-article links, generating disambiguation trees for entities and estimating the page rank of Wikipedia concepts to serve as a measure of popularity. The set of approaches combined together can contribute to a number of steps in a broader unified framework for automatically adding new concepts to the Wikitology knowledge base.

WIKITOLOGY:
A NOVEL HYBRID KNOWLEDGE BASE DERIVED FROM WIKIPEDIA

by
Zareen Saba Syed

Thesis submitted to the Faculty of the Graduate School
of the University of Maryland in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2010

To My Parents.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

World knowledge may be available in different forms such as relational databases, triple stores, link graphs, meta-data and free text. Humans are capable of understanding and reasoning over knowledge represented in different ways and are influenced by different social, contextual and environmental factors. By following a similar model, we can integrate a variety of knowledge sources in a novel way to produce a single hybrid knowledge base enabling applications to better access and exploit knowledge hidden in different forms. We have been exploring the use of Web-derived knowledge bases through the development of Wikitology [39] - a hybrid knowledge base of structured and unstructured information extracted from Wikipedia augmented by RDF data from DBpedia and other Linked Data resources.

The Web is dynamic and constantly evolving. It is not only evolving with respect to the physical composition, services, efficiency and commercial exploitation but also evolving with respect to the type of information being added to it, resulting in richer structure and content. The traditional web which was a static "read only web" permitted flow of content from the provider to the viewer evolved into Web 2.0 the "read write web" characterized by collaborative content development and sharing using blogs, photo and link sharing sites, on-line forums, bookmarking sites, social networking sites and collaborative editing using Wikis etc. One thing that

sets these "Web 2.0" sites apart from traditional Web pages and resources is that they all have rich underlying network structures that provide metadata and context. Their standard hyperlinks are enriched by social networks, comments, trackbacks, advertisements, tags, RDF data and metadata. The trend is now moving forward to the next step and that is to exploit the collaboratively developed content generated as a result of Web 2.0 applications for developing richer applications.

As the Web continues to evolve, we expect that the ways people interact with it, as content consumers as well as content providers, will also change. The result, however, will continue to represent an interesting and extravagant mixture of not only underlying networks such as networks of individuals, groups, documents, concepts, opinions, beliefs, advertisements, and scams, but also a variety of data representations such as free text, links, media, tables, triples etc. These interwoven networks of hybrid data present new opportunities and challenges for analyzing, modeling and extracting information and knowledge from them and would require new approaches for integrating information from heterogeneous sources and in heterogeneous representations such as structured, semi-structured and unstructured so that they could be directly exploited by end applications.

Some applications may require querying information in the form of a relational database or triple store, whereas, others might need to process free text or benefit from exploiting information in multiple forms using several complex algorithms at the same time. For example, in order to exploit the information available in free text and knowledge available in a triple store, a possible approach is to convert one form into another, such as using natural language processing techniques to extract

triples or facts from free text to populate a knowledge base. This will enable the applications to query the knowledge available in free text and the triple store using SPARQL-like [82] queries. However, populating a knowledge base with information extracted from text is still an open problem and active research is taking place in this direction. Furthermore, in many cases, unstructured free text may be more understandable and conceivable by humans who are the producers as well as the consumers of the web as compared to highly structured representations such as complex ontologies. We foresee the web continuing to grow as a network of heterogeneous rather than homogenous data representations. Efforts for converting unstructured data into structured representation would still blossom but will unlikely be able to completely replace or represent information contained in unstructured or semi-structured forms without any loss of information.

Another approach is to augment the information available in the form of free text with triples from the knowledge base, enabling the applications to access the knowledge by submitting free text queries to an information retrieval index. However, in this case we will lose much of the information that is available through the highly structured triple representation and other benefits such as reasoning over the knowledge.

We approach this problem in a different way and favor an approach that does not depend on converting one form of data into another and benefits from the hybrid nature of the data that is available in different forms.

Another major challenge concerning handling hybrid data besides integration of different data sources and representations is to also provide a search interface

to search over the integrated data. Most of the search engines today, support only keyword based search queries, which are easy to use, however, may not express the user requirements or intentions. The web is composed of hybrid data representations and hence there is a need to develop approaches for search engines to support hybrid queries over structured, semi-structured and unstructured data available on the web. Through the development of Wikitology knowledge base we demonstrate a novel approach for integrating different kinds of data structures and providing an integrated query interface to applications.

1.1 Advantages of Deriving a Knowledge Base from Wikipedia

There are many advantages in deriving a knowledge base from Wikipedia. We discuss different aspects of Wikipedia below which make it an attractive resource to develop a Knowledge Base.

Explicit Conceptualization: Wikipedia has explicit conceptualization. In general an article in Wikipedia excluding the redirects and disambiguation pages represents a unique concept within Wikipedia. The whole article typically talks about the main subject or concept. The URI's of Wikipedia entries serve as reliable identifiers for ontology concepts [36]. Using text similarity algorithms and other techniques, it is easy to associate a set of Wikipedia pages or concepts with a short or long piece of text.

Broad Coverage: The current English Wikipedia has over three million articles and 200 thousand categories, resulting in a concept space exhibiting broad cov-

erage and developed through a collaborative process. Wikipedia is domain independent and can be considered as the largest public ontology available on the web [36].

Easy to Understand: The intended meaning of the pages, as concepts, is self evident to humans, who can read the text and examine the images on the pages.

Consensus Based Concept Space: Cooperation between large groups is bound to issues of conflict, one way of resolving such conflicts is using group consensus for decision making. Wikipedia serves as a prominent example of a community effort to collaboratively producing and maintaining general knowledge. The case for Wikipedia is very challenging as the Wikipedian community is composed of participants from different parts of the world with different backgrounds, intents as well as attitudes. However, Wikipedia is successfully employing a system of evolving guidelines and policies, which provide principles, processes and strategies for collaboration that are created and managed by the community itself again through consensus. The same principles may be extended and adapted to other kinds of collaborative community efforts on the web as the web evolves.

Easy to Contribute and Maintain: Wikipedia provides a way to allow ordinary people to contribute knowledge as it is familiar and easy to use. Guidelines and requirements are available for editing it. The collaborative development process leads to a consensus model that is kept current and up-to-date by the community without cost to any organization and evolves through contributed

efforts.

High Quality of Articles: Evaluations have shown that the quality of the scientific entries in Wikipedia is comparable to those in more established encyclopedia such as Britannica [54], [46].

Availability and Interlinking of Hybrid Data: Wikipedia proves to be an invaluable resource for generating a hybrid knowledge base due to the availability and interlinking of structured and unstructured encyclopedic information. However, Wikipedia is designed in a way that facilitates human understanding and contribution by providing inter-linking of articles and categories for better browsing and search of information, making the content easily understandable to humans but requiring intelligent approaches for being exploited by applications directly. In case of Wikipedia, structured knowledge is available in the form of MySQL tables with metadata, inter-article links, category hierarchy, article-category links and infoboxes whereas, unstructured knowledge is in the form of free text of articles. Exploiting the structured, semi-structured and unstructured information in a useful way will expose greater knowledge hidden in different forms to applications.

Availability in Multiple Languages: Finally, Wikipedia exists in many languages with links between articles that are intended to denote the same thing, providing opportunities to use it in applications involving multiple languages.

A knowledge base incorporating information available in different forms can better meet the needs of real world applications than one focusing and exposing

knowledge in a more restricted way such as through SQL, SPARQL or keyword queries only. Exploiting Wikipedia and related knowledge sources to develop a novel hybrid knowledge base brings advantages inherent to Wikipedia. Wikipedia provides a way to allow ordinary people to contribute knowledge as it is familiar and easy to use. This collaborative development process leads to a consensus model that is kept current and up-to-date and is also available in many languages. Incorporating these qualities in knowledge bases like Cyc [61] will be very expensive in terms of time, effort and cost. Efforts like DBpedia [17], Freebase [23] and Linked Open Data [22] are focused on making knowledge available in structured forms. Wikitology hybrid knowledge base can complement these valuable resources by integrating knowledge available in other forms and providing much more flexible access to knowledge.

1.2 Thesis Statement

We can create a hybrid knowledge base from Wikipedia and other related knowledge sources by automatically generating knowledge about the world. The multiple data representations available through the hybrid knowledge base can effectively support a diverse set of common use cases such as, Document Concept Prediction, Cross Document Co-reference Resolution, Entity Linking and Interpreting Tables.

1.3 Contributions

Following are the main contributions of the thesis:

1. A Novel Hybrid Knowledge Base, Wikitology

2. Novel Approaches for solving a diverse set of common use-cases

3. A Hybrid KB Architecture, Querying Interface and API

4. Approaches for Automatic Enrichment of the knowledge base

We give an overview of the contributions below.

1.3.1 A Novel Hybrid Knowledge Base, Wikitology

Wikitology is a unique Knowledge Base that contains structured and unstructured information derived from Wikipedia and related resources. It currently contains more than 1.7 million concepts and is integrated with DBpedia Ontology, YAGO Ontology and WordNet and can be easily expanded to include other resources that already link to Wikipedia concepts. Structured information is often sparse. It can be augmented with semi-structured and un-structured information to overcome data sparseness. The knowledge available through Wikitology in multiple representations can effectively complement each other and support a wider range of applications with different needs.

1.3.2 Novel Approaches exploiting Wikitology

We have developed, implemented and evaluated novel approaches that exploit structured and unstructured knowledge available through Wikitology to effectively solve a variety of use-cases commonly defined by external groups or organizations

which include:

Document Concept Prediction: The task is to predict concepts represented in a text document. Our approach [93] exploits the structured (category links and page links graphs) and unstructured representations (free text) available through Wikitology to predict the concepts in documents by augmenting text similarity measures with spreading activation graph algorithm. Our evaluation shows that exploiting both structured and unstructured representations gives better performance [93].

Cross Document Co-reference Resolution: The task is defined as a part of ACE 2008 Annotation tasks [7] to find out if entity mentions in different documents refer to the same underlying entity. We use specialized entity documents (EDOCs) extracted from the input documents for each entity and distribute different parts of the EDOC to different components in Wikitology [38] and generate several features to represent similarity and dissimilarity between pairs of entities. Wikitology was used as a component of a broader system [64] developed by JHU Human Language Technology Center of Excellence which also incorporated feature sets from other sources. The system was evaluated on ACE 2008 data set. The evaluation showed that Wikitology based features, derived by exploiting structured and unstructured representations, effectively contributed to improving the performance of the overall system [64].

Entity Linking: The task is defined as part of Text Analysis Conference - Knowledge Base Population Track [65] to link entity mentions in documents to right

entities in the knowledge base. Wikitology was exploited using four variants of Query modules [39]. It was observed that the query module that exploited additional fields available through Wikitology and boosted certain fields achieved better performance as compared to simple query modules. Wikitology Query Interface enabled imposing structural constraints on the type of entity which also improved the results [39].

Interpreting Tables: The task is to convert information represented as tables into a linked data representation explicitly linking cell values and column headers to unique concepts in the knowledge base. In this case the input to Wikitology is a structured representation i.e. a tabular representation instead of free text as in previous three tasks. Structured data is often sparse. However, since Wikitology integrates both structured and unstructured representations, we constructed a specialized query module, to distribute different parts of the table to different fields representing free text as well as structured information in Wikitology in order to link cell values to concepts in the knowledge base. The initial cell value linking information was used as input to another algorithm to link the column header to the right class represented in one of the four vocabularies integrated in Wikitology i.e. DBpedia, WordNet, Freebase and YAGO. Our evaluation showed that Wikitology effectively supported the task of linking information in table cells as well as the column headers to the concepts in the knowledge base [94].

1.3.3 A Hybrid KB Architecture, Querying Interface and API

One of the challenges in developing a hybrid knowledge base is the selection of an appropriate data structure to represent data available in different forms and provide a query interface giving an integrated view. We present a novel architecture for representing knowledge available in different forms and an integrated query interface for querying the hybrid KB as well as a Java API, so that applications can harvest the knowledge hidden in different forms. For example, through Wikitology Query Interface it is possible to construct a hybrid query against information in the infoboxes, linked concepts as well as the free text of the article with structural constraints (for eg. spouse=Obama) and retrieve ranked results based on relevance. None of the existing resources derived from Wikipedia currently support hybrid querying with structured and unstructured knowledge. The specialized knowledge representation available in Wikitology enables applications to access and exploit hybrid data representations through a well defined, organized, general and unified query interface supporting a rich query language and returning ranked results based on relevance rather than simple boolean matches.

1.3.4 Approaches for Automatic Enrichment of the knowledge base

We have developed approaches addressing real world problems that not only demonstrate the use and value of our Wikitology KB but also directly contribute to enriching the Knowledge Base itself in different ways. The set of approaches consist of predicting categories, inter-article links and redirects for articles on new

concepts; linking to external resources through Entity Linking; generating Disambiguation Trees and predicting Page Rank of Wikipedia concepts. In addition to that, we have developed a novel approach for discovering ontology elements from Wikipedia article links which is completely unsupervised and unrestricted and does not require any training data or predefined set of relations. It assigns meaningful slot labels using WordNet and can suggest slots and fillers not existing in the extant infoboxes and hence contribute to enriching the knowledge base with new slots and fillers for entities. Existing approaches like YAGO and LEILA [91] use specialized heuristic for each kind of relation or require a seed relation to extract facts whereas, Kylin [102] learns from existing infoboxes and is therefore restricted to the kind of relations present in the infoboxes. Our approach does not require any seed and is independent of the infoboxes. We have also introduced a novel feature set composed of discovered ranked slots which performs better than other existing features sets for Entity Classification in Wikipedia. We use the ranked slot feature set to discover entity classes and the class hierarchy. The different approaches combined together can contribute to a number of steps in a broader unified framework for automatically adding an article on a new concept in Wikipedia and hence in the Wikitology KB.

1.4 Thesis Outline

The rest of the thesis is organized as follows.

- In Chapter 2 we discuss the background and review literature on Wikipedia. We discuss its structure and content in detail. We review different existing

approaches that have exploited Wikipedia as a knowledge resource. We then specifically discuss existing knowledge resources that have been derived from Wikipedia.

- In Chapter 3 we introduce the Wikitology knowledge base and give an overview of its evolution and application driven development. We present a comparison of the Wikitology knowledge base with similar existing resources, their limitations and how Wikitology knowledge base is novel and can complement existing knowledge resources.

- In Chapter 4 we present novel approaches that exploit Wikitology for solving a variety of problems and drive the incremental development of the Wikitology knowledge base. We present approaches for document concept prediction, entity classification, named entity cross-document coreference resolution, entity linking to knowledge base entities and interpreting information in tables.

- In Chapter 5, we describe in detail Wikitology's novel hybrid architecture integrating a variety of data structures and providing an integrated query interface as well as an API for the knowledge base for supporting existing as well as new approaches for solving a variety of problems.

- In Chapter 6, we present approaches for automatically enriching Wikitology knowledge base. We present a new unsupervised approach for discovering ontology elements from inter-article links in Wikipedia. We also discuss an approach for automatically generating disambiguation trees for entities. We

describe how different approaches when combined together can support a number of steps in a unified framework for enriching Wikitology with articles on new concepts.

- In Chapter 7, we discuss the different approaches to evaluating knowledge bases and our task based approach for evaluating Wikitology. We discuss and summarize the performance of Wikitology for different tasks on corpora belonging to a variety of genres.

- Finally in Chapter 8, we conclude this thesis by laying out future work.

Chapter 2

Background and Related Work

Wikipedia is an encyclopedia developed by a community of users and is freely available online. It is growing continuously and new content is being added to it daily by users around the globe. This encyclopedia comprises of millions of articles. The corpus is composed of several collections in different languages such as: English, French, German, Dutch, Chinese, Spanish, Arabic and Japanese. Each collection is a set of XML documents built using Wikipedia.

Wikipedia has gained extraordinary attention from researchers belonging to a variety of disciplines just in few years due to its structure, vast content and dynamic nature as it is growing continuously. Wikipedia has been a subject of research by scientists belonging to a wide variety of communities ranging from Social Science to Artificial Intelligence.

Wikipedia contains structured and un-structured data which is manually entered by humans. There are guidelines and policies available for editing Wikipedia however, since there is no automatic mechanism of imposing them, therefore in practice there are many inconsistencies and issues. In addition to that, Wikipedia has been designed for being used by humans, offering facilities and structure for better collaboration, browsing and representation of information, making the content more understandable and accessible to humans but requiring intelligent approaches for be-

ing directly exploited by applications. In general, making Wikipedia more machine understandable/readable requires either refining and enriching the existing content or intelligent approaches that can exploit relevant information in Wikipedia without getting significantly affected by the noise. In section 2.1 we discuss the Wikipedia structure and content in detail highlighting its usefulness and also the inconsistencies and challenges faced when using the information present in Wikipedia. In section 2.3 we give a review of the existing approaches targeted towards refining and enriching Wikipedia. In section 2.4 we give an overview of different approaches exploiting Wikipedia as a knowledge source and in section 2.5 we introduce other knowledge resources that have been derived from Wikipedia or aid in enriching Wikipedia.

2.1 Wikipedia Structure and Content

Wikipedia is organized in a way to facilitate collaboration, searching and browsing of information. It has a search interface which responds to simple search queries. It also facilitates other ways of browsing such as by using inter-article links and category hierarchy. There are different kinds of structured information available in Wikipedia, which are listed below:

1. Structure of Wikipedia Article
2. Category Hierarchy
3. Inter-article Links
4. List pages

5. Disambiguation Pages

6. Redirection Pages

2.1.1 Structure of Wikipedia Article

Wikipedia articles have structured content along with free text such as infoboxes, see-also lists, external links, section headings, article table of content, links to media and categories. Infoboxes are the most structured form of information and are composed of a set of subject-attribute-value triples that summarize or highlight the key features of the concept or subject of the article. Resources like DBpedia [17] and Freebase [23] have harvested this structured data and have made it available as triples for semantic querying.

2.1.2 Category System

Documents of the Wikipedia XML collections are organized in a hierarchy of categories defined by the authors of the articles to organize information; it cannot be considered a formal ontology. It is not complete or perfect. There are lots of inconsistencies in it and the relationships are defined loosely [108]. However, it can be considered a simple ontology due to explicit conceptualization [36]. We briefly describe general properties of Wikipedia Category hierarchy below.

Figure 2.1: Structure of a typical Wikipedia Article with Title, Contents, Infobox, Table of Contents with different sections and Categories.

2.1.3 Category Hierarchy

Categories usually fall under the Category "Categories" in Wikipedia; however, there are also disconnected categories from the large component rooted at "Categories". According to Nov, 2005 dump 1069 categories are disconnected from large component out of total 78,977 categories [53].

Thesaurus: The Wikipedia category system is a taxonomy for arranging articles into categories and sub-categories [98]. However, this taxonomy is not a strict hierarchy or tree of categories, but allows multiple categorizations of topics simultaneously, i.e., various categories have more than one super-category. It is shown that Wikipedia category system is a thesaurus that is collaboratively developed and used for indexing Wikipedia articles [53]. Multiple views of categories are present to facilitate browsing to related topics even in different domains.

Cycles: The Wikipedia Category guidelines strongly discourage cycles however, due to the absence of an automatic verification system cycles are also present in the category system [108]. As Wikipedia is edited manually, checking cycles is tedious therefore, such discrepancies might occur without the knowledge of the editors.

Main Articles: Categories may be associated with main articles on that topic, for example; the category on Biology is associated with the main article "Biology".

Loose Subsumption Relation: The Wikipedia category system does not strictly fol-

low the subsumption relation [97]. Few examples from Wikipedia 2007 are given below [97]:

- Philosophy is categorized under "Abstraction and Belief" (though it "deals-with" that) in addition to "Humanities" and "Science".
- Another example is: Geography → Geography by place → Regions → Regions of Asia → Middle East → Dances of Middle East."Dances of Middle East" is not a location.

There is no strict subsumption relation between the category and the article concept. The categories linked with articles serve more to tag articles and facilitate browsing.

Number of Super and Sub Categories: There is no restriction in Wikipedia for the number of super or sub categories a category can have.

Categories per Article: A single article can be present in many categories; the article on "George W. Bush" is present in 34 categories. There are also articles with no categories associated with them.

Article-Category Links: Articles may be linked to categories at any level in the category hierarchy though it is recommended by the Wikipedia guidelines to not associate articles with top most categories. Articles might also point to a category and its sub-category at the same time.

Administrative Categories: There are also many administrative categories present in Wikipedia. The articles may be associated with informative categories as well

Table 2.1: Wikipedia Category Statistics (Wikipedia March 2006 dump).

Measure	Value
No. categories	111287
No. articles	1079246
Avg. articles per category	25.7
Levels	14
Categories with multiple parents	76578
No. parents	23978
Avg. no. parents	2.0
Max. parents for any given child	56
No. leaf categories	87309
Avg. no. children	4.64
Max. children	1760
Avg breadth	8559.5
Max breadth	33331
Avg depth	5.8
Max depth	13
Fanout factor	0.78
Tangledness	0.69

as administrative categories for example, "Articles_with_unsourced_statements".

Categories Statistics: Yu et al. [108] discussed some general characteristics of Wikipedia Categories based on Wikipedia March 2006 dump which are given in Table 2.1. The different observations regarding the category hierarchy are given below [108]:

- Wikipedia categories to article ratio is 1:10
- Number of levels in Category hierarchy is 14
- Quite broad with Avg. breadth at a given level = 8559.5
- Number of Wikipedia categories having multiple parents = 69
- The co-occurrences of categories within individual articles have a power-law distribution and when mapped reveal the nicely clustered semantic structure of Wikipedia [53].

Category Structure w.r.t Clustering Properties: Capocci et al. [26] studied the Wikipedia

Category Structure and Clustering properties of the underlying graph. They partitioned the graph based on the built-in Categories and the one obtained by partitioning algorithm by considering the article links. The category size distribution and the cluster size distribution were quite similar however, the categorization of Wikipedia articles was found to be quite different from that obtained by clustering, which showed that all links in Wikipedia do not necessarily imply similarity or relatedness relation. They did not use edge weights for the clustering algorithm, using edge weights based on cosine similarity of articles may help in finding better clusters.

2.1.4 Inter-Article Links

The articles within Wikipedia are inter-linked. It has been observed that the Wikipedia article links graph generally resembles the World Wide Web graph [110]. Voss analyzed the link structure of Wikipedia March 2005 snapshot [98]. The analysis showed that distribution of in-links and out-links follows power law and the links form a scale-free network. Bellomi et al. worked on finding most authoritative pages in different areas like People, Countries, Cities etc., using HITS and PageRank algorithms [19].

Wikipedia articles form a single connected graph without isolated components or outliers [100]. Few nodes have a very high degree and many with a low degree of links shows power law distribution, especially for backward links [75]. It has also been reported that Wikipedias in different languages are only loosely interlinked

[53].

2.1.5 Lists

Wikipedia contains lists pages, which usually contain links to instances of some concept, for example, "list of actors" will point to articles on actors [28]. Chernov et al. employed List pages in Wikipedia as Categories as they contain links to instances of particular concept that can be identified by analyzing the title of the list [28].

2.1.6 Disambiguation Pages

An interesting structure present in Wikipedia is the disambiguation page. Disambiguation pages enlist ambiguous names that refer to two or more entities within Wikipedia. For example, the disambiguation page on George Harrison enlists 12 persons referred to by the same name.

2.1.7 Redirects

A redirect page exists for alternative spellings of a name that can be used to refer to an entity in Wikipedia. For example the article on "Condoleezza Rice" has several redirects to it such as "Condoleeza Rice", "Condoleza rice", "Condie Rice" etc.

2.1.8 Other Content and Data

Wikipedia also has other content such as user pages, talk pages, discussion pages, revision history and meta data about articles.

2.2 Wikipedia in-built Search

Wikipedia has a search interface for its users however, the search capabilities are limited to title and full text search [28].

2.3 Existing Approaches for Refining and Enriching Wikipedia

In this section we review the existing work on refining and enriching different information structures available in Wikipedia.

2.3.1 Refining and Enriching Wikipedia Category Hierarchy:

Yu et. al. applied a method for eliminating tangledness in Wikipedia Categories by using Dijkstra's algorithm for single source shortest path [108]. For similarity they used tf-idf cosine measure based on the article titles with in categories of a given subtree. They kept the subtree that was most semantically equivalent. After a user study they concluded that tangledness was desirable for better browsing by humans in general knowledge application areas like Wikipedia.

Chernov et al. analyzed relevant measures for inferring semantic relations between Wikipedia categories [28]. They concluded that in-links are a better measure of semantic connection as compared to out-links. Since, all hyperlinks in Wikipedia

are not semantically significant or related; they introduced measures to distinguish irregular and navigational links from semantically related links. They proposed two measures for predicting Semantic relatedness between categories, 1) Number of links between articles of categories 2) Connectivity Ratio (No. of links/Category Size), applied to in-links or out-links. They accepted that their results might be skewed due to Countries Category, since Country pages usually have more in-links as compared to out-links, as most of the articles mention some country. They suggested applying link analysis algorithms for establishing the semantic authorities among categories as future work. They gave the example of link "Republics to Non-violent Revolutions" which could help in answering the query "Find Countries which had Democratic Non-violent Revolutions?". The link "Republics to Non-violent Revolutions" could be obtained by using LinkSourceCategory-LinkTargetCategory structure. However, they argued that this simple rule cannot be applied as it will generate many useless links for example "Egg to Countries". They suggested filtering out unimportant links using the link density and link structures between categories.

Todd et al. used co-occurrences of categories in articles as a measure of similarity [53]. Ponzetto and Strube derived a large scale taxonomy from Wikipedia Categories by labeling the relations as "ISA" and "NotISA" using connectivity property of the network and lexico-syntactic patterns [80]. They compared their taxonomy with ResearchCyc [4] and WordNet [68] and got competitive results. YAGO ontology [91] builds an "ISA" hierarchy by linking Wikipedia leaf categories to WordNet hierarchy.

2.3.2 Refining Wikipedia Category Article Membership:

Suchanek et al. demonstrated that if the Head of the Category of the Entity Article is plural then this can establish Instance-of relation with high accuracy [91], for example Albert Einstein is in Category "Naturalized Citizens of United States" which implies instance-of relation. In Chapter 4 we discuss the heuristics we use to associate a WordNet type with Wikipedia concepts by exploiting the "ISA" patterns in the first sentence of article and associated category mappings to WordNet.

2.3.3 Refining and Enriching Wikipedia Inter-article Links:

The research work by Kinzler discusses the importance of basic link types like synonyms, homonyms however; it did not provide any experimental results or practical solutions [56]. In Chapter 4 we discuss our work with Wikipedia Article Links Graph [93] for document concept prediction. We introduce link weights based on cosine similarity between connected articles. Our results showed that link weights helped in finding semantically related concepts.

2.3.4 Refining and Enriching Wikipedia Infoboxes:

Kylin [**?**] generates infoboxes for articles by learning from existing infoboxes. KOG [106] automatically refines the Wikipedia infobox ontology and integrates Wikipedia's infobox-class schemata with WordNet. In Chapter 6 we present our approach to discovering ontology elements from Wikipedia page links that can help in enriching the Wikipedia infoboxes with new relations.

2.4 Use of Wikipedia as a Knowledge Resource

The depth and coverage of Wikipedia has attracted the attention of researchers who have used it as a knowledge resource for several tasks, including text categorization [42], co-reference resolution [79], predicting document topics [86], automatic word sense disambiguation [66], searching synonyms [58] and computing semantic relatedness [90], [43], [69]. In Chapter 4 we present different Wikipedia based novel approaches for a variety of tasks namely, named entity classification, document concept prediction, cross document coreference resolution, entity linking and interpreting tables. We will review in detail the related work corresponding to each task in the respective sections in Chapter 4.

2.5 Other Open Knowledge Sources

There are different knowledge sources that are either derived from Wikipedia or aid in enriching Wikipedia itself. In chapter 5 we discuss how we have incorporated related information in different knowledge sources into our Wikitology knowledge base. We give a brief introduction to related knowledge resources below.

2.5.1 DBpedia

DBpedia is a community effort focused on extracting structured information from Wikipedia and making it available for human and machine consumption on the web [17]. The extraction process involves mapping of the relations already present in Wikipedia relational database tables onto RDF, additional information

is extracted directly from articles and info-box templates. The data is extracted from info-boxes using pattern matching techniques and contains very specific information, whereas the other data sets such as Page links contain meta-data without specific semantics. Recently the most common info-boxes in Wikipedia have been used to create a shallow, cross-domain ontology in DBpedia, with 170 classes and 940 properties (http://wiki.dbpedia.org/Ontology). A new info-box data extraction method has been developed through manual mappings of Wikipedia info-boxes to the DBpedia ontology. The mappings are done using fine grained rules on parsing info-box values and also handle different info-boxes for the same class in Wikipedia (350 templates mapped to 170 classes and 2350 properties to 940 ontology properties). The DBpedia Ontology currently contains about 882 instances (http://wiki.dbpedia.org/Ontology).

There are tens of millions of links and attributes in articles that DBpedia can't figure out. DBpedia currently does not exploit the textual content of the articles whereas, a lot of information is present in the textual content of the articles and can be used to extract more information about entities. There is a need to discover the label or type of link between articles. Some basic statistics available for DBpedia in 2007 are given in Table 2.2 [17].

2.5.2 The Semantic MediaWiki Project

The Semantic MediaWiki project [59] is focused on promoting reuse of information, enhancing search and browsing facilities in Wikis. Semantic MediaWiki is

About: Barack Obama

An Entity in Data Space: dbpedia.org

Barack Hussein Obama II (born August 4, 1961) is the 44th and current President of the United States. The first African American to hold the office, he previously served as the junior United States Senator from Illinois from January 2005 until he resigned after his election to the presidency in November 2008. Obama is a graduate of Columbia University and Harvard Law School, where he was the president of the Harvard Law Review.

Property	Value
dbpedia-owl:Person/almaMater	dbpedia:Columbia_University dbpedia:Bachelor_of_Arts dbpedia:Occidental_College dbpedia:Juris_Doctor dbpedia:Harvard_Law_School
dbpedia-owl:Person/birthDate	1961-08-04 (xsd:date)
dbpedia-owl:Person/birthName	Barack Hussein Obama II
dbpedia-owl:Person/birthPlace	dbpedia:St._Petersburg_Times dbpedia:Hawaii dbpedia:Honolulu
dbpedia-owl:Person/child	dbpedia:Family_of_Barack_Obama#Immediate_family
dbpedia-owl:Person/individualisedPnd	132522136
dbpedia-owl:Person/nationality	dbpedia:United_States
dbpedia-owl:Person/occupation	dbpedia:Professor dbpedia:Constitutional_law dbpedia:Author dbpedia:Lawyer dbpedia:Community_organizing
dbpedia-owl:Person/party	dbpedia:Democratic_Party_(United_States)
dbpedia-owl:Person/religion	dbpedia:Associated_Press dbpedia:Jeremiah_Wright_controversy dbpedia:Msnbc.com dbpedia:Trinity_United_Church_of_Christ dbpedia:Christianity
dbpedia-owl:Person/residence	dbpedia:White_House

Figure 2.2: Demonstration of DBpedia resource on Barack Obama.

Table 2.2: Property related statistics in DBpedia Infobox Ontology.

Items	Count
Things	More than 1.95 million
Persons	More than 80,000
Places	More than 70,000
Music Albums	More than 35,000
Films	More than 12,000
Links to images	657,000
Links to related external webpages	1,600,000
Links to other RDF datasets	180,000
Wikipedia Categories	207,000
YAGO Categories	75,000

an enhancement of the MediaWiki software used by Wikipedia. It introduces ways to add structured data into Wikis by using specific syntax by authors and constrains the authors towards syntactical and structural consistency as well as homogeneity. DBpedia exploits the already existing structure whereas the Semantic MediaWiki provides software for authors to add structure themselves.

