Exploiting a Web of Semantic Data for Interpreting Tables

Zareen Syed, Tim Finin, Varish Mulwad, and Anupam Joshi
Computer Science and Electrical Engineering
University of Maryland, Baltimore County
Baltimore MD 21250
{zarsyed1, finin, varish1, joshi}@umbc.edu

ABSTRACT

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 describe techniques 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. The techniques have been prototyped for a subset of linked data that covers the core of Wikipedia.

Keywords

Semantic Web, linked data, human language technology, information retrieval

1. INTRODUCTION

Twenty-five years ago Doug Lenat started a project to develop Cyc [17] as a broad knowledge base filled with the common sense knowledge and information needed to support a new generation of intelligent systems. The project was visionary and ambitious, requiring broad ontological scope as well as detailed encyclopedic information. While the research and development around Cyc has contributed much to our understanding of building complex, large-scale knowledge bases relatively few applications have been built that exploit it.

Not long after the Cyc project began, Tim Berners-Lee proposed a distributed hypertext system based on standard Internet protocols [2]. The Web that resulted fundamentally changed the ways we share information and services, both on the public Internet and within organizations. One success story is Wikipedia, the familiar Web-based, collaborative encyclopedia comprising millions of articles in dozens of languages. The original Web proposal contained the seeds of another effort that is beginning to gain traction: a Semantic Web designed to enable computer programs to share and understand structured information easily as a Web of Data.

Resources like Wikipedia and the Semantic Web's linked data [3] are now being integrated to provide experimental knowledge bases containing both general purpose knowledge as well as a host of specific facts about significant people, places, organizations, events and other entities of interest. The results are finding immediate applications in many areas, including improving information retrieval, text mining, and information extraction.

2. BACKGROUND AND MOTIVATION

It is unlikely that the Web of data will be realized by having millions of people manually entering RDF data. We need systems that can automatically generate linked data from existing sources, be they unstructured (e.g., free text), semi-structured (e.g., text embedded in forms or wikis) or structured (e.g., data in spreadsheets and databases).

Extracting data and its implicit schema from tables is a problem that is common to several areas of computer science, including databases [26] where it is long been a part of the database integration problem, Web systems [5] where search engines need to know how to index tabular data and the Semantic Web [17] where organizations want to publish their traditional databases as linked RDF data. While a number of pragmatic systems have been built, there are many difficult research issues yet to be solved.

Several manual systems exist for specifying a mapping from relational tables and spreadsheets to RDF [4]. In earlier work we developed RDF123 [13] as an application and web service for converting data in simple spreadsheets to an RDF graph. Users control how the spreadsheet's data is converted to RDF by constructing a graphical RDF123 template that specifies how each row in the spreadsheet is converted as well as metadata for the spreadsheet and its RDF translation. The template can map spreadsheet cells to a new RDF node or to a literal value. Labels on the nodes in the map can be used to create blank nodes or labeled nodes, attach a XSD datatype, and invoke simple functions (e.g., string concatenation). The template itself is stored as a valid RDF document encouraging reuse and extensibility.

While systems like RDF123 are both practical and useful, they suffer, in general, from several shortcomings. First, they require

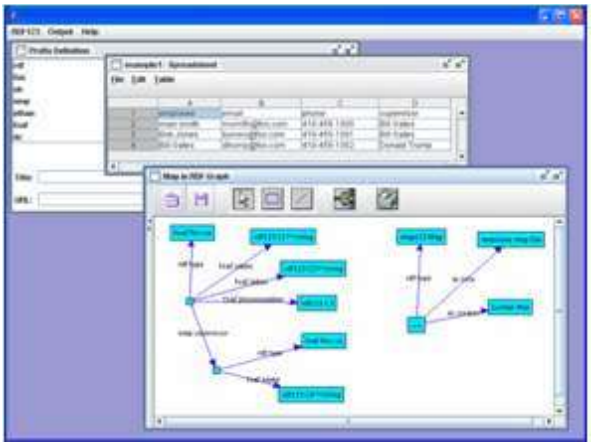

Figure 1. RDF123 is an application and web service for converting data in simple spreadsheets to an RDF graph.

