

APPROVAL SHEET

Title of Thesis: User Interaction Framework for TABEL, a domain independent framework for Inferring Semantics of Tables.

Name of Candidate: Ratnadeep Gajanan Mangalvedhekar
Master of Science, 2016

Thesis and Abstract Approved: _____
Dr. Tim Finin
Professor
Department of Computer Science and Electrical
Engineering

Date Approved: _____

ABSTRACT

Title of the document: User Interaction Framework for TABEL, a domain independent framework for Inferring Semantics of Tables.

Ratnadeep Gajanan Mangalvedhekar, M. S., 2016

Directed by: Professor Dr. Tim Finin
Department of Computer Science and Electrical Engineering

Data, structured in tabular form is omnipresent through all communication, research and analysis. Tables appear in print media, handwritten notes, computer software, white papers, architectural enhancement, and on the Web. Although the Web offers millions of tables, their interpretation is rarely evident to machines from the table itself. A novel way of automatically inferring the intended meaning of tables and representing it as RDF (Resource Description Framework) linked data has been implemented previously, but without user involvement through the process of interpretation. In this work we describe a user feedback framework to enable user involvement through the process of interpretation. We describe the design and implementation of a User Interaction Model, which allows the user to interact with the aforementioned system as a web service. The UI permits users to view and modify the system-generated interpretations for column headers and cell values and provide feedback if it is incorrect. We include a usability evaluation of the user interaction model to illustrate the effectiveness of the proposed model and subsequently discuss its role in improving the quality of the generated interpretations and in eliminating occasional misinterpretations.

USER INTERACTION FRAMEWORK FOR TABEL, A DOMAIN
INDEPENDENT FRAMEWORK FOR INFERRING SEMANTICS OF TABLES

By
Ratnadeep Gajanan Mangalvedhekar

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, Baltimore County, in partial fulfillment
of the requirements for the degree of
Master of Science
2016

© Copyright by
Ratnadeep Gajanan Mangalvedhekar
2016

Table of Contents

Table of Contents.....	i
List of Figures.....	iii
List of Illustrations.....	iv
Introduction.....	1
Tables on the Web.....	1
Inferring semantics of a Table.....	1
Motivation.....	3
Contributions.....	5
Related Work.....	7
Karma.....	7
Haystack.....	8
mSpace.....	9
System Architecture.....	10
Monolithic Architecture.....	10
Motivation for using REST.....	11
RESTful Architecture.....	12
Data Interchange format.....	13
Query and Rank resource.....	16
Joint Inference resource.....	17
Search on knowledge base.....	19
Query Openlink VIRTUOSO SPARQL endpoint.....	22
User Interaction Framework.....	25
Representation of Input.....	25
Representation of Output.....	26
User Interaction Scenarios.....	28
Preprocessing.....	30
Candidate selection for Column Header and Cell value.....	31
Candidate Definition.....	33
Search Knowledge Base.....	35
Participants.....	38
Usability Metrics.....	39
Tasks.....	40
Results.....	41
Conclusion.....	46
Future Work.....	48

References.....	50
-----------------	----

List of Figures

Fig.1 - Monolithic TABEL architecture.....	18
Fig. 2 - RESTful architecture for TABEL resources.....	21
Fig. 3 - Sequence diagram for query-and-rank.....	25
Fig. 4 - Sequence diagram for joint-inference	26
Fig. 5 - Query knowledge base sequence diagram	28
Fig. 6 - Lucene query executes on the server to return a list of ‘n’ candidate.....	28
Fig. 7 - Openlink VIRTUOSO endpoint query sequence diagram.....	31
Fig. 8 - Representation of the Input table.....	34
Fig. 9 - Representation of the Output Table.....	36
Fig. 10 - Individual Components of the User Interaction Model.....	37
Fig. 11 - User interaction scenarios.....	39
Fig. 19 - Outline of the usability evaluation process.....	59
Fig. 20 - User Interview Questionnaire.....	61

List of Illustrations

Fig. 12 - An illustration demonstrating the process of selecting a subset of the table.....	38
Fig. 13 - An illustration demonstrating omission of unwanted/inconsequential columns.....	39
Fig. 14 - An illustration demonstrating the process of selection of a candidate for Column header from a list of annotation.....	40
Fig. 15 - An illustration demonstrating the process of selection of a candidate from a list of annotations for cell value.....	41
Fig. 16 - An illustration showing a popover for Definition/Comment obtained from VIRTUOSO SPARQL query endpoint	42
Fig. 17 - An illustration demonstrating the process of querying VIRTUOSO for a description for the entity (Alan Turing) in question.....	43
Fig. 18 - An illustration demonstration the process of search for a specific candidate from the underlying knowledge base and adding it to list of candidates.....	45

Introduction

Tables on the Web

Today, the World Wide Web has become the a primary source of knowledge and information in the last two decades, replacing traditional encyclopedias and reference books. Although, most pages are primary text is written in a narrative form such as news stories, blogs, reports, etc., significant amounts of information is also encoded in structured forms such as tables embedded in pages and documents. One estimate suggests that the Web contains over 150 million high quality relational html tables (Cafarella et al. 2008). Tables are prevalent throughout research activities, analysis of data and documentations. Context of the data drives the nomenclature for defining tables. Further, they differ in type, form, flexibility, annotation, depiction and use. Tables are also a key tool in outlining important data and results in documents.

However, current available text processing systems do not work well for tables. Tables lack the grammatical context from the adjoining text. Regular text processing systems rely on this context to infer the meaning of set of words. The concise nature of table, which enables easy absorption of information for humans, renders it difficult for the machine to understand.

Inferring semantics of a Table

Preliminary research in the area of processing focused mainly on developing novel ways of extraction of tables from documents and web pages (Hurst 2006; Embley, Lopresti,

& Nagy 2006) with more recent research attempting to interpret their semantics. Existing work in table interpretation either partially infer the semantics (Venetis et al. 2011; Wang et al. 2012) based on what application is built on top or only focus on a particular domain such as the Web (Limaye, Sarawagi, & Chakrabarti 2010). TABEL, is a domain independent and extensible framework for inferring the semantics of tables and representing it as RDF Linked Data (Mulvad, Finin & Joshi 2013).

The following phases play a vital role in inferring the semantics of tables -

1. Preprocessing Phase - Input table first goes through a preprocessing phase which includes modules to handle a number of pragmatic issues such as acronym recognizing and expanding acronyms, stylized literal values, recognizing commonly encoded data such as addresses, telephone numbers, zip codes, etc.
2. Query and Rank Phase - The table is then processed by the Query and Rank module which queries background Linked Data sources to generate an initial ranked lists of candidate assignments for headers, data cells and relations between headers. The system uses Linked Data sources or knowledge bases for generating candidates. The knowledge bases can be adapted and changed based on the domain of the table.
3. Joint Inference - Once candidate assignments are generated, the joint inference module simultaneously infers the semantics of headers, data cells and relations between headers by representing a table as a probabilistic graphical model to capture correlation between subparts of a table and performing inference over the model. After the semantics are inferred, RDF Linked Data triples are generated.

A set of possible human intervention scenarios, “human in the loop”, in each of the above phases is described in TABEL and a brief account of the possible consequences of such an intervention is also provided. This provides us with the foundational requirement to design and develop a user interaction model, which is discussed in the subsequent sections of this document.

Motivation

Today, many areas are in demand of sophisticated user interaction models and visualization techniques, and applications based on Semantic Web are not an exception. As size and complexity of Ontologies and Linked Data in the Semantic Web constantly grow, so does the number of users from diverse backgrounds and application areas. Providing users with an intuitive user experience can significantly aid the understanding of the domains and knowledge represented by ontologies and Linked Data. There is no one-size-fits-all solution and different use cases demand different interaction models and visualization techniques. Ultimately, providing better user interaction models and visual representations will nurture user engagement and eventually lead to higher quality results in different applications employing ontologies and to the proliferation of Linked Data usage.

As ontologies grow in size and complexity, the demand for comprehensive visualization and user interaction framework also rises. In particular, user interfaces are an integral part of ontology engineering, to help bridge the gap between domain experts and ontology engineers. Ontology visualization is not a new topic and a number of approaches have become available in recent years, with some being already well-established, particularly in the field of ontology modeling. In other areas of ontology engineering, such as ontology alignment and debugging, although several tools have recently been developed, few provide a

graphical user interface, not to mention navigational aids or comprehensive visualization techniques.

Ontology engineers usually possess domain and knowledge representation expertise necessary to deal with the complex abstract concepts of large-scale ontologies. This is not necessarily the case with potential consumers of applications built around the core concept of Linked Data. They usually can come from extremely diverse backgrounds and have varying levels of expertise. Since “Semantic Web” is in early stages of adoption, currently, the main Linked Data consumers are technology-experienced users, especially from the research community. One of the main reasons for the lack of mainstream adoption is the absence of sophisticated user interaction models and visualizations techniques that are needed to assist various kinds of users, who pursue diverse goals and pose individual requirements. In the presence of a huge network of interconnected resources, one of the challenges faced by the Linked Data community is the visualization of the multidimensional datasets to provide for efficient overview, exploration and querying tasks, to mention just a few. With the focus shifting from a Web of Documents to a Web of Data, changes in the interaction paradigms are in demand as well. Novel approaches also need to take into consideration the technological challenges and opportunities given by new interaction contexts, ranging from mobile and touch interaction to visualizations on large displays, and encompassing highly responsive web applications.

It can be argued that, although the semantic web has been developed for an automated environment (machine based), eventually humans are its final intended consumers. Therefore, critical aspects of such a user interaction models are

1. They should be capable of making all the richness of the underlying data models available to the end user at the interaction level, with minimum constraints.

2. Such interfaces, built to drive user interaction, should be flexible enough to offer to users different ways of interacting with their data and this flexibility should not be at the cost of user's cognitive burden.

In following chapters, we present one such user interaction model which allows the user to interact with TABEL (A domain independent and Extensible Framework for inferring semantics of Tables) as it disambiguates an input table by utilizing the underlying Linked Data.

Contributions

In this thesis we present a generic, web based user interaction model developed with the objective of enabling “Human in the loop” paradigm proposed in TABEL. To the best of our knowledge, we are the first to propose such a user interaction model for TABEL.

Features -

1. The user interaction model provides a feedback framework enabling users to guide the system (TABEL) as it attempts to infer the semantics. We also discuss the changes/enhancements made to the existing TABEL architecture enabling it to support the development of this user interaction model.
1. This feedback framework allows the users to interact with the system during the following phases
 - a. During the preprocessing phase
 - b. Before the inference phase, during the query and rank phase
 - c. During the inference phase, after each iteration of the semantic message passing

d. after the inference phase

1. The user is provided with the ability to query underlying Linked Data Knowledge Base used by TABEL during the inference in-order to replace/correct any erroneous assignments.
2. The model also provides the users with the ability to view the definition entities by executing SPARQL queries over HTTP against the dbpedia Virtuoso SPARQL query endpoint.

Finally, we conclude by providing a detailed usability evaluation of the proposed system. We carry out the evaluation by observing participants interact with the system and gathering subjective input from the participants through interviews, surveys and feedback.