2.5.3 Linked Open Data

Linked Open Data is a community effort that focuses on publishing existing open license datasets as linked data on the web, interlinking things between different data sources and developing client applications that can use the linked data. Wikipedia defines Linked Data as "a term used to describe a recommended best practice for exposing, sharing, and connecting pieces of data, information, and knowledge on the Semantic Web using URIs and RDF." Currently more than 2 billion RDF triples have been interlinked by around 3 million RDF links [22]. Different browsers for linked data have been introduced, a number of applications have

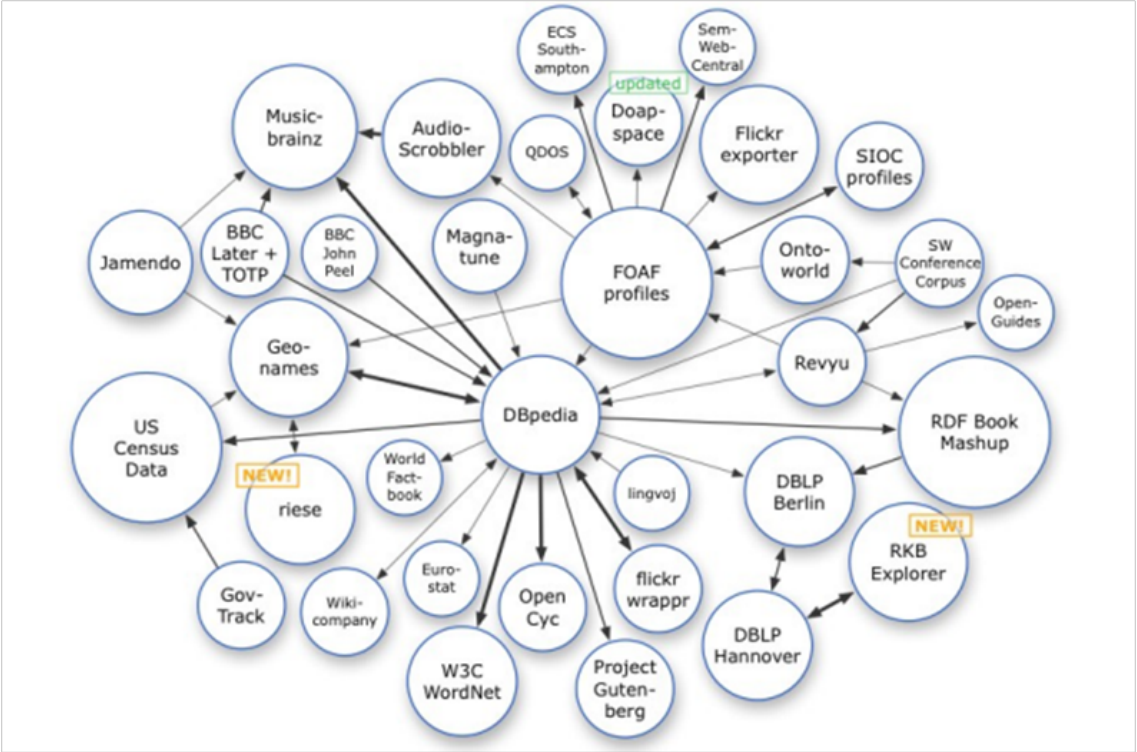


Figure 2.3: Linked Open Data datasets on the web (april 2008).

been developed that use the linked data through mashups, search engines have been developed that take benefit from the available linked data. Additional effort is required to publish more open data, increase the amount of links between them and also improve the quality of links. Currently DBPedia is serving as a core for linking other open data on the web [22].

2.5.4 Freebase

Freebase [23] is a scalable, graph-shaped database of structured general human knowledge, inspired by Semantic Web research and collaborative data communities such as the Wikipedia. Freebase currently has more than 125,000,000 tuples, over 4000 different types and over 7000 properties. Freebase allows public read and

Figure 2.4: Freebase Topic on Barack Obama along with structured data.

write access through an HTTP-based graph-query API for research, the creation and maintenance of structured data, and application building. Access is free and all data in Freebase has a very open license (Creative Commons, GFDL). Freebase has also launched an interface for open-linked data. There is a big overlap between Freebase and DBpedia therefore 2.4 million RDF links have been added to DBpedia pointing at corresponding things in Freebase. Many applications are using Freebase ranging from Powerset [29], the new semantic search engine from Microsoft, to Parallax (http://mqlx.com/~david/parallax/) a pivoting data browser.

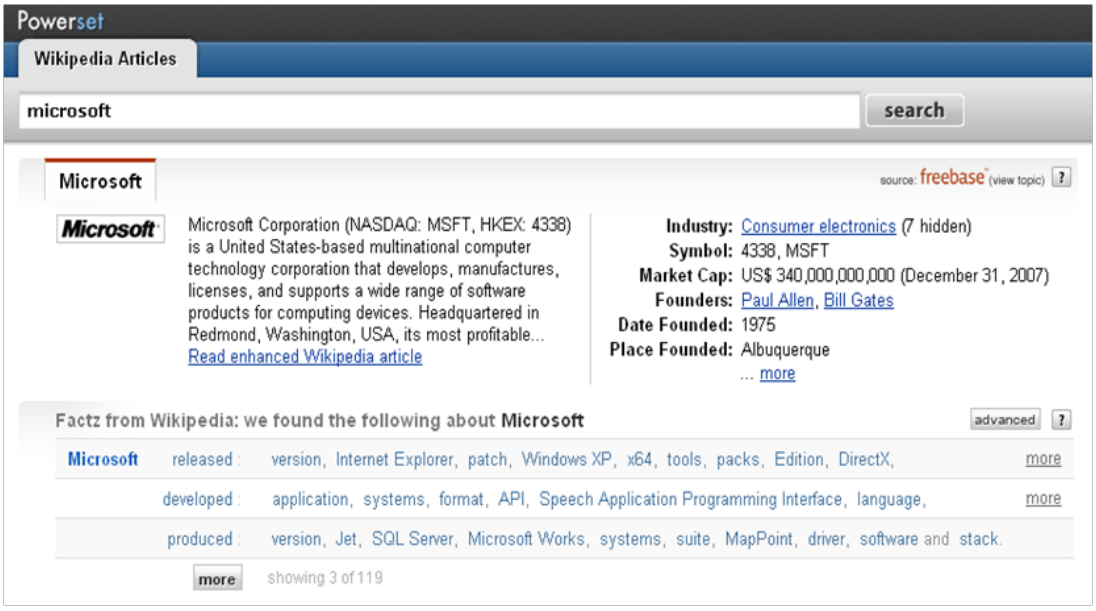


Figure 2.5: Demonstration of Powerset Factz available online.

2.5.5 Powerset

Powerset is an online search engine for querying Wikipedia using Natural Language Queries. Powerset performs a linguistic analysis of the sentences within Wikipedia and produces semantic representations from the processed sentences. Some of the semantic processing steps done in Powerset include predicate argument structure extraction, pronoun and coreference resolution and finding synonyms, hypernyms and resolving word senses using Wordnet [68]. There are different levels of semantic representations available in Powerset which are a result of different stages of the linguistic analysis processes. One of the representations that is available online is the "Factz" representation which represents facts extracted from sentences in the form of subject, predicate and object. For most entities like persons, places and things, Powerset shows a summary of facts from across Wikipedia (Figure 2.5).

Powerset Factz are facts that are extracted from sentences within Wikipedia

and are based on the predicate argument structure. The Factz may represent relations between named entities or just nouns. For example,

- $< BankofAmerica >< acquired >< bank >$
- $< BankofAmerica >< acquired >< MerrillLynch >$
- $< BankofAmerica >< received >< interest >$

The Factz in Powerset are directly extracted from sentences and do not currently incorporate information in structured data in Wikipedia such as tables, infoboxes and links.

2.5.6 WordNet

WordNet [68] is a lexical database for a wide range of common words in English Language and has been developed manually at the Cognitive Science Laboratory of Princeton University. WordNet contains English words which are grouped based on having the same sense, into sets of synonyms called synsets. Words having different senses might be present in multiple synsets. It provides short definitions and also various semantic relations between synsets such as hypernymy (subClassOf) and meronymy (partOf). WordNet 3.0 contains 82,115 synsets for 117,798 unique nouns. It also contains other types of words such as adjectives and verbs.

2.5.7 YAGO

YAGO (Yet Another Great Ontology) is a light weight ontology developed by extracting facts from Wikipedia and unifying with WordNet [91]. YAGO ex-

tracts facts from Wikipedia using a carefully designed combination of rule-based and heuristic methods. Wikipedia Category hierarchy is very noisy [17] hence YAGO builds an "ISA" hierarchy only considering Wikipedia leaf categories and linking them to WordNet's manually created taxonomy. To extract the "Type" relation, YAGO does a shallow linguistic parsing of the category name and if the head of the category name is a plural it is considered a conceptual category. For example, "Naturalized citizens of the United States" is broken into pre-modifier (Naturalized), head (citizens) and post-modifier (of the United States). This gives the TYPE relation, its domain and range. They exploit the redirects in Wikipedia to extract the MEANS relation. YAGO stores linked individuals as CONTEXT. YAGO incorporates several other relations which are extracted using specialized heuristics for each type of relation on the categories and infoboxes of Wikipedia articles. For example, if the article is in the category ending with "_births" (e.g. 1879_births) or "_deaths", then the BornInYear or DiedInYear relation is extracted for the individual.

Chapter 3

Introduction to Wikitology Knowledge Base

Wikipedia proves to be an invaluable resource for generating a hybrid knowledge base due to the availability and interlinking of structured and un-structured encyclopedic information ranging from highly structured triples in the infoboxes to un-structured free text available in the content of the articles. In addition to that it has other types of information structures available such as the category hierarchy, list pages, Redirects, Disambiguation Pages and inter-article links. Info-boxes are the most structured form of information and are composed of a set of subject-attribute-value triples which summarize or highlight the key features of the concept or subject of the article. Resources like DBpedia [17] and Freebase [23] have harvested this structured data and have made it available as triples for semantic querying. While infoboxes are a source of readily available structured information within Wikipedia, the free text of the article contains a lot more information about the entity.

Wikipedia is being used extensively to provide background knowledge for a wide variety of tasks such as text categorization [42] , co-reference resolution [79], predicting document topics [86] , automatic word sense disambiguation [66], searching synonyms [58] and computing semantic relatedness [90], [43], [69], information extraction [99], [107], [76], information retrieval [35], question answering [14] and named entity recognition [55].

Efforts like DBpedia [17] , Freebase [23] and Linked Open data [22] are focused on making knowledge available and accessible in the form of triples whereas, we have developed a hybrid KB incorporating structured and un-structured information in a novel way for applications requiring more flexible access to knowledge contained in free-text as well as structured data. The encyclopedic content of Wikipedia and presence of structured and un-structured data linked to concepts in the encyclopedia make it an ideal resource for developing a hybrid Knowledge Base that can be exploited by applications for solving a variety of problems. In order to exploit the knowledge available in un-structured form such as free text of the article and the knowledge available in a highly structured form such as ontology triples, a possible approach is to convert one form into another, such as using natural language processing techniques to extract triples from free text to populate a knowledge base. This will enable the applications to query the knowledge available in free text and the triple store using SPARQL-like [82] queries. However, populating a knowledge base with information extracted from text is still an open problem and active research is taking place in this direction. Another approach is to augment the knowledge available in the form of free text with triples from the knowledge base, enabling the applications to access the knowledge by submitting free text queries to an information retrieval index. However, in this case we will lose much of the information that is available through the highly structured triple representation and other benefits such as reasoning over the knowledge.

We approach this problem in a different way and favor an approach that does not depend on converting one form of data into another and benefits from the hybrid

nature of the data that is available in different forms.

There exists a great need for systems that can store, manage, query and handle data available in different formats. The Claremont Report on Database Research [13] has identified "The Interplay of Structured and Unstructured Data" as one of the new focus areas of research. Storing the context which may be available in multiple forms such as text, links, related entities etc. is very important as it helps in interpreting the meaning especially if the data is less precise. We find our approach to developing a hybrid knowledge base closer in direction to the Dataspaces approach [41] which is aimed at providing basic functionality over all data sources regardless of how integrated they are, for example providing a keyword search over all data sources as provided in existing desktop search systems. In case more complex operations are required such as relational queries or data mining then additional incremental effort can be applied in a "pay-as-you-go" fashion whereas, traditional data integration systems require semantic integration between terms in each schema before any services can be provided. This results in significant upfront effort to setup a data integration system as opposed to the Dataspaces approach.

Three main properties of our approach that are common with the Dataspace Support Platforms (DSSP) [41] are:

1. A DSSP is required to deal with data and applications in a wide variety of formats accessible through many systems with different interfaces. In case of Wikitology, the data available in different formats includes triples, free text and link graphs which can be accessed using different independent systems.

2. In addition to DSSP offering an integrated means of searching, querying, updating, and administering the Dataspace, often the same data may also be accessible and modifiable through an interface native to the system hosting the data. Thus, unlike a DBMS, a DSSP is not in full control of its data. In case of Wikitology, in addition to providing a hybrid querying interface, the data available in different formats can be accessed with different interfaces native to the format of data. For example, the triples can be directly accessed and queried through the triple store, the links through graph data structures and the text through an IR index.

3. A DSSP provides basic functionality over all of the data regardless of integration and may support Complex operations incrementally. In a similar fashion, Wikitology has evolved with integration of different data formats based on the needs of the applications.

Figure 3.1 shows the overview of Wikitology KB exploiting information available in different kinds of data structures, integrating them and providing a hybrid query interface to applications. In this chapter, we give an overview of the evolution of the Wikitology hybrid knowledge, approaches for evaluating and enriching the knowledge base and a comparison of Wikitology with similar existing resources derived from Wikipedia.

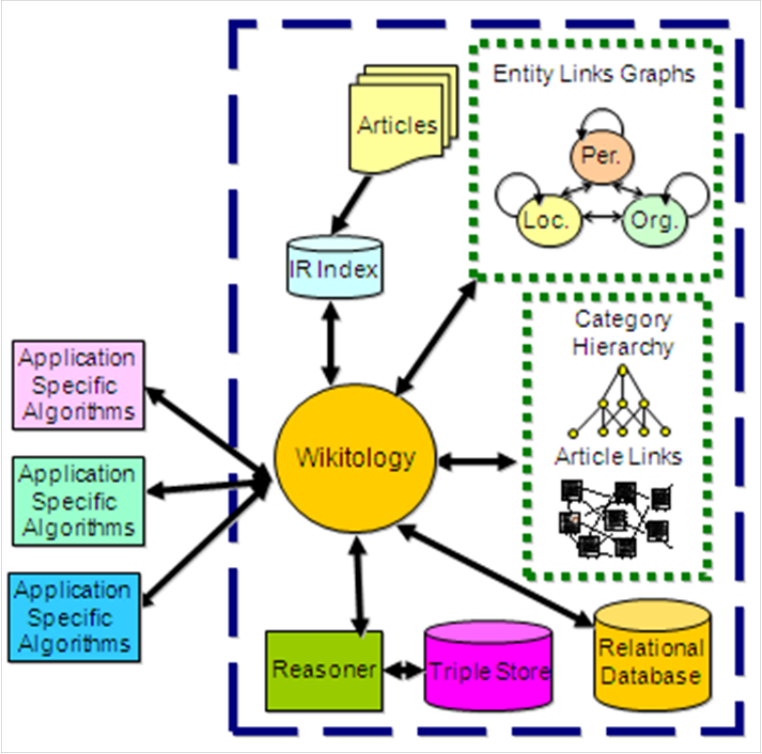


Figure 3.1: Wikitology is a hybrid knowledge base integrating information in structured and unstructured forms.

3.1 Evolution of Wikitology Knowledge Base

Research projects like Cyc [61] have resulted in the development of a complex broad coverage knowledge base however, relatively few applications have been built that really exploit it. In contrast, the design and development of Wikitology KB has been incremental and has been driven and guided by a variety of applications and approaches that exploit the knowledge available in Wikipedia in different ways. This evolution has resulted in the development of a hybrid knowledge base that not only incorporates and integrates a variety of knowledge resources but also a variety of data structures, and exposes the knowledge hidden in different forms to applications through a single integrated query interface. Initial work done in this

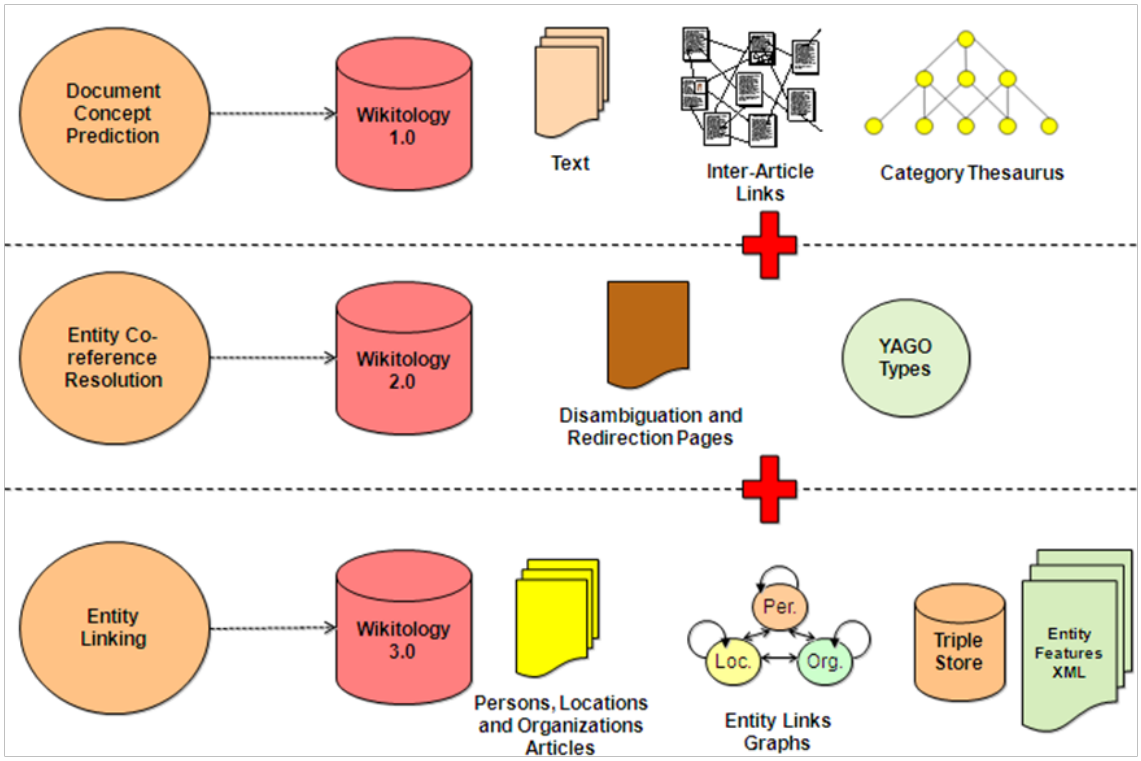


Figure 3.2: Evolution of the Wikitology System showing the different data structures incorporated over time and the different applications targeted.

direction resulted in the development of Wikitology 1.0 [93] system which was a blend of the statistical and ontological approach for predicting concepts represented in documents. Our algorithm first queries the specialized index and then exploits the page links and category links graphs using spreading activation algorithm for predicting document concepts.

The results of our first system were quite encouraging which motivated us to enhance the original system and incorporate knowledge from other knowledge resources such as YAGO and develop Wikitology 2.0 system [38] which was employed and evaluated for cross-document entity co-reference resolution task defined as a part of ACE [1]. We further enhanced Wikitology [39] using Freebase types (Persons, Locations and Organizations) to support the entity linking task defined as part

of TAC KBP 2009 [65]. We extended the entity linking approach for interpreting entities in tables [94] and incorporated WordNet types, linked concepts, DBpedia ontology and PageRank approximations into our knowledge base. A comparison of the enhancements done in different versions of Wikitology and the applications that exploited our system is given in Figure 3.2.

3.2 Evaluating the Knowledge Base

We demonstrate the value of the derived knowledge base by evaluating it against problem specific intelligent approaches that exploit Wikitology for a diverse set of use cases: namely, document concept prediction, cross document co-reference resolution defined as a task in Automatic Content Extraction (ACE) [1], entity linking to KB entities defined as a part of Text Analysis Conference Knowledge Base Population Track 2009 [65] and interpreting tables [94]. These use cases directly serve to evaluate the utility of the knowledge base for different applications and also show how the knowledge base can be exploited in different ways. Based on our work we have also developed a Wikitology API that applications can use to exploit this unique hybrid resource for solving a variety of real world problems.

3.3 Automatic Enrichment of Knowledge Base

The different use cases that exploit Wikitology for solving real world problems also contribute to enriching the knowledge base automatically. The document concept prediction approach can predict inter-article and category-links for new

Wikipedia articles. Cross document co-reference resolution and entity linking provide a way for linking information related to entities in other resources such as news articles as external links and also for suggesting redirects. In addition to that we have also developed specific approaches aimed at automatically enriching the Wikitology KB by unsupervised discovery of ontology elements using the inter-article links, generating disambiguation trees for persons and estimating the page rank of Wikipedia concepts to serve as a measure of popularity. In Chapter 6, we propose a framework for automatically enriching Wikitology with articles on new concepts. We discuss how the different Wikitology based approaches can contribute and support a number of steps in the framework, which could be extended to automatically add articles on new concepts to Wikipedia and hence the Wikitology knowledge base in the future.

3.4 Wikitology v/s Existing Wikipedia based Knowledge Resources

There are different knowledge resources that have been derived from Wikipedia, which mainly include DBpedia [17], Freebase [23], Linked Open Data [22], Powerset [29] and YAGO [91]. We have reviewed these resources in Chapter 2. Table 3.1 summarizes the features that distinguish Wikitology from existing knowledge resources derived from Wikipedia.

DBpedia, Freebase, Linked Open Data and YAGO exploit the structured data available in Wikipedia infoboxes, templates and categories and provide knowledge in the form of triples and support structured querying over the triples. YAGO in

Table 3.1: Summary of features distinguishing Wikitology from existing knowledge resources derived from Wikipedia.

Resource	Type of Data	Data Storage Representation	Querying Support	Relevance Score
DBpedia	Triples	Triple Store	SPARQL	No
Freebase	Graph	Graph	MQL	No
Linked Open Data	Triples	Triple Store	SPARQL	No
Powerset	Facts and Semantic Representations extracted from sentences	-	Natural Language Queries	-
YAGO	Triples	Triple Store	SPARQL-like	No
Wikitology	Free text, Link graphs, Triples	Specialized IR Index associated data structures	Hybrid queries with free text and structural constraints	Yes Yes

addition provides a framework for adding new facts by linguistic analysis of sentences in Wikipedia but requires as input the target relation to extract. Powerset does a linguistic analysis of sentences within Wikipedia, extracts facts and supports natural language queries. It does not currently incorporate structured information available in Wikipedia such as in infoboxes, categories or inter-article links. Wikitology is unique as it incorporates both structured and un-structured information in Wikipedia and provides a flexible and rich query interface for hybrid querying of content along with structure. It also outputs ranked results with a relevance score which can be more informative for a variety of applications.

3.5 Conclusion

Developing the novel hybrid knowledge base and evaluating it using different real world applications guided us in improving the knowledge base in order to better serve the needs of variety of applications especially in the Information Retrieval and Extraction domain. In the next chapter we discuss different approaches for solving real world problems that directed and contributed to the final design of Wikitology. We discuss the detailed design and implementation of the Wikitology knowledge base in Chapter 5 and present approaches for automatically enriching Wikitology in Chapter 6.

Chapter 4

Novel Wikitology Based Approaches for Common Use Cases

In this chapter, we discuss a variety of approaches that exploit Wikipedia in different ways for solving some common use cases and have led to the incremental development of the Wikitology knowledge base. We initially demonstrate how we can exploit Wikipedia for classifying articles on named entities and also discuss the challenges in directly using the Wikipedia Category system for named entity classification. We then discuss our approach for predicting generalized and specialized concepts present in documents which resulted in the development of Wikitology 1.0 system. The original Wikitology system was enhanced by incorporating knowledge from other knowledge resources and resulted in the development of Wikitology 2.0 which was employed for cross-document entity co-reference resolution task. Further enhancements were made that resulted in Wikitology 3.0 system which was exploited for the Entity Linking task and for interpreting information in tables. In this chapter we discuss these common use cases that exploit Wikitology. For each use case we review the relevant literature, discuss our approach, implementation, experiments, results and conclusions.

4.1 Named Entity Classification

4.1.1 Introduction

In this study we focused on analyzing and employing the structure and content of Wikipedia for Named Entity Classification as Location, Person or Event for Wikipedia articles on entities. Wikipedia articles primarily contain two types of links, namely internal links and external links. Internal links are those that point to articles within Wikipedia and external links point to web pages on the internet. The internal links within an article may be of different types, for e.g., they may be pointing to a location, a person, an event, a term definition and articles with related content or concepts. Such links are clearly understood by humans but cannot be interpreted by machines, hence semantically enriching Wikipedia with link labels based on the type of entity they link to, would improve the machine readability of knowledge contained within Wikipedia articles. This study was aimed at discovering the challenges in generating training data from Wikipedia automatically or semi-automatically and evaluating the training data for classifying Wikipedia articles on named entities based on their type.

4.1.2 Problem Statement

Given a Wikipedia article on a Named Entity, classify it into a Person, Location or Event Class.

4.1.3 Related Work

Named Entity Recognition and Classification (NERC) is an important task in Information Extraction i.e., to get structured information from unstructured text such as newspapers. The target of NERC is to find entities in text and label them with class tags such as PERSON, LOCATION, and ORGANIZATION etc. Our research is focused on labeling links between articles based on the type of entity in the target article. A survey conducted for research period from 1991 to 2006 in NERC showed that the latest systems mostly rely on machine learning techniques for such tasks [74]. The survey emphasized that the choice of features is as important as the choice of the technique itself. Application of supervised machine learning techniques is dependent on the availability of a large annotated corpus. Such a resource is usually unavailable and therefore often semi-supervised or unsupervised machine learning techniques are employed. Supervised learning techniques include Hidden Markov Models, Decision Trees, Maximum Entropy Models, Support Vector Machines and Conditional Random Fields. The most famous technique in case of semi-supervised learning is bootstrapping, in which case a set of seeds is given as input to start the learning process. For example, to identify disease names the system may start by seeds of few disease names and use it to get sentences with those names and learn the context and then the system finds other instances with similar context. This process may be repeated several times to get a large number of contexts and disease names.

Other techniques include use of regular expressions to get the seeds for starting

training process. Patterns of proper names followed by noun phrases in apposition have also been used. In case of unsupervised learning the most common approach that is used is clustering. It has also been proposed to use ensemble based methods for text classification [78]. The basic idea behind ensemble based methods is to combine predictions of multiple classifiers and use it as a single classifier which should have a greater accuracy than the individual classifiers. Ensembles are generally created using two methods 1) Bagging 2) Boosting. Bagging technique is based on bootstrapping whereas Boosting focuses on producing a series of classifiers based on choosing the misclassified examples more often, so that new classifiers are produced that work better on misclassified examples.

Beneti et al. worked on automatically generating fine grained Named Entity Classification by using Wikipedia categories in addition to other methods [20]. They used Lingpipe [15] for initial classification of named entities as Person, Location and Organization and then used that information to find Wikipedia pages on those entities, the categories of those pages were then filtered out using a ranking system to get the most relevant categories to those entities. They concluded that their approach had promising results. Kyriakopoulou and Kalamboukis applied clustering on the training and test data to add meta-information to the simple BOW (Bag of words) model [60]. They reported substantial improvements for classification methods especially when using small training set. Ghosh and McCallum used Wikipedia as a repository of named entities with a relation extraction system for disambiguation of entities in Wikipedia [45]. Gabrilovich and Markovitch employed Wikipedia pages as concepts to enrich the feature space for classifying documents and reported

qualitative improvements [42].

Kliegr proposed an unsupervised algorithm which expressed Entities and Classes as Wordnet Synsets and Wikipedia was employed for real-time hypernym discovery to map uncommon entities to WordNet [57]. Watanabe et. al. categorized Named Entities in Wikipedia by structuring anchor texts and dependencies between them induced by the HTML structure (for example, a list or table and anchor text of inter-article links) into a graph and trained Graph Based Conditional Random Fields for obtaining models for classifying Named Entities in Wikipedia [101]. Toral et al. proposed to automatically build and maintain gazetteers for Named Entities by analyzing the entries of Wikipedia with the aid of a noun hierarchy from WordNet [95]. For every noun in the first sentence of the Wikipedia article on a Named Entity they follow the hypernym branch of that synset until the root is reached or the considered synset is reached i.e. Person, Location or Organization. They also apply a weighing algorithm and certain heuristics to improve their results. In the later case they consider the entity as belonging to that synset or class.

4.1.4 Approach

4.1.4.1 Semi-automatic generation of Training Data

Wikipedia articles may be associated with a number of categories. Wikipedia Category Graph has a structure of a thesaurus [100], it's not a strict hierarchy and one category could have several super-categories, however, there exists a super-category and sub-category relationship between related categories. Wikipedia Cat-

egory network was extracted and used in the process of generating the training sets. JUNG Graph library [1] was used to parse the category graph of Wikipedia and extract all sub-categories for a given category.

Training data set for Persons: It was initially planned to directly use the Wikipedia category hierarchy for extracting training data related to "Persons" class for classifying links as "Persons". It was proposed that the top level category in Wikipedia with the title of "People" would be used by fetching articles randomly from its sub-category tree. The category "People" does not have any articles associated directly with it, however it has several sub-categories (44 sub-categories in Wikipedia 2006) which in turn have other sub-categories and so on. One interesting category is "List of People". The "List of People" category has further sub-categories based on nationality, religion, wealth, ideology etc. It was proposed that 100 articles would be selected randomly from within this category falling directly or indirectly in this category which will be achieved by traversing the category graph. However, after a closer analysis it was observed that fetching training data directly from the sub-categories may introduce a lot of noise since all the articles directly or indirectly associated with the Wikipedia "People" category do not represent Wikipedia articles on people for e.g.:

- The category "Animal" was found as one of the sub-category in the category hierarchy under the root category "People" [104].

- The category "List of people" had a category "List of people by wealth" but

[1] http://jung.sourceforge.net/

many of the pages in the later category are not on people such as "List of Wealthiest Foundations", "List of Companies by revenue" (Wikipedia 2006).

Using such training data may result in classifying wealthiest foundations as persons and so on. Therefore, it was planned to use a slightly different approach. After observing several articles related to persons on Wikipedia, it was noted that the articles on persons in Wikipedia usually have the word "born" in their first sentence, there may be some articles on people without having the word "born" in the first sentence such as the article on "Aristotle". There might be a very few cases where the word "born" may not refer to a person for e.g. The article on the movie "born again", however, it was decided that this heuristic could be used to fetch pages on persons by ignoring a little noise. The following method was used to extract training data on "persons":

1. Index Wikipedia articles using Lucene [52].

2. Find all articles with the word "born" in them.

3. Further filter those articles to only those which have the word "born" in their first sentence. (so that we know that the whole article is about a person rather than any article on another topic just mentioning some person anywhere in the text). Table 4.1 shows the statistic on the number of pages with the word born in them.

4. Further filter those articles to exclude the pages starting with "List%". (There are several pages like that in Wikipedia for e.g. "list of american people"',

Table 4.1: Wikipedia statistics, articles with word born in them.