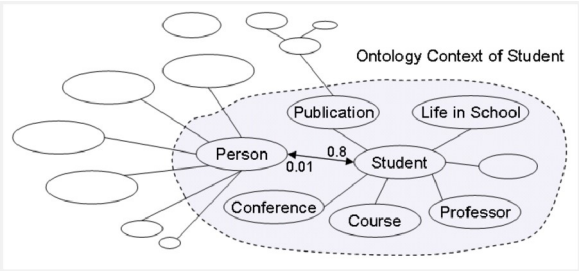

Figure 2. Han et al. [14] describe a system that suggests terms for relational tables based on the words in their schema column headers. This graph shows the "ontology context" for the word "student". Each oval stands for an ontology and the gray area denotes the ontology context of student ontology. Some connected lines in the gray area are omitted for simplicity. The Person ontology has a conditional probability of 0.8 accompanying the Student ontology while the other way only has a conditional probability of 0.01.

significant manual work to specify the mapping from the table to an appropriate representation in RDF. Second, users who specify the mapping must decide what RDF terms to use to describe the instances in the table and the relationships between them. For example, if the table described people, the RDF data should include instances of a person class. But there are many ontologies that are widely used that contain a class intended to represent people, e.g., Cyc, FOAF, etc. Choosing from among them, or even knowing what the possibilities are, is a major problem, especially for users unfamiliar with the Semantic Web who only want to convert their data into some RDF vocabulary. Third, they offer no support for producing linked data – i.e., RDF data where objects are referenced not be strings (e.g., "Baltimore") but by RDF instances (e.g., http://dbpedia.org/resource/Baltimore) that are part of or linked to part of the linked data cloud. These problems are interlinked and a solution to one involves solving the others.

In earlier work we tackled the second problem, recommending a set of terms to use to describe the objects and relationships in the table. Interpreting a table means recognizing the concepts in the table and mapping columns to the properties of the concepts according to the information in the table. Column titles have very important information for finding the underlying concepts. They work actually as identifiers for the concepts. They can be the names of the concepts, the names of the attributes of the concepts, the names of the connections between the concepts (attributes and connections can all be viewed as properties of a concept). The column titles (the identifiers), taken as a whole, can be mapped to concepts and can disambiguate. For example, if the column titles include "Beetle" and "Habitat", then we can conclude that the "Beetle" is insect beetle but not car beetle. We can also learn concepts from the instance rows.

For instance, suppose a table column contains the title medicine and the entries *Motrin, Advil, Alleve* and *Voveron*. Using Linked Data sources or Wikipedia, we can find out while these are all names of medicines, they are specifically a category called *Non Steroidal Anti Inflammatory Drugs.* We may also find out from existing drug ontologies or linked open data sources that medicines have doses, and conditions they are prescribed for. This might help us interpret the other columns, and also establish that some column title represents a class, and others define properties that this class has.

Linked data sources and Wikipedia can also provide a sanity check for our interpretations. Suppose we see a column with the entries *Nilgiri, Aravali, Jwalamukhi,* and *Karakoram* in a table, and another column that has numbers that seem to be latitude and longitude. We could infer from DBpedia that these are names of mountains. Not only is this fact inferred using linked data and Swoogle [6], but we can also check to see if the latitudes and longitudes provided match those that DBpedia lists. If they do not, we would avoid an incorrect interpretation. We could then search for other interpretations and find that these are also the names for dorms at IIT-Delhi. This interpretation would make the latitude/longitude entries correct, and might also be consistent with a column entitled residents in the table.

Of course, doing all this assumes that we have a machine representation of concepts. This is where existing semantic web search engines such as Swoogle can be useful, since they index a variety of ontologies based on the terms in them. We can learn what properties a class is associated with from the indexed ontologies and data documents. We can learn the co-occurrences of two concepts/classes from RDF data documents. We can also search for ontologies that might be based on synonyms of the words used in the table [13] by using open sources such as WordNet.

3. APPROACH

We have been exploring the use of Web-derived knowledge bases through the development of Wikitology [9] - 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. 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, social bookmarking sites and collaborative editing using Wikis etc. 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 richer applications.

Another challenge confronting the Web is the integration of information from heterogeneous sources and in heterogeneous representations such as structured, semi-structured and un-structured representations.