Related Work

Underlying goal of any semantic web based application is to make the web more usable for its users to facilitate their primary activities of information retrieval, information management, and information presentation. User interfaces, through which users gain access to semantic information, is a key in achieving this goal. In order to take advantage of the added structure and semantics to the content, we need extended user interaction experience through innovative user interfaces.

Several attempts have been made to implement a viable user interaction model on semantic web applications. Various tools and applications have been developed to assist users create, manipulate, retrieve, present, organize, and manage semantic information. These tools are mainly designed to aid content creators create semantic information through different design activities like designing or visualizing ontologies, creating RDF files for resources, searching semantic data [Swoogle], and creating semantic metadata. On the other hand applications, leveraging the power of semantic web, are focused towards helping users perform activities such as information retrieval and management. Applications that fall into this category are mainly, Karma, Haystack, mSpace, Gnowsis, Fenfire, etc.

Karma

Karma (Knoblock et al. 2012; Szekely et al. 2013; Knoblock et al. 2013) is an information integration tool that enables users to quickly and easily integrate data from a variety of data sources including databases, spreadsheets, delimited text files, XML, JSON, KML and Web APIs. Users integrate information by modeling it according to an ontology of

their choice using a graphical user interface that automates much of the process. Karma learns to recognize the mapping of data to ontology classes and then uses the ontology to propose a model that ties together these classes. Users then interact with the system to adjust the automatically generated model. During this process, users can transform the data as needed to normalize data expressed in different formats and to restructure it. Once the model is complete, users can published the integrated data as RDF or store it in a database.

Haystack

Haystack seeks to apply semantic web technologies to personal information management. It allows individuals to manage their information in the ways that makes the most sense to them. Haystack exhibits three novel functionalities to facilitate users in their information management tasks.

1. It incorporates and exposes all types of information in a single, coherent manner. It provides a single, uniform interface for viewing and organizing of e-mail, instant messages, contact information, web pages, documents, news, music, images, blog feeds, etc.
2. In haystack every entity whether it is a simple text in an email message or an email message itself is considered as information object. Any of these information objects can be right clicked for its context menu, allowing immediate access to all the operations that make sense for that object. The interface also lets users define their own information objects to incorporate any non-standard types of information. Users can readily define attributes of these new objects that help them categorize and retrieve information, and add new relationships to objects. An information object or

operation can also be downloaded from outside applications that will be immediately available to use.

3. Haystack gives users flexibility to modify standard as well as user defined information objects irrespective of its type and application it belongs to.

mSpace

mSpace is a semantic web application developed at School of Electronics and Computer Science (ECS) at the University of Southampton to facilitate information access, browsing, and organization given that the user has limited domain knowledge. This is achieved through exploring various relationships in information through semantic web technologies, and allowing users to manipulate information categorization to suit their interests. Researchers have developed a beta version, called mSpace classical music browser, to access and browse music information, but the framework can also be applied to any type of information.

System Architecture

Monolithic Architecture

In the baseline implementation of TABEL, the input table goes through a series of modules sequentially, providing very little or no scope for user interaction as shown in the figure below. This essentially limits our ability to develop a user interaction model to implement the “human in the loop” paradigm proposed by in the TABEL literature.

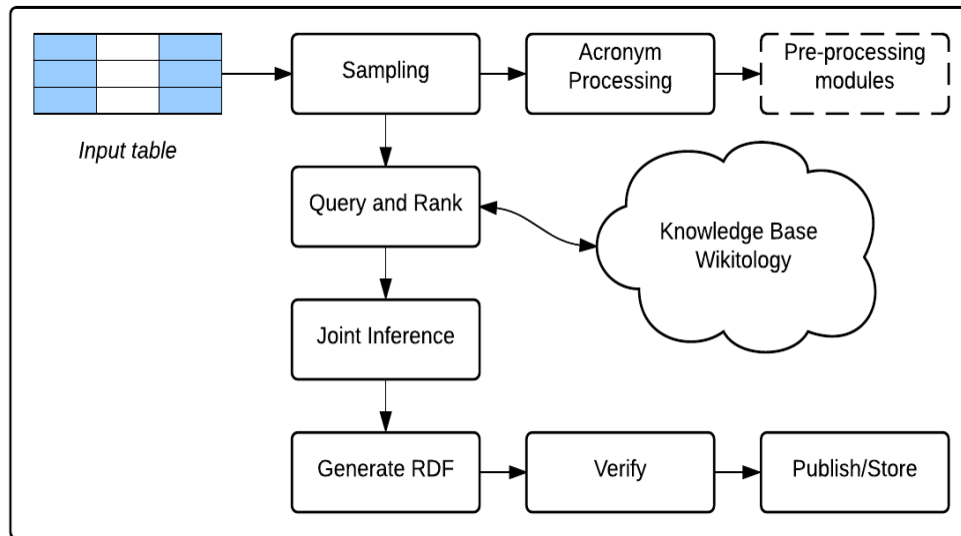


Fig. 1 - Monolithic TABEL architecture

It supports user interactions at the following phases :

1. During the input phase, when the user inputs the table to be processed
2. After the entire process of disambiguation ends, when the user can inspect the results generated by the system.

To overcome these limitations, we design a REST based architecture, which allows us to create the necessary resources to enable the user interaction throughout the course of generating the semantics. Each of the TABEL algorithms, like query-and-rank, and joint-inference are developed into REST based resources accessible over HTTP. In order to further simplify the interaction between the user interaction model and the RESTful web-services, we have designed and developed a JSON based custom data interchange object.

Motivation for using REST

One of the most important REST principles for Web applications is for interaction between the client and server to be stateless. Each request from the client to the server must contain all of the information necessary to understand the request. The client wouldn't notice if the server were to be restarted at any point between the requests. Additionally, stateless requests are free to be answered by any available server, which is appropriate for an environment such as cloud computing. In addition to this, the client has the ability to cache the data to improve performance.

On the server side, the application state and functionality are divided into multiple resources. A resource is usually item of interest, a conceptual identity that is exposed to the clients. In our case, include application objects, database records, algorithms, and so on. All resources share a uniform interface for the transfer of state between client and server, using standard HTTP methods such as GET, PUT, POST, and DELETE. We primary use POST for processing and GET for querying resources. Hypermedia is the engine of the application state, and resource representations are interconnected by hyperlinks.

Another important REST principle, guiding our design design, is the layered system, which means a component cannot see beyond the immediate layer with which it is interacting. By restricting knowledge of the system to a single layer, a boundary is placed on the overall system complexity, promoting substrate independence.

We focus on the application of RESTful architectural constraints to our overall design, enabling it to scale well to a large number of clients. It also reduces interaction latency between clients and servers, which is essential to provide users with a desirable experience. The uniform interface, as described in the next sections, simplifies the overall system architecture and improves the visibility of the interactions between subsystems.

In the subsequent sections, we describe the creation of a layered system for TABEL, where the core functionality is operationally split into multiple resources and is made accessible over HTTP. We also describe a custom data interchange format developed to support the interaction between client and server.

RESTful Architecture

In this section, we describe the RESTful architecture design for TABEL, enabling the creation of a user interaction model. As shown the figure below, the core components of our architecture are -

1. Query and Rank resource
2. Joint Inference resource
3. Knowledge Base search resource
4. Definition resource
5. Knowledge base

6. Supporting Database
7. Openlink VIRTUOSO SPARQL query service

The query-and-rank resource, joint-inference resource, search-knowledge-base resource and the definition resource are deployed independently as their respective, self-contained resource units accessible over HTTP. These components provide the core functional elements required by the system to infer the semantics of the table.

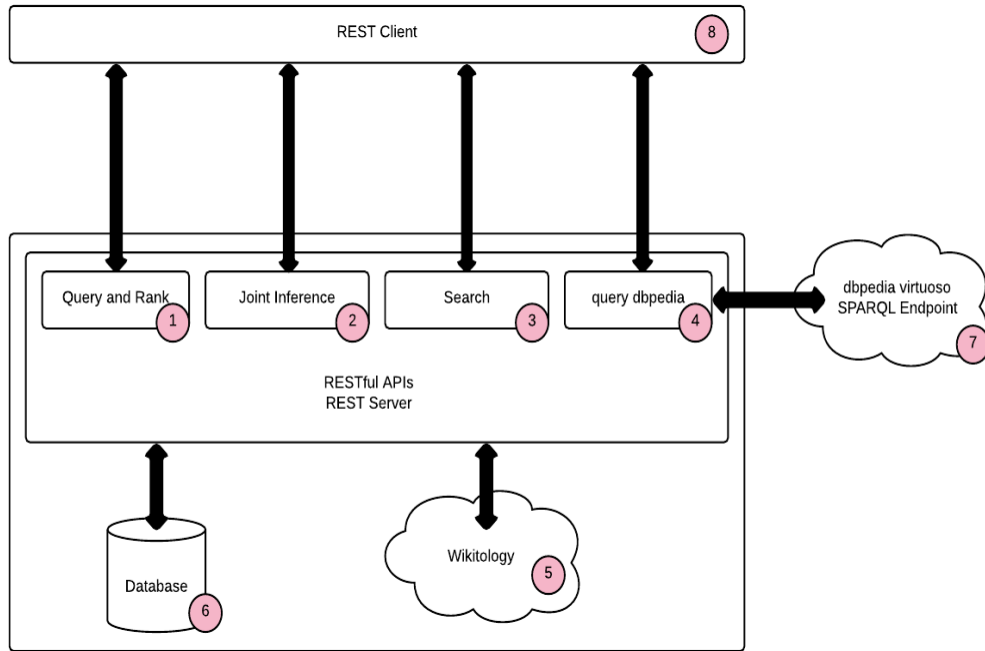


Fig. 2 - RESTful architecture for TABEL resources

Data Interchange format

We design a single, custom JSON based data interchange format to represent the input and the output table. The table is made up of column-header-objects and cell-value-objects as shown below:

Table Object

```
{
  "column-headers":
  [{"column-header-object"},
  {"column-header-object"},...,
  {"column-header-object"}],

  "cell-values":[
  [{"cell-value-object"},
  {"cell-value-object"},...,
  {"cell-value-object"}],
  [{"cell-value-object"},
  {"cell-value-object"},...,
  {"cell-value-object"}],
  .
  .
  [{"cell-value-object"},
  {"cell-value-object"},...,
  {"cell-value-object"}]]
}
```

Column Header Object

```
{
  "input":"","
  "current":"","
  "candidates":[],
  "dbpedia-top":"","
  "yago-top":"","
  "annotations-dbpedia":[],
  "annotations-yago":[]
}
```

Each column-header-object is made of the following data members -

1. input - This is the initial value of the column header from the input phase.
2. current - This is the current value assigned to the column header by a user operation.
3. candidates - This is the list of n candidates generated by the query-and-rank resource.