Description	No. of pages
No. of Wikipedia pages with the word born in them	271514
No. of Wikipedia pages with the word born in their first sentence	142758

"list of people by birth" etc. such pages don't directly represent the entity "Person").

Training data set for Locations: It was earlier decided to use the Wikipedia top level category on "Geography and Places" to extract training data on articles related to Locations however it had the same problem as mentioned earlier for the "People" category, i.e., not all articles belonging directly or indirectly under this category are related to locations. For example the following hierarchy was observed:

Georaphy → Geography by place → Regions → Regions of Asia → Middle East → Dances of Middle East (Wikipedia 2006)

The category "Dances of Middle East" does not contain pages on location. Hence another approach was used to automatically generate training data on "locations" from Wikipedia.

1. Fetch 'n' pages from Wikipedia randomly.
2. Use any entity recognizer to find words in those articles that represent locations.
3. Filter the predicted locations manually as the entity recognizer does not predict with 100% accuracy.

4. Use those words to find Wikipedia articles on those topics which would essentially represent articles on locations, since those words have already been recognized as locations in step 2.

We used StandfordNER [40] software to find the words representing locations in text of articles. However, there was a lot of noise in the predicted locations and several words related to persons and organizations and other entities were labeled by the software as locations, hence we manually filtered out such words and only used the manually filtered location words to fetch their corresponding Wikipedia pages. Many location pages in Wikipedia were ambiguous as they referred to the locations at different places but having the same name, hence we also filtered the disambiguation pages in Wikipedia and used only those pages that referred to an unambiguous location.

Training data set for Events: It was earlier decided to use the top level category in Wikipedia named as "Historical Events" which has one of its sub-category as "Events" which has several sub-categories and so on. However, the sub-categories under this category also show several inconsistencies such as:

Events $\rightarrow$ Events by year $\rightarrow$ Lists of leaders by year (Wikipedia 2006)

The presence of category "Lists of leaders by year" under the category "Events" includes persons. This would introduce incorrect classification in the training data itself. It might result in labeling certain articles on "persons" as "events". We also observed that the deeper the category tree, the more the inconsistencies, hence, we used the following method to find a representative training set related to events:

1. Start from the Events Category tree at the root and include the sub-categories within the depth of 10.

2. Fetch 'n' pages from the pruned Events category tree.

3. Manually filter the retrieved pages.

4.1.4.2 Feature Sets

The text in Wikipedia articles is usually organized under section headings and sub-headings, but not all Wikipedia articles have headings in them, some articles don't have any headings in their text. It was also noted that the text in the first paragraph should be sufficient to indicate whether the page belongs to a person, location or event. There were two benefits in using this approach, firstly, using the text of just the first paragraph would reduce the amount of processing as compared to using the whole article text, and secondly, our system would not be affected by the length of articles whether they are long or short. It was also observed that the articles belonging to different categories may have different stop words used in them, which could be used as distinguishing features. For example, in case of an article related to a person, the chances of having the stop words "He", "She" in the first paragraph are very high. Therefore the following feature sets were considered:

1. Article Section Headings

2. Text in first paragraph

3. Stop words in first paragraph

4. Different combinations of the above mentioned feature sets

4.1.4.3 Classification

According to the literature, Support Vector Machines [21] are one of the most famous methods for text classification tasks. We started by analyzing our training set by using simpler techniques first, like Clustering and K-Nearest Neighbors before we constructed the classifier to get more understanding of our data. By using k-means clustering in weka [48] and hierarchical clustering in Lingpipe software [15] we were not able to get a satisfactory class to cluster assignment, similar was the case in predicting the class using K-Nearest Neighbor technique. Therefore we directed our work towards using SVM for text classification task and compared it with Nave Bayes and Decision Trees to observe how well does SVM perform in terms of accuracy as compared to the other two methods on our given training data. We were expecting the best results from SVM according to its fame in the available literature.

4.1.5 Experiments and Evaluation

We used 25 Wikipedia articles for each class (Persons, Locations, Events), i.e., 75 articles in total. We could have used a larger training set but since it required manual filtering we restricted ourselves to the available training set. We extracted different feature sets from those articles and constructed our training data. We evaluated the classifiers using k-fold (k = No. of training documents)

Table 4.2: Results showing accuracy obtained using different feature sets. H: Headings Only, F: Words in first paragraph, H+F: Headings + Words in first paragraph, H+F+S: Headings + Words in first paragraph + Stop words.

Algorithms	H	F	H+F	H+F+S
SVM (Linear Kernel)	73.3%	86.6%	86.6%	89.3%
Nave Bayes	78.6%	66.6%	68.0%	68.0%
Decision Trees (Id3)	74.6%	80.0%	74.6%	77.3%

cross validation. The results obtained using different feature sets and classification algorithms are given in Table 4.2. It was observed that a combination of features related to text of first paragraph, headings and stop words gave best performance. Amongst the three machine learning techniques used i.e., SVMs, Nave Bayes and Decision Trees, SVM performed the best overall. In case of individual classes SVM did the best for the PERSONS class, for the LOCATIONS class both SVM and Decision Trees gave the same accuracy whereas for EVENTS the best accuracy was given by Nave Bayes. Confusion Matrices for Headings + First Paragraph + Stop Words using different algorithms are given in Table 4.3, Table 4.4 and Table 4.5.

Table 4.3: Confusion matrix using SVM.

Persons	Locations	Events	Classified as
23	0	2	Persons
0	24	1	Locations
3	2	20	Events

Table 4.4: Confusion matrix using Decision Trees

Persons	Locations	Events	Classified as
23	1	1	Persons
2	14	9	Locations
0	4	21	Events

Table 4.5: Confusion matrix using Nave Bayes

Persons	Locations	Events	Classified as
16	0	9	Persons
0	13	12	Locations
3	0	22	Events

4.1.6 Conclusions

In this case study we generated training data from Wikipedia for named entity classification. We initially planned to use the Wikipedia categories for generating the training data, however, we observed that the Wikipedia category hierarchy is very noisy. We have introduced different semi-automatic approaches for generating training data for classifying persons, locations and events. We also experimented with different feature sets. Our results showed that using the section headings, text in first paragraph and stop words as features gave the best accuracy. With respect to the type of entities, persons were classified with highest accuracy followed by locations and then events using SVMs.

4.2 Document Concept Prediction

4.2.1 Introduction

Characterizing what a document is "about" is a task common to many applications, including classification, retrieval, modeling a user's interests, and selecting appropriate advertisements. The work we report on in this case study was motivated by the requirements of the following application:

> A team of analysts is working on a set of common tasks, with each analyst focused on several different areas and working sometimes independently and sometimes in a tightly coordinated group. Collaboration in such a setting is enhanced if the individual analysts maintain an awareness of what their colleagues have been working on. As new members join the team or return to it after a temporary assignment or vacation, it is important for them to acquire the context of who has been working on or is interested in what. A way to describe the topics on which an analyst focuses is through an analysis of the documents, or portions of documents that she has been reviewing, reading or writing. In short, if we know what she has been reading, we have a good handle on what she is working on.

One general approach to describing what a document is about is to use statistical techniques to describe the words and phrases it contains. This is the basis of information retrieval, which has had enormous practical success. Another approach

is to tag the document with relevant terms that represent semantic concepts important to the document. This is typically used in information science using terms from a standard classification hierarchy such as the Dewey Decimal System [32] or ACM Computing Classification System [30]. More recently, many Web 2.0 systems have allowed users to tag documents and Web resources with terms without requiring them to come from a fixed vocabulary. In a social media context (e.g., del.icio.us or Flickr) an implicit ontology of tags can emerge from the community of users and subsequently influence the tags chosen by individuals, reinforcing a notion of a common ontology developed by the community.

As discussed earlier, an advantage of using the "ontology" approach, whether based on a designed or emergent ontology, is that the terms can be explicitly linked or mapped to semantic concepts in other ontologies and are thus available for reasoning in more sophisticated language understanding systems such as OntoSem [77] and Powerset [29], or specialized knowledge-based systems, or in Semantic Web applications.

Using the traditional approach of a controlled, designed ontology has many disadvantages beginning with the often difficult task of designing and implementing the ontology. Once that it done, it must be maintained and modified, an important process in domains where the underlying concepts are evolving rapidly. ACM's CCS, for example, undergoes periodic reorganization and redesign and yet as a classification of computer science concepts, it always seems to be out of date or even quaint. As a final problem, consider the process a person must follow in assigning ontology terms to a document. She has to be familiar with all of the

possible choices or have some way to browse or search through them. She has to understand what each of the terms means, either the original meaning intended by the ontology designer or the possibly different current meaning as used by her community. Finally, she has to select the best set of terms from among the many relevant choices the ontology may present to her.

The use of an implicit ontology emerging from the tagging choices of a community of individuals solves some of these problems, but also has significant disadvantages. Some of these are inherent and others are being addressed in the research community and may ultimately admit good solutions. These problems are worth addressing because the result will be an ontology that (1) represents a consensus view of a community of users and (2) is constructed and maintained by the community without cost to any organization. It remains unclear how the terms in such an ontology should be organized structurally, understood informally by end users, or mapped to a more formal ontology such as Cyc [61] or popular Semantic Web ontologies like FOAF [34].

We have developed a system that is a blend of the two approaches based on the idea of using Wikipedia as an ontology. Specifically, each non-administrative Wikipedia page is used as a term in an ontology. These include Wikipedia articles describing individuals (Alan Turing), concepts (Emissions trading), locations (Barbados), events (collapse of the World trade Center), and categories (microbiology). Using Wikipedia as an ontology has many advantages: it is broad and fairly comprehensive, of generally high quality, constructed and maintained by tens of thousands of users, evolves and adapts rapidly as events and knowledge change, and free and

"open sourced". Moreover, the meaning of any term in the ontology is easy for a person to understand from the content on the Web page. Finally, the Wikipedia pages are already linked to many existing formal ontologies though efforts like DBpedia [17] and Semantic MediaWiki [59] and in commercial systems like Freebase and Powerset.

The underlying concept of an article cannot be assessed by merely considering the words that appear in that article, in addition to that, finding out if two articles are conceptually related is an even more challenging problem and requires a lot of background domain knowledge, common sense as well as information about the context. Humans have the inborn ability to relate concepts semantically however it is still a very difficult problem for computers, which can be made easier by augmenting background domain knowledge for such tasks, which would certainly improve the accuracy and quality of prediction. Wikipedia proves to be an invaluable source for such background domain knowledge. We employ the spreading activation algorithm in our approach. A brief introduction to spreading activation algorithm is given below.

4.2.1.1 Spreading Activation

Spreading Activation is a technique that has been widely adopted for associative retrieval [31]. In associative retrieval the idea is that it is possible to retrieve relevant documents if they are associated with other documents that have been considered relevant by the user. In Wikipedia the links between articles show asso-

ciation between concepts of articles and hence can be used as such for finding related concepts to a given concept. The algorithm starts with a set of activated nodes and in each iteration the activation of nodes is spread to associated nodes. The spread of activation may be directed by addition of different constraints like distance constraints, fan out constraints, path constraints, threshold etc. These parameters are mostly domain specific.

In this study we consider a Wikipedia category as a representative of a generalized concept. The title of a Wikipedia article may be considered as a specific or specialized concept. The links between different articles are considered as links between different concepts. We have implemented different approaches to exploit the Wikipedia article texts, category network and page links graph for predicting concepts related to documents.

4.2.2 Problem Statement

Given a document or set of documents, predict the concepts related to the document(s).

4.2.3 Related Work

The depth and coverage of Wikipedia has attracted the attention of researchers who have used it as a knowledge resource for several tasks, including text categorization [42], co-reference resolution [79], predicting document topics [86], automatic word sense disambiguation [66], searching synonyms [58] and computing semantic

relatedness [90], [43], [69]. To the best of our knowledge, Wikipedia has not yet been directly used to predict concepts that characterize a set of documents.

While this is similar to the task of assigning documents to a class or category, it differs in a significant way. In categorization, the task is to predict the category of a given document however, predicting common concepts for a set of documents may include documents belonging to very different categories but having some concept in common. For example a user searching for information related to growing a flowering plant may consider reading different articles on seeds, fertilizers, herbicides, manure, gardening etc, all these articles may belong to different categories yet share a common concept that all are related to the plant. However, in certain cases in which the set of documents belong to the same category, we may be able to introduce the predicted common concept as a new sub-category.

We find our problem very similar in direction to computing semantic relatedness between concepts with the addition that we focus on predicting a concept that is common as well as semantically related to a set of documents. In this section we give a brief review of related work.

The initial work done on employing Wikipedia for computing semantic relatedness was by Strube and Ponzetto and realized in a system named WikiRelate! [90]. They used information from Wordnet, Wikipedia and Google in computing degrees of semantic similarity and reported that Wikipedia outperforms Wordnet. However, they obtained the best results when evidence from all three resources was integrated. They used different measures for computing semantic relatedness including measures based on paths, information content, and text overlap.

Gabrilovich and Markovich used concept space derived from Wikipedia to compute semantic relatedness between fragments of natural language text, and reported the performance to be significantly better than other state of the art methods [43]. They named their approach "Explicit Semantic Analysis" (ESA) as they use concepts that are explicitly defined by users. Their method employs machine learning technique to represent the meaning of a text fragment as a weighted vector of concepts derived from Wikipedia. Relatedness is then measured through the comparison of concept vectors using conventional metrics such as cosine similarity.

The success of their experiments gives support to our method, which also initially utilizes the Wikipedia concept space, although in a different manner. Instead of using machine learning techniques, we directly compute the related concepts based on the cosine similarity between the input document and Wikipedia articles and then use those concepts as our initial activated nodes in spreading activation. The key difference is that we are not interested in merely finding the semantic relatedness between documents but in finding a semantically related concept that is also common to a set of documents. Wikipedia Link Vector Model is an approach that is similar to ESA that eliminates the need for processing Wikipedia article text [69]. This method computes the semantic relatedness between terms based on the links found in their corresponding Wikipedia articles. The reported results, however, give less accuracy than ESA.

4.2.4 Approach

We downloaded the Wikipedia XML snapshot of 4 November 2006 and extracted 2,557,939 Wikipedia articles. The text of each article was indexed using the Lucene text search engine library [52] under the standard configuration. We ignored the history, talk pages, user pages, etc. We also downloaded the Wikipedia database tables in order to create the category links graph and the article links graph. Major administrative categories (e.g., "All articles with unsourced statements") were identified and removed from the category links graph. Any remaining administrative categories appearing in the prediction results were excluded. We implemented three different methods for our study, which are described and discussed below.

4.2.4.1 Method 1: Article Text

In the first method we use the test document or set of related documents as search query to the Lucene Wikipedia index. After getting top N matching Wikipedia articles (based on cosine similarity) for each document in the set, we extract their Wikipedia categories and score them based on two simple scoring schemes.

In scoring scheme one, we simply count the number of times that each Wikipedia category was associated with one of the N results.

In scoring scheme two, we take into account the cosine similarity score between the test document and matching Wikipedia articles. The score for each category is the sum of the cosine similarity scores of the matching articles that are linked to

the category.

4.2.4.2 Method 2: Text and Categories with Spreading Activation

In the second method we also use the Wikipedia category links network for prediction of related concepts. We take the top N Wikipedia categories predicted as a result of method one scoring scheme one and use them as the initial set of activated nodes in the category links graph. After 'k' pulses of spreading activation, the category nodes are ranked based on their activation score. The activation function composed of the input function and the output function is given in equation 4.1 and 4.2 respectively.

$$I_j = \sum_i O_i \tag{4.1}$$

$$O_j = \frac{A_j}{D_j \times k} \tag{4.2}$$

Where the variables are defined as:

O_i : Output of Node i connected to node j

A_j : Activation of Node j

k : Pulse No.

D_j : Out Degree of Node j

4.2.4.3 Method 3: Text and Links with Spreading Activation

In the third method we take the top N matching Wikipedia articles (based on cosine similarity) to each test document as the initial set of activated nodes in the article links graph. During our preliminary experiments we observed that there were many irrelevant links between articles based on the fact that a title of one article appears as a word in the other for example, an article that mentions the name of a country (e.g., Canada) often has a link to the article on that country even though the country is mentioned in passing and is unrelated to the article's topic.

Hence to refine the links in the article links graph we filter out all links between documents whose cosine similarity scores are below a threshold (e.g., 0.4) so that the spreading activation would be more directed. We use three variations of node activation functions for spreading activation. The objective was to see if there is any difference in the prediction results through each function.

Activation Function 1:

$$I_j = \sum_i O_i w_{i,j} \tag{4.3}$$

$$O_j = \frac{A_j}{k} \tag{4.4}$$

Activation Function 2:

$$I_j = \sum_i O_i \tag{4.5}$$

$$O_j = 1 \tag{4.6}$$

Activation Function 3:

$$I_j = \sum_i O_i \tag{4.7}$$

$$O_j = \frac{A_j}{k} \tag{4.8}$$

Where the variables are defined as:

O_i : Output of node i connected to node j

A_j : Activation of node j

$w_{i,j}$: Weight on edge from node i to node j

k : Pulse No.

D_j : Out Degree of node j

4.2.5 Experiments and Results

We conducted three different kinds of experiments. Our first experiment was focused at simply observing how well the Wikipedia categories represent concepts in individual test documents. For this we ran experiments using methods one and two. The second experiment was an extension to the first experiment in that it included a set of test documents rather than individual documents. The third experiment was targeted towards finding if a common concept could be predicted for a set of documents using the article links graph given that the concept is not already defined as a Wikipedia category.

4.2.5.1 Ex 1: Predicting the topic of a single document using Methods 1 and 2

In this experiment we took several articles representing various subjects and areas directly from Internet and used methods 1 and 2 to predict concepts related to individual articles. The results of the top three predictions in order of their rank for a few of those articles are given in Table 4.6. We also give the actual titles of those test articles to evaluate the predictions.

The top most ranked prediction using both methods and scoring schemes in most of the cases match the title of the document or concept related to the title of the test document. We observed that the results don't significantly differ using the different methods and scoring schemes, however using spreading activation either results in the same prediction or a prediction of a more generalized concept that is evident in the results. In case of document 3, using three pulses for spreading activation resulted in prediction of a much generalized category named "Main topic classifications". Since the category graph is directed, i.e., from sub-categories to super-categories, it is expected that increasing the number of pulses will result in spreading activation to more generalized categories.

4.2.5.2 Ex 2: Predicting the topic of a set of documents using Methods 1 and 2

In this experiment, two test cases were run. For the first test case we took a set of ten documents related to Genetics from the Internet and tried to predict

Table 4.6: Top three concepts predicted for a single test document using Method 1 and Method 2.

No.	Test Document Title	Method 1 Scoring Scheme 1	Method 1 Scoring Scheme 2	Method 2 Pulses=2	Method 2 Pulses=3
1	Geology	Geology, Stratigraphy, Geology of the United Kingdom	Geology, Stratigraphy, Science occupations	Earth sciences, Geology, Physical sciences	Earth sciences, Natural sciences, Physical sciences
2	Atomic Bombings of Nagasaki	Atomic bombings of Hiroshima and Nagasaki, Manhattan Project, Nuclear warfare	Atomic bombings of Hiroshima and Nagasaki, Manhattan Project, Nuclear warfare	Atomic bombings of Hiroshima and Nagasaki, Warfare by type, Manhattan Project	Wars by country, Military history of the United States, Nuclear technology
3	Weather Prediction of thunder storms (*CNN Weather Prediction)	Weather Hazards, Winds, Severe weather and convection	Weather Hazards, Current events, Types of cyclone	Meterology, Nature, Weather	Main topic classifications, Fundamental, Atmospheric sciences

a common or general concept covering all documents. For each document in the test set, the ten top matching Wikipedia articles were retrieved resulting in initial activation of 100 nodes for spreading activation in category links graph. The results of this experiment, shown in Table 4.8, are also very encouraging and a related concept common to all is predicted in almost all cases. We observed that increasing the spreading activation pulses results in prediction of more generalized categories. For example, if we consider the top most ranked predictions, in case of method 1 the prediction is "Genetics" however, in case of spreading activation with 2 pulses the prediction is "Biology" and with three pulses the prediction is "Nature" which is an even broader concept than "Biology".

This same experiment was repeated for the second test case, for which we took ten articles related to different antibiotics from internet. The top most ranked predictions using method 1 are "Antibiotics" using scoring scheme 1 and "Macrolide antibiotics" using scoring scheme 2. In case of Method 2 with spreading activation the top ranked predictions are "Medicine" with two pulses and "Biology" with three pulses. It is again observed that increasing the pulses results in prediction of a more generalized concept.

4.2.5.3 Ex 3: Predicting Common Concepts using Page Links and Categories

The objective of this experiment was to see if it is possible to predict a concept common to a set of Wikipedia articles themselves, given that the concept is

Table 4.7: Titles of documents in the test sets.

Test Set 1	Test Set 2	Test Set 3
1. Basic Genetics	1. Azithromycin	1. Crop rotation
2. Exceptions to Simple Inheritance	2. Cephalexin	2. Permaculture
3. Mendel's Genetics	3. Ciprofloxacin	3. Beneficial insects
4. Probability of Inheritance	4. Clarithromycin	4. Neem
5. Basics of population genetics	5. Doxycycline	5. Lady Bird
6. Chromosomes	6. Erythromycin	6. Principles of Organic Agriculture
7. Coat Color Genetics	7. Levofloxacin	7. Rhizobia
8. Genes	8. Ofloxacin	8. Biointensive
9. Inbreeding and Linebreeding	9. Tetracycline	9.Intercropping
10.Structure of DNA	10. Trimethoprim	10. Green manure

Table 4.8: Common concept prediction results for a set of documents

Set	Method 1 Scoring Scheme 1	Method 1 Scoring Scheme 2	Method 2 Pulses=2	Method 2 Pulses=3
1	Genetics Classical genetics Population genetics	Genetics Classical genetics Population genetics	Biology Genetics Life	Nature Academic disciplines Main topic classification
2	Antibiotics Macrolide antibiotics Organofluorides	Macrolide antibiotics Antibiotics Organofluorides	Medicine Antibiotics Medical specialties	Biology Human Health sciences
3	Agriculture Sustainable technologies Crops	Agriculture Sustainable technologies Crops	Skills Applied sciences Land management	Knowledge Learning Industries

not already represented as a Wikipedia category by using the article text and links. For this experiment we picked a set of related Wikipedia articles belonging to different categories but sharing some common concept as input. We picked different Wikipedia articles related to the concept of "Organic Farming" which is not represented as a category in Wikipedia. We used all three proposed methods to see if method 3, which also uses the page links information, can predict a common concept that is more specialized than the concepts predicted using methods 1 and 2.

The top ranked predictions using method 1 and 2 (Table 4.8) are "Agriculture", "Skills" and "Knowledge" whereas by using method 3 and different activation functions (Table 4.9), the top ranked predictions are "Organic farming" and "Per-

Table 4.9: Common concept prediction results for a set of documents related to "Organic farming" using Method 3 with different pulses and activation functions.

Pulses	Activation Function 1	Activation Function 2	Activation Function 3
1	Organic farming Sustainable agriculture Organic gardening	Organic farming Sustainable agriculture Agriculture	Permaculture Crop rotation Green manure
2	Organic farming Permaculture Crop rotation	Permaculture Organic farming Sustainable agriculture	Organic farming Sustainable agriculture Organic gardening

maculture" (Permaculture: means Permanent Agriculture, which is also a related concept to Organic farming). These results show that it is possible to predict concepts common to a set of documents belonging to different categories by utilizing the article link structure of Wikipedia. We further observed that using method 1 and 2 the best common concept that is predicted is very generalized i.e., "Agriculture" whereas by utilizing the article links we are able to predict a more specialized common concept. Using method 3 we can analyze Wikipedia as a whole to introduce new sub-categories within Wikipedia and aid in enriching its category hierarchy. We used three activation functions in our method 3 however; we do not note any significant difference in predictions using the different activation methods for this experiment.

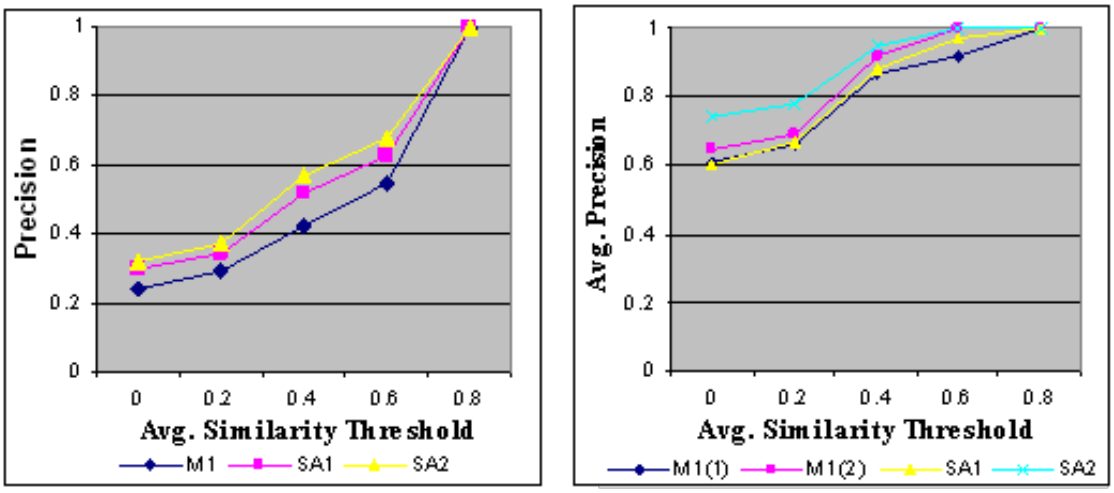
4.2.6 Evaluation

The experiments described in the previous section produced results that were encouraging, but serve only as an informal evaluation. The scale was small and the accuracy of the results were based on our own, possibly biased, judgments. We designed and ran a more formal evaluation by creating a test set of 100 articles ran-

domly from Wikipedia. We removed references to those articles from our Wikipedia index, article links graph and category graph. We then used our system to find related articles and categories for each of the 100 articles. The results were compared to the actual categories and article links found in Wikipedia, which we took to be the "ground truth", wielding measures of precision, recall and F-measure.

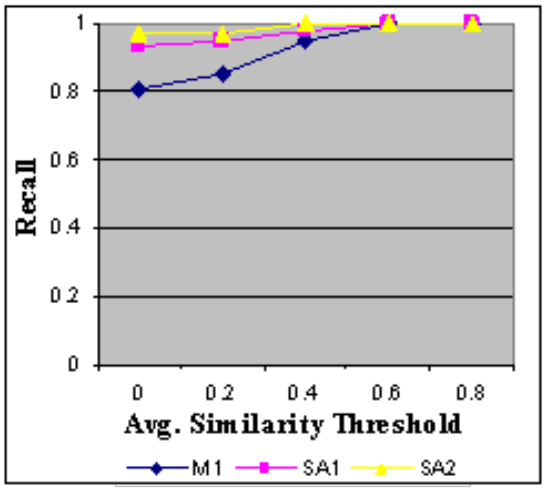
For evaluating the category prediction, for each Wikipedia test article we retrieved top ten similar articles from Wikipedia index based on cosine similarity between the documents. We took the average of the cosine similarity score between the test article and the top ten similar Wikipedia articles and sorted the test articles based on that score. We computed precision, average precision, recall and F-measure at different similarity score thresholds for all methods. For example, at 0.5 average similarity threshold we computed all metrics for the subset of test documents that had a score of greater than or equal to 0.5. For computing these metrics we included the top three level categories to the actual categories of the test documents so that if our method predicts a category that is a super-category at a distance of three then we consider it to be an accurate prediction.

Figure 4.1 shows our results. We observed that higher the average similarity scores the better the precision, average precision and recall for all methods. A comparison of the different methods using the F-measure metric shows that the method using spreading activation with two pulses (SA2) almost always performs better than other methods at different average similarity thresholds and also for the test document set as a whole. Measuring the average precision gives us an idea of our ranking schemes. We observed that in all cases the average precision is better

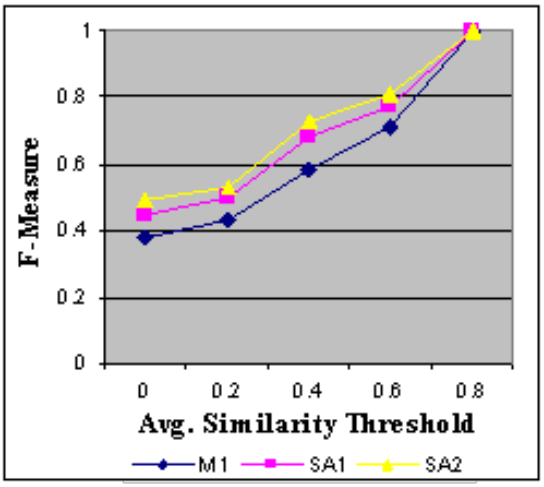


(a) Precision

(b) Average Precision



(c) Recall



(d) F-measure

Figure 4.1: These graphs show the precision, average precision, recall and f-measure metrics as the average similarity threshold varies from 0.1 to 0.8. Legend label M1 is method 1, M1(1) and M1(2) are method 1 with scoring schemes 1 and 2, respectively, and SA1 and SA2 represent the use of spreading activation with one and two pulses, respectively.

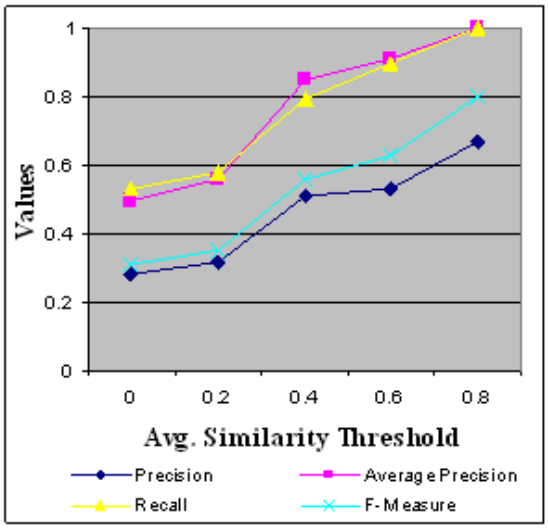


Figure 4.2: In the concept prediction task, precision, average precision and recall improve at higher similarity thresholds, with average precision remaining higher than precision, indicating that our ranking scheme ranks relevant links higher than irrelevant links.

than the precision for all methods indicating that our scoring scheme gives a higher score to the relevant results. The best average precision is given by method SA2 and is always higher than other methods. In case of Recall, SA2 gives highest recall at all thresholds. We also observe that M1(2) gives higher average precision than M1(1) hence, showing that scoring scheme 2 based on cosine similarity is superior to scoring scheme 1 based on number of occurrences. M1(1) also outperforms SA1 in case of average precision however, it is always lower than for SA2.

To evaluate our method for related concept prediction using Wikipedia article text and links, we used our test articles and removed their references from Wikipedia page links graph. Using our method we predicted the links of those articles and compared them with the actual links within Wikipedia using precision, average precision and recall. For each test article we retrieved top five similar ar-

ticles and ran spreading activation with one pulse and activation function number 2 with unit edge weights. Figure 4.2 shows our results for related concept prediction. We again observed that similar to the category prediction case the precision, average precision and recall improve at higher average similarity thresholds. However, at lower similarity thresholds the precision and recall are greatly affected. We also observe that the average precision is always significantly higher than precision, indicating that our ranking scheme ranks relevant links higher than irrelevant links.