Wikitology is not unique in using Wikipedia to form the backbone of a knowledge base, see [22] and [25] for examples. However, it is unique in integrating knowledge available in different structured and un-structured forms and providing an integrated query interface to applications enabling them to exploit broader types of knowledge resources such as free text, relational tables, link graphs and triples using a single interface. The core idea exploited by most approaches is to use references to Wikipedia articles and categories as terms in an ontology. For example, the reference to the Wikipedia page on weapons of mass destruction can be used to represent the WMD concept and the page on Alan Turing that individual person.

These basic Wikipedia pages are further augmented by category pages (e.g., biological weapons) representing concepts associated and other categories. Finally, Wikipedia pages are rich with other data having semantic impact, including links to and from other Wikipedia articles, links to disambiguation pages, redirection

City	Mayor	State	Population
Boston	T. Menino	MA	610,000
New York	M. Bloomberg	NY	8,400,000
Philadelphia	M. Nutter	PA	1,500,000
Baltimore	S. Dixon	MD	640,000
Washington	A. Fenty	DC	595,000

Figure 3. The data model implicit in a simple table can be inferred by analyzing the column headers and string values and the result mapped into appropriate target ontologies as linked data.

links, in- and out-links, popularity values computed by search engines, and history pages indicating when and how often a page has been edited. Wikipedia's infoboxes flesh out the nascent ontology with links corresponding to properties and relations.

There are many advantages in deriving a knowledge base from Wikipedia such as having broad coverage (over 3M articles); a consensus based concept space developed and kept current by a diverse collection of people; high quality of content [16] and availability in multiple languages. 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.

Wikitology has been used successfully for a variety of tasks such as Concept Prediction, Cross Document Coreference Resolution [10] and Entity Linking [9]. An example of an application that further illustrates the potential of this system is extracting linked data from the content found in structured documents such as spreadsheets, relational tables, HTML tables and structured reports. The process involves and integrates data model extraction, ontology alignment, entity extraction and linking, and interpretation. Consider, as an example, processing the simple table shown in Figure 3. The column headers strongly suggest the type of information in the columns: *city* and *state* might match classes in a target ontology such as DBpedia; *mayor* and *population* could match properties in the same or related ontologies.

The DPpedia ontology associates the *mayor* property with the domain *city* and *population* with populated place, a class that includes both cities and states. Examining the data values, which initially are just strings, provides additional information that can confirm some possibilities and disambiguate between possibilities for others. For example, the strings in column one can be recognized as entity mentions and linked to known entities in the Wikitology knowledge base [9] that are instances of the *DBpedia:Place* class. Additional analysis can automatically generate a narrower description in OWL-DL, such as major cities located in the United States. A named entity recognizer can identify the column two values as person mentions and Wikitology can resolve some of these to individuals in its KB. Moreover, from DBpedia we can find confirming evidence for many of the individual relationships between these people and places.

Finally, consider the population column. The likely target ontology schema supports its interpretation as a property from places to integers, but does not help in choosing whether it should be associated with the cities (column one) or their states (column two). Several heuristics can be employed, such as preferring the closest column or the first column but these can, as in this case, suggest different selections. Here, the DPpedia background KB offers direct evidence that the integer values are closer to the populations of the cities than the states. Moreover, the KB can help

```
@prefix dbp: <http://dbpedia.org/resource/> .
@prefix dbpo: <http://dbpedia.org/ontology/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix cyc: <http://www.cyc.com/2004/06/04/cyc#>

dbp:Boston dbpo:PopulatedPlace/leaderName
dbp:Thomas_Menino;
   cyc:partOf dbp:Massachusetts;
   dbpo:populationTotal "610000"^^xsd:integer .
dbp:New_York_City …
…
```

Figure 4. The linked data (serialized in Turtle) that might be generated from the table in figure one resolves values to familiar linked data entitles when possible.

disambiguate between the two population-related properties associated with cities – one for the city proper and another for its broader metropolitan effort. The data aligns best with the known values for the narrower legal city entity.