4. dbpedia-top - This is the most relevant annotation for the current column header from the dbpedia knowledge base as generated by each iteration of joint-inference resource.
5. yago-top - This is the most relevant annotation for the current column header from the yago knowledge base as generated by each iteration of joint-inference resource.
6. annotations-dbpedia - This is a list of other relevant annotations from dbpedia, generated by the joint-inference resource and ordered according to their relevance.
7. annotations-yago - This is a list of other relevant annotations from yago, generated by the joint-inference resource and ordered according to their relevance.

Cell Value Object

```
{  
  "input": "",  
  "current": "",  
  "candidates": [],  
  "annotation-top": "",  
  "redirects": [],  
  "annotations": []  
}
```

Each column-header-object is made of the following data members -

1. input - This is the initial value of the column header from the input phase.
2. current - This is the current value assigned to the column header by a user operation.
3. candidates - This is the list of n candidates generated by the query-and-rank resource.

4. annotation-top - This is the most relevant annotation for the current cell value from the knowledge base as generated by each iteration of joint-inference resource.
5. annotations - This is a list of other relevant annotations for the current cell value, as generated by the joint-inference resource and ordered according to their relevance.
8. annotations-yago - This is a list of other relevant annotations from yago, generated by the joint-inference resource and ordered according to their relevance.

Query and Rank resource

The Query and Rank module generates an initial set of candidate assignments for column headers and data cells using appropriate underlying Knowledge Base (KB). The KB used here depends on the domain of the table. In the current implementation we use Wikitology, a hybrid KB as our underlying KB as it provides excellent coverage for general purpose topics such as places, organizations, music, movies, politics, and sports. These can be complemented or replaced with domain specific ones; for example SNOMED CT and UMLS can be used as a compliment or replacement in the case of medical tables.

This web-service is designed to accept the table to be processed as the input and return an initial ranked list of candidate assignments for headers, data cells and relations between headers. The system generates this list by querying the background Linked Data sources. This gives the user the ability to select a specific candidate.

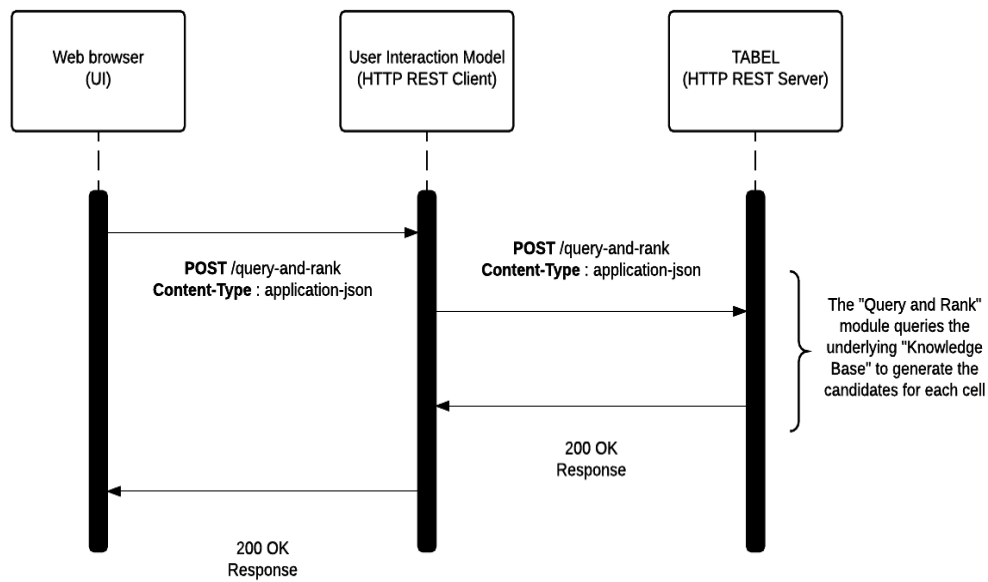


Fig. 3 - Sequence diagram for query-and-rank

Resource URL

<http://tabel.localhost:8080/query-and-rank>

Resource Information

Response Format : JSON

Parameters

table (required) : Input table in the custom JSON format described in the previous section in this chapter.

Joint Inference resource

This web-service is designed to accept the output of the previous query and rank module, with or without user inputs. The joint inference module simultaneously infers the semantics of headers, and data cells by representing a table as a probabilistic graphical model to capture correlation between sub-parts of a table and performing inference over the model.

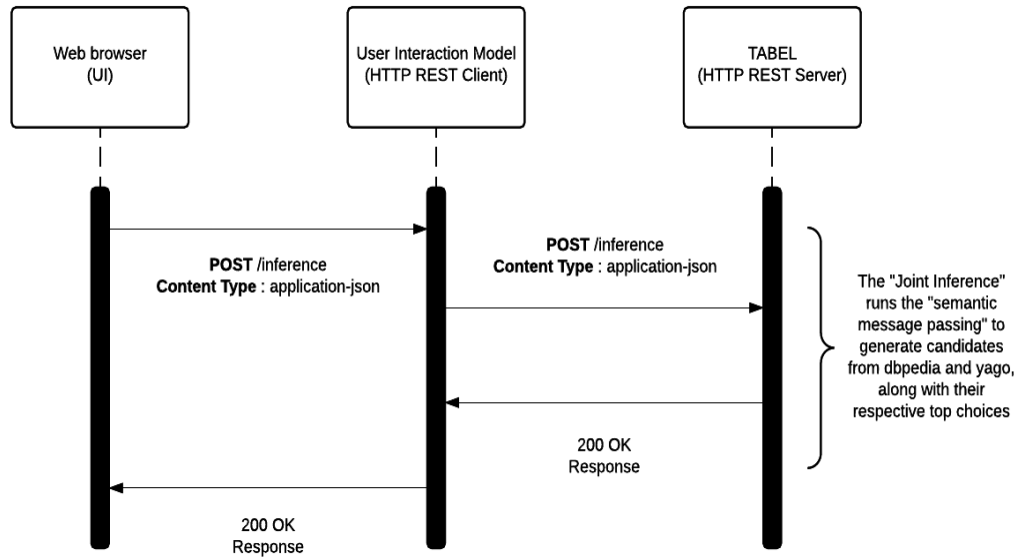


Fig. 4 - Sequence diagram for joint-inference

Resource URL

<http://tabel.localhost:8080/joint-inference>

Resource Information

Response Format : JSON

Parameters

- table-with-candidates (required) : Table with candidates generated from the query-and-rank phase with or without any user modification/reassignments

In addition to the core web services, the system also supports to two additional web-services

1. Search on knowledge base
2. Query Openlink VIRTUOSO SPARQL endpoint

Search on knowledge base

This web-service is designed to accept a user entered search query and returns a list of candidates matching the user input. This list of candidates is ordered based on their relevance to the user input. Idea this resource should be capable of querying any underlying knowledge base, but for the purpose of this thesis, it is designed to work over the Wikitology index already available with the initial implementation of TABEL.

The user can access the search utility within the candidates dropdown for all the column headers and cell values. A detailed illustration of this is provided in the next chapter. As the user begins entering the query string, a HTTP GET call is initialed with each keystroke, narrowing down the results with each subsequent entry. The other parameters of this query are column header, and the values of other cells in the same row. On the server this translates into a lucene query that is described in the figure below, and the sequence of events are shown the following sequence diagram.

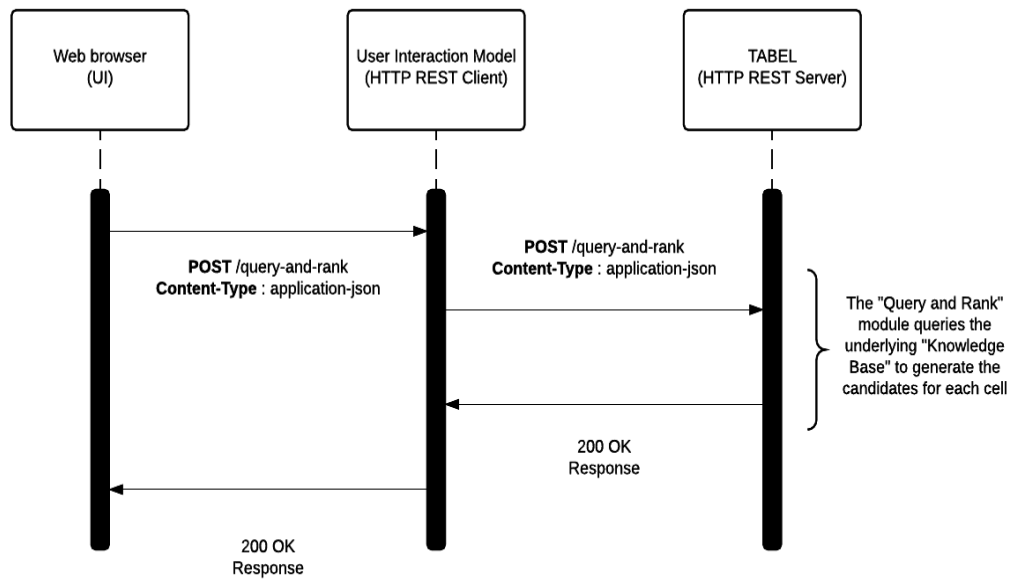


Fig. 5 - query knowledge base sequence diagram

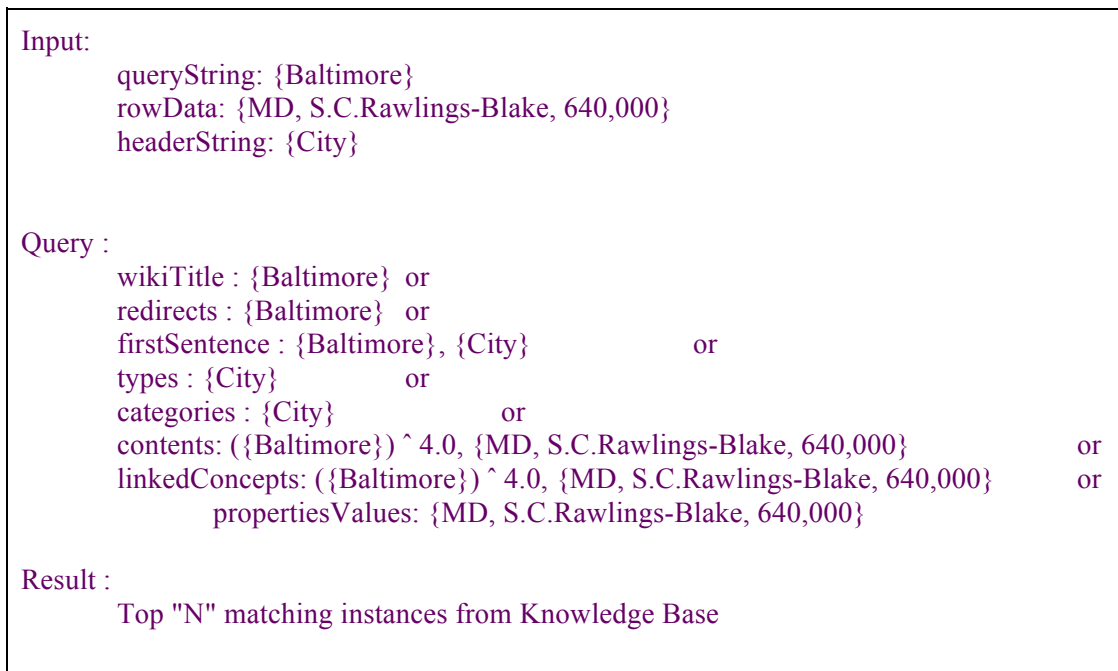


Fig. 6 - Lucene query executes on the server to return a list of 'n' candidates

Resource URL

<http://tabel.localhost:8080/query-knowledge-base>

Resource Information

Response Format : JSON

Parameters

- query (required): User inputted query string
- rowData (required): Other row values in the same row as the cell for which this query is being performed
- Header (required) : Column header of the cell for which the query is being performed

Example Request

GET

<http://tabel.localhost:8080/search-knowledge>

<base?query=sample&rowData=cell1,cell2,cell3&header=columnHeader>

Example Response

```
{"candidates" :  
["candidate", "candidate", "candidate", ..., "candidate"]}
```

Query Openlink VIRTUOSO SPARQL endpoint

The Virtuoso SPARQL query web service was initially developed with the goal of implementing the SPARQL Protocol for RDF (W3C Recommendation, January, 15 2008). It has been updated to support SPARQL 1.1, providing SPARQL query-processing for RDF data available on the open Internet. The query service implementation extends the standard protocol by providing multiple output-formats alongside the standard XML results serialization.