4.2.7 Discussion

We have conducted three sets of experiments and also evaluated our methods using Wikipedia articles themselves. In the first set of experiments we only utilized the Wikipedia page texts to predict the category or concept related to a document. We gave each test document as input to Wikipedia articles index and got ten similar Wikipedia articles. We utilized the category information related to the matching articles to predict the category or concept of the test document using different scoring schemes. We experimented with a few documents and observed that the prediction was satisfactory for all of them. We also repeated the experiments with a group of documents related to a particular concept or topic instead of a single document and found the results to be encouraging in predicting the category of a group of related documents.

In the second set of experiments, in addition to using the Wikipedia article text we also applied spreading activation algorithm on the category links graph.

The purpose of applying spreading activation was to find out if we could extract a generalization of the concept or a common concept presented in the test document or set of test documents. We observed that depending on the input parameters of spreading activation, it helped in predicting nodes representing a broader or more generalized concept as compared to the initial prediction of concept. This method was observed to be useful in predicting the super-categories or super-concepts of the test documents.

In the third set of experiments we also included the article links information. The purpose of the experiment was to investigate if it is possible to predict a common concept for a set of documents given that the concept is not already represented as a Wikipedia category. Our general observation was that the concepts that are sufficiently represented in Wikipedia usually have a category associated with them, however, there may be certain cases where several pages may be related to a particular concept and that concept may not be represented as a category. To study this we took few such examples from Wikipedia and ran spreading activation on the article links graph to predict a related concept to a set of documents.

The results of experiments for predicting more specialized concepts related to a group of documents were also encouraging. Such a concept could be considered as representing a specialized topic related to a set of documents in contrast to a generalized topic or category. If the group of documents under consideration belongs to the same category then the predicted specialized concept could be used in defining a new Wikipedia subcategory whereas, if the group of documents does not belong to the same category then the specialized concept could be used in defining a new

relationship between those documents. For example, if we have an article related to a person and another article related to a location, we might be able to predict that the particular person and location are related to each other given a particular event which involved that person and occurred at the respective location however, we would not want to classify that person and location under that event.

An interesting application of the different methods that we have implemented and evaluated is that these methods could be used in recommending the categories and article links for new Wikipedia articles, or even in automatically building an enterprise Wiki from a given corpus by running our algorithms that utilize the category and article links information already present in Wikipedia.

Since we are currently employing Wikipedia as our knowledge base, predicting common concept to a set of documents is highly dependent on different factors inherent to Wikipedia:

- To what extent is the concept represented in Wikipedia: For example, there exists a category for the fruit "apple" however there is no category for "mango" since apple and its different varieties are discussed in detail in Wikipedia whereas for mango such information is limited to a few varieties.

- Presence of links between semantically related concepts: Since Wikipedia is developed by its users and not necessarily by experts hence the author of an article may not be able to add links to all other semantically related articles, and also doing that manually is infeasible in itself.

- Presence of links between irrelevant articles: Articles may be linked to other

Wikipedia articles irrelevant to their topics or concepts. For example articles mentioning a name of a country may be linked to that country's Wikipedia page. An article that mentions a term may be linked to the article defining and giving details on that term.

Hence the accuracy of our method is largely dependent on the above three factors. However, we have shown through our evaluation that the greater the similarity between a test article and its similar Wikipedia articles the better the prediction. Therefore the average similarity score may be used to judge the accuracy of prediction. For factors 2 and 3 related to the presence and absence of semantically related links between articles we could use the existing semantic relatedness measures to introduce additional links between semantically related articles or to filter out links between irrelevant articles.

4.2.8 Conclusions

In this use case we described the use of Wikipedia and spreading activation to find generalized or common concepts related to a set of documents using the Wikipedia article text and hyperlinks. We started our experiments with the prediction of concepts related to individual documents, extended them to predict concepts common to a set of related documents, and used the text and links of uncategorized Wikipedia articles to predict extant Wikipedia articles to serve as a category term. We have discussed the results of our experiments and have also evaluated those using random articles from Wikipedia itself. Our experiments show that it is possible

to predict concepts common to a set of documents by using the Wikipedia article text and links. We have also discussed some possible solutions for improving our results. Where earlier work has been directed towards computing semantic relatedness between text fragments, we have focused on a more challenging task of finding semantically related concepts common to a set of documents.

4.3 Cross Document Co-reference Resolution

4.3.1 Introduction

In this section, we describe the use of the Wikitology knowledge base as a resource for cross-document named entity coreference resolution task. Cross-document coreference resolution is the identification of entity mentions in different documents that refer to the same underlying entity. An entity is anything that might be referred to; however, for our purposes we will concentrate on named entities-those that are mentioned by name (e.g., "Barack Obama"). Such entities may also have nominal mentions (e.g., "the country's president"), pronominal mentions (e.g., "he"), or additional named mentions (e.g., "Barry").

Wikitology was used to define features that were part of a system [64] implemented by the Johns Hopkins University Human Language Technology Center of Excellence for the 2008 Automatic Content Extraction cross-document coreference resolution evaluation [1] organized by National Institute of Standards and Technology. In this task, systems had to extract entities and relations from a set of documents in English and Arabic and to identify which ones referred to the same entities or relations in the world.

4.3.2 Problem Statement

Given entity mentions in different documents, find out if they refer to the same underlying entity.

4.3.3 Approach

4.3.3.1 Wikitology 2.0

For use in the ACE cross document coreference task, we constructed an enhanced version of the Wikitology system as a knowledge base of known individuals and organizations as well as general concepts. This was used as a component of a system developed by the JHU Human Language Technology Center of Excellence [64].

For the ACE task, a system had to process 20,000 documents, half in English and half in Arabic and extract the entities and relationships mentioned in each, after performing intra-document coreference resolution (e.g., recognizing that "Secretary Rice", "Dr. Rice" and "she" referred to the same entity). Within each language set, systems then had to identify the document entities and relationships that refer to the same object or relationship in the world. For example, recognizing that "Secretary Rice" in one document and "Condoleezza Rice" in another refer to the same person but that these are not coreferent with a mention of "Dr. Rice" in a third document that in fact refers to Susan Elizabeth Rice, Barack Obama's nominee for the office of United States Ambassador to the United Nations.

The BBN Serif system [24] was used to extract intra-document entities and relations which were represented using the APF format [2]. Intra-document entities and relations information extracted from the output was processed by Wikitology

[2]APF is the ACE Program Format, an XML schema used to encode system output for ACE information extraction evaluations. It specifies, for example, various types and subtypes for entities and relations extracted from text documents.

to produce vectors of matching Wikitology terms and categories. These were then used to define twelve features that measured the similarity or dissimilarity of a pair of entities.

The Wikitology 2.0 knowledge base system used the Lucene information retrieval library and MySQL database and ran in two environments: on a single Linux system and on a Linux cluster for high performance. We used the cluster to process the small documents representing the approximately 125 thousand entities that Serif found in the ACE English test collection. The basic operation takes a text document and to return two ranked lists with scores: one for the best Wikitology article matches and another for the best category matches. Parameter settings determine what kind and how much processing is done, the maximum length of the vectors and thresholds for a minimum quality match.

4.3.3.2 Enhancements to Wikitology

For ACE 2008 we enhanced Wikitology in several ways and added a custom query front end to better support the ACE 2008 task. Starting with the original Wikitology, we imported structured data in RDF from Dbpedia [17] and Freebase [23]. Most of these data were in fact derived from Wikipedia, but have been mapped into various ontologies and re-imported in structured form. The structured data was encoded in a RDFa-like format in a separate field in the Lucene index object for the Wikipedia page. This allows one to query the system using both text (e.g., an entity document) and structured constraints (e.g., rdfs:type=YAGO:Person).

We enriched the text associated with each article with titles of Wikipedia "redirects". A redirect page is a pseudo page with a title that is an alternate name or mis-spelling for the article (e.g., Condoleeza_Rice for Condoleezza_Rice and Mark_Twain for Samuel_Clemons). An access to a redirect page results in the system returning the canonical page. The result is that the Wikitology pages for a term are indexed under these variant titles.

We extracted type information for people and organizations from the Freebase system. We found that the classification for these in Freebase was both more comprehensive and more accurate than that explicitly represented in either Wikipedia or DBpedia. This included information on about 600,000 people and 200,000 organizations. This information was stored in a separate database and used by the ACE Wikitology query system.

We extracted information from Wikipedia disambiguation pages to identify Wikitology terms that might be easily confused, e.g., the many people named Michael Jordan that are in Wikipedia. This information was stored in a separate table and used in the Wikitology feature computation.

4.3.3.3 Processing entity documents

We used special "entity documents" or EDOCs extracted from the APF (ACE Program Format) for the English documents as input to the Wikitology system. Each entity in a given document produced one EDOC that includes the following data as a semi-structured block of text: longest entity mention, all name mentions,

<DOC>
<DOCNO>ABC19980430.1830.0091.LDC2000T44-E2</DOCNO>
<TEXT>
Webb Hubbell
PER
Individual
NAM: "Hubbell" "Hubbells" "Webb Hubbell" "Webb_Hubbell"
NOM: "Mr . " "friend" "income"
PRO: "he" "him" "his"

, . abc's accountant after again ago all alleges alone also and arranged attorney avoid been b before being betray but came can cat charges cheating circle clearly close concluded conspiracy cooperate counsel counsel's department did disgrace do dog dollars earned eighty-nine enough eva sion feel financial firm first four friend friends going got grand happening has he help him his hope house hubbell hubbells hundred hush income increase independent indict indicted indictme nt inner investigating jackie jackie_judd jail jordan judd jury justice kantor ken knew lady la te law left lie little make many mickey mid money mr my nineteen nineties ninetyfour not nothing now office other others paying peter_jennings president's pressure pressured probe prosecutor s questions reported reveal rock saddened said schemed seen seven since starr statement such tax taxes tell them they thousand time today ultimately vernon washington webb webb_hubbell were what's whether which white whitewater why wife years

</TEXT>
</DOC>

Figure 4.3: Entity document capture information about entities found in documents, including mention strings, type and subtype, and text surrounding the mentions.

all nominal mentions, all pronominal mentions, APF type and subtype, all words within 15 tokens of each mention. The EDOCs were used to find candidate matches in the Wikitology knowledge base. Figure 4.3 shows an example of the EDOC for the entity for Webb Hubbell.

The EDOCS were processed by a custom query module for Wikitology that mapped the information in the EDOC into different components of Wikitology entries. The EDOC's name mention strings are compared to the text in Wikitol-

ogy's title field, giving a slightly higher weight to the longest mention, i.e., "Webb Hubbell" in our example. The EDOC type information is mapped into the Wikitology type information (YAGO terms imported from Dbpedia) and matched against the RDF field of each Wikitology entry. Finally the name mention strings with boost (^4) along with contextual text surrounding the mentions is matched against the text of the Wikitology entries. The Wikitology module returns two vectors: one for matches against article entries and the other against category articles.

4.3.3.4 Knowledge based features

We used an enhanced version of the Wikitology system [93] as a knowledge base of known individuals and organizations as well as general concepts. Entity information extracted from the Serif output was processed by Wikitology to produce vectors of matching Wikitology terms and categories. We produced twelve features based on Wikitology: seven that were intended to measure similarity of a pair of entities and five to measure their dissimilarity.

The similarity measures were all based on the cosine similarity of the article or category vectors for each entity and differed in the lengths of the vectors considered and whether they were Boolean or real-valued. For example, features 20 is true of both entities have type PER and their top article matches are identical. Feature 22 is the cosine similarity of the entities top five article matches, and 29 are applied to PER entities only and 28 and 30 to ORG entities. The four boolean features (20,21,26,28) have weighted versions (31,32,29,30) that factor in how strong the

```
Article Vector for ABC19980430.1830.0091.LDC2000T44-E2

1.0000 Webster_Hubbell
0.3794 Hubbell_Trading_Post_National_Historic_Site
0.3770 United_States_v._Hubbell
0.2263 Hubbell_Center
0.2221 Whitewater_controversy

Category Vector for ABC19980430.1830.0091.LDC2000T44-E2

0.2037 Clinton_administration_controversies
0.2037 American_political_scandals
0.2009 Living_people
0.1667 1949_births
0.1667 People_from_Arkansas
0.1667 Arkansas_politicians
0.1667 American_tax_evaders
0.1667 Arkansas_lawyers
```

Figure 4.4: Each entity document is tagged by Wikitology, producing a vector of article tabs and another vector of category tags.

Table 4.10: Twelve features were computed for each pair of entities using Wikitology, seven aimed at measuring their similarity and five for measuring their dissimilarity.

Name	Range	Type	Description
APL20WAS	0,1	sim	1 if the top article tags for the two entities are identical, 0 otherwise
APL21WCS	0,1	sim	1 if the top category tags for the two entities are identical, 0 otherwise
APL22WAM	[0..1]	sim	The cosine similarity of the medium length article vectors (N=5) for the two entities
APL22WcM	[0..1]	sim	The cosine similarity of the medium length category vectors (N=4) for the two entities
APL24WAL	[0..1]	sim	The cosine similarity of the long length article vectors (N=8) for the two entities
APL31WAS2	[0..1]	sim	match of entities top Wikitology article tag, weighted by avg(score1,score2)
APL32WCS2	[0..1]	sim	match of entities top Wikitology category tag, weighted by avg(score1,score2)
APL26WDP	0,1	dissim	1 if both entities are of type PER and their top article tags are different, 0 otherwise
APL27WDD	0,1	dissim	1 if the two top article tags are members of the same disambiguation set, 0 otherwise
APL28WDO	0,1	dissim	1 if both entities are of type ORG and their top article tags are different, 0 otherwise
APL29WDP2	[0..1]	dissim	Match both entities are of type PER and their top article tags are different, weighted by 1-abs(score1-score2), 0 otherwise
APL30WDP2	[0..1]	dissim	Dissimilar two ORG entities match different Wikitology orgs, weighted by 1-abs(score1-score2)

matches are.

4.3.4 Experiments and Evaluation

The ACE 2008 evaluation was a cross-document coreference resolution task over a collection of 10,000 English and 10,000 Arabic language documents of several genres (e.g., newspaper stories, and newsgroup postings). In such a task, one

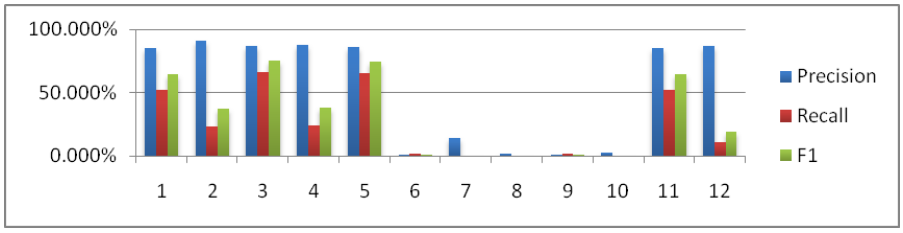


Figure 4.5: The twelve Wikitology-based features varied in their usefullness in disambiguating entity references in the collection of 10,000 English language documents used in the 2008 xdoc task. Features APL20WAS, APL22WAS, APL24WAL and APL29WDP2 enjoyed good F1 measures.

must determine whether various named people, organizations or relations from different documents refer to the same object in the world. For example, does the "Condoleezza Rice" mentioned in one document refer to the same person as the "Secretary Rice" from another?.

The larger system to which we contributed had a number of different components, including the Serif information extraction system developed by BBN. The overall approach worked as followed, focusing on the analysis of entities for English for ease of explanation. Serif was used to extract a set of entities for each of the 10K documents, producing approximately 125,000 entities. A heuristic process was used to select about one percent of the 16x109 possible pairs as being potentially coreferent. For each of these 160M pairs, over 30 features were computed as input to an SVM-based classifier that decided whether or not the pair was coreferent. The resulting graph of coreference relations was then reduced to a collection of equivalence sets using a simple technique of finding connected components.

An analysis of Wikitology based features showed that several of these KB features were among those highly weighted by the final classifier (Figure 4.5) . We also analyzed a system constructed with only the Wikitology features on a smaller

set of documents and entities for which human judgments are available. This gives us some additional indication of how well the features worked as a group.

We constructed an SVM based on the twelve Wikitology features using a data set of 154 entity documents mapping to 52 distinct external entities. For this dataset, we had human judgments for the cross-document entity co-reference task that mapped each entity into an external set, resulting in 241 entity pairs that referred to the same external entity. These entity pairs were used as positive examples. The negative examples were generated using the same entity document set in the following way:

1. The 154 EDOCs were paired with each other exhaustively resulting in (154*154 = 23716 pairs).

2. From the generated pairs the following pairs were removed:

    - Pairs between same EDOCs eg. edoc1,edoc1 i.e, 154 pairs.
    - Pairs present in positive examples i.e, 241 pairs which were manually labeled earlier and also removing the symmetric pairs to positive pairs (241 symmetric pairs) i.e. 241+241=482 positive pairs in total.
    - After removing the above mentioned pairs, the remaining pairs (23716-154-482= 23080) were labeled as negative examples and their symmetric pairs were also removed resulting in 23080/2=11540 negative pairs.

A test set was generated in the same way using another set of 115 entity documents mapping to 35 distinct external entities with 234 pairs labeled as positive

Table 4.11: Evaluation results for Cross Document Entity Coreference Resolution using Wikitology Generated Features.

match	TP rate	FP rate	precision	recall	F	ROC area
yes	.722	.001	.966	.722	.826	.861
no	.999	.278	.99	.999	.994	.861

examples through human judgments. The negative examples were created in the same way as mentioned above for the training set. The total number of pairs in the test set were 6,555 and were composed of 234 pairs as positive examples and 6321 pairs as negative examples.

We used SVM to classify pairs of document entities as either co-referent or not and used 10-fold cross validation to evaluate the result. Of the 6,555 pairs in the collection, 6,484 (98.9%) were correctly and 71 (1.08%) incorrectly classified. Table 4.11 presents some of the key statistics.

4.3.5 Conclusions

An enhanced version of the Wikitology system was constructed as a knowledge-base resource for use in the cross-document entity co-reference resolution task that was the focus of the 2008 Automatic Content Extraction evaluation. This was used to define features that contributed to a system developed by the Human Language Technology Center of Excellence. Our evaluation shows that these features are indeed useful in providing evidence for the cross-document entity coreference resolution task.

4.4 Entity Linking

4.4.1 Introduction

In this section, we describe the enhancements done in Wikitology 3.0 and the approach used for Entity Linking task defined in the Knowledge Base Population (KBP) track of the 2009 Text Analysis Conference [65]. One part of the KBP track is to link entities found in text to those in an external knowledge base. In particular, the entity linking task is defined in the problem statement below.

4.4.2 Problem Statement

Given an entity mention string and an article with that entity mention, find the link to the right Wikipedia entity if one exists.

4.4.3 Approach

We used Freebase resource [23] to label Wikipedia articles on entities as persons, locations and organizations. Freebase is an online database of the world's information developed by the community and contains facts in different categories such as people, locations, books, movies, music, companies, science etc. Freebase also has a lot of data that is imported from DBpedia [17]. Using "person", "location" and "organization" type specified in Freebase resource, we were able to construct a list of Wikipedia articles that were on Persons, Locations and Organizations with more than 13,000 organizations, 550,000 persons and 350,000 locations. We updated

the Wikitology index and included a field for "Entity Type". We implemented different querying approaches for our specialized Wikitology index which are described below.

In approach 1 we query the index using the entity mentions against the titles and redirects field and then search within the returned results by querying the given entity document against the Wikipedia article contents field.

In approach 2 we query the index using the entity mentions against the titles and redirects field and the given entity document against the Wikipedia article contents field.

In approach 3 we query using the entity mentions against the title and redirects fields and then search within the returned results by querying the entity mentions against the title and redirects fields and the given entity document against the Wikipedia article contents field.

In approach 4 we query using the entity mentions against the title and redirects fields and then search within the returned results by querying the given entity document against the Wikipedia article contents field and against the YAGO types field and the entity mentions against the title, redirects and contents field with a boost of 4.

The different approaches return a list of ranked Wikipedia entities and the top most is selected as the correct match. To detect if the test entity is a new entity and doesn't exist in our Wikitology system we learn a threshold and if the score of the top most entity is below the threshold we report a nil match or a "New Entity".

4.4.3.1 Heuristics based on Entity Types

We developed a few heuristics based on the type of entity for persons and locations. In case of persons, we used approach 4 however, if a name initial was present in the entity mention we included that in the query by introducing the initial followed by a wild card in the query to also match any entity names that contain the full name rather than the initial, this helped in giving a higher score to "Chris Broad" for the entity mention "C Broad" as compared to "Dan Broad".

For locations, we replaced all adjectival forms of place names with noun forms by using the list of adjectival place names in Wikipedia so words like "Australian" would get replaced by "Australia" in the entity mention strings. We also used another heuristic for matching locations. In Wikipedia place names are often followed by names of locations that would help in disambiguating them. For example, Baltimore is a name of several places in Wikipedia. The different Baltimores are distinguished by having another place name in the title such as "Baltimore, Ohio" or "Baltimore, Indiana". To exploit this information present in the titles of Wikipedia pages, we extracted all the locations from the test entity article and if any of the locations appeared in the title of the top five matching locations, that Wikipedia entity was selected as the matching entity.

4.4.4 Experiments and Evaluation

In order to evaluate the entity linking task we used the Wikinews corpus from October 2009 dump [103], which consists of news articles that are linked manu-

ally by contributors to relevant Wikipedia articles. We extracted all the links to Wikipedia articles and the surface text associated with the links as entity mentions. We created a test set of 1000 articles and entity mentions for persons, locations and organizations each by randomly selecting articles which had links to persons, locations and organizations in Wikipedia.

4.4.4.1 Evaluation for Existing KB Entities

We conducted different experiments to evaluate the different approaches. For experiments number 1 to 5 we used separate Wikitology indices for Persons, Locations and Organizations whereas for experiment 6 we used the full index with Persons, Locations and Organizations without any information about the entity type of the test entity being queried. The entity type of an entity can be predicted using the entity mention and the related article by locating the entity mention in the article and using any NER system to label the type of entity. Once the entity type is known, the entity could be queried against the respective index. In case it is not possible to know the entity type in advance, then the query could be directed to the full index as we do in experiment 6.

Table 4.12 reports the accuracy obtained for Persons, Locations and Organizations using the different approaches. We observed that amongst the four different approaches, approach 4 in general gave the highest accuracy for all the three entity types i.e. 95.2%, 86.2% and 86.1% for Persons, Locations and Organizations respectively. The specialized approaches for Persons and Locations further improved the

Table 4.12: Accuracy obtained for Entity Linking Task for entities that exist in Wikipedia using different approaches.

Exp No.	Approach	Person	Location	Organization
1	Approach 1	66.8%	51.6%	56.8%
2	Approach 2	94.2%	85.0%	84.7%
3	Approach 3	95.1%	85.8%	85.5%
4	Approach 4	95.2%	86.2%	86.1%
5	Specialized Approach	96.9%	93.3%	-
6	Approach 4 (Full Index)	94.9%	85.0%	82.8%

accuracy from 95.2% to 96.9% and from 86.2% to 93.3% for Persons and Locations. The improvement in accuracy was seen more in case of Locations as compared to Persons. Using approach 4 on the full index with all the three types of entities resulted in a slight drop in the accuracy for all the three entity types as compared to when approach 4 is used on individual indices for the entity types.

4.4.4.2 Evaluation for New KB Entities

In order to detect if the entities are not present in Wikitology we wanted to learn a threshold that we could use to distinguish between an existing entity in the knowledge base and a new entity. Our approach for Entity Linking for existing entities is totally unsupervised, however, to detect new entities we use a very small set of labeled examples.

We constructed a list of scores for positive and negative entity matches using the following approach. We used approach 4 to query Wikitology and retrieved top ranking entities using our test data of 1000 entity mentions and entity articles for people, locations and organizations each. In case the top most match was not the

Table 4.13: Accuracy obtained for positive and negative entity matches using learned score thresholds.

Entities	Negative	Positive	Total
Persons	84.0%	92.1%	88.1%
Locations	59.8%	83.4%	71.8%
Organizations	92.2%	67.8%	79.0%
All	78.7%	81.1%	79.9%

right one, we labeled that score as a negative match. If the top most match was a correct match we label the score as a positive match and the score of the second ranking entity as a negative match, because in case the first entity was not in the Wikitology knowledge base it would have predicted the second entity as the top most match which would be an example of negative match.

We used decision tree algorithm in Weka [48] to learn a score to split the positive and negative matches. We learned the threshold using only 5% of the data as the training set and then tested it using the 95% of the remaining scores. In Table 4.13 we report the accuracy obtained for the positive and negative entity matches using the learned thresholds for people, locations and organizations dataset separately and then for the combined dataset.

The highest accuracy for predicting a positive and negative match using thresholds separately for each type of entity was for persons (88.1% accuracy) with positive matches being predicted more accurately (92.1%) as compared to negative matches (84.0%), followed by organizations (79.0% accuracy) in which case the accuracy for predicting a positive match (67.8%) was much lower than for predicting a negative match (92.2%). For locations (71.8%) the accuracy for predicting a positive match

(83.4%) was higher than for predicting a negative match (59.8%). When a single threshold was used to detect positive and negative matches for the three type of entities, the overall accuracy was 79.9%, with the accuracy of positive match being slightly higher than the accuracy for negative match prediction.

4.4.5 Conclusions

In this case study we have presented an approach for linking entities mentioned in documents to entities in Wikitology. We have employed the Wikitology knowledge base for the entity linking task and our results show that the Wikitology based approach gives an accuracy of above 80% for linking persons, locations and organizations to the right entities in Wikipedia. We also developed an approach to identify if the entities do not exist in Wikipedia and got an accuracy of about 80% for different types of entities. The entity linking approach described in this study cannot only be incorporated in variety of systems requiring to link entities to entities in the KB such as for getting additional context, but also serves to evaluate the performance of Wikitology hybrid knowledge base.

4.5 Interpreting Tables

4.5.1 Introduction

Much of the world's knowledge is contained in structured documents like spreadsheets, database relations and tables in documents found on the Web and in print. The information in these tables might be much more valuable if it could be appropriately exported or encoded in RDF, making it easier to share, understand and integrate with other information. This is especially true if it could be linked into the growing linked data cloud. We have exploited Wikitology as a part of a larger system "T2LD" to automatically infer a (partial) semantic model for information in tables using both table headings, if available, and the values stored in table cells and to export the data the table represents as linked data. We first give an overview of the approach and then descibe in detail the steps in which Wikitology directly contributed and how it performed for the given task.

4.5.2 Approach

4.5.2.1 From Tables to Linked Data

Producing linked data to represent a table is a complex task that requires developing an overall interpretation of the intended meaning of the table as well as attention to the details of choosing the right URIs to represent both the schema as well as instances. We break the task down into the following tasks:

- Associating classes or types with each column based on the column header

and values and selecting the best one

- Linking cell values to entities, if appropriate
- Associating properties that represent the implied relations between columns and selecting the best ones

While these could be addressed serially, the problems are intertwined. T2LD system approaches the problems mostly serially with some interactions. The labels in the table headers, if present, as well as the values in the rows can be exploited for interpreting the information contained in the tables. Linking the table headers as well as the instances in the rows to concepts in a knowledge base can aid in providing more context and links to other related concepts. The main steps in the approach are as follows:

1. Initial Entity Linking: For each row in the table Query Wikitology using as input table headers and cell values
2. Use initially linked cell values to predict class label for the columns in the table
3. Requery Wikitology including information about predicted column labels
4. Final Entity Linking: Link cell values to concepts in Wikitology with new evidence
5. Identify relations between columns
6. Output Linked Data

4.5.2.2 Exploiting Wikitology 3.0

We exploited Wikitology 3.0 targeted towards Entity Linking task for interpreting tables. The simple Entity Linking task is defined as: given an entity mention string and an article with the entity mention, link it to the right Wikipedia entity. The entity mention represents the entity to link and the article provides additional context about that entity. In the case of tabular data, the individual entry in a particular row and a column represents the entity mention. Instead of a document, the different parts of the table serve as the context to disambiguate the entity or concept. Similar to the entity linking task it is not trivial to link table headers or values to concepts in the KB as the same concept may be expressed in a variety of ways in the table headers as well as data rows and there might not be enough context available in the table.

A table may be defined as a two-dimensional presentation of logical relationships between groups of data [96]. It is often the case that the table column header represents the type of information in that column such as cities, countries, artists, movies etc. whereas, the values in the columns represent the instances of that type. The values in the rows of the tables may represent related information to the instances in the same row.

There are different fields available in Wikitology's specialized IR index that can be exploited for inferring a (partial) semantic model for information available in tabular forms. We process the information in the table using a custom query module for Wikitology that maps the information in different parts of the table to

different components of Wikitology index.

4.5.2.3 Enhancements to Wikitology

In addition to different fields representing different types of content extracted from Wikipedia and related resources, we also introduced a field with PageRank approximations of Wikipedia articles. Earlier work on Entity Linking task [38] identified Google PageRank as an important feature for linking entities mentioned in text to Wikipedia entities. However, it is not possible to query Google for PageRank of all Wikipedia articles, therefore we have developed an approach for approximating the PageRank for Wikipedia articles and incorporate the predicted PageRank in Wikitology 3.0.

We selected 4296 random articles from Wikipedia and queried Google for their PageRank resulting in the distribution shown in column two of Table 4.15. We used these to train a classifier using the number of in- and out-links and the article length as classification features and the PageRank as the class label. We used 10-fold cross-validation to evaluate four common classification algorithms: SVM, decision trees with and without pruning and Nave Bayes. The pruned Decision Tree algorithm gave the highest accuracy of 53% followed by 52%, 50.34%, 49.8% and 41% for an un-pruned decision tree, Linear Regression, SVM and Nave Bayes, respectively 4.14.

The accuracy obtained per PageRank class using decision tree (J48) algorithm is given in Table 4.15 third column. We recomputed the accuracy and considered a prediction to be correct if there is a difference of one between the predicted and

Table 4.14: Accuracy for PageRank prediction for different classification Algorithms.

Algorithm	Accuracy
SVM	49.8%
Nave Bayes	41.5%
Decision Tree (J48)	52%
Decision Tree (J48 Pruned)	53.14%
Linear Regression	50.34%

actual PageRank, which we considered to be sufficient for our application. This resulted in an overall accuracy of about 94% with a distribution shown by the final column in Table 20. Since majority of the Wikipedia articles have PageRank values between three and four, having a high accuracy for these classes indicates that we can approximate the PageRank for majority of Wikipedia articles with high accuracy.

We examined the different feature combination to discover which feature combination was best, with results shown in Table 20. The combination of in-links and page length gave the best accuracy (54.81%) closely followed by in-links only (54.03%). For our 1 accuracy, the best result was obtained using in-links only (95.69%). We are currently using the decision tree (J48 Pruned) classification algorithm using in-links only feature set to approximate the PageRanks for Wikipedia articles.

Feature Contributions: We also experimented with different feature combination to estimate which feature combination was contributing the most. The results are given in Table 4.16. In case of accuracy InLinks + Page Length gave the best results (54.81) which were followed by InLinks only (54.03). When the Accuracy was computed considering a prediction to be correct if it is within +/- 1 range of

Table 4.15: Accuracy per PageRank class using Decision Tree (J48 pruned) algorithm.

PageRank	No. of Articles	Accuracy	Accuracy (+/-1)
0	44	0.062	0.05
1	18	0.080	0.28
2	100	0.191	0.68
3	1129	0.422	0.95
4	2124	0.640	0.98
5	794	0.394	0.91
6	84	0.420	0.75
7	3	0	0.67

Table 4.16: Feature contributions for PageRank approximations.