A knowledge base incorporating information available in different forms can better meet the needs of real world applications than one exposing knowledge in a more restricted way such as through SQL, SPARQL or 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. Incorporating these qualities in knowledge bases like Cyc will be very expensive in terms of time, effort and cost. Efforts like DBpedia, Freebase, YAGO, and linked data are focused on making knowledge available in structured forms. Our novel hybrid KB can complement these valuable resources by integrating knowledge available in other forms and providing flexible access to knowledge.

4. WIKITOLOGY

One of the challenges in integrating information available in different forms is the selection of an appropriate data structure to represent or link data available in different forms and provide a query interface giving an integrated view.

Most of the knowledge available in Wikipedia is in the form of natural language text. Approaches for indexing and querying IR indices are more efficient and scalable as compared to triple stores. Therefore, an information retrieval (IR) index is our basic information substrate. The articles in Wikipedia represent concepts. We represent the concepts (articles) as documents in the index using the Lucene IR Library [15].

To integrate it with other forms of knowledge, we enhance the IR index with fields containing instance data taken from other data structures and links to related information available in other data structures such as graphs or triples in an RDF triple store which are related to the Wikipedia concept.

Using a specialized IR index enables applications to query the knowledge base using both text (e.g., words in a document) and structured constraints (e.g., rdfs:type = yago:Person).

The different fields present in the Wikitology index include title, redirects, first sentence of article, article content, Wikipedia categories, types (Freebase entity types, DBpedia and Yago types), linked concepts, DBpedia infobox properties and values.

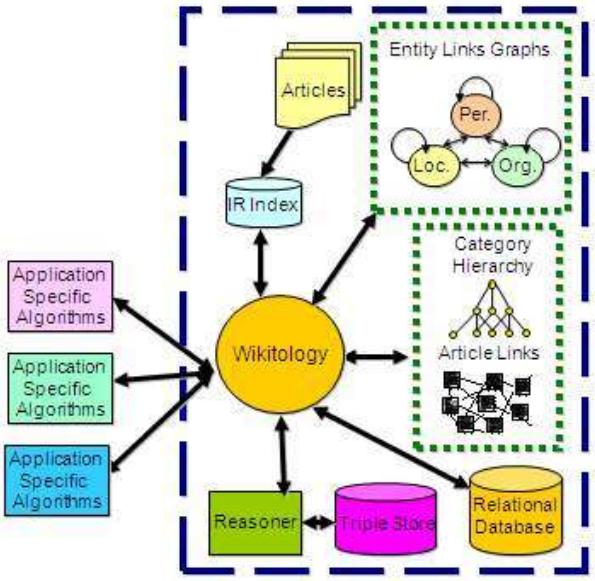

Figure 5. Wikitology is a hybrid knowledge based storing information in structured and unstructured forms and reasoning over it using a variety of techniques.

In addition to different fields representing different types of content extracted from Wikipedia and related resources, we have also introduced a field with page rank approximations of Wikipedia articles. Earlier work on Entity Linking task [19] defined in the Knowledge Base Population track at the 2009 Text Analysis Conference [20] identified Google page rank as an important feature for linking entities mentioned in text to Wikipedia entities. However, it is not possible to query Google for page rank of all Wikipedia articles, therefore we have developed an approach for approximating the page rank for Wikipedia articles. We discuss our approach below.

4.1 Page Rank Approximations

We selected 4296 random articles from Wikipedia and queried Google for their PageRank resulting in the distribution shown in column two of Table 1. We used these to train a classifier using the number of in- and out-links and the article length as classification features and the page rank as the class label. We used 10-fold cross-validation to evaluate four common classification algorithms: SVM, decision trees with and without pruning and naïve Bayes. The pruned decision tree algorithm gave the highest accuracy of 53% followed by 52%, 50% and 41% for an unpruned decision tree, SVM and naïve Bayes, respectively.

Page rank	Articles	Accuracy	± 1 Accuracy
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	0.000	0.67

Table 1. We sampled Wikipedia articles and used them to develop a decision tree to predict their page rank based on the umber of in and out-links.

Features	Accuracy	± 1 Accuracy
All features	53.14	93.78
All - Page Length	53.91	95.30
All – InLinks	51.46	93.23
All - OutLinks	54.81	95.34
In-links	54.03	95.69
Page Length	49.95	94.20
Out-links	49.46	94.13

Table 2. An analysis of our features revealed that only using the number of in-links provided our best approximation for Google's PageRank metric for a Wikipedia article.