Our query web-service is designed to create an access point to leverage the OpenLink Virtuoso SPARQL query resource. The SPARQL query endpoint for the DBpedia data-set is publically accessible at <http://dbpedia.org/sparq>. This web-service allows the user to query the OpenLink Virtuoso SPARQL protocol endpoint with the candidate and obtain a definition/description about the candidate, helping the user make a more informed choice.

When the user clicks on the definition link, the javascript running inside the web browser initiates a HTTP GET request with the value currently selected as the mandatory query param. This GET request is then received by the HTTP Client, running the User interaction model, which inturn re-interprets it as a POST request which is then forwarded to the OpenLink's VIRTUOSO endpoint at <http://dbpedia.org/sparq>. with the appropriate headers and parameters. The VIRTUOSO endpoint, then, processes the requests and responded with the required data. This response it then forwarded onto the user's web-browser by the User Interaction Model (HTTP client). This sequence of events is shown in the sequence diagram below.

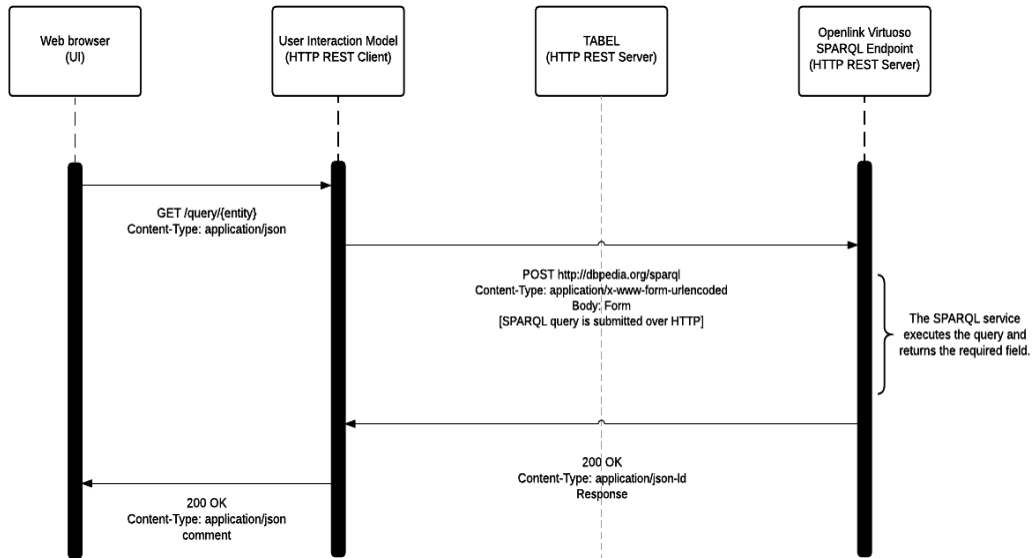


Fig. 7 - Openlink VIRTUOSO endpoint query sequence diagram

Resource URL

<http://tabel.localhost:8080/query-virtuoso>

Resource Information

Response Format : JSON

Parameters

- query (required) : Column header or cell value for which definition is sought.

Example Request

GET

http://tabel.localhost:8080/query-virtuoso?query=Alan_Turning

Example Response

```
{  
  "definition" : "Alan Mathison Turing OBE FRS (/ˈtʃʊərɪŋ/; 23 June 1912 – 7 June  
    1954) was a pioneering English computer scientist, mathematician, logician,  
    cryptanalyst and theoretical biologist. He was highly influential in the development  
    of theoretical computer science, providing a formalisation of the concepts of  
    algorithm and computation with the Turing machine, which can be considered a  
    model of a general purpose computer. Turing is widely considered to be the father of  
    theoretical computer science and artificial intelligence."  
}
```

In the next chapter we describe the design and implementation of user interaction model built to leverage the RESTful APIs described in this section.

User Interaction Framework

In this chapter we describe the User Interaction Model build to leverage the APIs described in the previous chapter. Although TABEL, like other semantic web systems, has been developed for a machine based automated environment, eventually human beings are the intended consumers. Therefore one critical objective of developing the UIM is to make all the richness of the underlying data model and to make it available at the interaction level, with minimal constraints and a relatively short learning curve.

In the subsequent sections of this chapter, we describe the design and implementation of the various features developed as a part of this user interaction model.

Representation of Input

A table allows a form of generalization of information from an unlimited number of different social and scientific contexts. It provides a familiar way to convey information that otherwise might not be obvious or readily understood. Since TABEL was primarily developed to process data encoded in tables, we have designed all the necessary interactive components around the traditional, row and column based table structure.

City	State	Abbreviation	Office Holder
Baltimore	Maryland	MD	S. C. Rawling Blake
San Francisco	California	CA	Edwin Lee
New York	New York State	NY	M Bloomberg
Boston	Massachusetts	MS	Jim Kennedy
Philadelphia	Pennsylvania	PA	Ed Murray
Seattle	Washington	WS	Marty Walse
Denver	Colorado	CO	Michael Hancock

Fig. 8 - Representation of the Input table

Representation of Output

Each single valued cell and column header is replaced with a drop down (figure) containing annotations and candidates generated by the system at the end of query and rank phase and joint inference phase. They also include a link (figure) to Openlink Virtuoso SPARQL endpoint, which enables the users to view the definitions of the candidate, which in-turn empowers the users to make a better choice. These detailed are shown the subsequent figures.

City ▼ 🔗	State ▼ 🔗	Abbreviation ▼ 🔗	Office Holder ▼ 🔗
Baltimore ▼ 🔗	Maryland ▼ 🔗	MD ▼ 🔗	S. C. Rawlings Blake ▼ 🔗
San Francisco ▼ 🔗	California ▼ 🔗	CA ▼ 🔗	Edwin Lee ▼ 🔗
New York ▼ 🔗	New York State ▼ 🔗	NY ▼ 🔗	M. Bloomberg ▼ 🔗
Boston ▼ 🔗	Massachusetts ▼ 🔗	MS ▼ 🔗	Jim Kennedy ▼ 🔗
Philadelphia ▼ 🔗	Pennsylvania ▼ 🔗	PA ▼ 🔗	Ed Murray ▼ 🔗
Seattle ▼ 🔗	Washington ▼ 🔗	WS ▼ 🔗	Marty Walse ▼ 🔗
Denver ▼ 🔗	Colorado ▼ 🔗	CO ▼ 🔗	Michael Hancock ▼ 🔗

Fig. 9 - Representation of the Output Table

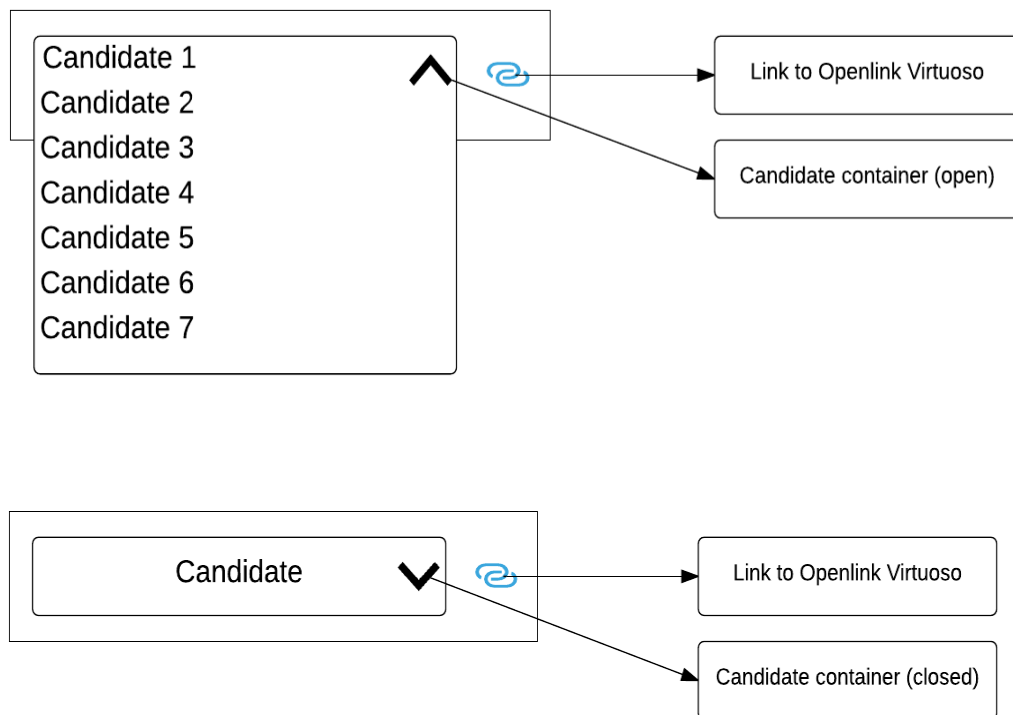


Fig. 10 - Individual Components of the User Interaction Model

User Interaction Scenarios

The scope for user interaction can be categories broadly into 3 main categories, mainly, before processing, during processing, and after convergence. During the preprocessing phase, the user has the ability to sample the input table before submission, by dragging and selecting a smaller subsection of the table. This is particularly helpful in an event where the user is dealing with larger data-tables.

During the processing phase, the user has the ability to interact with the process's query-and-rank module, by POSTing user feedback to the resource at /query-and-rank. The user can rerun the query-and-rank algorithm until he is satisfied with the result. Next, the user has the ability to submit the output of query-and-rank to joint-inference module, which is accessible at /joint-inference. The joint-inference module, tries infer the semantics by using a process called "semantic message passing." The module runs the semantic message passing process multiple times, and after each iteration, the inferred semantics are accessible to the user for review and feedback. At this juncture, the user can either review the results and stop to change the candidate assignments and re-run joint-inference in the quest to achieve better convergence.