Features	Acc.	Acc.(+/- 1)
All	53.14	93.78
All - Page Length	53.91	95.30
All InLinks	51.46	93.23
All - OutLinks	54.81	95.34
InLinks	54.03	95.69
Page Length	49.95	94.20
Out Links	49.46	94.13

the PageRank, the highest accuracy was obtained by InLinks only (95.69). Based on our results we used Decision Tree (J48 Pruned) classification algorithm using inlinks only feature set to approximate the PageRanks for Wikipedia articles and included that in the newer version of our Wikitology specialized index.

4.5.2.4 Wikitology Query Module for Initial Entity Linking and Class Prediction

In order to link the instances in the table rows to the entities in Wikitology we get the context for the instances in the following way. The table header suggests

the type of the instance, whereas, the values in the same row suggest concepts and literals associated with the instance.

Our custom query module maps the instance mention to the "title" field. The "title" field contains the Wikipedia article or concept titles. The instance mention is also mapped to the "redirects" field which contains the Wikipedia redirects to the article or concept. A Wikipedia redirect page is a pseudo page with a title that is an alternate name or misspelling for the article (e.g., Condoleeza_Rice for Condoleezza_Rice and Mark_Twain for Samuel_Clemons). An attempt to access a redirect page results in the Wikipedia server returning the canonical page. The result is that the Wikitology pages for a term are effectively indexed under these variant titles.

The table header is mapped to the "types" field. The "types" field contains the DBpedia classes, YAGO classes, WordNet classes and Freebase classes (i.e. Person, Location and Organization) associated with the concept.

The entity mention and the column header is mapped to the "firstSentence" field. It is often the case that the first sentence in Wikipedia usually defines the concept and mentions the type of the concept as well.

The values in the same row are mapped to the "contents" field and the "linkedConcepts" field in Wikitology, giving a boost of 4.0 to the instance mention itself. The "contents" field contains article text including the title, redirects, infobox properties and values as well as the linked categories. The "linkedConcepts" field enlists all the linked concepts in Wikipedia. We also map the row values to the "propertiesValues" field which contains the DBpedia infobox properties and value pairs for

a concept.

The Wikitology query module returns a vector of top N matching Wikitology entities. To associate the best class with a set of strings in a column we do the following:

1. Let 'S' be the set of 'k' strings in a table column (e.g., Baltimore, Philadelphia, New York, Boston ...).

2. We use Wikitology to get the top 'N' possible Wikipedia entities for each string and their types or classes.

3. Let 'C' be the set of all associated classes for a column (e.g., place, person, populatedPlace).

4. From the top 'N' Wikipedia entities predicted for each string in the table, we vote for each class in the set C based on the entity's rank, i.e. 1/R (1 for 1st, 1/2 for 2nd , 1/3 for 3rd and so on). We also incorporate the normalized PageRank of the top 'N' Wikipedia entities using a weighted sum (w = 0.25):

$$Score = w \times \frac{1}{R} + (1 - w) \times (PageRank) \tag{4.9}$$

5. We create a (sparse) matrix V[i,j] with the value we assign to interpreting string i as being in class j. The default value V[i,j]=0 is used when string i is not mapped to class j at all.

6. To pick the best class for the column, we find the class j that maximizes the sum of V[i,j] for 0<i<length(C).

We repeat the process for four types of classes existing in Wikitology i.e., DBpedia class, YAGO class, WordNet and Freebase (Freebase type currently includes Person, Location and Organization only). We normalize the weight for each class with the number of strings present in a column. A class label is considered for selection as a best class for the column only if its normalized weight is more than a threshold weight. We use 0.3 as the threshold.

4.5.2.5 Wikitology Query Module for Final Entity Linking

We use the predicted types as additional evidence to link the instances to the right Wikipedia entities. We requery Wikitology by updating the original query described in section 4.5.2.4, by adding the predicted types mapped to the "typesRef" field in the index using an AND clause. The "typesRef" field is an un-tokenized field and supports exact match. Querying using the typesRef field with an AND clause restricts the results to only those entities whose type exactly matches the input type. For each cell we retrieve the top N matching Wikitology concepts.

The T2LD system generates a feature vector for each entry in the top N concepts based on Wikitology ranked score, PageRank, Page Length, Lavenshtein Distance and Dice Score. The feature vector is then passed on to SVM Rank classifier in the T2LD system and the top most ranked concept is selected. The top ranked concept along with its svm rank score and difference of scores between top two svm ranked concepts is passed on to the next classifier producing a "yes" or "no" label to identify if the predicted concept should be linked or if a matching concept doesnt

Table 4.17: Test Data Set for Interpreting Tables.

Description	Count
Number of Tables	16
Total Number of Rows	199
Total Number of Columns	52
Total Number of Entities to be Linked	611

exist in Wikitology KB. The T2LD system contains other approaches for exploiting relations in DBpedia and identifying relations between table columns.

4.5.3 Experiments and Evaluation

The T2LD system exploits Wikitology for two tasks i.e., 1) Column label or class prediction and 2) Cell value linking. We discuss the performance of T2LD system for these two tasks. The test data set was generated from Google Squared [47], Wikipedia and tables on the Web. The statistics for the data set are given in Table 4.17. The distribution of type of entities in columns is given in Table 4.18. Three human judges was asked to evaluate the results for class label prediction and cell linking. The evaluation results for class label prediction and cell value linking are given in Table 4.19. For class label prediction the system gave best accuracy on Location class i.e. 90.48% and an overall accuracy of 76.92%. For cell value linking task the best accuracy was obtained for Persons i.e. 83.05%, however, the overall accuracy was 66.12% as the system did not perform well for other types of entities such as movies, nationalities, songs etc.

Table 4.18: Entity Type Distribution in Test Data Set for Interpreting Tables.

Entity Type	Distribution
Person	20%
Location	45%
Organization	10%
Other	25%

Table 4.19: Evaluation Results for Class Label Prediction and Cell Linking.

Entity Type	Class Label Accuracy	Cell Linking Accuracy
Person	76.92%	83.05%
Location	90.48%	80.43%
Organization	66.67%	61.90%
Other	58.33%	29.22%
All	76.92%	66.12%

4.5.4 Conclusions

Realizing the Web of Data vision requires making a significant amount of useful data available in RDF and linked into the growing linked data cloud. Achieving this will require developing techniques to automate or mostly automate the extraction of linked data from unstructured, semi-structured and structured sources. We have given an overview of the T2LD system and specifically discussed the steps in which Wikitology Knowledge Base is exploited to automatically infer a (partial) semantic model for information in tables using both table headings, if available, and the values stored in table cells. The techniques have been prototyped for a subset of linked data that covers the core of Wikipedia. Evaluation on the test data shows that the techniques are promising especially for Person, Location and Organization Entity types and can be employed for automated extraction of linked data from structured data in the form of tables.

Chapter 5

Wikitology Hybrid KB Design, Query Interface and API

5.1 Introduction

We initially introduced the Wikitology knowledge base in chapter 3 and in chapter 4 we presented novel Wikitology based approaches for solving a variety of real world problems that guided and directed the final design of the Wikitology knowledge base. In this chapter, we discuss in detail the design and architecture of Wikitology knowledge base for storing, handling and querying multiple data representations available in Wikipedia and related resources. Wikipedia proves to be an invaluable resource for generating a hybrid knowledge base due to the availability and interlinking of structured and un-structured encyclopedic information ranging from highly structured triples in the info-boxes to un-structured free text available in the content of the articles. We first review existing approaches of accessing and querying Wikipedia and also discuss different knowledge resources derived from Wikipedia, we then describe in detail the architecture and design of Wikitology hybrid KB and introduce the Wikitology API for querying the knowledge base. We finally demonstrate how different approaches already discussed in detail in chapter 4 can access and exploit the knowledge in Wikitology using the API.

5.2 Related Work

Wikipedia provides a simple search interface in which a user can directly enter the search term to retrieve an article. A perl module [92] is also available for automatically querying Wikipedia however, it is discouraged as it may significantly increase load on the server. There are some other approaches which rely on web crawling [88], however, these approaches are not suitable for large scale applications. Another approach is to upload Wikipedia on a local server however, it also poses overheads associated with sending request, processing of request by the web server and returning results. The Wikipedia xml dumps can also be directly accessed using the perl module Parse::MediaWikiDump [84] however, since the dumps are huge (several GBs) the times is not constant for accessing articles in different parts of the xml corpus.

Zesch et al. [109] have developed a java based Wikipedia library (JWPL) to provide efficient access to Wikipedia by applications. They process the Wikipedia dump and store it in an optimized schema in a database, explicitly storing information about article links and categories which is not explicitly available in the XML dumps. Since their api is based on a relational database, there is no relevance score available for approximate matches. In their PageQuery one can specify a regular expression to match the title of the page however, it is not possible to match any keywords in the contents of the articles. It is also not possible to access information in the infoboxes using the JWPL currently.

Wikipedia has been exploited in different ways to generate knowledge resources

that can be used by applications for solving different problems. DBpedia is a community effort focused on extracting structured information from Wikipedia and making it available for human and machine consumption on the web [17]. The extraction process involves mapping of the relations already present in Wikipedia relational database tables onto RDF, additional information is extracted directly from article templates and info-boxes. Recently the most common info-boxes in Wikipedia have been used to create a shallow, cross-domain ontology in DBpedia, with using manual mappings of Wikipedia info-boxes to the DBpedia ontology.

Suchanek et al. developed the YAGO ontology [91] by extracting facts from Wikipedia and unifying with WordNet. They developed several heuristics to extract facts from Wikipedia infoboxes and categories and linked the leaf categories with WordNet to represent ISA hierarchy for their ontology. Freebase [23] is a collaboratively created graph database of structured human knowledge. It imports structured data from Wikipedia and also allows users to directly contribute structured data to Freebase. Linked Open Data [22] is a community effort that focuses on publishing existing open license datasets as linked data on the web, interlinking things between different data sources and developing client applications that can use the linked data. DBpedia is serving as a core for linking other open data sets on the web.

All these resources store structured data mainly in the form of triples and permit structured queries only, whereas, in Wikitology KB we integrate both structured and un-structured data and provide an integrated query interface enabling free-text queries with structural constraints and also returning ranked results with relevance scores.

Powerset [29] is a search engine based on Wikipedia. It extracts facts from sentences in Wikipedia through deep linguistic analysis. It accepts natural language queries and retrieves the answers from Wikipedia articles. It also provides an interface for searching for facts, however, it does not support a query with both facts and free-text and does not currently integrate the structured data available in Wikipedia via the infoboxes, categories, inter-article links etc. which can provide more information and context.

5.3 Wikitology Architecture and Design

A major challenge in developing a hybrid knowledge base is the selection of an appropriate data structure to represent or link data available in different forms that would not only serve to store and represent hybrid data but also provide a query interface giving an integrated view.

In case of structured data domain one can define a perfect schema for storing and querying the data. The results returned by the query are usually from exact matches and determined by binary decisions, whereas, in case of IR systems the data is unstructured usually in the form of free-text and the queries return results that are ranked based on some relevance measure generally expressed by a floating point score between 0 and 1.

Based on the need and importance of integrating un-structured data with structured data, several database systems have started offering full-text search capabilities as well. Arslan and Yilmazel [16] compared the performance of text

retrieval in IR Library "Lucene" [52] and relational databases (MySQL[73] and postgreSQL [11]) for Turkish language. They reported that the performance of relational databases is very low without linguistic pre-processing and even after linguistic pre-processing, Lucene performed better than the database systems (MySQL and postgreSQL). They also reported that the ranking algorithms in IR library are more robust and when they used long queries for databases the search time was impractical. The indexing times were also reported as much slower than Lucene.

Several Triple stores have also started integrating full-text search capabilities for literals in the triples. Queries in structured query languages such as SPARQL are not powerful enough for very large data sets. SPARQL provides complete string matching or regular expression based filtering which are slow operations [71] whereas, state of the art IR approaches are more scalable and provide keyword based searches as well as ranked results with relevance score. Most of the triple stores have incorporated IR techniques by simply indexing the RDF literals. RDFStore [83] maintains one inverted index for all literals. It only permits simple keyword queries and does not provide any ranking of results. YARS [50] also uses an inverted index for all literals however, it only permits queries in N3 notation and does not provide any scoring. 3store [49] is a mySQL based implementation of triple store and uses the full-text index for each column which allows for simple keyword queries, Boolean operators, phrase queries and also a relevance measure. Minack et al. developed LuceneSail [71] to incorporate full text search capabilities in RDF store by combining their RDF indexing store Sesame2 with Lucene. Their system enables full text queries over the RDF literals within structured queries using virtual properties.

Kowari [105] also integrates Lucene support but only implements a subset of RDQL [87].

For Wikipedia, most of the knowledge is available in the form of natural language text as content of encyclopedic articles. Approaches for indexing and querying IR indices are more efficient and scalable as compared to triple stores [71]. Therefore, we select an information retrieval (IR) index as our basic data structure. To integrate it with other forms of knowledge, we enhance the IR index with fields containing related instance data from other data structures such as graphs, tables or triples. We also store the references to related instances in other data structures for applications that might need to run data-structure specific algorithms. Our specialized IR index enables applications to query the knowledge base using either simple free text queries or complex queries with or without structural constraints over multiple fields in the index.

5.3.1 Lucene IR Library

Apache Lucene [52] is a powerful, open source, cross platform, IR library written in java which provides full-featured text search with high performance and scalability and is freely available. It provides efficient algorithms for ranked searching, supports many powerful query types such as phrase queries, wildcard queries, proximity queries, range queries, fielded searching (e.g., title, author, contents), date-range searching, sorting by any field, multiple-index searching with merged results and also allows simultaneous update and searching. Tools like Luke [8] provide

a Graphical User Interface that can be used to manually query the index using the rich query language available in Lucene. The Lucene index is composed of Document Objects. Each Document is a collection of fields (name/value pairs). The field represents a piece of data that can be queried and retrieved from the index during search. Lucene supports different kinds of options related to storing and indexing fields depending on the need of the application. Lucene uses an inverted index for each indexed field.

5.3.2 Scalability

Lucene supports distributed indexing and querying over indices in parallel over remote machines, which can be achieved using different configurations and architectures. Experiments using Nutch [72] a search engine based on Lucene, have reported it to be highly scalable. We currently have Wikitology running on a single machine however, if more knowledge resources are to be integrated with Wikitology in the future then it could be run in parallel using different architectures.

5.3.3 Ranking Function

Wikitology KB currently uses the default ranking function provided by Lucene in the org.apache.lucene.search.DefaultSimilarity class which correlates to the Vector Space Model [85]. The documents and queries are both represented as vectors of terms weighted by TfIdf scheme (Equation 5.1). The similarity of the document

vector to the query vector is computed using the cosine similarity (Equation 5.2).

$$tfidf_i = tf_i \times \log \frac{D}{Df_i} \tag{5.1}$$

tf_i = term frequency (term counts) or number of times a term i occurs in a document.

df_i = document frequency or number of documents containing term i

D = number of documents in a corpus

$$Sim(D, q) = cos\theta = \frac{d \bullet q}{\mid d \mid\mid q \mid} \tag{5.2}$$

D = Document vector

q= Query vector

The ranking function can be overridden depending on the requirements of the application. It might be inconvenient or even impractical to change the ranking of results returned by text searches in databases. Therefore, implementing Wikitology using Lucene not only provides a flexible query interface but also permits applications to change the rankings according to their needs by overriding the functions in Lucene classes.

5.3.4 Extending and Updating the Index

Documents in the index can be updated, however, Lucene currently updates a document by first deleting it and then adding it to the index, this might be expensive for large number of dynamic updates. New Wikipedia dumps are usually available on monthly basis. Resources depending on Wikipedia such as DBpedia and Freebase

usually update Wikipedia based data once a month. It is more practical to update the index from the new dump every month instead of dynamically updating the index each time an article is edited in Wikipedia.

5.3.5 Wikitology Specialized IR Index

We downloaded the Wikipedia mySQL tables and xml article dump from March 2008. Each Wikipedia article represents a concept. We represent each Wikipedia concept as a Lucene Document. We store information and context related to the concept in different fields in the Lucene Document. We use different field options available in Lucene for representing different kinds of information related to the concept.

Lucene provides the option to store the contents of a field in the index , in which case the contents of a field are not only available for searching but can also be retrieved from the index and used in some application however, it also results in the increase in size of the index. We add certain data values as both tokenized and un-tokenized fields in the Lucene index. Both kinds of fields can be searched however, the untokenized fields are stored as a single term and useful for the cases where we are implementing some constraint on the search that requires an exact match with the single term in the field or for using the value of that field as a reference to an instance in an external data structure in which case we require an exact match. The presence of reference fields (having references to related instances in different data structures) enables the applications to exploit and harvest related

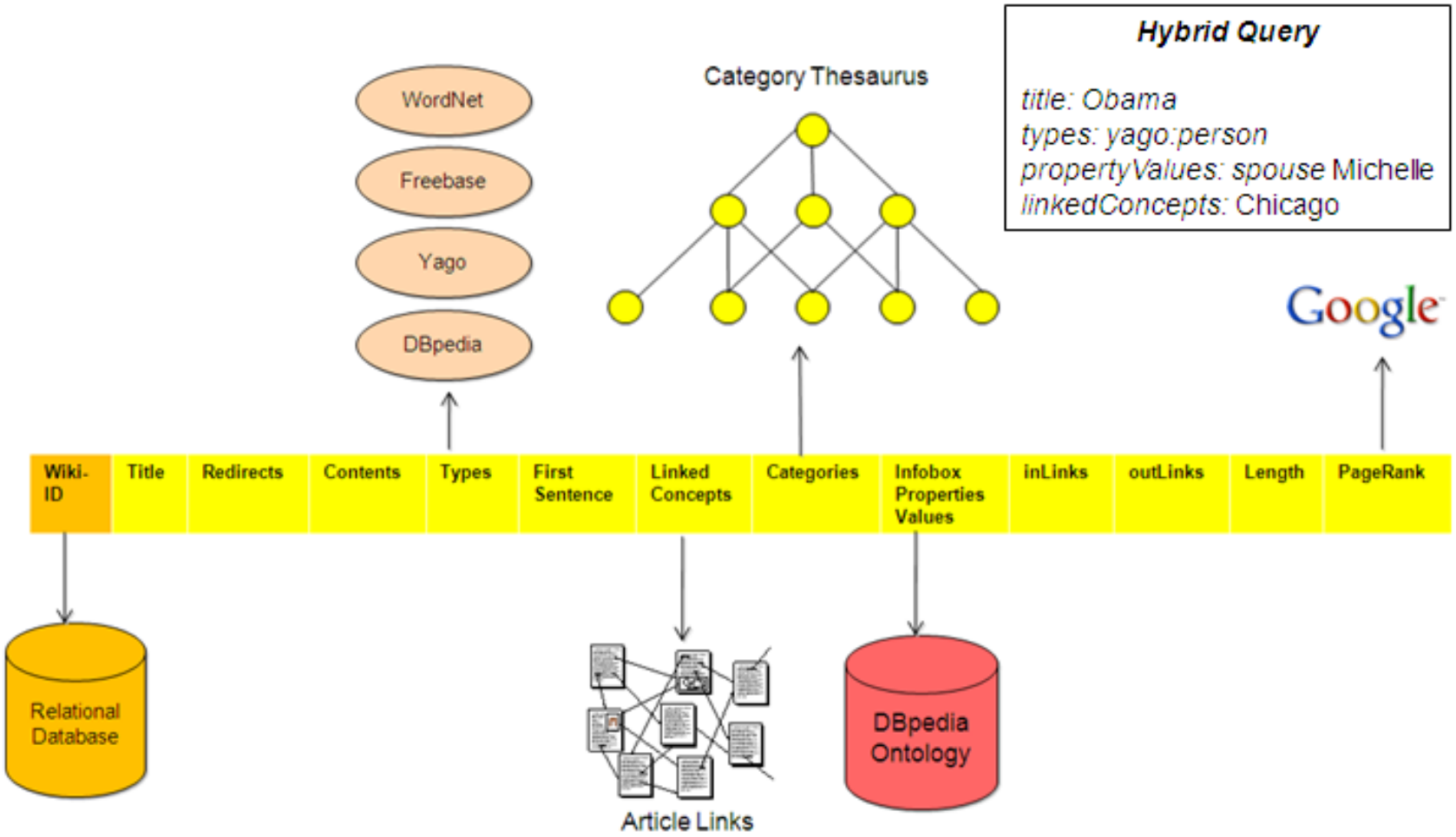


Figure 5.1: Wikitology uses a specialized information retrieval index comprising text, instance fields and reference fields with references to data available in other data structures.

information available in other forms by running data structure specific algorithms.

An overview of our specialized IR index is given in Figure 5.1. Below we describe the different kinds of fields in Wikitology Index and the way we store them in the Index.

Wiki-Id: The Wiki-Id field has the ID associated with each Wikipedia article or concept in Wikipedia. The Wiki-Id field serves as a reference to related data in Wikipedia MySQL tables as well as other tables generated having relevant information to the concept, for example, we generated a table with disambiguation entries for a concept which can be referenced using the Wiki-Id. The Wiki-Id field is added as stored and un-tokenized field in the index.

Title: We add the title of the Wikipedia concept as a tokenized field (wikiTitle) as well as un-tokenized stored field in the index (wikiTitleRef). The un-tokenized field helps in matching and retrieving an exact title and can be used to reference associated triples in the DBpedia and YAGO Ontologies.

Redirects: The redirects field contains the redirects to the concept in Wikipedia. A Wikipedia redirect page is a pseudo page with a title that is an alternate name or misspelling for the article. We add it as a tokenized field to the index.

First Sentence: We extract the first sentence of the Wikipedia article on the Concept as it often defines the concept. We add it as tokenized field (firstSentence) in the index. We store the sentence in the index as some applications might need to retrieve the full sentence and do some processing on it such as linguistic analysis.

Contents: We add the contents of the Wikipedia article on the Concept as a tokenized field (contents). We do not store the contents in the index as it will

significantly increase the size of the index. The contents field contains all the information present in the Wikipedia article and available in the Wikipedia markup of the article which includes article text, categories, infobox properties, both surface form of the links and the titles of the linked articles. We also add the redirects to the title concept in the contents field.

Types: We imported structured data in RDF from the YAGO ontology [91] and the DBPedia Infobox Ontology [17]. The structured data was encoded in an RDFa-like format in the types field for the Wikipedia page. This enables one to query the Wikitology knowledge base using both text (e.g., an entity document) and structured constraints (e.g., rdfs:type = YAGO:President). Freebase resource [23] contained a more comprehensive list of Named Entities (Persons, Locations and Organizations) as compared to YAGO and DBpedia ontology, we therefore generated a list of Wikipedia articles on Persons, Locations and Organizations by extracting all Wikipedia articles defined under the corresponding types in the Freebase resource [23]. We also added the DBpedia WordNet Mappings [5] that are manually created for about 400,000 Wikipedia articles. As Wikipedia has more than 2 million articles we used the Wikipedia Categories to WordNet mappings [81] to heuristically assign a WordNet type to any remaining Wikipedia articles (see VI.3.1.1). We add the type information both as a tokenized field (types) and un-tokenized stored field (typesRef) so that it can be used to reference related types in other resources. For the un-tokenized field we store the field by prefixing the type with the resource name from which we got that type such as wordnet:president or freebase:person.

Categories: The categories field contains the list of associated categories with

the Wikipedia article. We add it as a tokenized field (categories) and un-tokenized stored field (categoriesRef) so that it can be used to refer to Category node in the Category hierarchy or the same resource in the DBpedia Ontology.

Linked Concepts: This field lists the out-linked concepts. This field can be used to retrieve linked concepts and also to impose structural constraints while querying (e.g., linkedConcepts = Michelle_Obama, linkedConcepts = Chicago). We add this information as both tokenized field (linkedConcepts) and un-tokenized stored field (linkedConceptsRef) so that linked concepts are available both for exact and approximate matching in the query.

Infobox Triples: Info-boxes are the most structured form of information and are composed of a set of subject-attribute-value triples which summarize or highlight the key features of the concept or subject of the article. We incorporate the concept properties available via the DBpedia Infobox Ontology [17] (which has been manually developed using the infoboxes from Wikipedia) in our specialized index. This enables the applications to impose structural constraints related to the properties of the concepts when querying Wikitology. For example, wikiTitle:("Baltimore Ravens") AND propertiesValues:("coach John Harbaugh")

Different approaches could be followed to index triples in an IR index. Below we describe some common approaches and then discuss our approach.

One approach is to store each rdf statement as a lucene document with Subject, Predicate and Object in three fields. This option is not suitable for our design as we represent each Wikipedia Concept as a Document in the Lucene index secondly, it will also significantly increase the number of documents that Lucene will need to

index. Another approach is to store each resource i.e. Concept as a document with a separate field for each property. This approach will not permit one to make a query using "any" or "all" of the predicates. An alternate is to have an "all" field having all the object values in a single field similar to option adopted by LuceneSail [71] for storing literals in rdf statements. Yet another approach is to store all predicates and objects in a single field for example:

http://dbpedia.org/ontology/Person/spouse=http://dbpedia.org/resource/Barack_Obama

In our approach we add the properties and values in the infoboxes as a tokenized field (propertiesValues) and an un-tokenized stored field (propertiesValuesRef). The benefit of this approach is that querying becomes simpler. In case different properties are stored in different fields the user must be aware of the property names in order to use them in the query. In total there are 641 properties available in DBpedia infobox ontology (Table 5.1). The average number of properties for Persons, Locations and Organizations range from 7 to 8.

Keeping the properties values as un-tokenized field will permit exact matches to properties and their values (see Query 1). In many cases the applications might not have exact information regarding the properties and also the values, in that case they can query the tokenized field in the index with words that might appear in values (see Query 2). In case the applications have some information regarding the words appearing in the property but not exact information then the words in properties and values can be queried using a phrase query which would search for the words in vicinity and does not need to have an exact match (see Query 3).

```
Query 1:

propertiesValuesRef:
    http://dbpedia.org/ontology/Person/spouse=
    http://dbpedia.org/resource/Barack_Obama

Query 2:

propertiesValues:
    Barack

Query 3:

propertiesValues:
    "spouse Barack"
```

Figure 5.2: Demonstrating different queries to propertiesValues and propertiesValuesRef fields in Wikitology index.

Table 5.1: Property related statistics in DBpedia Infobox Ontology.

Description	Count
Total Number of Properties in DBpedia Ontology	641
Total Properties for all Person	249
Total Properties for all Locations (Place)	263
Total Properties for all Organizations	228
Avg. No. of Properties per Person	6.12
Avg. No. of Properties per Location	7.9
Avg. No. of Properties per Organization	6.3

The properties present in the infoboxes might have literal values or another Wikipedia concept as an object. We treat the title of the Wikipedia concept as a literal value so that it can be tokenized and searched along with other literals. We can also add related properties to Wikipedia concepts available via other linked Ontologies such as YAGO [91] or Linked Open Data [22] in a similar way.

PageRank Approx. Field: We trained a classifier based on a set of Wikipedia articles for which we had the Google page rank available using the in-links, out-links

and page length as features. We use the classifier to classify a Wikipedia article into a page rank class ranging from 0 to 9. We store the predicted page rank of a Wikipedia concept in the PageRank field. The predicted Page Rank can be used as a measure of popularity for a Wikipedia article and can serve as a useful feature for certain applications.

Statistics: Some applications might need to directly impose constraints on the in-degree, out-degree, page length of Wikipedia articles to retrieve, therefore we store these statistics in the index as well. We also disable length normalization for certain fields where normalizing a field based on the length of the field is not required such as for redirects, categories, propertiesValues as well as for all reference fields. Table 5.2 shows the different fields in Wikitology's specialized index and the field options used to index different fields.

5.3.6 Other Data Structures in Wikitology

Wikitology's specialized index contains references to instances in other data structures so that applications may be able to run data-structure specific algorithms when needed. We describe those data structures and resources below.

5.3.6.1 Triple Store

We have implemented a triple store using Jena RDF library [27] in java and MySQL database at the backend. We have uploaded the DBpedia Infobox Ontology and YAGO Ontology in the triple store. Applications can query the triples using

Table 5.2: Wikitology Index Structure and Field Types.

No.	Field	Tokenized	Stored	Normalized
1	wikiID	No	Yes	n/a
2	wikiTitle	Yes	No	No
3	wikiTitleRef	No	Yes	n/a
4	redirects	Yes	Yes	No
5	firstSentence	Yes	Yes	No
6	contents	Yes	No	Yes
7	types	Yes	No	No
8	typesRef	No	Yes	No
9	categories	Yes	No	No
10	categoriesRef	No	Yes	No
11	linkedConcepts	Yes	No	No
12	linkedConceptsRef	No	Yes	No
13	propertiesValues	Yes	No	No
14	propertiesValuesRef	No	Yes	No
15	inLinks	No	Yes	n/a
16	outLinks	No	Yes	n/a
17	length	No	Yes	n/a
18	pageRank	No	Yes	n/a

the SPARQL queries via the jena api and exploit the information in the triples.

5.3.6.2 InterArticle Links Graph

We downloaded the pagelinks table from Wikipedia download site and extracted the inter-article links graph, which is composed of more than 2 million nodes and more than 87 million edges. The graph is too big to be loaded into memory of a regular desktop machines (with 2 GB RAM). We therefore store the page links as a directed graph in a separate index which does not require loading the whole graph into memory.

5.3.6.3 CategoryLinks Graph

We have extracted a directed category links graph from mySQL tables available for Wikipedia. We filtered out administrative categories using a set of heuristics. The category graph is available as an edgelist file and can be accessed using Wikitology API or any other graph library.

5.3.6.4 Entity Links Graph

We have derived an Entity Links Graph using Freebase Types and Wikipedia page links. We extracted the links between articles that were on persons, locations and organizations that we labeled earlier using the Freebase resource. We generated person, location and organization graphs with links between entities of each type along with a general graph having interlinks between these individual graphs. These graphs may be useful for entity linking and named entity disambiguation tasks.

5.3.6.5 Database Tables

We have uploaded the mySQL dump for Wikipedia in a mySQL database. We also have some specialized tables constructed using data from the original tables for more efficient querying.

5.3.6.6 WordNet via JWNL

We have a local copy of WordNet to use with Wikitology that can be accessed using the JWNL library [33]. Since Wikipedia Concepts are associated with

WordNet types via YAGO ontology, DBpedia WordNet mappings and our heuristic mapping, the applications can use additional information available directly in WordNet along with Wikitology in their algorithms.

5.4 Wikitology API

We have also developed a java based api for accessing and querying Wikitology and retrieving information in the specialized index and related data structures. The object oriented interface revolves around mainly three object types. 1) Wikitology: To connect to the Wikitology KB and to search queries 2) Concept: The Wikipedia Concept (article) and information related to Concept 3) Category: The Wikipedia category and information related to the Category. The api contains functions for accessing information related to the Wikipedia Concepts, Categories, Category Links, Page Links, Entity Links graphs and also accepts Lucene Query objects. Lucene supports single term or phrase queries (a term consists of more than just one word), multiple field queries (searching in multiple fields at the same time), wild card searches (* and ?), fuzzy searches (finds terms that are similar to the given one), proximity searches (terms appear in a specified distance), range searches (all terms alphabetically between two given terms) and boosting terms (terms of a query may be weighted differently to distribute importance among them) in a query. These queries can even be combined to more complex queries by using boolean operators.