The accuracy obtained per PageRank class using decision tree (J48) algorithm is given in Table 1's third column. We recomputed the accuracy and considered a prediction to be correct if there is a difference of one between the predicted and 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 1. 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 2. 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 page ranks for Wikipedia articles.

4.2 Discovering Ontology Elements from Wikipedia page links

We have developed an unsupervised and unrestricted approach to discovering an info-box like ontology by exploiting the inter-article links within Wikipedia which represent relations between concepts. In our approach we consider the linked concepts as candidate fillers for 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" mentions *pop*, *rock* and *soul* as fillers for the slot Genre and all three of these are a type of Genre. Based on this observation, we discover and exploit "ISA"' relation between fillers (linked concepts) and WordNet nodes to serve as candidate slot labels.

We first identify and rank candidate slots and fillers for an entity, then classify the entities based on ranked slots and finally derive a class hierarchy. We have compared the predicted slots/properties for different entity classes (such as Politician, Basketball Player etc.) with DBpedia infobox ontology and Freebase resource. For the set of properties found to be common we manually evaluated the subject object pairs of the property by verifying the ground truth from respective Wikipedia articles. The average accuracy for

the properties was found to be above 80%. 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 and can be used to enrich the existing infobox ontology as well as the Wikitology knowledge base itself.

5. 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
- Analyzing the table globally decompose or normalize it into one or more sub-tables, if appropriate, and identify the key of each resulting sub-table.

While these tasks could be addressed serially, the problems are intertwined. Our current prototype 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 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.

We used an earlier version of the Wikitology knowledge base in the entity linking task defined in the Knowledge Base Population track of Text Analysis Conference 2009 [20]. The 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 [24]. 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.

We discuss our approach to first linking the instances in the table to Wikitology knowledge base entries to predict the Class for a table column, we then use the predicted class labels as additional evidence to link the instances in the column. After linking the instances in the table we can enumerate the relations present between columns and select a particular relation to describe the relationship existing between pairs of columns. We describe our approach below.

5.1 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

Algorithm 1: **WikitologyLinkInstance**

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"

Output: Top N Matching Concepts Vector

```
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)
TopNConceptVector = Wikitology.searchQuery(Query,TopN)
```

Return TopNConceptVector

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, 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 page rank of the top 'N' Wikipedia entities using a weighted sum (w = 0.25):

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

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.

5.2 Entity Linking

We use the predicted types as additional evidence to link the instances to the right Wikipedia entities. We re-query Wikitology by updating the original query in Algorithm 1, by adding the predicted types mapped to the "typesRef" field in the index. The "typesRef" field is an un-tokenized field and supports exact match. Querying using the typesRef field restricts the results to only those entities whose type exactly matches the input type. We then select the top most predicted entity by Wikitology which matches one of the types predicted, as the right match.

5.3 Relation Enumeration

To describe the relations between columns in a table we take all pairs of entities present in the same row (which have been already linked to Wikipedia) and query DBpedia for the set of relations between each pair. For example the relations between Massachusetts and Boston are:

http://dbpedia.org/ontology/largestCity

http://dbpedia.org/ontology/PopulatedPlace/largestCity

http://dbpedia.org/ontology/capital

http://dbpedia.org/ontology/PopulatedPlace/capital

http://dbpedia.org/property/capital

http://dbpedia.org/property/largestcity

In this way we can enumerate the set of relations between a pair of columns by listing the relations between pair of entities in the respective columns for each row.

Table Topic	Columns
US States	State, Capital City, Largest City, Governor
US Presidents	President, Party, Vice President, Preceded by, Succeeded by
Basketball players	Name, College, Team, Nationality, Place of Birth
US cities	City, State, Mayor, Population
Rivers in Europe	River, Length, Basin Countries, Mouth, Origin

Table 3. We used a small collection of tables each with four or five columns to evaluate our initial system.

5.4 Relation Selection

After relation enumeration we have a step for relation selection to select a representative relation existing between a pair of columns in a table. The relation that appears the maximum number of times between the pairs of entities in the columns can be selected as a representative relation.