Once the process of inference comes to an end, the user can make changes to the final result by querying the underlying knowledge base. The interaction scenarios described above are shown in the Figure 11 below.

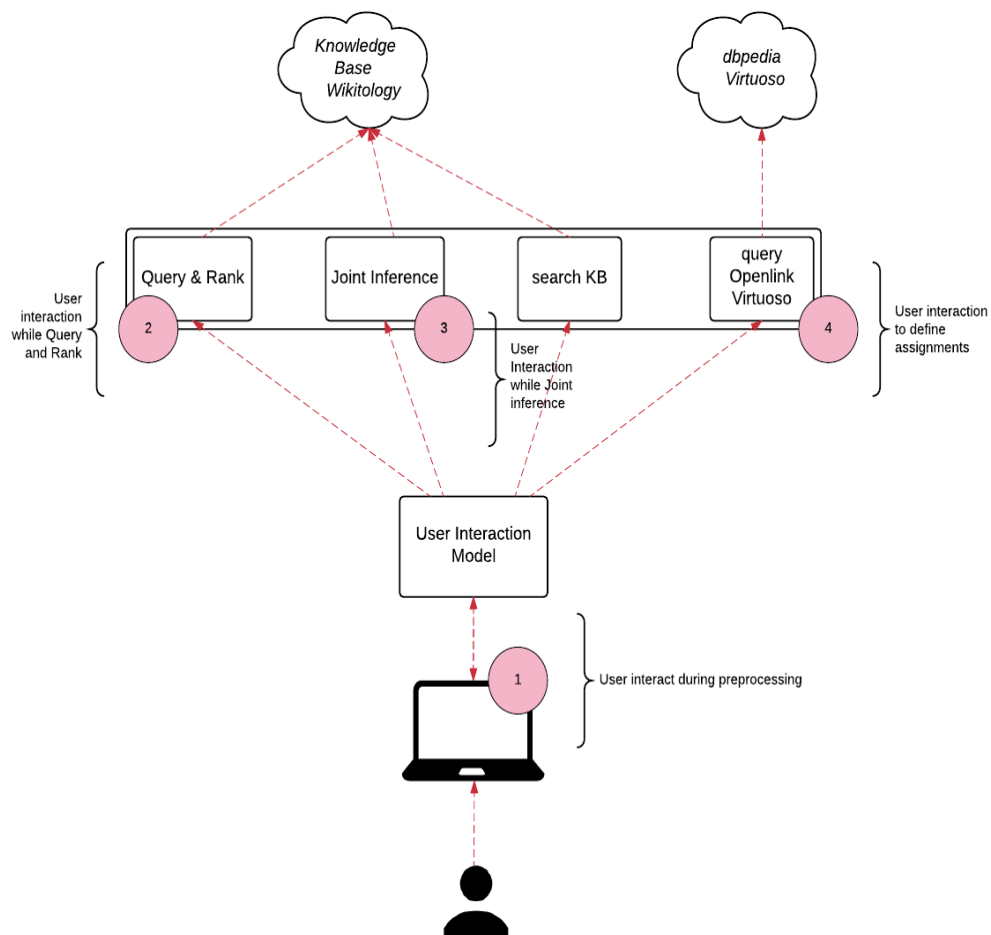


Fig. 11 - User interaction scenarios

Preprocessing

In the preprocessing phase, the user interaction model gives the user, the ability to ‘drag and select’ a relevant section of the input table for processing. This is illustrated in the figure below where the user had selected only the first 3 rows for processing along with the column headers. Also, note that the ‘Population’ column is automatically ignored since it was not a part of user selection. This feature is important because it enable to process larger tables in relatively short periods of time

City	State	Abbreviation	Office Holder	Population
Baltimore	Maryland	MD	S. C. Rawlings Blake	620,961
San Francisco	California	CA	Edwin Lee	852,469
New York	New York State	NY	M. Bloomberg	8,491,079
Boston	Massachusetts	MS	Mary Walse	655,884
Philadelphia	Pennsylvania	PA	Jim Kennon	1,560,297
Seattle	Washington	WA		
Denver	Colorado	CO		

City	State	Abbreviation	Office Holder
Baltimore	Maryland	MD	S. C. Rawlings Blake
San Francisco	California	CA	Edwin Lee
New York	New York State	NY	M. Bloomberg

Fig. 12 - An illustration demonstrating the process of selecting a subset of the table

Based to the input table, the user can omit inconsequential columns as illustrated in the figure below, where the user has chosen to omit column ‘Sl. No.’ and column ‘Population’ since these constants don’t have any visible impact on the final result. This

feature is important important because, it optimizes the processing by enabling the query-and-rank and joint-inference resources to focus on important portion/relevance of the table.

Sl. No.	City	State	Abbreviation	Office Holder	Population
1	Baltimore	Maryland	MD	S. C. Rawlings Blake	620,961
2	San Francisco	California	CA	Edwin Lee	852,469
3	New York	New York State	NY	M. Bloomberg	8,491,079
4	Boston	Massachusetts	MS	Marty Walse	655,884
5	Philadelphia				
6	Seattle				
7	Denver				

City	State	Abbreviation	Office Holder
Baltimore	Maryland	MD	S. C. Rawlings Blake
San Francisco	California	CA	Edwin Lee
New York	New York State	NY	M. Bloomberg
Boston	Massachusetts	MS	Marty Walse
Philadelphia	Pennsylvania	PA	Jim Kennedy
Seattle	Washington	WS	Ed Murray
Denver	Colorado	CO	Michael Hancock

Fig. 13 - An illustration demonstrating omission of unwanted/inconsequential columns

Candidate selection for Column Header and Cell value

The illustrations shown in this section demonstrate the selection of a candidate for a given column header (figure) and a cell value (figure). The user can click on the ‘open arrow’ to expose the list of available candidates. Since, the system generates relevant classes from dbpedia and yago, we have maintained separators in the list to enable to user to visually identify the two different sets. Once the user has reviewed all the available options and made a decision, he can then ‘click’ on the desired candidate to update the current assignment. The user has the option of changing the assignments for any number of data cells and column headers.

This interaction of selecting an assignment for a cell or a column header, that is different from the initial assignment gives users the unique ability of guiding the system through to better inferred semantics. These interactions are shown in the subsequent illustrations.

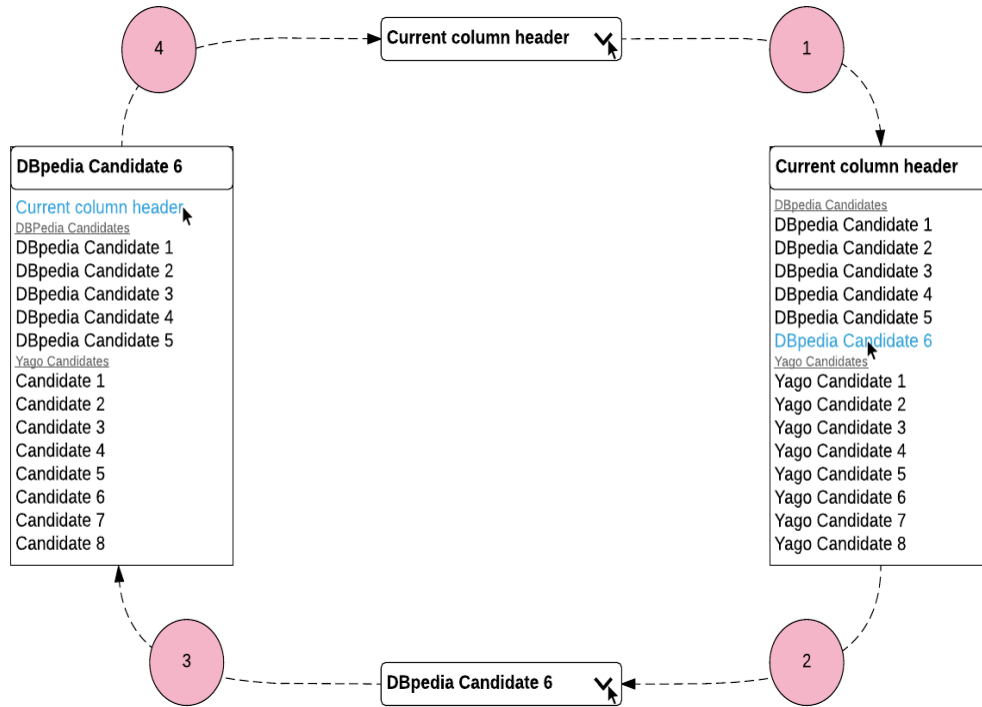


Fig. 14 - An illustration demonstrating the process of selection of a candidate for Column header from a list of annotations

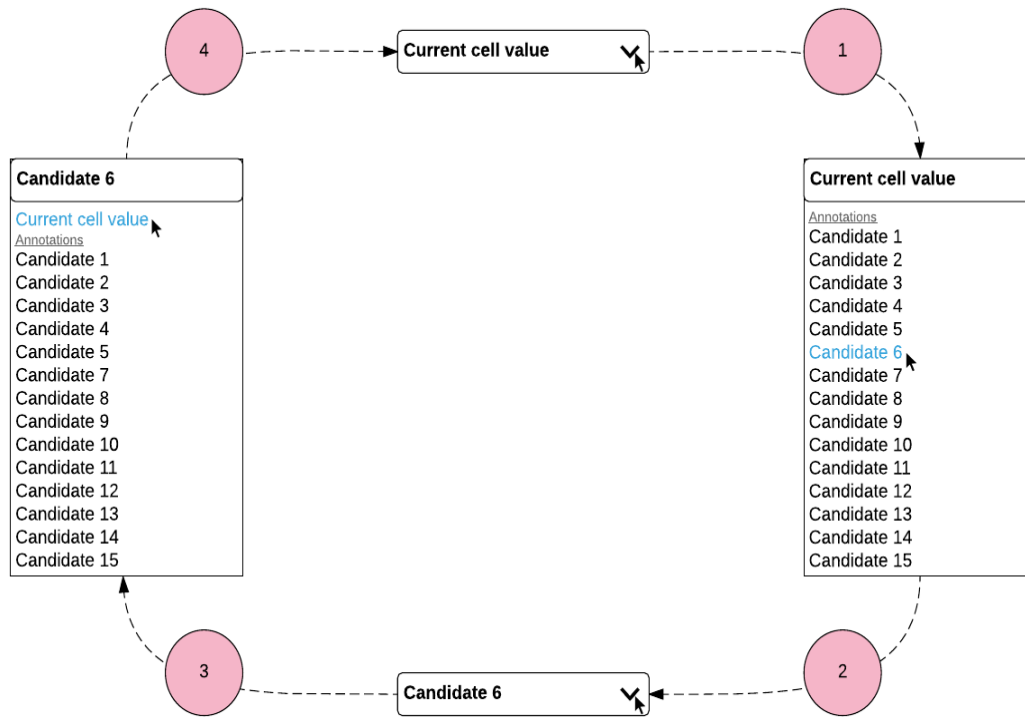


Fig. 15 - An illustration demonstrating the process of selection of a candidate from a list of annotations for cell value.