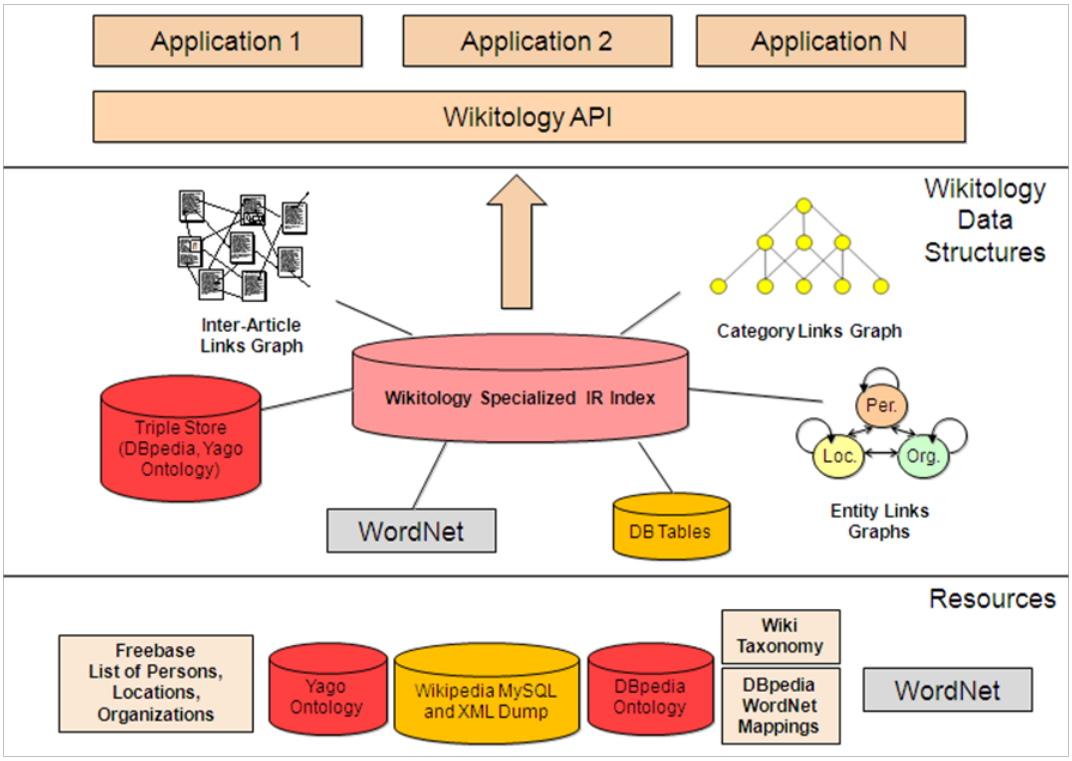


Figure 5.3: Overview of Wikitology API showing different resources that are incorporated in Wikitology and an API providing query interface and access to Wikitology.

5.5 Example Applications

Using a specialized IR index enables applications to query the knowledge base using either simple free text queries or complex queries over multiple fields with or without structured constraints and the presence of reference fields enables the applications to exploit and harvest related information available in other forms by running data structure specific algorithms. We have already presented different approaches that exploit Wikitology for a variety of tasks in chapter 4. In this section, we give a brief overview of how those approaches can be supported and implemented using the Wikitology API.

5.5.1 Document Concept Prediction

We have discussed in detail our Document Concept Prediction system in section 4.2, which is based on a blend of the statistical and ontological approach for predicting concepts represented in documents. Our algorithm first queries the specialized index and then exploits the page links and category links graphs using spreading activation algorithm for predicting document concepts. The Wikipedia articles represent specialized concepts whereas, the Wikipedia categories represent generalized concepts.

We give as input a test document or set of documents to our system, get top N similar Wikipedia articles and use them as seed nodes to run spreading activation algorithm on the article links graph and category links graph to predict relevant specialized concepts and generalized concepts respectively. The code in Algorithm 1 demonstrates how we exploit the Wikitology KB for predicting generalized concepts in a document using the Wikitology API.

5.5.2 Cross Document Named Entity Co-reference Resolution

Our approach for cross document named entity coreference resolution (Section IV.4) was targeted towards the ACE cross document co-reference task, and was used as a component of a system developed by the JHU Human Language Technology Center of Excellence [64].

In our approach, we used special "entity documents" or EDOCs extracted from the Serif's [24] APF output for the English documents as input to our system

Algorithm 1 Predict Generalized Concepts in Document

```
Input:        Document "doc"
              Initialization Parameters File "Param"
              Number of Top N Documents "TopN"
              Spreading Activation Pulses "k"
Output:       List of Predicted Generalized Concepts

// initialize Wikitology
1 Wikitology wikitology = new Wikitology(Param)

// initialize Category Graph
2  CategoryGraph cGraph = new CategoryGraph(Param)

// set the fields list to query for Top N Similar documents
3 String[] WikitologyQueryFields= {"contents"}

// retrieve a List of Top 'N' Similar Documents to "doc" by
// querying the fields in WikitologyQueryFields list
4  List DocumentList =
        wikitology.getTopNSimilarDocs(doc, TopN,
                                      WikitologyQueryFields)

// get the list of categories associated with the Documents List
5  List CategoriesList = wikitology.getDocCategories(DocumentList)

//  Run Spreading Activation algorithm and get a list of predicted
// concepts using the CategoriesList as seed nodes and k pulses
6   List GConcepts  =
         Wikitology.SpreadingActivation(cGraph,CategoriesList, k)

7   return GConcepts
```

based on the Wikitology knowledge base. Each entity in a given document produced one EDOC that included the longest entity mention, all name mentions, all nominal mentions, all pronominal mentions, APF type and subtype and all words within 15 tokens of each mention. The EDOCs were used to find candidate matches in Wikitology.

The EDOCs were processed by a custom query module for Wikitology that mapped the information in the EDOC into different components of Wikitology index. The EDOC's name mention strings are compared to the text in Wikitology's title field, giving a slightly higher weight to the longest mention. The EDOC type information is mapped into the Wikitology type information terms imported from DBpedia which are expressed using the YAGO ontology and matched against the RDF field of each Wikitology entry. Finally the name mention strings along with contextual text surrounding the mentions are matched against the text of the Wikitology entries.

The Wikitology module returns two vectors: one for matches against article entries and the other against category articles (Algorithm 2), which are used to produce features to measure the similarity and dissimilarity of a pair of entities.

5.5.3 Entity Linking

Our approach for Entity Linking (Section IV.5) was targeted towards the TAC KBP task [39] in 2009. One part of the KBP track is to link entities found in text to those in an external knowledge base. In particular, the entity linking task is defined

Algorithm 2 Get Wikitology Concept and Category Feature Vectors for Cross-Document Co-reference Resolution

```
Input:        Entity Document "EDOC"
              Initialization Parameters File "Param"
              Number of Top N Entries to use as features  "TopN"
Output:       Concept Feature Vector, Category Feature Vector

// initialize Wikitology
1  Wikitology wikitology = new Wikitology(Param)

// formulate Query by mapping different parts of the EDOC to
// different fields in Wikitology giving a higher boost (4.0)
// to the Longest Name Mention in the EDOC
2  Query query =
                  title:    (EDOC.NameMentions)
                  title:    (EDOC.LongestNameMention)^4.0
                  type:     (EDOC.Type)
                  contents: (EDOC.NameMentions)
                  contents: (EDOC.Context)

// retrieve top N Concepts and their relevance scores to represent
// feature weights using the EDOC Query
3 Map<Concept,double> ConceptVector =
                              wikitology.searchQuery(query, TopN)

// generate a Category vector based on Categories linked to the
// Concepts in the Concept Vector and assign feature weights as a
// sum of the relevance scores for all linked Concepts in the
// Concept Vector
4 Map<Category,double>  CategoryVector =
                              wikitology.getCategoryVector(
                                                  ConceptVector)

5 Output:  ConceptVector and CategoryVector
```

as: given an entity mention string and an article with that entity mention, find the link to the right Wikipedia entity if one exists.

We tested different approaches to querying Wikitology KB using the entity mention string and input article, the approach that gave the best results for all the entity types was approach 4. In which case we query using the entity mentions against the title and redirects fields and then search within the returned results by querying the given entity document against the Wikipedia article contents field and against the Types field and the entity mentions against the title, redirects and contents field with a boost of 4 (Algorithm 3).

5.5.4 Interpreting Tables

In section 4.5, we have discussed our approach to interpreting information in tables. We developed a custom query module that maps different parts of the table to different components in the Wikitology specialized index.

Our custom query module maps the instance mention to the "title" field. The "title" field contains the Wikipedia article or concept titles. The instance mention is also mapped to the "redirects" field which contains the Wikipedia redirects to the article or concept. The table header is mapped to the "types" field. The "types" field contains the DBpedia classes, YAGO classes, WordNet classes and Freebase classes (i.e. Person, Location and Organization) associated with the concept. The entity mention and the column header is mapped to the "firstSentence" field. It is often the case that the first sentence in Wikipedia usually defines the concept and

Algorithm 3 Link Entity to the correct Wikipedia Entity

```
Input:        Entity Mention "EMention"
              Initialization Parameters File "Param"
              Document "Doc"
Output:       Matching Wikipedia Entity

// initialize Wikitology
1  Wikitology wikitology = new Wikitology(Param)

// Formulate Queries to Wikitology
2  Query_1=
            title:     (EntityMention)
            redirects: (EntityMention)
3  Query_2=
            title:     (EntityMention)
            redirects: (EntityMention)
            contents:  (EntityMention)^4
            contents:  (Doc)
            types:     (Doc)

// query using Query_2 on the results returned by Query_1
4 Map<Concept,double> ConceptVector =
                         wikitology.searchQuery( Query_2,  Query_1)

// return the top most match to the Wikipedia Concept
5 Return getTopConcept(ConceptVector)
```

mentions the type of the concept as well.

Algorithm 4 Link Instance in Table

```
Description:  Different parts of the table are mapped to different
              components of the Wikitology Index to get Top N
              matching Wikitology Concepts to the Instance in the
              table.
Input:        Instance Mention "Mention"
              Instance Row Data "RowData"
              Instance Column Header "ColHeader"
              Number of Top N Entries to return "TopN"
              Initialization Parameters File "Param"
Output:       Top N Ranking Concepts Vector

// initialize Wikitology
1  Wikitology wikitology = new Wikitology(Param)

// Formulate Query to Wikitology
2 Query =
            title:           (Mention)
            redirects:       (Mention)
            firstSentence:   (Mention)
            firstSentence:   (ColHeader)
            linkedConcepts:  (RowData)
            linkedConcepts:  (Mention)^4.0
            types:           (ColHeader)
            propertiesValues:(RowData)
            contents:        (Mention)^4.0
            contents:        (RowData)

// Get Top N Concept Vector matching the Query
2 TopNConceptVector = Wikitology.searchQuery(Query,TopN)

3 Return TopNConceptVector
```

The values in the same row are mapped to the "contents" field and the "linkedConcepts" field in Wikitology, giving a boost of 4.0 to the instance mention itself. The "contents" field contains article text including the title, redirects, infobox properties and values as well as the linked categories. The "linkedConcepts" field enlists

all the linked concepts in Wikipedia. We also map the row values to the "propertiesValues" field which contains the DBpedia infobox properties and value pairs for a concept. Algorithm 4 demonstrates our query module for querying Wikitology.

5.6 Conclusions

In this chapter, we have presented the design and architecture of the Hybrid Wikitology knowledge base and demonstrated how different kinds of applications can query the KB and harvest the Knowledge available in different forms. The existing Knowledge bases or api's only permit structured queries to Wikipedia, however, our KB is unique in providing access to different forms of data via a single hybrid query interface supporting the rich query language available in the Lucene IR Library and returning ranked results based on relevance score rather than boolean matches.

Chapter 6

Approaches for Automatically Enriching Wikitology

In Chapter 4, we discussed different approaches that exploit Wikitology for some common use cases such as Document Concept Prediction [93], Cross Document Coreference Resolution [38] Entity Linking to KB entities [39] and Interpreting Tables [94]. These use cases not only serve to solve different real world problems but also contribute to enriching Wikipedia and hence the Wikitology KB. In general, automatically adding an article on a new concept to Wikipedia would require predicting the categories, inter-article links and infobox properties and values for the new article. There might also be a need to associate one or more redirect pages with the new article. Furthermore, if the concept title is ambiguous with other existing concepts it would also require updating the relevant disambiguation page.

The document concept prediction approach can predict inter-article links and categories for new Wikipedia articles. Cross document co-reference resolution and entity linking are more specialized approaches for specifically linking entity mentions in Wikipedia articles or external articles to the entity articles in Wikipedia.

In addition to that we have also developed specific approaches aimed at automatically enriching the Wikitology KB by unsupervised discovery of ontology elements which can complement the existing infoboxes and also suggest new slots and fillers; generating disambiguation trees for entities and estimating the page rank

of Wikipedia articles to serve as a measure of popularity. The broader goal of developing approaches for enriching Wikitology is to integrate the set of approaches into a unified framework that would support automatic enrichment of Wikipedia and the Wikitology KB with new concepts.

In this Chapter we first discuss the different approaches specifically targeted towards enriching Wikitology KB and then propose a broader unified framework for adding new concepts to Wikipedia and hence Wikitology KB and discuss how the different Wikitology based approaches can contribute to a number of steps in the broader framework.

6.1 Unsupervised techniques for discovering ontology elements from Wikipedia article links

One of the biggest challenges faced by the Semantic Web vision is the availability of structured data that can be published as RDF. One approach is to develop techniques to translate information in spreadsheets, databases, XML documents and other traditional data formats into RDF. Another is to refine the technology needed to extract structured information from unstructured free text. We describe our work on a system that can discover ontological elements as well as data from a free text with embedded hyperlinks.

Infoboxes in Wikipedia are a readily available source of structured data, however, the free text of the article contains much more information about the entity. Barker et al. [18] unified the state of the art approaches in Natural Language Pro-

cessing and Knowledge Representation in their prototype system for understanding free text. Text resources which are rich in hyperlinks especially to knowledge based resources (such as encyclopedias or dictionaries) have additional information encoded in the form of links, which can be used to complement the existing systems for text understanding and knowledge discovery. Furthermore, systems such as Wikify [67] can be employed to link words in free text to knowledge resources like Wikipedia and thus enrich the free text with hyperlinks.

We describe an approach for ontology discovery based on the idea of unrestricted discovery [89] of relations from links in the free text of the Wikipedia articles, without specifying a relation or set of relations in advance. Through our approach we can discover new slots and fillers that may not be available in the Wikipedia infoboxes. Our results demonstrate that there are certain types of properties which are evident in the link structure of resources like Wikipedia that can be predicted with high accuracy using little or no linguistic analysis. The discovered properties can be further used to discover a class hierarchy. Our experiments have focused on analyzing people in Wikipedia, but the techniques can be directly applied to other types of entities such as organizations, places and products in text resources that are rich with hyperlinks.

The techniques we describe are not suggested as an alternative to natural language processing or information extraction, but as an additional source of evidence that can be used to extract ontological elements and relations from the kind of text found in Wikipedia. This approach might be particularly useful in "slot fillings" tasks like the one in the Knowledge Base Population track at the 2009 Text

Analysis Conference [65].

6.1.1 Contributions

We see several contributions that this work has to offer:

- Unsupervised and unrestricted ontology discovery. We present an automatic approach that does not require a predefined list of relations or training data. The analysis uses inter-article links in the text and does not depend on existing infoboxes, enabling it to suggest slots and fillers that do not exist in any extant infoboxes.

- Meaningful slot labels. We use WordNet [68] nodes to represent and label slots enabling us to exploit WordNet's hypernym and hyponym relations as a property hierarchy.

- Entity classification and class labeling. We introduce a new feature set for entity classification, i.e. the discovered ranked slots, which performs better than other feature sets extracted from Wikipedia. We also present an approach for assigning meaningful class label vectors using WordNet nodes.

- Deriving a class hierarchy. We have developed an approach for deriving a class hierarchy based on the ranked slot similarity between classes and the label vectors.

Below we describe the details of the approach, mention closely related work, present and discuss preliminary results and provide some conclusions and possible

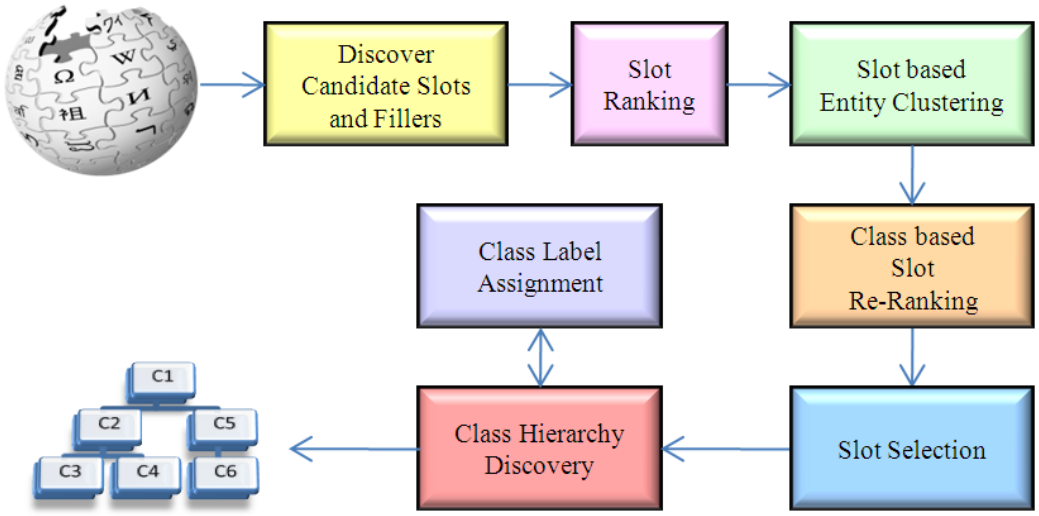


Figure 6.1: The ontology discovery framework comprises a number of steps, including candidate slot and filler discovery followed by slot ranking, slot selection, entity classification, slot re-ranking, class labeling, and class hierarchy discovery.

next steps.

6.1.2 Approach

Figure 6.1 shows our ontology discovery framework and its major steps. These include discovering candidate slots and fillers, slot ranking, slot selection, entity classification, slot re-ranking, class labeling, and class hierarchy discovery. We describe each of this in the rest of this section.

6.1.2.1 Discovering Candidate Slots and Fillers

Most Wikipedia articles represent a concept, i.e., a generic class of objects (e.g., Musician), an individual object (e.g., Michael_Jackson), or a generic relation or property (e.g., age). Inter-article links within Wikipedia represent relations between concepts. In our approach we consider the linked concepts as candidate fillers for

Figure 6.2: The Wikipedia infobox for the Michael_Jackson article has a number of slots from appropriate infobox templates.

slots related to the primary article/concept. There are several cases where the filler is subsumed by the slot label for example, the infobox present in the article on "Michael_Jackson" (Figure 6.2) mentions pop, rock and soul as fillers for the slot Genre and all three of these are a type of Genre. The Labels slot contains fillers such as Motown, Epic and Legacy which are all Record Label Companies. Based on this observation, we discover and exploit "ISA" relation between fillers (linked concepts) and WordNet nodes to serve as candidate slot labels.

In order to find "ISA" relation between a concept and a WordNet synset we use manually created mappings by DBpedia, which links about 467,000 articles to

synsets. However, Wikipedia has more than two million articles, therefore, to map any remaining concepts we use the automatically generated mappings available between WordNet synsets and Wikipedia categories [81]. A single Wikipedia article might have multiple categories associated with it and therefore multiple WordNet synsets. The Wikipedia category system serves more as a way to tag articles and facilitate navigation rather than to categorize them. The article on Michael Jordan, for example, has 36 categories associated with it. In order to select an individual WordNet synset to represent the type of the concept we use the following two heuristics:

- Category label extraction. Since the first sentence in Wikipedia articles usually defines the concept, we extract a category label from the first sentence using patterns based on POS tags similar to Kazama and Torisawa [55].

- Assign matching WordNet synset. We assign the WordNet synset if any of the words in the synset matches with the extracted category label. We repeat the process with hypernyms and hyponyms of the synset up to three levels.

6.1.2.2 Slot Ranking

All slots discovered using outgoing links might not be meaningful, therefore we have developed techniques for ranking and selecting slots. Our approach is based on the observation that entities of the same type have common slots. For example, there is a set of slots common for musical artists whereas, a different set is common for basketball players. The Wikipedia infobox templates based on classes also provide

a set of properties or slots to use for particular types of entities or concepts.

In case of people, it is a common observation that there are a set of slots that are generalized, i.e., they are common across all types of persons. Examples are name, born, spouse, etc. There are also sets of specialized slots which are generally related to a given profession. For example, the slots for basketball players have information related to basketball related activities and Musical Artists have slots with music related activities. The slots for "Michael_Jordan" include Professional Team(s), NBA Draft, Position(s) and slots for "Michael_Jackson" include Genres, Instruments and Labels.

Another observation is that people engaged in a particular profession tend to be linked to others within the same profession. Hence the maxim "A man is known by the company he keeps." For example, basketball players are linked to other basketball players and politicians are linked to other politicians. We rank the slots based on the number of linked entities of the same type (in our case persons) having the same slots. We generated a list of person articles in Wikipedia by getting all Wikipedia articles under the Person type in Freebase. We randomly select up to 25 linked persons (which also link back) and extract their candidate slots and vote for a slot based on the number of times it appears as a slot in a linked person normalized by the number of linked persons to assign a slot score.

6.1.2.3 Entity Classification and Slot Re-Ranking

We use the ranked candidate slots to classify entities and then further rank the slots based on number of times they appear among the entities in the cluster. We use complete link clustering using the slot similarity function defined below.

$$sim_{slot}(p_i, p_j) = cos(slot(p_i), slot(p_j)) \tag{6.1}$$

Slot Similarity Function: Cosine similarity between tf.idf weighted slot vectors, where the slot score represents the term frequency component and the inverse document frequency is based on the number of times the slot appears in different individuals.

We also collapsed location expressing slots (country, county, state, district, island etc.) into the slot labeled location by generating a list of location words from WordNet as these slots were causing the persons related to same type of geographical location to cluster together.

After clustering we re-score the slots based on number of times they appear among the individuals in the cluster normalized by the cluster size. The output of clustering is a vector of scored slots associated with each cluster.

6.1.2.4 Slot Selection

To filter out any irrelevant slots we have an approach for slot selection. Our intuition is that specialized slots or attributes for a particular entity type should be somehow related to each other. For example, we would expect attributes like

league, season and team for basketball players and genre, label, song and album for musical artists. If an attribute like album appears for basketball players it should be discarded as it is not related to other attributes. We adopted a clustering approach for finding attributes that are related to each other. For each pair of attributes in the slot vector we computed a similarity score based on how many times the two attribute labels appear together in Wikipedia person articles within a distance of 100 words as compared to the number of times they appear in total and weigh it using weights of the individual attributes in the slot vector. This metric is captured in the following equation where Df is the document frequency and wt is the attribute weight.

$$sim_{attr}(a_i, a_j) = wt(a_i) \times wt(a_j) \times \frac{Df(a_i, a_j, 100)}{Df(a_i) + Df(a_j)} \tag{6.2}$$

We did some initial experiments using single link and complete link and found single link to be more appropriate for slot selection. We got clusters at a partition distance of 0.9 and selected the largest cluster from the set of clusters. In addition to that we also added any attributes exceeding a 0.4 score into the set of selected attributes. Selected ranked slots for Michael Jackson are given in Table 6.1.

6.1.2.5 Class Labeling

Assigning class labels to clusters gives additional information about the type of entities in a cluster. We generate a cluster label vector for each cluster which represents the type of entities in the cluster. We compute a list of person types by

Table 6.1: Fifteen slots were discovered for musician Michael Jackson along with scores and example fillers.

Slot	Score	Fillers Example
Musician	1.00	ray_charles, sam_cooke ...
Album	0.99	bad_(album), ...
Location	0.97	gary,_indiana, chicago,
Music_genre	0.90	pop_music, soul_music, ...
Label	0.79	a&m_records, epic_records, ...
Phonograph_ record	0.67	give_in_to_me, this_place_hotel ...
Act	0.59	singing
Movie	0.46	moonwalker
Company	0.43	war_child_(charity),
Actor	0.41	stan_winston, eddie_murphy ...
Singer	0.40	britney_spears,
Magazine	0.29	entertainment_weekly,
Writing_style	0.27	hip_hop_music
Group	0.21	'n_sync, RIAA
Song	0.20	d.s._(song)

taking all hyponyms under the corresponding person sense in WordNet. That list mostly contained the professions list for persons such as basketball player, president, bishop etc. To assign a WordNet type to a person in Wikipedia we matched the entries in the list to the words in the first sentence of the person article and assigned it the set of types that matched. For example, for Michael Jordan the matching types found were basketball_player, businessman and player.

We assigned the most frequent sense to the matching word as followed by Suchanek et al. [91] and Wu and Weld [106], which works for majority of the cases. We then also add all the hypernyms of the matching types under the Person node. The vector for Michael Jordan has entries basketball_player, athlete, businessperson, person, contestant, businessman and player. After getting matching types and their hypernyms for all the members of the cluster, we score each type based on

the number of times it occurs in its members normalized by the cluster size. For example for one of the clusters with 146 basketball players we got the following label vector: player:0.97, contestant:0.97, athlete:0.96, basketball_player:0.96. To select an individual label for a class we can pick the label with the highest score (the most generalized label) or the most specialized label having a score above a given threshold.

6.1.2.6 Discovering Class Hierarchy

We employ two different feature sets to discover the class hierarchy i.e. the selected slot vectors and the class label vectors. We also combine both functions using their weighted sum. The similarity functions are described below.

The common slot similarity function is the cosine similarity between the common slot tf.idf vectors where the slot score represents the tf and the idf is based on the number of times a particular slot appears in different clusters at that iteration. We re-compute the idf term in each iteration. We define the common slot tf.idf vector for a cluster in which only those slots have non-zero weight that have non-zero weight for all members of the cluster. The label similarity function is the cosine similarity between the label vectors for clusters. The hybrid similarity function is a weighted sum of the common slot and label similarity functions.

$$sim_{comm\_slot}(c_i, c_j) = cos(comm\_slot(c_i), comm\_slot(c_j)) \tag{6.3}$$

$$sim_{label}(c_i, c_j) = cos(label(c_i), label(c_j)) \tag{6.4}$$

$$sim_{hyb}(c_i, c_j) = w_c \times sim_{comm\_slot}(c_i, c_j) + w_l \times sim_{label}(c_i, c_j) \tag{6.5}$$

$$\text{where, } w_c + w_l = 1$$

6.1.3 Experiments and Evaluation

For our experiments and evaluation we used the Wikipedia dump from March 2008 and the DBpedia infobox ontology created from Wikipedia infoboxes using hand-generated mappings [17]. The Person class is a direct subclass of the owl:Thing class and has 21 immediate sub-classes and 36 subclasses at level two. We used the persons in different classes in DBpedia ontology at level two to generate data sets for experiments.

There are several articles in Wikipedia that are very small and have very few out-links and in-links. Our approach is based on the out-links and availability of information about different related things on the article, therefore, in order to avoid data sparseness, we randomly select articles with greater than 100 in-links and out-links, at least 5KB page length and having at least five links to entities of the same type that link back (in our case persons).

We first compare our slot vector features with other features extracted from Wikipedia for entity classification task and then evaluate their accuracy. We then discover the class hierarchy and compare the different similarity functions.

6.1.3.1 Entity Classification

We did some initial experiments to compare our ranked slot features with other feature sets extracted from Wikipedia. We created a dataset composed of 25 different classes of Persons present at level 2 in the DBpedia ontology by randomly selecting 200 person articles from each class. For several classes we got less than 200 articles which fulfilled our selection criteria defined earlier.

We generated twelve types of feature sets and evaluated them using ground truth from DBpedia ontology. We compare tf.idf vectors constructed using different feature sets. (1) Ranked slot features, where tf is the slot score. (2) Words in first sentence of an article. (3) Associated categories. (4) Assigned WordNet nodes (see section 2.2). (5) Associated categories tokenized into words. (6) Combined Feature Sets 1 to 5 (All). Feature sets 7 to 11 are combinations excluding one feature set at a time. (12) Unranked slots where tf is 1 for all slots. We applied complete link clustering and evaluated the precision, recall and F-measure at different numbers of clusters ranging from 1 to 100. Table 6.2 gives the precision, recall and number of clusters where we got the maximum F-measure using different feature sets.

Feature set 10 (All - 2) gave the best F-measure i.e. 0.74, whereas, feature set 1 (Ranked Slots only) gave the second best F-measure i.e. 0.73 which is very close to the best result. Feature set 12 (unranked slots) gave a lower F-measure i.e. 0.61 which shows that ranking or weighing slots based on linked entities of the same type performs better for classification.

Table 6.2: Comparison of the precision, recall and F-measure for different feature sets for entity classification. The k column shows the number of clusters that maximized the F score.

No.	Feature Set	k	P	R	F
1	Ranked Slots	40	0.74	0.72	0.73
2	First Sentence	89	0.07	0.53	0.12
3	Categories	1	0.05	1.00	0.10
4	WordNet Nodes	87	0.40	0.22	0.29
5	(3 tokenized)	93	0.85	0.47	0.60
6	All (1 to 5)	68	0.87	0.62	0.72
7	(All - 5)	82	0.79	0.46	0.58
8	(All - 4)	58	0.78	0.63	0.70
9	(All - 3)	53	0.76	0.65	0.70
10	(All - 2)	58	0.88	0.63	0.74
11	(All - 1)	57	0.77	0.60	0.68
12	(1 unranked)	34	0.57	0.65	0.61

6.1.3.2 Slot and Filler Evaluation

We selected all the sub-classes of Person class at level 2. We randomly selected 200 person articles using the same criteria defined earlier to avoid data sparseness. For several classes we got fewer than 200 articles. We discarded classes for which we got fewer than 20 articles. Our final data set comprised 28 classes and 3810 articles.

We used our ranked slots tf.idf feature set and ran complete link clustering algorithm producing clusters at partition distance of 0.8. We re-scored the slots based on the number of times they appeared in the cluster members normalized by the cluster size. We applied slot selection over the re-scored slots for each cluster. In order to evaluate our slots and fillers we mapped each cluster to a DBpedia class based on the maximum number of members of a particular DBpedia class in our cluster.

In total our approach predicted 124 unique properties corresponding to differ-

Table 6.3: Manual evaluation of discovered properties.

Property	Accuracy
automobile_race,championship,expressive_style,fictional_character label,racetrack,team_sport,writing_style	1.00
academic_degree,album,book,contest,election,league, phonograph_record,race	0.95
tournament	0.94
award,movie,novel,school,season,serial,song	0.90
car,church,game,musical_instrument,show,sport,stadium	0.85
broadcast,telecast	0.80
hockey_league	0.75
music_genre,trophy	0.70
university	0.65
character,disease	0.60
magazine	0.55
team	0.50
baseball_club,club,party	0.45
captain	0.30
coach	0.25
Avg. Accuracy:	0.81

ent classes and out of 124 properties we were able to find 46 properties that existed in either DBpedia ontology or Freebase for the corresponding class. We initially tried to evaluate the discovered slots by comparing them with DBpedia ontology and Freebase however, we were able to find an overlap in the subject and object pairs for very few properties. Therefore we decided to evaluate the pairs manually. We randomly selected 20 subject object pairs for each of the 46 properties from the corresponding classes and manually evaluated if the relation was correct by consulting the corresponding Wikipedia articles. The accuracy obtained for each property is given in the Table 6.3.

The highest accuracy of 100% was obtained for the following eight properties: auto-mobile_race, championship, expressive_style, label, racetrack, team_sport, fic-

Table 6.4: Evaluation results for class hierarchy prediction using different similarity functions.

Similarity Function	k (L=2)	F (L=2)	k (L=1)	F (L=1)
sim_{slot}	56	0.61	13	0.55
sim_{com_slot}	74	0.61	15	0.65
sim_{label}	50	0.63	10	0.76
sim_{hyb} $w_c = w_l = 0.5$	59	0.63	10	0.76
sim_{hyb} $w_c = 0.2, w_l = 0.8$	61	0.63	8	0.79

tional_character, writing_style. In total 33 properties had accuracy of greater than or equal to 80%. For few properties the accuracy was 60% or below which are coach, captain, baseball_club, club, party, team and magazine. The average accuracy for the 46 relations was 81%.