Candidate Definition

This interaction was designed with the objective of user assistance and empowerment, by providing them with the option of making an informed decision when they decide to change the system generated assign. We provide a link to Openlink VIRTUOSO SPARQL query end-points for all the candidates generated by the system.

When the user clicks on 'link' for any specific cell, a SPARQL query is run on the VIRTUOSO endpoint via HTTP. This query return the 'comment' section of the dbpedia

resource, which describes the candidate in question. At present, we query to obtain the ‘comment’, but this feature can be extended to enable a deeper integration between the two systems, by giving users the ability to pick from a variety of opens for each candidate.

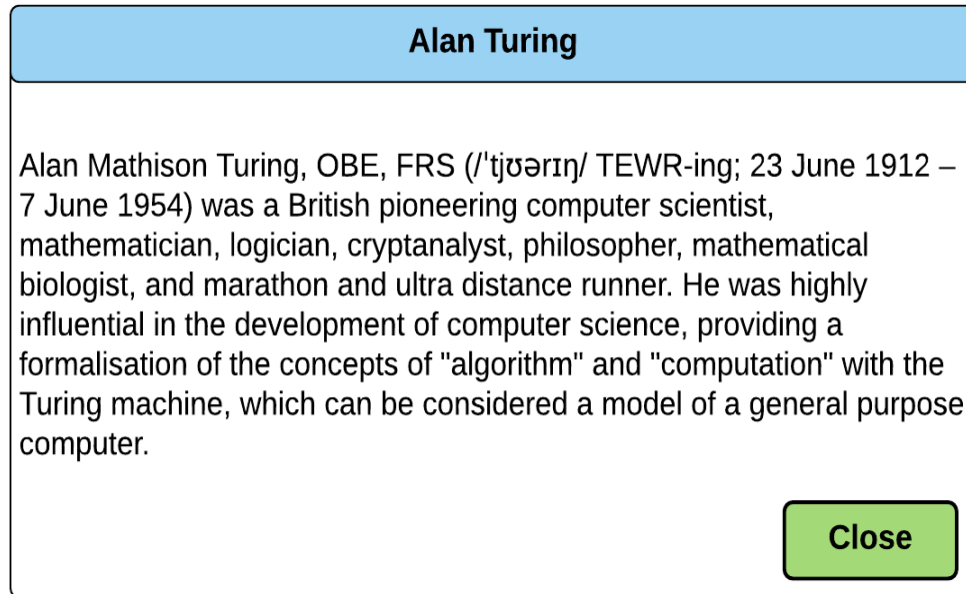


Fig. 16 - An illustration showing a popover for Definition/Comment obtained from
VIRTUOSO SPARQL query end-point

This interaction is illustrated in the Figure 17 below. The user clicks on the link to query VIRTUOSO in-order to view the ‘comment.’ When the system receives the response, the ‘comment’ is displayed as a popover on the screen.

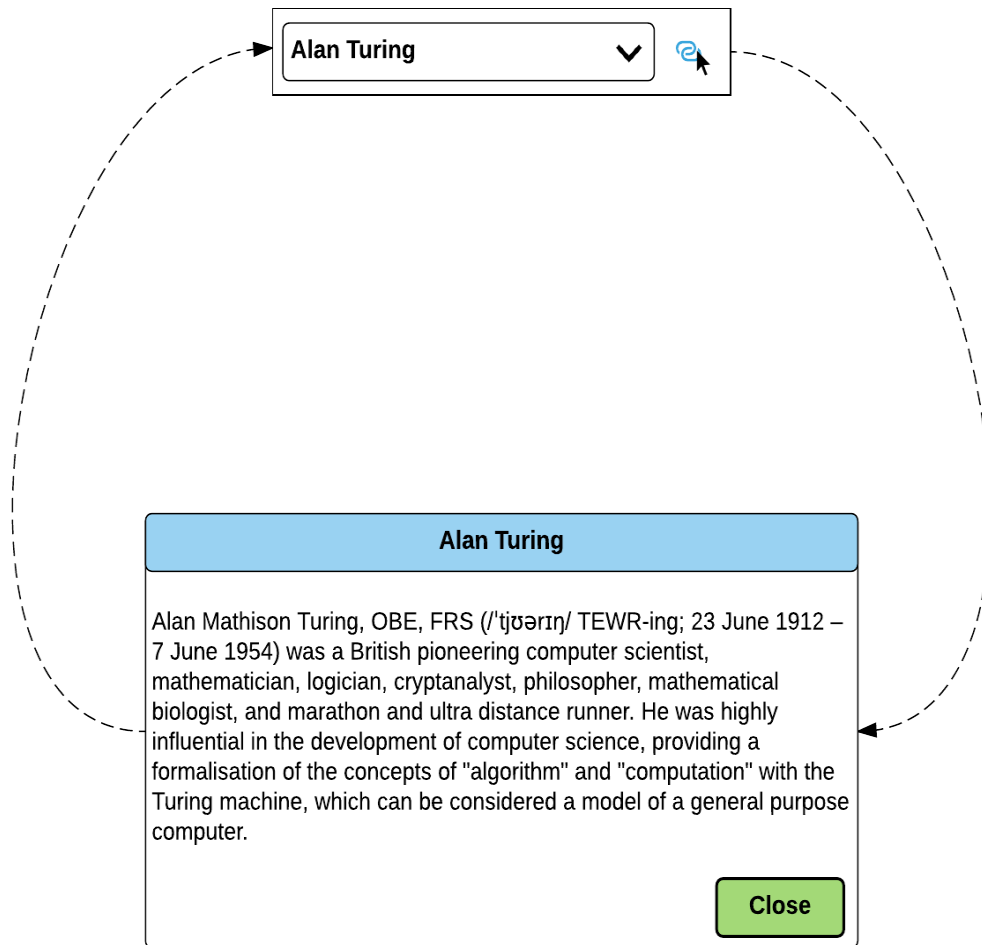


Fig. 17 - An illustration demonstrating the process of querying VIRTUOSO for a description for the entity (Alan Turing) in question

Search Knowledge Base

This specific interaction was designed with the goal of providing users the capability of querying the underlying Knowledge Base for better assignments. This in-turn enables the user to guide the system as it tries to infer semantics of the input table by providing feedback

at each step of the way, after each iteration. This interaction is not limited in accessibility to any specific core resources, like query-and-rank or joint-inference. This feature is horizontally available throughout the entire process, i.e., from the output of query-and-rank, through any number of iterations of joint-inference, to finally when the system converges but the user wants to modify some assignments.

When the user reviews the candidates/annotations generated by the system and realizes that she might have a better assignment for that specific cell, the user can then click on the ‘Search Knowledge Base’ option. This presents the user with a popover, and provides her with the option of entering free form text by typing it in. As the user begins typing, a query is made over HTTP with the ‘query string’ entered by the user, column header, and cell values from that row. The response received is displayed under the text input field and user is given an option of selecting the one he thinks is the most appropriate. This value is appended to the top of the existing list of assignments.

The following illustration (Figure 18), describes the various steps involved in this interaction.

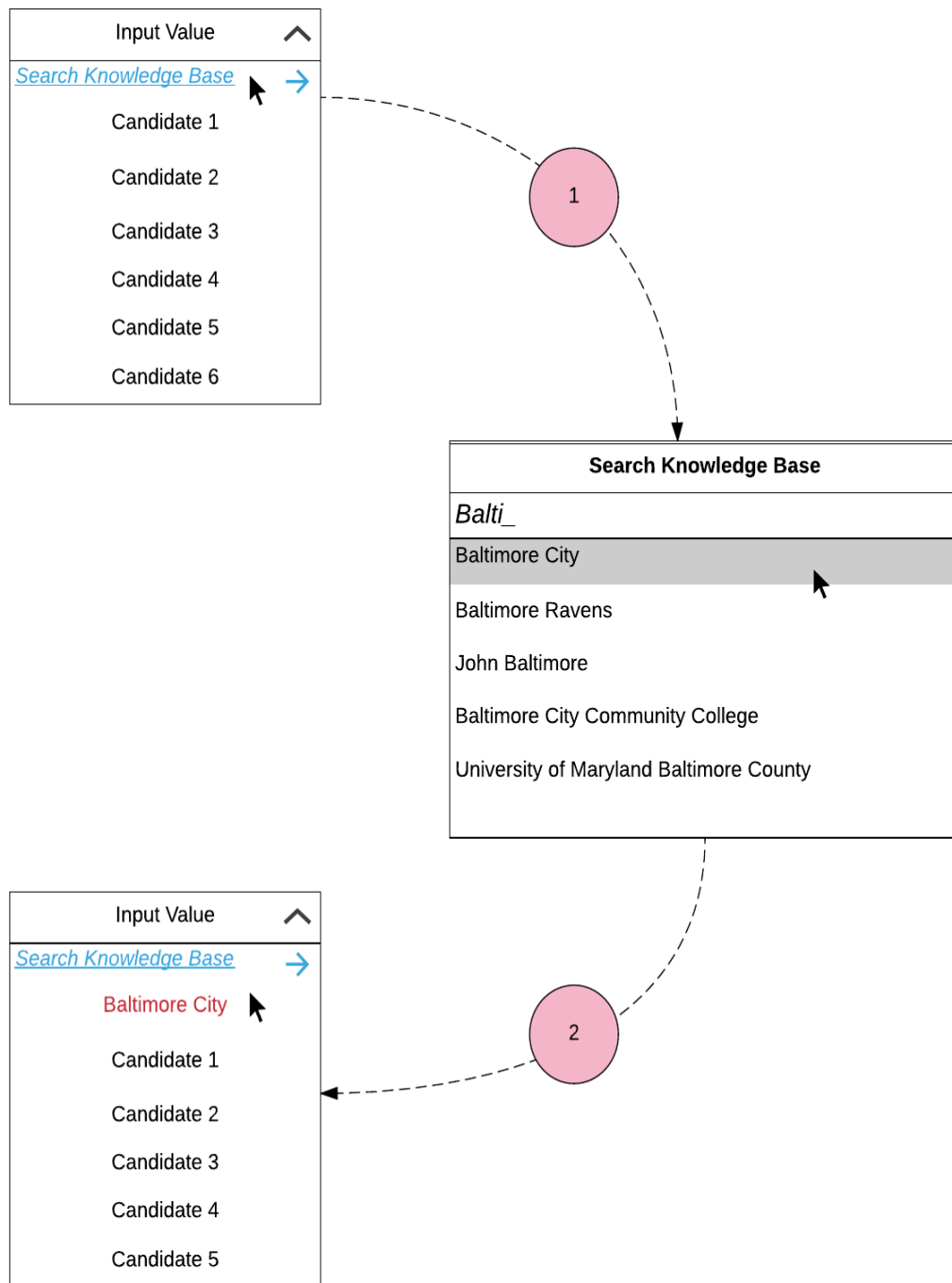


Fig. 18 - An illustration demonstration the process of search for a specific candidate from the underlying knowledge base and adding it to list of candidates

Evaluation

We begin by describing the aspects of the User Interaction Model to be evaluated along with the Target Users. We also include the user metrics and include the various evaluation methods. We also include experiments carried out with the help of target users. Finally, we end the chapter by including the analysis and interpretation of Usability Data.

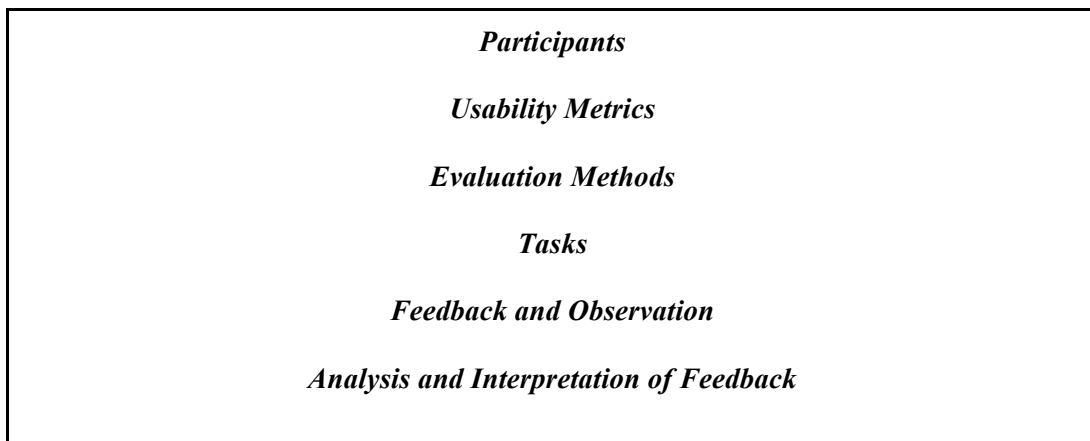


Fig. 19 - Outline of the usability evaluation process

Participants

Since this thesis focuses on the intersection of Web Application, User Experience, and Semantic Web, we recruited 8 users with varied experience in their understanding of the topics mentioned previously. Each user was asked to rate their understanding in each of the subjects from 1 to 5, 1 being no experience and 5 meaning highly proficient. The participants were aged between 24 to 32.

We interviewed 15 participants and asked them the following questions (Figure 20)

<p><i>How would you rate your understanding of the following areas -</i></p> <p><i>1. WWW and Web applications</i></p> <p><i>2. Programming</i></p> <p><i>3. Ideas and core concepts of Semantic Web</i></p>
--

Fig. 20 - User Interview Questionnaire

We asked the users to rate themselves on a scale of 5, where 1 meant least or no understanding of the topic(s) mentioned in the question and 5 meant high proficiency in the mentioned topics. At the end of this exercise we picked users who rated themselves 3 or above in (1) and (2) and we ensured that 50% of users had and no prior understanding of the ideas and concepts of Semantic Web.

Usability Metrics

Usability metrics are a crucial component of the usability evaluation. The goal in selecting these metrics is to choose a minimal number of metrics that reveal maximum amount of usability detail for the UI under study. ISO Standard 9241 recommends using effectiveness, efficiency, and satisfaction measures described below -

1. ***Effectiveness*** is the accuracy and completeness with which users achieve a specified goals. Example metrics include: percentage of goals achieved, and functions learner.

2. ***Efficiency*** assesses the resources expended in relation to the accuracy and completeness with which users achieve goals.
3. ***Satisfaction*** reflects users' freedom from discomfort and positive attitudes towards using an interface. Example metrics include: ratings for satisfactions

Tasks

In this section, we define the tasks which participants were asked to complete as a part of this usability evaluation. We aligned these tasks to highlight specific features of the user interaction model.

Task 1 : Input preprocessing task - In this task, 5 out of the 8 users were asked to “drag and select” a subsection of the input table, while the remaining 3 users were asked to “drop/omit columns” irrelevant to the process of inferring the semantics. At the end of the preprocessing task the users were asked to submit their respective inputs for processing.

Task 2 : Assignments review and change task I - At the end of the “query-and-rank” process, the 4 users were asked to review the assignments made by the system. They were then asked to change/override the assignments made by the system, by selecting an assignment they thought was more appropriate in the given context for the tabel cell and/or column header.

Task 3 : Assignments review and change task II - In this task, the remaining 50% of the users were asked to review the assignments made by the system. But, later, they were asked to change/override the assignments made by the system, by querying the underlying

Knowledge Base for the value they considered more relevant for the table cell and/or column header.

Task 4 : Define entity task - Here the users were asked to define some entities from data cells and column headers, by clicking on the “definition” link present in each table cell.

For each user we recorded the following metrics -

1. Total number of tasks completed
2. Task or Feature specific feedback, and satisfaction rating
3. Overall satisfaction rating

Results

In this section we describe the results from the usability evaluation. This is categorised into 4 sections, starting with user feedback, followed by user satisfaction, and ending with overall satisfaction.

User feedback

The user feedback is gathered and ordered by the level of frequency of the following type of comments. We grouped the issues in four distinct categories, according to the features under testing.

No of users	Input and Preprocessing	"Query and Rank" and "Joint Inference"	Search Knowledge Base	Query VIRTUOSO
1 - 3	"Can this system be used for domain specific tables?"	"Can this process be used on incomplete datasets?"	"Search feature isn't immediately accessible as it is hidden under the drop down"	"Can I see additional information about the entity"
4 - 6	"Would it be possible to import a specific table that I might be interested in?"	"Enables feedback loop successfully"	"The entity from the search result cannot be readily used to query VIRTUOSO without adding it to list of candidates"	
7+		"Slow response time"		

Fig. 21 - Feature specific feedback

User Satisfaction

Finally, each user was asked to rate all the features on the scale of 1 to 5 for satisfaction, where 1 was meant to indicate that the feature was not performing as the user had expected, and 5 meant that the user was satisfied with feature, since it did exactly what the user had expected it to do.

The questions and the captured satisfaction ratings are described in the subsequently.

On the scale of 1 - 5, how satisfied are you with the preprocessing features available?

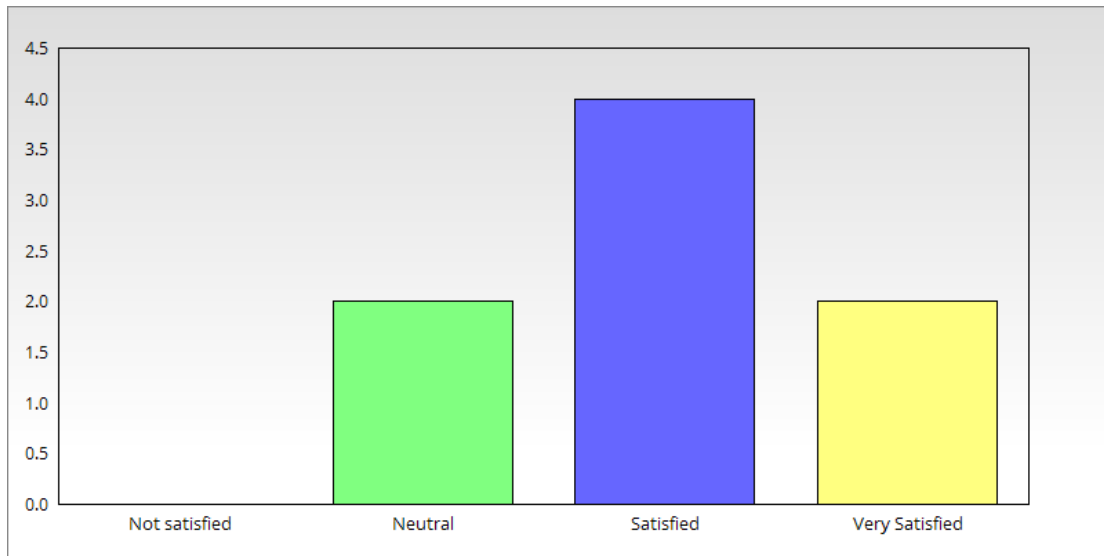


Fig. 22 – User satisfaction rating for preprocessing feature

On the scale of 1 - 5, how satisfied are you with the core inference feature?

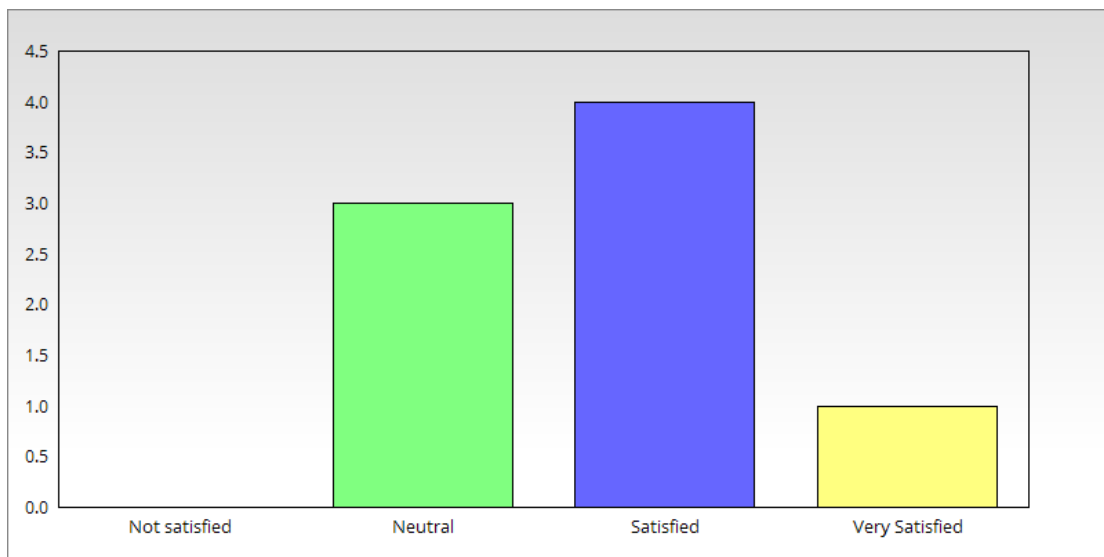


Fig. 23 – User satisfaction rating for core inference feature

On the scale of 1 - 5, how satisfied are you with the feature that lets you query the knowledge base?