6.1.3.3 Discovering Class Hierarchy

In order to discover the class hierarchy we took all the clusters obtained earlier at partition distance of 0.8 and their corresponding slot vectors after slot selection. We experimented with different similarity functions and evaluated their accuracy by comparing the results with the DBpedia ontology. We ran complete link clustering algorithm using different settings of the similarity functions. We evaluated the different similarity function settings by comparing the output hierarchy with level 2 and level 1 class hierarchy present below the Person node (considered at level 0). The highest F measure obtained for L2 and L1 and the 'k' (number of clusters) for which we got the highest F-measure using a particular similarity function are given in the Table 6.4.

The highest F-measure both at level 2 (0.63) and level 1 (0.79) was obtained

by sim_{hyb} with $w_c = 0.2$, $w_l = 0.8$ and also at lowest number of clusters at L1 (k=8). The sim_{hyb} ($w_c = w_l = 0.5$) and sim_{label} functions gave almost the same F-measure at both levels. The sim_{com_slot} function gave better performance at L1 (F=0.65) than the base line sim_{slot} (F=0.55) which was originally used for entity clustering. However, both these functions gave the same F-measure at L2 (F=0.61).

6.1.4 Discussion

In case of property evaluation, properties for which the accuracy was 60% or below include coach, captain, baseball_club, club, party, team and magazine. For the magazine property (corresponding to Writer and ComicsCreator class) we observed that many times a magazine name was mentioned in a Wikipedia article because it published some news about a person rather than that person contributing any article in that magazine. For all the remaining properties we observed that these were related to some sort of competition. For example, a person played against a team, club, coach or captain.

The political party relation is a similar case, where articles frequently mention a politician's party affiliation as well as significant opposition parties. For such properties, we need to exploit additional contextual information to judge whether the person competed "for" or "against" a particular team, club, coach or party. Even if the accuracy for fillers for such slots is low, it can still be useful to discover the kind of slots associated with an entity.

We also observed that there were some cases where the property was related

to a family member of the primary person such as for properties like disease, school and university. Certain other properties such as spouse, predecessor, successor, etc. require more contextual information and are not directly evident in the link structure. However, our experiments show that there are certain properties that can be predicted with high accuracy using the article links only and can be used to enrich the existing infobox ontology or for other purposes.

While our work has mostly experimented with person entities, the approach can be applied to other types as well. For example, we were able to discover software as a candidate slot for companies like Microsoft, Google and Yahoo!, which appeared among the top three ranked slots using our slot ranking scheme and corresponds to the products slot in the infoboxes of these companies.

For class hierarchy discovery, we have exploited the specialized slots after slot selection. One way to incorporate generalized slots in the hierarchy is to consider all slots for class members (without slot selection) and recursively propagate the common slots present at any level to the level above it. For example, if we find the slot team to be common for different types of Athletes such as basketball players, soccer players etc. we can propagate it to the Athlete class, which is one level higher in the hierarchy.

6.1.5 Related Work

Unsupervised relation discovery was initially introduced by Hasegawa et al. [51]. They developed an approach for unsupervised relation discovery by clustering

pairs of entities based on intervening words represented as context vectors. Shinyama and Sekine [89] generated basic patterns using parts of text syntactically connected to the entity and then generated a basic cluster composed of a set of events having the same relation.

Several approaches have used linguistic analysis to generate features for supervised or un-supervised relation extraction such as [76], [37] and [107]. In our approach we mainly exploit the link structure of Wikipedia and demonstrate that there are several relations that can be discovered with high accuracy without the need of features generated from a linguistic analysis of the Wikipedia article text.

Suchanek et al. [91] used Wikipedia categories and infoboxes to extract 92 relations by applying specialized heuristics for each relation and incorporated the relations in their YAGO ontology, whereas our approach does not use specialized heuristics based on the type of relation. Kylin [**?**] generated infoboxes for articles by learning from existing infoboxes. Our approach can discover new fillers for several existing slots and also discover new slots for infoboxes. KOG [106] automatically refined the Wikipedia infobox ontology and integrated Wikipedia's infobox-class schemata with WordNet. Since we already use the WordNet nodes for representing slots, it eliminates the need for several refinement steps taken by KOG to refine the infoboxes.

All the three systems YAGO, Kylin and KOG, rely on relations present in the infoboxes. Our approach can complement these approaches by discovering new relations evident in inter-article links in Wikipedia. For example, we could add slots like songs and albums to the infobox schema for Musical Artists, movies for the

Actors infobox schema, and party for the Politicians schema.

6.1.6 Conclusions

People have been learning by reading for thousands of years. The past decade, however, has seen a significant change in the way people read. The developed world now does much of its reading online and this change will soon be nearly universal. Most online content is read as hypertext via a Web browser or custom reading device. Unlike text, hypertext is semi-structured information, especially when links are drawn from global namespace, making it easy for many documents to link unambiguously to a common referent.

The structured component of hypertext augments the information in its plain text and provides an additional source of information from which both people and machines can learn. Our work is aimed at learning useful information, both about the implicit ontology and facts, from the links embedded in collection of hypertext documents.

Our approach for discovering structured data similar to infoboxes within Wikipedia is fully unsupervised and does not require having a pre-defined catalogue of relations. We have discovered several new slots and fillers which are not already present in existing Wikipedia infoboxes and also a scheme to rank the slots based on linked entities of the same type. We have compared our results with ground truth from the DBpedia infobox ontology and Freebase for the set of properties that were common and manually evaluated the accuracy of the common properties. Our results show

that there are several properties that can be discovered with high accuracy from the link structure in Wikipedia and can also be used to discover a class hierarchy. The same approach may be employed for discovery of slots from non-Wikipedia articles by linking them to Wikipedia concepts using existing systems like Wikify [67].

6.2 Disambiguation Trees

We have developed an approach to disambiguate mentions that refer to a Wikipedia entity. Wikipedia has special, manually created disambiguation pages for sets of entities with identical or similar names. For example, the disambiguation page "Michael_Jackson_(Disambiguation)" lists 36 different entities to which the string "Michael Jackson" might refer. A short description is given for each, such as

Michael Jackson (actor) (born 1970), Canadian actor, best known for his role as Trevor on Trailer Park Boys

that identifies one or more facts that can help distinguish it from others in the set. Not all confusable entities had such disambiguation pages and an automated process for creating them could both contribute to Wikipedia and also support entity linking.

6.2.1 Problem Statement

Given a set of ambiguous Wikipedia entities, extract disambiguating features to automatically generate machine understandable representations of Disambiguation pages.

6.2.2 Approach

We initially developed a prototype for creating disambiguation pages for people. We modeled this problem as a multiple class classification problem where each person with a confusable name is considered an individual class. We extract nouns and adjectives from the first sentence of the person articles and use them to construct a decision tree. Most commonly, the nodes that were selected to split on referred to either the person's nationality or profession.

We enhanced our approach by using a domain model [44] of nationalities and professions constructed from Wikipedia's list pages "list_of_nationalities" and "list_of_professions". These were used to extract the nationality and profession of persons by selecting nouns and adjectives as features that appeared in the first sentence and were in one of the domain models. Using the nationality and profession as features, we constructed a decision tree using Weka [48] for different Persons having a confusable name. When we were not able to extract a profession or nationality of the entity from the first sentence, we gave that feature a value of "0". We refer to these decision trees that help in disambiguating entities as disambiguation trees. We constructed several disambiguation trees for different sets of persons having the same name. Disambiguation trees constructed as a result of three of our experiments on persons having name "Michael Jackson", "Michael Jordan" and "George Harrison" are shown in Figure 6.3, Figure 6.4 and Figure 6.5.

Out of 21 different people named "Michael Jackson" we were able to disambiguate 15 of them using just the profession and nationality features. For three of

```
Profession = musician: Michael_Jackson
Profession = 0
|  Nationality = english: Michael_Jackson_(footballer)
|  Nationality = 0: Michael_Jackson_(Anglican_bishop)
Profession = guitarist: Michael_Gregory_(jazz_guitarist)
Profession = sheriff: Michael_A._Jackson_(sheriff)
Profession = journalist: Michael_Jackson_(journalist)
Profession = player: Michael_Jackson_(basketball)
Profession = executive: Michael_Jackson_(television_executive)
Profession = writer: Michael_Jackson_(writer)
Profession = professor: Michael_Jackson_(anthropologist)
Profession = footballer: Michael_Jackson_(rugby_league)
Profession = scientist: Michael_A._Jackson
Profession = soldier: Michael_Jackson_(American_Revolution)
Profession = actor
|  Nationality = english: Michael_J._Jackson_(actor)
|  Nationality = canadian: Michael_Jackson_(actor)
```

Figure 6.3: Automatically compiled disambiguation tree for persons named Michael Jackson.

the remaining six we were unable to extract the profession and nationality features from the first sentence using our domain models either because the profession and nationality were not mentioned or our domain model did not contain a matching entry. We could generate a more comprehensive list of professions by adding new domain model entries using resources such as DBpedia, Freebase and YAGO ontology or by extracting them from text using patterns.

For one person we were not able to extract the profession feature and the nationality was not discriminatory enough to distinguish it from others. For two of them the profession and the nationality were the same, e.g., Michael_A._Jackson and Michael_C. _Jackson have nationality "British" and profession "Scientist". Similarly, Michael_Jackson_(basketball) and Michael_Jackson_(wide_receiver) have nationality

```
Profession = politician: Michael_Jordan_(Irish_politician)
Profession = footballer: Michael_Jordan_(footballer)
Profession = player: Michael-Hakim_Jordan
Profession = researcher: Michael_I._Jordan
Profession = actor: Michael_B._Jordan
```

Figure 6.4: Automatically compiled disambiguation tree for persons named Michael Jordan.

```
Profession = sailor: George_H._Harrison
Profession = swimmer: George_Harrison_(swimmer)
Profession = banker: George_L._Harrison
Profession = 0
|  Nationality = english: George_Harrison_(civil_servant)
|  Nationality = irish: George_Harrison_(Irish_Republican)
Profession = guitarist: George_Harrison
Profession = president: George_Harrison_(executive)
Profession = editor: Harrison_George
```

Figure 6.5: Automatically compiled disambiguation tree for persons named George Harrison.

"American" and profession "Player". The ambiguity can be reduced with a finer-grained professions hierarchy. For example, Michael_A._ Jackson is described as a "Computer Scientist" and Michael_C._Jackson as a "Systems Scientist". The entity Michael_Jackson_(basketball) is said to be a "Basketball Player" whereas Michael_Jackson_ (wide_receiver) is a "Football Player".

Of six people named Michael Jordan, we were able to disambiguate five of them using the profession and nationality features. The one that could not be classified (Michael_Jordan) had the same profession and nationality as Michael_Hakim_Jordan. Both have the nationality "American" and profession as "Player" and at fine grained level both are "Basketball Players". There is a need of additional features in order to disambiguate between them.

Out of thirteen people named "George Harrison" we were able to disambiguate eight using the profession and nationality features. For five of them we were not able to extract the profession and nationality features from the first sentence. Our disambiguation trees show that the profession and nationality features are very useful in disambiguating different entities with the same name in Wikipedia. We can extract the profession and nationality features from the first sentence in articles about people in most of the cases.

From the profession and nationality attributes, profession is selected as the first node to split on by the decision tree algorithm in all of the disambiguation trees we discussed, which shows that the profession feature is more discriminatory and helpful in disambiguating people with the same name as compared to nationality of a person in Wikipedia.

For locations with confusable names, we have observed that another location can help in disambiguating it. For example, the disambiguation page on "Springfield" refers to 64 different place names in Wikipedia. In almost all cases it is disambiguated by using another place name or location which is usually mentioned in the first sentence of the respective article. Therefore, disambiguation trees for locations can be generated using other locations mentioned in the first sentence. We also observed that in several cases the organizations are also disambiguated based on their locations. For example, different countries have political parties with the same name and the titles of the Wikipedia articles on such political parties are often followed by their locations.

Other features that may be discriminatory are the type of entity for example,

in case of locations the type may be a city, county, country, airport etc. For persons, the type may be based on his profession such as basketball player, singer, author etc. In case of organizations the type may be a political party, sports team, university etc. The first sentence in articles on entities in Wikipedia often mentions the type of the entity. In section 6.1.2.1, we have already discussed how we associate concepts in Wikipedia with WordNet nodes by extracting the type information from the first sentence. In general the nouns and adjectives in the first sentence of articles on concepts can provide discriminatory features for disambiguating in majority of the cases.

6.2.3 Evaluation

In Wikipedia, different pages cannot share the same title. In case the title of the new concept is already existing for another concept, Wikipedia guidelines recommend adding a disambiguating tag after the ambiguous title in parentheses or sometimes after a comma . For example, "Springfield,_California","Springfield,_Colorado", "Michael_Jackson_(singer)", "Michael_Jackson_(jazz_guitarist)". We exploit the information in parentheses or after a comma for ambiguous named entities in Wikipedia for evaluating our extracted disambiguation features.

For locations and organizations, we extracted the locations mentioned in first sentence and WordNet types to represent disambiguating features. For persons, in addition to locations and WordNet types we also extracted their profession mentioned in the first sentence by matching the nouns in first sentence with list of

professions generated using all the nodes appearing under the "Person" synset in WordNet.

In order to evaluate the extracted disambiguating features we compare the features with the words that appear after commas "," or in brackets in the titles of ambiguous Wikipedia articles. We downloaded the disambiguation links from DBpedia data sets , which contains a list of concepts that are linked to disambiguation pages in Wikipedia and are ambiguous. We randomly selected 100 ambiguous persons, locations and organizations from that list which had a disambiguation tag associated. We extracted disambiguation features for the selected entities using our approach. If the disambiguation tags match with any of our extracted disambiguation features for an entity, we consider it accurate and refer to it as a "Direct Match". In case our feature set contains a related but relatively generalized concept as compared to the disambiguation tag we consider it a "Generalized Match" for example, if the disambiguation tag is "(Antrim_County,_Michigan)", we consider "Michigan" to be a Generalized Match. However, if the concept is relatively specialized with respect to the disambiguation tag we consider it a "Specialized Match" for example, we consider "guitarist" as a Specialized Match for the disambiguation tag "musician". The accuracy obtained with respect to "Direct Match", "Generalized Match" and "Specialized Match" is given in Table 6.5.

Table 6.5: Disambiguation Features evaluation for different entity types.

Entity Type	Direct Match	Generalized Match	Specialized Match	Total
Persons	57	18	9	84
Locations	72	15	5	92
Organizations	67	5	1	73

6.3 A Broader Framework for automatically enriching Wikitology

The different Wikitology based approaches combined together can contribute to a number of steps in a broader unified framework for adding an article on a new concept in Wikipedia and hence in the Wikitology KB (Figure 6.6). Future work may be targeted towards filling the gaps in the framework and evaluating the overall performance of different steps in the unified architecture.

6.3.1 Linking to Existing Concept

In order to add a new article to Wikipedia, the first step in our approach is to check if an article already exists on that concept. We can use our entity linking approach to determine this. Our Entity Linking approach was originally targeted for the entity linking task introduced in the KBP track of TAC 2009. The task is defined as given an entity mention string and an article, link it to the right entity in Wikipedia and if the entity doesn't exist then add it to the KB. Our approach is general enough and can be applied for linking other concepts in addition to named entities.

In case an article on the input concept already exists we can update the existing article and add a reference to the new article in the "external links" section and refrain from adding duplicate concepts. If an article on the concept does not already exist we follow the steps to add the article to Wikipedia and Wikitology KB.

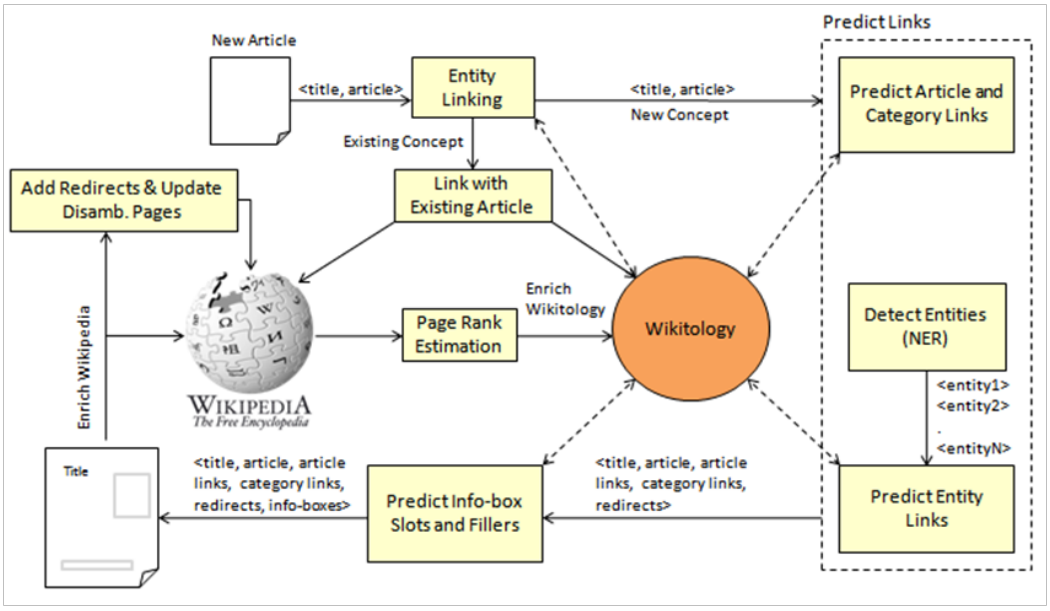


Figure 6.6: Approach for adding new articles to Wikipedia showing different Wikitology based approaches contributing towards automatically enriching Wikipedia and hence the Wikitology Knowledge Base itself.

6.3.2 Predicting Categories and Inter-article Links

The second step in our approach predicts the categories and the inter-article links for the new article. We can employ our Document Concept Prediction approach [93] for predicting generalized concepts i.e. categories as well as specialized concepts i.e. related Wikipedia articles for a new document.

For a given document as input we retrieve top N similar Wikipedia articles from the Wikitology index and use them as seed nodes for spreading activation algorithm on the page links graph to predict specialized concepts related to the input document. The categories associated with the top N similar articles are used as seed nodes for spreading activation on the category links graph for predicting generalized concepts or categories for the input document.

We have evaluated the system by predicting the categories and article links of

existing Wikipedia articles and compared them with the ground truth using measures for precision, average precisions, recall and f-measure. We observed that the greater the average similarity between the new document and the retrieved top N Wikipedia articles, the better the prediction. The average similarity score can also be used as a weight on the links to represent the confidence in accuracy of the predictions. Our approach for predicting inter-article links currently predicts the concepts to link to in Wikipedia, however, the approach would need to be extended to identify the concept mentions in the article to actually place the hyperlinks.

While our concept prediction approach can predict related concepts/articles to link, we also have specialized approaches for linking different types of entities to the relevant Wikipedia articles on entities [39]. We can use any existing NER system to identify the different entities (persons, locations and organizations) mentioned in the new article and then for each entity we can use our entity linking approach to link to the right Wikipedia article. We can also identify the redirects to articles as a side effect of entity linking. The different entity mentions that link to the same entity can serve as redirects to that entity provided they are unique and do not exist as redirects for any existing concepts.

6.3.3 Predict Infobox Slots and Fillers

Our approach for discovering ontology elements from Wikipedia article links can be exploited to discover structured data similar to the infoboxes in Wikipedia. We have discussed that there are certain properties which are evident in the article

links and can be discovered with high accuracy. However, there are certain properties that require more context such as successor, predecessor, population etc. Systems like YAGO and Kylin can complement our approach for predicting such properties whereas, our approach can complement these systems by discovering new slots and fillers for existing infoboxes.

6.3.4 Related Work

To our knowledge there doesn't exist a single unified system that supports all steps of adding a new article automatically to Wikipedia. However, there are a few systems that support individual steps of adding a new article and can complement the different steps proposed in our framework. We discuss those systems below. Schonhofen [86] developed an approach for identifying document topics using Wikipedia category network and evaluated the approach by predicting categories of Wikipedia articles. Wikify [67] is an approach that annotates any given text with links to Wikipedia using keyword extraction and word sense disambiguation. Milne and Witten [70] use a machine learning approach for cross referencing documents within Wikipedia. Both the systems can complement our approach for discovering inter-article links for the new article.

Systems like YAGO [91] and Kylin [**?**] can be used to predict infoboxes for entities. Our system for ontology discovery can complement these existing systems by discovering certain existing as well as new relations evident in inter-article links in Wikipedia. For example, we could add slots like songs and albums to the infobox

schema for Musical Artists, movies for the Actors infobox schema, and party for the Politicians schema.

6.3.5 Conclusions

We have presented different approaches that can enrich Wikipedia and hence the Wikitology KB and can support a number of steps in the proposed framework for adding new articles/concepts in Wikipedia. Our approaches can predict whether the concept already exists in Wikipedia, the categories, inter-article links, redirects as well as certain infobox slots and fillers for the new article. The different approaches have been evaluated individually. We have also presented an approach for generating disambiguation trees for ambiguous entities and a way to predict the popularity based on Google page rank. These approaches integrated together can contribute to a number of steps in a broader unified framework for enriching Wikipedia and hence the Wikitology Knowledge Base with new concepts.

Chapter 7

Evaluation Summary

In chapter 4 we discussed different approaches that exploit the Wikitology knowledge base. We presented a detailed evaluation of the approaches in the respective sections wielding measures of accuracy, precision, recall and f-measure. The evaluation of the approaches that exploit Wikitology directly serves to evaluate the utility of the knowledge base for different real world tasks. In this chapter we first review the different approaches that can be used to evaluate knowledge bases in general and then discuss our task based evaluation of Wikitology. We then present a summary of evaluation and discuss the performance of Wikitology using different approaches for different tasks and finally conclude this chapter.

7.1 Approaches for Evaluating Knowledge Bases

There are different approaches for evaluating knowledge bases that exist as ontologies. In short, they can be classified as any one of the following [108]:

Gold Standard Evaluation (Comparison to an existing Ontology): In this approach, the given ontology is compared to a benchmark ontology. Meadche and Staab [63] demonstrated gold standard ontology evaluation in their work. Ponzetto and Strube derived a large scale taxonomy from Wikipedia Categories by labeling the relations as "ISA" and "NotISA" using connectivity property of the

network and lexico-syntactic patterns [80]. They compared their taxonomy with ResearchCyc [4] and WordNet [68] and got competitive results.

Criteria based Evaluation (by humans): Ontologies can also be evaluated based on different criteria such as consistency, completeness, conciseness, expandability and sensitivity, however automating this procedure is difficult [25] and even if the ontology meets the standards, it might still not be useful for a particular application.

Task based Evaluation (Application based): In this case, the Ontology is evaluated based on how well it can complete a task in the context of that task or application. A disadvantage of this approach is that the evaluation is very specific to the application and the results might change over different applications.

Comparison with Source of Data (Data driven): In this case the ontology is compared with the source of data to see how well the ontology covers the given domain [62].

Using a Reasoning Engine: A reasoning engine could also be used to evaluate ontologies in terms of inconsistencies present in them.

7.2 Evaluating Wikitology Knowledge Base

Wikitology is a hybrid knowledge base composed of structured and un-structured information integrated in a novel way. It is difficult to evaluate Wikitology by comparing it with any existing resource (such as a bench mark or Gold Standard) due to

the hybrid nature. Criteria based evaluation discussed above is difficult, requires a lot of human labor and may still not be able to evaluate the utility of the knowledge base for different applications. Reasoning engines may only be used to evaluate the triples in Wikitology.

The task based evaluation seems more appropriate for our case as it does not involve human labor like other cases and also serves to evaluate the utility of the knowledge base with respect to the task. However, evaluating against just one task would make it very specific to that application therefore, we adopted the task based evaluation approach and evaluated Wikitology against a variety of tasks rather than a single one, which not only serves to evaluate the hybrid knowledge base and its utility for applications but also demonstrates how the knowledge base could be exploited in a variety of ways by different applications.

Research projects like Cyc [61] have resulted in the development of a complex broad coverage knowledge base however, relatively few applications have been built that really exploit it. In contrast, the design and development of Wikitology KB has been incremental and has been driven and guided by a variety of applications and approaches that exploit Wikitology in different ways. Exploiting and evaluating Wikitology using different applications would highlight its utility and usefulness for applications.

In chapter 4, we discussed different approaches which exploit Wikitology for solving different real world problems, namely, document concept prediction, cross document co-reference resolution defined as a task in Automatic Content Extraction (ACE) [1], entity linking to KB entities defined as a part of Text Analysis Conference

Knowledge Base Population Track 2009 [65] and interpreting tables [94]. In the remaining part of this chapter we briefly review the different approaches used to evaluate the performance of Wikitology knowledge base, summarize and discuss the evaluation results and suggest the kind of applications that can benefit from Wikitology.

7.3 Evaluation for Document Concept Prediction task

We exploited the initial version of Wikitology knowledge base to predict individual document topics as well as concepts common to a set of documents. Our approach exploited the structured information in the form of a loose ontology i.e. the inter-category and inter-article links and un-structured information in the form of free text in Wikitology. Wikipedia articles served as specialized concepts whereas, the Wikipedia categories served as generalized concepts with inter-article, article-category and inter-category links representing relations between concepts. These concepts interlinked with each other exposed knowledge in the form of loose concept ontology. The article text present as contents field in the Wikitology's specialized index served as a way to map free text in test documents to concepts in Wikitology. Wikitology knowledge base was exploited by algorithms to select, rank and aggregate concepts.

We evaluated the system by predicting the categories and article links of existing Wikipedia articles and compared them with the ground truth. We then computed measures for precision, average precisions, recall and F-measure. We observed

that the greater the average similarity between the test documents and the retrieved top N Wikipedia articles, the better the prediction. We got a precision of 0.62, recall of 0.94 and f-measure of 0.75 for Generalized Concept prediction, at an average similarity threshold of 0.5, whereas, for specialized concept prediction the values for precision, recall and f-measure were 0.59, 0.88 and 0.67 respectively.

We observed that while the Wikipedia category graph can be used to predict generalized concepts, the article links graph helped by predicting more specific concepts and concepts not in the category hierarchy. Our experiments showed that it is possible to suggest new category concepts identified as a union of pages from the page link graph. Such predicted concepts could also be used to define new categories or sub-categories within Wikipedia.

Our experiments and evaluation demonstrate that Wikitology can be exploited successfully using the approaches we described to predict generalized and specialized concepts related to documents. Measures such as, average similarity between the test article(s) and top N Wikipedia articles can serve as a measure of accuracy for the prediction.

7.4 Evaluation for Cross Document Coreference Resolution Task

We constructed an enhanced version of the Wikitology system as a knowledge base of known individuals and organizations as well as general concepts for use in the ACE cross document co-reference task. This was used as a component of a system developed by the JHU Human Language Technology Center of Excellence

[64]. For our ACE task we enhanced Wikitology in several ways and added a custom query front end to better support the cross document co-reference resolution task. Starting with the original Wikitology, we imported structured data in RDF from DBpedia [17], Freebase [23] and YAGO ontology [91]. Most of the data in DBpedia and Freebase were in fact derived from Wikipedia, but have been mapped onto various ontologies and reimported in structured form. The structured data was encoded in an RDFa-like format in a separate field in the Lucene index object [52] for the Wikipedia page. This allows one to query Wikitology using both text (e.g., an entity document) and structured constraints (e.g., rdfs:type=YAGO:Person).

We enriched the text associated with each article with titles of Wikipedia "redirects". A Wikipedia redirect page is a pseudo page with a title that is an alternate name or misspelling for the article. We extracted type information for people and organizations from the Freebase system. This information was stored in a separate database and used by the ACE Wikitology query system. We extracted data from Wikipedia's disambiguation pages to identify Wikitology terms that might be easily confused, e.g., the many people named Michael Jordan that are in Wikipedia. This information was stored in a separate table and used in the Wikitology feature computation for a feature indicating that two document entities do not refer to the same individual. We used special "entity documents" or EDOCs extracted from the Serif's [24] APF output for the English documents as input to our system based on the Wikitology knowledge base. Each entity in a given document produced one EDOC that included the longest entity mention, all name mentions, all nominal mentions, all pronominal mentions, APF type and subtype and all words within

15 tokens of each mention. The EDOCs were used to find candidate matches in Wikitology.

The EDOCs were processed by a custom query module for Wikitology that mapped the information in the EDOC into different components of Wikitology entries. The Wikitology module returns two vectors: one for matches against article entries and the other against category articles. We produced twelve features based on Wikitology: seven that were intended to measure similarity of a pair of entities and five to measure their dissimilarity.

To analyze and evaluate our approach and Wikitology knowledge base we constructed a training set and a test set from the EDOCs for which human judgments were available for the cross-document entity co-reference task and used SVM to classify a given pair of entity mentions as a positive match or a negative match to indicate if the two mentions refer to the same entity or not. Using Wikitology based features, the SVM classifier was able to classify the positive matches with a precision of 0.96, recall of 0.72 and f-measure of 0.82. Whereas, for negative matches the precision was 0.99, recall was 0.99 and f-measure 0.99. Our evaluation demonstrates that Wikitology based features are indeed useful and can be used effectively for the cross-document entity coreference resolution task with high accuracy.

7.5 Evaluation for Entity Linking Task

Wikitology knowledge base was employed for Entity Linking task defined as a part of Knowledge Base Population (KBP) track of the 2009 Text Analysis Con-

ference. One part of the KBP track is to link entities found in text to those in an external knowledge base. In particular, the entity linking task is defined as: given an entity mention string and an article with that entity mention, find the link to the right Wikipedia entity if one exists. We developed specialized query modules for distributing different parts of the query to different structured and un-structured components in Wikitology. We developed approaches based on the type of entity and also in general, for linking entities mentioned in text to the right Wikitology entities and also for detecting if the entity doesn't exist in Wikitology. We evaluated the Wikitology based approaches using the Wikinews corpus, which consists of news articles that are linked manually by contributors to relevant Wikipedia articles. We extracted all the links to Wikipedia articles and the surface text associated with the links as entity mentions. We created a test set of 1000 articles and entity mentions for persons, locations and organizations each by randomly selecting articles which had links to persons, locations and organizations in Wikipedia. We conducted a set of experiments for evaluating different approaches for entity linking and also for identifying new entities.

We observed that amongst the four different approaches, approach 4 in general gave the highest accuracy for all the three entity types i.e. 87.5%. The specialized approaches for Persons and Locations further improved the accuracy to 96.9% and 93.3% for Persons and Locations respectively.

In order to detect if the entities are not present in Wikitology we learned a score threshold to distinguish between an existing entity in the knowledge base and a new entity. We learned score thresholds for each type of entity and also an overall

threshold independent of the type of entity. The accuracy for identifying whether an entity did or did not exist in Wikitology was for 79.9%. This evaluation demonstrates that simple Wikitology based approaches that intelligently query Wikitology, can be used effectively for the entity linking task and also for identifying new entities.

7.6 Evaluation for Interpreting Tables

We have developed a Wikitology based approach to linking data in the table and table headers to concepts in the DBpedia ontology. The labels in the table headers, if present, as well as the values in the rows, can be used to interpret the information contained in the tables. Linking the table headers as well as the instances in the rows to concepts in a knowledge base can aid in providing more context and links to other related concepts. In the case of tabular data, the individual entry in a particular row and a column represents the entity mention. The different parts of the table serve as the context to disambiguate the entity or concept. Similar to the entity linking task it is not trivial to link table headers or values to concepts in the KB as the same concept may be expressed in a variety of ways in the table headers as well as data rows and there might not be enough context available in the table as compared to a document.