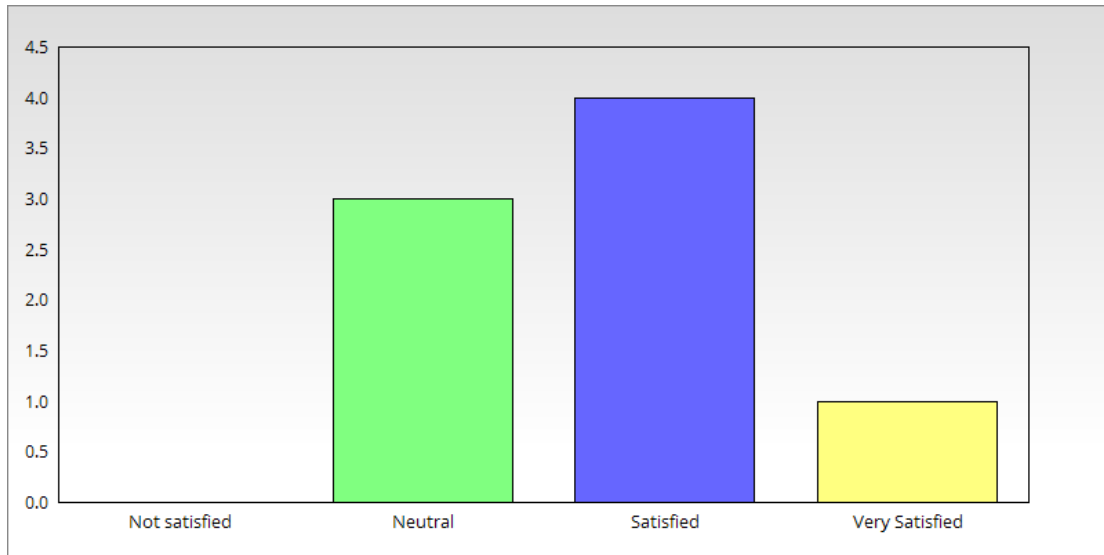


Fig. 24 – User satisfaction rating for knowledge base query feature

On the scale of 1 - 5, how satisfied are you with the feature that lets you view the intended meaning of an entity?

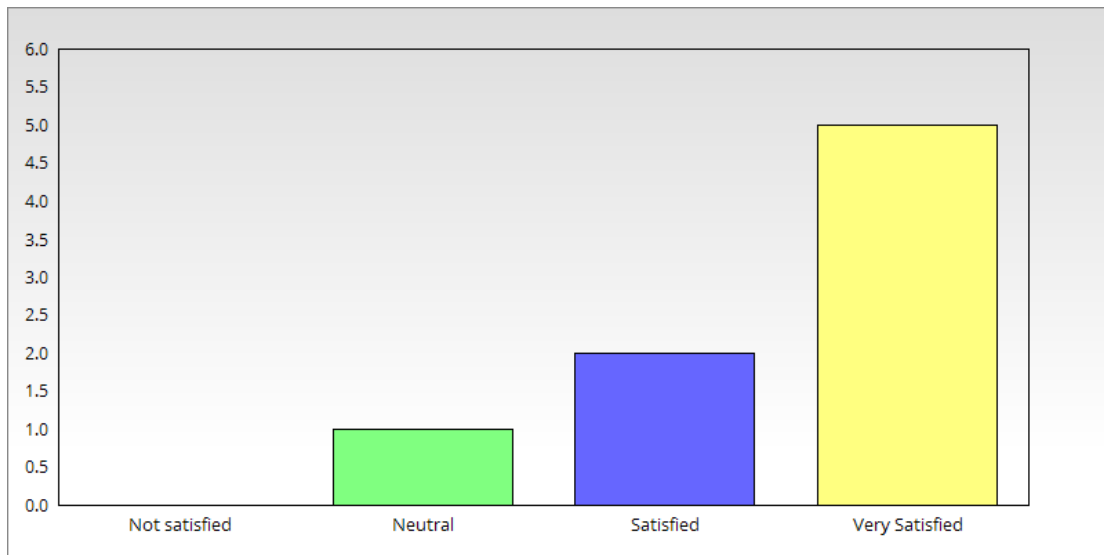


Fig. 25 – User satisfaction rating for VIRTUOSO query feature, which enables the user to view intended meaning of an entity

Overall Rating

Overall, on the scale of 1 - 5, how satisfied are you with the entire experience?

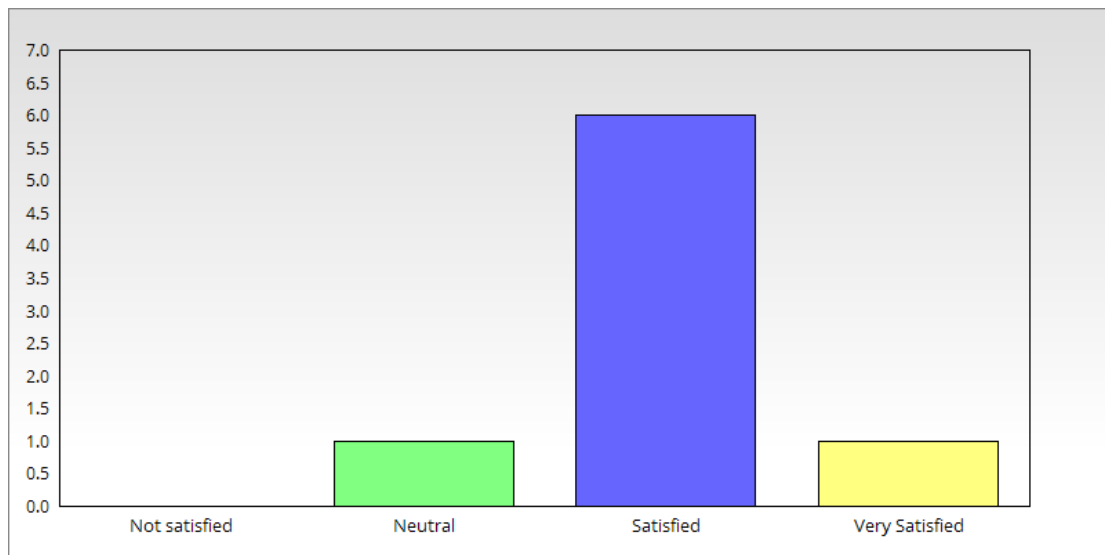


Fig. 26 – Over all user satisfaction rating for the entire application

Conclusion

There is a general consensus within the semantic web research community that the challenges to semantic web interfaces are -

1. Although semantic web based technologies provide a wide array of innovative functionalities, they require equally innovative and intuitive user interfaces to be adopted and accepted.
2. Interfaces need to be able to deal with different levels of granularity of data and information.
3. Interfaces are required for data-sets for which the schemas are not fully known at design time.

Developing a generic, web based user interaction model for semantic web applications, such as TABEL, would enable a wide array of users to access the system. We hope this, and many such attempts, would enable wider user adoption and would help semantic web concepts gain wider acceptance.

By focusing on developing and deploying TABEL as a RESTful service over HTTP, we ensure that the RESTful constraints are enforced. Some of the important advantages of this approach are -

1. Client - Server share a common uniform interface through which they communicate.
The uniform interface constraint is fundamental to the design of any RESTful web service. The uniform interface simplifies and decouples the architecture, which enables each part to evolve independently.

2. This separation of constraints for example, clients are not concerned with data storage, which remains internal to each server, so that the portability of client code is improved. Servers are not concerned with the user interface or user state, so that servers can be simpler and more scalable. Servers and clients may also be replaced and developed independently, as long as the interface between them is not altered.
3. We further ensure that the client - server communication is further constrained by no client context being stored on the server between requests. Each request from any client contains all the information necessary to service the request, and session state is held in the client.

In conclusion, our work in this thesis is primary focused on the intersection of semantic web, service oriented architecture, user interaction, and usability evaluation. We believe that this interdisciplinary approach is required in order to advance and consolidate the development of semantic web.

Future Work

Our existing work on designing and implementing a generic, web based user interaction model for TABEL. Some of the extensions that we propose are -

1. Enabling the user to import a table from a dataset accessible to or available with the user. This is can be done by designing and implementing a javascript based data import utility.
2. Design and develop a table creation module within the application user interface, which would enable users to have the independence of creating and experimenting with the system.
3. Implement a linked data publishing utility and create integration points for various linked data resources in order to enable users to create new linked data and enrich the existing.
4. Giving advanced users the ability to select a knowledge base specific to the domain of their dataset.

Integration with Multiple Knowledge Bases

The present implementation integrates well with wiktology as a knowledge base, which serves well for general purpose disambiguations as shown previously. But, we believe our restful architecture can be extended to successfully incorporate multiple such general

purpose knowledge bases as well as some special, domain specific ones. This is enable disambiguation across multiple domains.

Focus on Availability, Reliability and Scalability

Although we were able to successfully create of a user interaction model by converting a monolith into a suite of web-services, we truly believe this effort can be deploy this model at scale and to make it easily accessible to researchers, developers and other interested members of the semantic community.

References

- [1] Limaye, G.; Sarawagi, S.; and Chakrabarti, S. 2010. Annotating and searching web tables using entities, types and relationships. In Proc. 36th VLDB.
- [2] Mulwad, V. 2010. T2LD - An automatic framework for extracting, interpreting and representing tables as Linked Data. Master's thesis, University of Maryland, Baltimore County.
- [3] Mulwad, V. 2010. T2LD - An automatic framework for extracting, interpreting and representing tables as Linked Data. Master's thesis, University of Maryland, Baltimore County
- [4] Syed, Z., and Finin, T. 2011. Creating and Exploiting a Hybrid Knowledge Base for Linked Data. In Agents and Artificial Intelligence.
- [5] Syed, Z.; Finin, T.; Mulwad, V.; and Joshi, A. 2010. Exploiting a Web of Semantic Data for Interpreting Tables. In Proceedings of the 2nd Web Science Conference
- [6] Berners-Lee, T.; Hendler, J.; Lassila, O.; et al. 2001. The semantic web. Scientific American.
- [7] Berners-Lee, T. 2006. Linked data. <http://www.w3.org/DesignIssues/LinkedData.html>
- [8] Cafarella, M. J.; Halevy, A. Y.; Wang, Z. D.; Wu, E.; and Zhang, Y. 2008. Webtables: exploring the power of tables on the web.

- [9] Knoblock, C. A.; Szekely, P.; Ambite, J. L.; Gupta, S.; Goel, A.; Muslea, M.; Lerman, K.; Taheriyani, M.; and Mallick, P. 2012. Semi-automatically mapping structured sources into the semantic web. In Proceedings of the Extended Semantic Web Conference.
- [10] Knoblock, C. A.; Szekely, P.; Gupta, S.; Manglik, A.; Verborgh, R.; Yang, F.; and de Walle, R. V. 2013. Publishing data from the smithsonian american art museum as inked open data. In Proceedings of the ISWC 2013 Posters & Demonstrations Track, 129–132.
- [11] Langeegger, A., and Wob, W. 2009. Xlwrap - querying and integrating arbitrary spreadsheets with SPARQL. In Proc. 8th Int. Semantic Web Conf.
- [12] Levenshtein, V. I. 1966. Binary codes capable of correcting deletions, insertions, and reversals. Technical Report 8, Soviet Physics Doklady.
- [13] Limaye, G.; Sarawagi, S.; and Chakrabarti, S. 2010. Annotating and searching web tables using entities, types and relationships. In Proc. 36th VLDB.
- [14] Szekely, P.; Knoblock, C. A.; Yang, F.; Zhu, X.; Fink, E.; Allen, R.; and Goodlander, G. 2013. Connecting the Smithsonian American Art Museum to the Linked Data Cloud. In Proceedings of the 10th Extended Semantic Web Conference.