It is often the case that the table column header represents the type of information in that column such as cities, countries, artists, movies etc. whereas, the values in the columns represent the instances of that type. The values in the rows of the tables may represent related information to the instances in the same

row. We exploited different fields available in Wikitology's specialized IR index to infer a (partial) semantic model for information available in tabular forms. We processed the information in the table using a custom query module for Wikitology that mapped the information in different parts of the table to different components of Wikitology index.

We collected a test data set composed of 16 tables taken from Google Squared [47], Wikipedia and Web which had a total of 611 entities, and 52 columns. For each column Wikitology was exploited to predict class labels from four vocabularies i.e., Dbpedia Ontology, YAGO, WordNet and Freebase. In case of entity linking for entities present in table cells, out of 611 entities we were able to link 66.12% of entities accurately. For column label prediction the accuracy was 76.92%. Our evaluation on the test data set shows that our approach exploiting Wikitology is promising and Wikitology can be used effectively for automated extraction of linked data from structured data in the form of tables.

7.7 Discussion

We have evaluated Wikitology using a variety of approaches for different tasks. Wikitology has been evaluated not only using complex approaches which use specialized algorithms (such as spreading activation) in addition to output from Wikitology, but also for relatively simple approaches that are mainly based on the output of specialized query modules querying Wikitology in various ways and require some pre and post processing based on the need of application. This also demonstrates that

Wikitology's hybrid query interface can support a variety of applications which can query Wikitology in different ways for different tasks and require a little pre and post processing.

We evaluated Wikitology not only on Wikipedia articles (for Document Concept Prediction) but also on off-domain corpora such as ACE News Articles, Web People corpus as well as Wikinews. Our evaluation results show that Wikitology is not only suitable for encyclopedic articles but can be employed effectively for tasks on off-domain corpora as well. Table 7.1 summarizes the different tasks and approaches used to evaluate Wikitology, the corpora domains used for evaluation, the different Wikitology components exploited by applications and the results of different evaluation measures i.e. accuracy, precision, recall and f-measure depending on the type of application.

A common feature of the set of approaches that we have used to evaluate Wikitology is that they all exploit both structured and un-structured data in Wikitology. The different approaches query Wikitology in a variety of ways using a combination of structural constraints along with content, where the content part of the query ranges from full documents as input (for Concept Prediction, Entity Linking) to context words in EDOCS (for Coreference Resolution) to values in the table rows and column headers (for interpreting tables).

7.8 Target Applications for Wikitology

Wikitology is suitable for applications that would like to exploit:

Table 7.1: Summary of performance of Wikitology based approaches.

Task	Corpus Domain/Genre	Wikitology Components Exploited	Sub Task	Acc.%	P	R	F
Concept Prediction (@ 0.5 threshold)	Wikipedia Encyclopedia	Article content, Category Links and Article Links Graphs	Generalized Concepts	-	0.62	0.94	0.75
			Specialized Concepts	-	0.59	0.88	0.67
Cross Document Coreference Resolution	ACE, Web People	Article content, title, redirects, YAGO Types, Disambiguation pages	Positive Match	-	.97	.72	.83
			Negative Match	-	.99	.99	.99
Entity Linking	Wikinews News Articles	Article content, title, redirects, YAGO and Freebase Types	Existing Entities	87.5	-	-	-
			New Entities	79.9	-	-	-
Interpreting tables	Google Squared Tables	Article content, title, redirects, types, Infoboxes, first sentence, linked Concepts and PageRank	Interpreting Row Data	66.12	-	-	-
			Interpreting Column Headers	76.92	-	-	-

- general world knowledge available in Wikipedia
- a combination of structured and un-structured data in the form of articles, infoboxes, links, graphs, categories, first sentence, concept types available in Wikitology to solve problems
- returned ranked concepts based on relevance by Wikitology, by either using them directly as results or using them as input for other algorithms.

7.9 Conclusions

In this chapter, we have given an overview of general approaches for evaluating knowledge bases that exist as ontologies. We found the task based evaluation approach suitable and applicable to our hybrid knowledge base and summarized and presented the evaluation results based on a variety of tasks and corpora. We also discussed the types of applications for which Wikitology would be suitable.

Chapter 8

Conclusions

In this dissertation, we have presented Wikitology, a novel hybrid knowledge base derived from Wikipedia. This dissertation encompasses four main contributions: the Wikitology Hybrid Knowledge Base as a resource (Chapter 3), approaches that exploit structured and unstructured knowledge representation through Wikitology for solving a variety of real world problems (Chapter 4), the design of a hybrid architecture for representing multiple data structures and providing a unified query interface (Chapter 5), and approaches for automatically enriching the Wikitology knowledge base with new knowledge (Chapter 6).

These approaches, in combination, form a powerful set of tools along with the hybrid knowledge base that not only support different applications but can be used to directly expand and enrich the hybrid knowledge base automatically.

In the thesis statement we stated that we can create a hybrid knowledge base from Wikipedia and other related knowledge sources by automatically generating knowledge about the world, effectively supporting a diverse set of common use cases. Wikipedia is originally designed for humans. Applications need to employ intelligent approaches to access and harvest the knowledge available in Wikipedia for solving different problems. In this dissertation, we have presented novel approaches for solving a variety of real world problems by exploiting hybrid knowledge avail-

able in different forms such as free text, link graphs, categories and triples in a collaboratively developed knowledge resource, Wikipedia along with other related resources.

These novel approaches exploiting Wikipedia have guided and directed the incremental development of the Wikitology knowledge base which unifies a variety of hybrid knowledge sources and representations in an organized way and provides a rich integrated query interface thus enabling applications to better access and exploit knowledge hidden in different forms and in different resources.

This dissertation also presents a set of approaches that can generate structured data for automatically enriching Wikipedia and hence the Wikitology knowledge base by predicting inter-article links, categories, redirects, disambiguation trees and discovering ontology elements using the Wikipedia inter-article links.

Structured data is often sparse. Unstructured data or free text can complement the structured data to overcome data sparseness in many cases. Text similarity algorithms return ranked results based on relevance which is not the case for structured data. A knowledge base incorporating information available in different forms can better meet the needs of real world applications than one focusing and exposing knowledge in a more restricted way such as through SQL, SPARQL or simple keyword queries. Exploiting Wikipedia and related knowledge sources to develop a novel hybrid knowledge base brings advantages inherent to Wikipedia. Wikipedia provides a way to allow ordinary people to contribute knowledge as it is familiar and easy to use. This collaborative development process leads to a consensus model that is kept current and up-to-date and is also available in many languages. In-

corporating these qualities in knowledge bases like Cyc [61] will be very expensive in terms of time, effort and cost. Efforts like DBpedia, Freebase and Linked Open Data are focused on making knowledge available in structured forms.

Wikitology knowledge base can complement existing resources by integrating knowledge available in other forms and providing much more flexible access to knowledge. We have directly demonstrated through our work that we can use world knowledge accessible through Wikitology hybrid knowledge base system to go beyond the level of mere words and can predict the semantic concepts present in documents as well as resolve ambiguity in NER systems by mapping the entities mentioned in documents to unique entities in the real world. Wikitology knowledge base system can provide a way to access and utilize common-sense and background knowledge for solving a variety of real world problems.

8.1 Discussion

The broader goal of this research has been to develop approaches that can effectively exploit hybrid knowledge representations as well as resources and hence guide the development of a hybrid general purpose knowledge base that would unify different types of knowledge resources and provide an integrated query interface to applications for exploiting general knowledge for solving real world problems. Our evaluation results for solving a variety of problems demonstrates that structured and unstructured representations can complement each other and result in better performance of the system for a given task.

Through Wikitology we have provided a framework for combining and integrating different knowledge resources namely, Wikipedia, DBpedia, WordNet, YAGO and Freebase as well as Linked Open Data via DBpedia links. Through the Wikitology Query Interface, applications can exploit knowledge available in Wikipedia and related resources in a much more systematic, organized and unified way. The same framework may be used directly or extended to incorporate knowledge available in different forms (such as articles, tables or ontologies and triples) and across different types of corpora such as news articles, encyclopedic articles, biographies, blogs etc.

The structured and un-structured knowledge representations complement each other. We solved different problems by first intelligently mapping the problem to the conceptual space in Wikitology and then exploiting and reasoning over the different available representations using a variety of approaches. For example, incase of document concept prediction we used text similarity measures such as cosine similarity between TfIdf vectors to map the text in documents to concepts in Wikitology and then exploited the structured data available in the form of category link and page link graphs using spreading activation graph algorithm for ranking and aggregating concepts represented in documents. We also introduced edge weights for directing the spread of activation in case of page link graph. Our evaluation results demonstrate that exploiting both the structured representations along with the unstructured representation improved the results.

In case of cross document coreference resolution we generated special entity documents from the input documents and intelligently mapped the entity mentions

to concepts in Wikitology by distributing different parts of the EDOC to different components in the KB. Based on the mapped concepts returned by Wikitology we generated a set of features to identify if a pair of entity mentions refered to the same entity or not. Wikitology based features directly contributed to improving the performance of a broader system [64] developed by JHU Human Language Technology Center of Excellence which also incorporated a set of other features.

For the entity linking task, it was observed that the query module that exploited additional fields available through Wikitology and boosted certain fields achieved better performance as compared to simple query modules. Wikitology Query Interface enabled incorporating structural constraints such as the type of entity which also contributed to improving the results [39].

For interpreting tables, the information represented in tabular form was initially intelligently mapped to conceptual space in Wikitology through a specialized query module. We then ran well designed heuristics exploiting the table in conceptual space to infer the classes representing the different columns. Once the column headers were mapped to concepts, that additional evidence was used to requery Wiktology by imposing structural constraints on the class of concepts for linking cell values to concepts.

The different Wikitology based approaches targeting a variety of real world problems make it clearly evident that structured and unstructured knowledge representations complement each other and can improve the performance of a variety of systems.

The set of approaches presented in this thesis mainly exploit Wikipedia and

related resources, however, many of the approaches may be employed independently or can be used on other similar knowledge resources as well. Our document concept prediction approach can be employed on other corpora that have inter-article links and categories associated with them such as Microsoft Encarta [9], Britannica [3] Scholarpedia [12], subject specific encyclopedias, news corpuses such as New York Times [10], BBC news [2] or internal Wiki's to an organization such as Diplopedia [6] and corpora related to biographies. Our approach is not language dependent and hence can run on articles in other languages as well, which are interlinked and may have categories associated with them.

Our approaches for cross-document named entity co-reference resolution and entity linking have been evaluated on news articles. Both approaches take as input an entity mention and an article that mentions that entity as context for the entity. We do not use any corpus or language specific features and represent the entity article as Bag of Words (BOW) therefore our approach can be directly employed on corpora belonging to different genres and languages as well. We have also demonstrated this by applying the entity linking approach for linking entities in tables. Our approach for discovering ontology elements from Wikipedia page links can be applied to other knowledge resources having inter-linked encyclopedic articles or dictionaries.

8.2 Future Work

There are numerous opportunities in which our work can be extended and improved. Our work on discovering ontology elements can be improved by using other features in addition to article links such as text surrounding the links and features extracted through linguistic analysis of sentences. A more efficient and parallel architecture can be developed for Wikitology so that applications requiring large scale processing of data may be able to run more efficiently. The Wikipedia article links and category links graphs can be exploited in numerous ways for different applications. New features can be computed in addition to PageRank.

We have used the default similarity function available in Lucene library [52] for Wikitology however, the similarity function can be customized to meet the needs of specific applications. Wikitology may be linked with other non-encyclopedic resources that are related to concepts in Wikipedia such as product reviews, blogs, news articles, shopping sites, airline databases and support even a wider range of applications, search and browsing.

Another area worth exploring is developing approaches that can reason over the hybrid knowledge available in Wikitology so that applications can benefit from inference in addition to existing knowledge.

Bibliography

[1] Automatic content extraction (ace) evaluation.

www.nist.gov/speech/tests/ace/.

[2] Bbc news.

http://news.bbc.co.uk/.

[3] Britannica.

http://www.britannica.com.

[4] Cycorp inc., researchcyc. [online].

http://research.cyc.com/.

[5] Dbpedia wordnet classes.

http://wiki.dbpedia.org/Downloads34#wordnetclasses.

[6] Diplopedia.

http://www.state.gov/m/irm/ediplomacy/115847.htm.

[7] Ldc - projects - ace08 annotation tasks. http://projects.ldc.upenn.edu/ace/annotation/.

[8] Luke.

http://www.getopt.org/luke/.

[9] Microsoft encarta.

http://www.microsoft.com/uk/encarta/default.mspx.

[10] The new york times linked open data.

http://data.nytimes.com.

[11] Postgresql.

urlhttp://www.postgresql.org.

[12] Scholarpedia.

http://www.scholarpedia.org/.

[13] Rakesh Agrawal, Anastasia Ailamaki, Philip A. Bernstein, Eric A. Brewer, Michael J. Carey, Surajit Chaudhuri, Anhai Doan, Daniela Florescu, Michael J. Franklin, Hector G. Molina, Johannes Gehrke, Le Gruenwald, Laura M. Haas, Alon Y. Halevy, Joseph M. Hellerstein, Yannis E. Ioannidis, Hank F. Korth, Donald Kossmann, Samuel Madden, Roger Magoulas, Beng C. Ooi, Tim O'Reilly, Raghu Ramakrishnan, Sunita Sarawagi, Michael Stonebraker, Alexander S. Szalay, and Gerhard Weikum. The claremont report on database research. *Commun. ACM*, 52(6):56–65, 2009.

[14] David Ahn, Valentin Jijkoun, Gilad Mishne, Karin Müller, Maarten de Rijke, and Schlobach. Using wikipedia at the trec qa track. In *Proceedings of TREC 2004*, 2004.

[15] Alias-i. Lingpipe 3.9.0, 2008.

[16] Ahmet Arslan and Ozgur Yilmazel. A comparison of relational databases and information retrieval libraries on turkish text retrieval. In *2008 International Conference on Natural Language Processing and Knowledge Engineering*, pages 1–8. IEEE, October 2008.

[17] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *In 6th Int'l Semantic Web Conference, Busan, Korea*, pages 11–15, 2007.

[18] Ken Barker, Bhalchandra Agashe, Shaw Y. Chaw, James Fan, Noah Friedland, Michael Glass, Jerry Hobbs, Eduard Hovy, David Israel, Doo S. Kim, Rutu M. Mehta, Sourabh Patwardhan, Bruce Porter, Dan Tecuci, and Peter Yeh. Learning by reading: a prototype system, performance baseline and lessons learned. In *AAAI'07: Proceedings of the 22nd national conference on Artificial intelligence*, pages 280–286. AAAI Press, 2007.

[19] Francesco Bellomi and Roberto Bonato. Network analysis for wikipedia. 2005.

[20] Aspasia Beneti, Woiyl Hammoumi, Eric Hielscher, Martin Müller, and David Persons. Automatic generation of fine-grained named entity classifications. In *Technical report, University of Amsterdam*, 2006.

[21] Kristin P. Bennett and Colin Campbell. Support vector machines: Hype or hallelujah? *SIGKDD Explorations*, 2, 2003.

[22] Christian Bizer, Tom Heath, Kingsley Idehen, and Tim B. Lee. Linked data on the web (ldow2008). In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 1265–1266, New York, NY, USA, 2008. ACM.

[23] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human

knowledge. In *SIGMOD '08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250, New York, NY, USA, 2008. ACM.

[24] E. Boschee, R. Weischedel, and A. Zamanian. Automatic Information Extraction. In *Proceedings of the 2005 International Conference on Intelligence Analysis, McLean, VA*, pages 2–4, 2005.

[25] C. Brewster, H. Alani, S. Dasmahapatra, and Y. Wilks. Data driven ontology evaluation. In *Proceedings of International Conference on Language Resources and Evaluation*, Lisbon, Portugal, 2004.

[26] A. Capocci, F. Rao, and G. Caldarelli. Taxonomy and clustering in collaborative systems: the case of the on-line encyclopedia wikipedia. Oct 2007.

[27] J. J. Carroll, I. Dickinson, C. Dollin, D. Reynolds, A. Seaborne, and K. Wilkinson. Jena: implementing the semantic web recommendations. In *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, pages 74–83. ACM New York, NY, USA, 2004.

[28] Sergey Chernov, Tereza Iofciu, Wolfgang Nejdl, and Xuan Zhou. Extracting semantics relationships between wikipedia categories. In Max Völkel and Sebastian Schaffert, editors, *Proceedings of the First Workshop on Semantic Wikis – From Wiki To Semantics*, Workshop on Semantic Wikis. ESWC2006, June 2006.

[29] T. Converse, R. M. Kaplan, B. Pell, S. Prevost, L. Thione, and C. Walters. Powerset's Natural Language Wikipedia Search Engine.

[30] N. Coulter, J. French, T. Horton, N. Mead, R. Rada, A. Ralston, C. Rodkin, B. Rous, A. Tucker, P. Wegner, E. Weiss, and C. Wierzbicki. Computing classification system 1998: Current status and future maintenance report of the ccs update committee. In *In ACM Computing Reviews, 39(1):1-24*. ACM Press New York, NY, USA, January 1998.

[31] F. Crestani. Application of spreading activation techniques in informationretrieval. *Artif. Intell. Rev.*, 11(6):453–482, December 1997.

[32] M. Dewey. Abridged dewey decimal classification and relative index. Forest Press, 1990.

[33] J. Didion. Jwnl (java wordnet library).

http://sourceforge.net/projects/jwordnet/, 2004.

[34] Li Ding, Lina Zhou, Tim Finin, and Anupam Joshi. How the semantic web is being used: An analysis of foaf documents. In *In Proceedings of the 38th International Conference on System Sciences*, 2005.

[35] Ofer Egozi, Evgeniy Gabrilovich, and Shaul Markovitch. Concept-based feature generation and selection for information retrieval. In *AAAI'08: Proceedings of the 23rd national conference on Artificial intelligence*, pages 1132–1137. AAAI Press, 2008.

[36] As O. Elements, Daniel Bachlechner, and Katharina Siorpaes. Harvesting wiki consensus - using wikipedia entries. In *IEEE Internet Computing*, volume 11, pages 54–65, 2006.

[37] Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S. Weld. Open information extraction from the web. volume 51, pages 68–74, New York, NY, USA, December 2008. ACM.

[38] T. Finin, Z. Syed, J. Mayfield, P. McNamee, and C. Piatko. Using wikitology for cross-document entity coreference resolution. In *Proceedings of the AAAI Spring Symposium on Learning by Reading and Learning to Read*. AAAI Press, 2009.

[39] Tim Finin and Zareen Syed. Creating and exploiting a web of semantic data. In *InProceedings, Proceedings of the Second International Conference on Agents and Artificial Intelligence, January 2010*, January 2010.

[40] Jenny R. Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370, Morristown, NJ, USA, 2005. Association for Computational Linguistics.

[41] Michael Franklin. From databases to dataspaces: A new abstraction for information management. *SIGMOD Record*, 34:27–33, 2005.

[42] Evgeniy Gabrilovich and Shaul Markovitch. Overcoming the brittleness bottleneck using wikipedia: enhancing text categorization with encyclopedic knowledge. In *AAAI'06: proceedings of the 21st national conference on Artificial intelligence*, pages 1301–1306. AAAI Press, 2006.

[43] Evgeniy Gabrilovich and Shaul Markovitch. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *IJCAI'07: Proceedings of the 20th international joint conference on Artifical intelligence*, pages 1606–1611, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[44] Nikesh Garera and David Yarowsky. Structural, transitive and latent models for biographic fact extraction. In *EACL '09: Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 300–308, Morristown, NJ, USA, 2009. Association for Computational Linguistics.

[45] J. Ghosh and A. McCallum. *PhD Thesis: Learning for Information Extraction: From Named Entity Recognition and Disambiguation to Relation Extraction*. PhD thesis, 2007.

[46] Jim Giles. Internet encyclopaedias go head to head. *Nature*, 438(7070):900–901, December 2005.

[47] Google. *Google Squared*.

[48] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: an update. *SIGKDD Explorations*, 11(1):10–18, 2009.

[49] S. Harris and N. Gibbins. 3store: Efficient bulk RDF storage. In *Proceedings of the 1st International Workshop on Practical and Scalable Semantic Systems (PSSS'03)*, pages 1–20. Citeseer, 2003.

[50] A. Harth and S. Decker. Optimized index structures for querying rdf from the web. In *Proceedings of the 3rd Latin American Web Congress*, pages 71–80. Citeseer, 2005.

[51] Takaaki Hasegawa, Satoshi Sekine, and Ralph Grishman. Discovering relations among named entities from large corpora. In *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, pages 415+, Morristown, NJ, USA, 2004. Association for Computational Linguistics.

[52] E. Hatcher and O. Gospodnetic. Lucene in action, 2004.

[53] Todd Holloway, Miran Bozicevic, and Katy Börner. Analyzing and visualizing the semantic coverage of wikipedia and its authors: Research articles. *Complex.*, 12(3):30–40, January 2007.

[54] Meiqun Hu, Ee P. Lim, Aixin Sun, Hady W. Lauw, and Ba Q. Vuong. Measuring article quality in wikipedia: models and evaluation. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 243–252, New York, NY, USA, 2007. ACM.

[55] Jun'ichi Kazama and Kentaro Torisawa. Exploiting wikipedia as external knowledge for named entity recognition. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 698–707, 2007.

[56] D. Kinzler. Wikisense- mining the wiki. In *In Proceedings of Wikimania 2005. The First International Wikimedia Conference*. Wikimedia Foundation, 2005.

[57] Tomas Kliegr. Unsupervised entity classification with wikipedia and wordnet. In *Proceedings of the 2nd K-Space PhD Jamboree Workshop,TELECOM ParisTech, Paris, France*, 2008.

[58] A. Krizhanovsky. Synonym search in wikipedia: Synarcher. *Arxiv preprint cs/0606097*, 2006.

[59] Markus Krötzsch, Denny Vr, and Max Völkel. Semantic mediawiki. In *Proc. 5th International Semantic Web Conference (ISWC06*, volume 4273, pages 935–942, 2006.

[60] Antonia Kyriakopoulou and Theodore Kalamboukis. Text classification using clustering. In *In Proceedings of the ECML-PKDD Discovery Challenge Workshop*, 2006.

[61] Douglas B. Lenat and R. V. Guha. *Building Large Knowledge-Based Systems; Representation and Inference in the Cyc Project*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[62] A. Lozano-Tello and A. Gómez-Pérez. ONTOMETRIC: A Method to Choose the Appropriate Ontology. *Journal of Database Management*, 15(2), April 2004.

[63] Alexander Maedche and Steffen Staab. Measuring similarity between ontologies. In *EKAW '02: Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web*, pages 251–263, London, UK, 2002. Springer-Verlag.

[64] J. Mayfield, D. Alexander, B. Dorr, J. Eisner, T. Elsayed, T. Finin, C. Fink, M. Freedman, N. Garera, P. McNamee, and Others. Cross-Document Coreference Resolution: A Key Technology for Learning by Reading. In *AAAI 2009 Spring Symposium on Learning by Reading and Learning to Read*, 2009.

[65] Paul McNamee and Hoa T. Dang. Overview of the tac 2009 knowledge base population track. In *In Proceedings of the 2009 Text Analysis Conference*. National Institute of Standards and Technology, November 2009.

[66] R. Mihalcea. Using wikipedia for automatic word sense disambiguation. In *Proceedings of NAACL HLT*, volume 2007, 2007.

[67] Rada Mihalcea and Andras Csomai. Wikify!: linking documents to encyclopedic knowledge. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 233–242, New York, NY, USA, 2007. ACM.

[68] George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine Miller. Wordnet: An on-line lexical database. *International Journal of Lexicography*, 3:235–244, 1990.

[69] David Milne. Computing semantic relatedness using wikipedia link structure. In *Proc. of NZCSRSC'07*.

[70] David Milne and Ian H. Witten. Learning to link with wikipedia. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, pages 509–518, New York, NY, USA, 2008. ACM.

[71] E. Minack, L. Sauermann, G. Grimnes, C. Fluit, and J. Broekstra. The Sesame LuceneSail: RDF Queries with Full-text Search. *NEPOMUK Consortium, Technical Report*, 1, 2008.

[72] José E. Moreira, Maged M. Michael, Dilma Da Silva, Doron Shiloach, Parijat Dube, and Li Zhang. Scalability of the nutch search engine. In *ICS '07: Proceedings of the 21st annual international conference on Supercomputing*, pages 3–12, New York, NY, USA, 2007. ACM.

[73] A. B. MySQL. MySQL database server. *Internet WWW page, at URL:¡ http://www. mysql. com¿(last accessed 5/11/2003).*

[74] Nadeau, David, Sekine, and Satoshi. A survey of named entity recognition and classification. *Linguisticae Investigationes*, 30(1):3–26, January 2007.

[75] Kotaro Nakayama, Takahiro Hara, and Shojiro Nishio. Wikipedia mining for an association web thesaurus construction. In Boualem Benatallah, Fabio Casati, Dimitrios Georgakopoulos, Claudio Bartolini, Wasim Sadiq, and Claude Godart, editors, *Web Information Systems Engineering WISE 2007*, volume 4831 of *Lecture Notes in Computer Science*, chapter 27, pages 322–334. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[76] Dat P. T. Nguyen, Yutaka Matsuo, and Mitsuru Ishizuka. Subtree mining for relation extraction from wikipedia. In *NAACL '07: Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers on XX*, pages 125–128, Morristown, NJ, USA, 2007. Association for Computational Linguistics.

[77] Sergei Nirenburg, Stephen Beale, and Marjorie McShane. Evaluating the performance of the ontosem semantic analyzer. In Graeme Hirst and Sergei Nirenburg, editors, *ACL 2004: Second Workshop on Text Meaning and Interpretation*, pages 33–40, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[78] David Opitz and Richard Maclin. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, 11:169–198, 1999.

[79] Simone P. Ponzetto and Michael Strube. Exploiting semantic role labeling, wordnet and wikipedia for coreference resolution. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 192–199, Morristown, NJ, USA, 2006. Association for Computational Linguistics.

[80] Simone P. Ponzetto and Michael Strube. Deriving a large scale taxonomy from wikipedia. In *AAAI'07: Proceedings of the 22nd national conference on Artificial intelligence*, pages 1440–1445. AAAI Press, 2007.

[81] Simone P. Ponzetto and Michael Strube. Wikitaxonomy: A large scale knowledge resource. In *Proceeding of the 2008 conference on ECAI 2008*, pages 751–752, Amsterdam, The Netherlands, The Netherlands, 2008. IOS Press.

[82] E. Prud'Hommeaux, A. Seaborne, and Others. Sparql query language for rdf. *W3C working draft*, 4, 2006.

[83] A. Reggiori, D. W. van Gulik, and Z. Bjelogrlic. Indexing and retrieving Semantic Web resources: the RDFStore model. In *Workshop on Semantic Web Storage and Retrieval*, 2003.

[84] Tyler Riddle. Parse::mediawikidump.

urlhttp://search.cpan.org/~triddle/Parse-MediaWikiDump-0.40/, 2006.

[85] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):620, 1975.

[86] Peter Schönhofen. Identifying document topics using the wikipedia category network. *Web Intelli. and Agent Sys.*, 7(2):195–207, 2009.

[87] A. Seaborne. Rdql-a query language for rdf. *W3C Member submission*, 9:29–1, 2004.

[88] Bayle Shanks. Wikigateway: a library for interoperability and accelerated wiki development. In *WikiSym '05: Proceedings of the 2005 international symposium on Wikis*, pages 53–66, New York, NY, USA, 2005. ACM.

[89] Yusuke Shinyama and Satoshi Sekine. Preemptive information extraction using unrestricted relation discovery. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 304–311, Morristown, NJ, USA, 2006. Association for Computational Linguistics.

[90] Michael Strube and Simone P. Ponzetto. Wikirelate! computing semantic relatedness using wikipedia. In *AAAI'06: proceedings of the 21st national conference on Artificial intelligence*, pages 1419–1424. AAAI Press, 2006.

[91] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: A large ontology from wikipedia and wordnet. *Web Semant.*, 6(3):203–217, 2008.

[92] Ed Summers. Www:wikipedia.

urlhttp://search.cpan.org/~esummers/WWW-Wikipedia-1.9/, 2006.

[93] Z. Syed, T. Finin, and A. Joshi. Wikipedia as an ontology for describing documents. In *Proceedings of the Second International Conference on Weblogs and Social Media*. AAAI Press, 2008.

[94] Zareen Syed, Tim Finin, Varish Mulwad, and Anupam Joshi. Exploiting a web of semantic data for interpreting tables. In *Proceedings of the Second Web Science Conference*, April 2010.

[95] A. Toral and R. Munoz. A proposal to automatically build and maintain gazetteers for Named Entity Recognition by using Wikipedia. *EACL 2006*, 2006.

[96] C. Vanoirbeek. Formatting structured tables. In *In Proc. of Electronic Publishing'92, pp. 291*. Cambridge University Press, April 1992.

[97] Anne M. Vercoustre, James A. Thom, and Jovan Pehcevski. Entity ranking in wikipedia. In *SAC '08: Proceedings of the 2008 ACM symposium on Applied computing*, pages 1101–1106, New York, NY, USA, 2008. ACM.

[98] Jakob Voss. Measuring wikipedia. In *International Conference of the International Society for Scientometrics and Informetrics : 10th*. ISSI, July 2005.

[99] Gang Wang, Huajie Zhang, Haofen Wang, and Yong Yu. Enhancing relation extraction by eliciting selectional constraint features from wikipedia. In Zoubida Kedad, Nadira Lammari, Elisabeth Métais, Farid Meziane, and Yacine Rezgui, editors, *Natural Language Processing and Information Systems*, volume 4592 of *Lecture Notes in Computer Science*, chapter 29, pages 329–340. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[100] Rob Warren, Edo Airoldi, and David Banks. Network analysis of wikipedia.

[101] Yotaro Watanabe, Masayuki Asahara, and Yuji Matsumoto. A graph-based approach to named entity categorization in wikipedia using conditional random fields, 2007.

[102] Daniel S. Weld, Raphael Hoffmann, and Fei Wu. Using wikipedia to bootstrap open information extraction. *SIGMOD Rec.*, 37(4):62–68, 2008.

[103] Wikinews. Wikinews, the free news source, 2009.

[104] Wikipedia. Wikipedia, the free encyclopedia, 2008.

[105] D. Wood, P. Gearon, and T. Adams. Kowari: A platform for semantic web storage and analysis. In *Proceedings of the 14th International WWW Conference*. Citeseer, 2005.

[106] Fei Wu and Daniel S. Weld. Automatically refining the wikipedia infobox ontology. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 635–644, New York, NY, USA, 2008. ACM.

[107] Yulan Yan, Naoaki Okazaki, Yutaka Matsuo, Zhenglu Yang, and Mitsuru Ishizuka. Unsupervised relation extraction by mining wikipedia texts using information from the web. In *ACL-IJCNLP '09: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2*, pages 1021–1029, Morristown, NJ, USA, 2009. Association for Computational Linguistics.

[108] Jonathan Yu, James A. Thom, and Audrey Tam. Ontology evaluation using wikipedia categories for browsing. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 223–232, New York, NY, USA, 2007. ACM.

[109] Torsten Zesch, Christof Müller, and Iryna Gurevych. Extracting lexical semantic knowledge from wikipedia and wiktionary.

[110] V. Zlatic, M. Bozicevic, H. Stefancic, and M. Domazet. Wikipedias: Collaborative web-based encyclopedias as complex networks, Jul 2006.