6. EVALUATION

We have developed an initial prototype system to evaluate our approach. We collected a sample data set composed of 5 tables taken from Google Squared [11]. The table topics and columns are given in Table 3.

We have excluded the columns "Population" from the US cities tables and "Length" from the Rivers in Europe table when we calculated the accuracy. These columns contain integers, thus linking them as an entity would be inappropriate. These columns most likely describe a property of some other column in the table. Different approach will be needed to deal with such columns.

The results for class prediction for table columns are given in Table 4. For each column we predicted class labels from four vocabularies – DBpedia Ontology, Yago, WordNet and Freebase. The accuracies for correct prediction of class labels for a column from each of the four vocabularies are given in the table. The Freebase class labels predicted for a column were the most accurate, whereas the Yago and WordNet class labels predicted for a column were least accurate.

In the five tables, obtained from Google Squared, we had a total of 171 entities, which were classified into Persons, Places and

Entity Type	Cols	Accuracy (%)			
		DBpedia	Yago	WordNet	Freebase
Places	11	90.90	45.45	54.54	90.90
Persons	7	100	100	85.71	100
Organization	3	33.33	100	100	66.66
Total	**21**	**85.71**	**71.42**	**71.42**	**90.47**

Table 4. Results for the class prediction tasks for entities in a given column for the tables in our small collection.

Entity Type	Correct Predictions	Number of Entities	Accuracy
Person	59	65	90.76 %
Places	45	73	61.64 %
Organiza-tion	20	30	66.667 %
Total	**131**	**171**	**76.60 %**

Table 5. Results for the entity linking task for the entity mentions found in our sample tables.

Organizations. The results for entity linking are given in Table 5. Out of a 171 entities we were able to link 131 entities correctly which gives an accuracy of 76.60 %. Our system was able to link persons with the highest accuracy (90.76 %), however it has moderate success in linking Places and Organizations. On further evaluation, we have realized that certain columns in the tables will need further specialized querying for the entities to be linked correctly. The columns that have been labeled incorrectly are: "Nationality" from the Basketball players table, "Mayors" from the US Cities table, and "States" from the US Cities table.

The "States" column from the US cities table includes all the states as acronyms. Our system presently does not handle acronyms well. We plan on adding additional mechanisms to exploit a collection of acronym and abbreviations for common entities, e.g., states, stock ticker symbols, chemical compounds, etc. Instead of using the present mechanism for querying a better solution would be identifying that the column contains acronyms and then expanding the acronyms to full names before querying using the present mechanism**.**

The "Mayors" column from the US cities table includes the name as first name initial dot last name. We are currently not employing any specialized heuristics for person names, however, in our earlier work [9] 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".

The "Nationality" column present in the Basketball Players table gave a low accuracy for entity linking. The values in the nationality column are mostly adjectival forms of place names such as "Australian". In our earlier work with entity linking [9] we used a separate module to replace all adjectival forms of place names with nouns i.e. "Australian" will get replaced with "Australia", this helped in linking to the right country articles in Wikipedia.

7. DISCUSSION AND FUTURE WORK

We have developed an approach for linking data in the table and table headers to concepts in the DBpedia ontology. We have also proposed approaches for describing the relations evident in the tabular data through relation enumeration and relation selection. We have currently experimented with a small data set which we were able to verify manually. In the future we plan to run our experiments on larger data sets.

The technique of resolving strings against Wikitology is a productive one that works for many cases. However, we recognize the need to handle a number of special cases. We are developing a framework into which we can plug column type identifiers to identify common types such as numbers, dates, addresses, stock ticker symbols, ISBN numbers etc.

There might be many concepts available in structured data like tables that are not present in the DBpedia ontology. We would also like to explore approaches to discovering new concepts and resources as well as relations and properties not already existing in the ontology. Our work on predicting ontology elements from Wikipedia page links suggests that we can discover several properties/slots by exploiting the associated links. We can extend the same approach to predict the properties for entities in a table based on other entities present in the same row. We might also be able to exploit the evidence from multiple rows in the same table.

Since it is unreasonable to expect that any system will be completely accurate, we are evaluating strategies for handling data that cannot be linked to linked data entities. There are two general categories. In some cases it may be that neither the column header nor its values provide good evidence to predict the intended meaning. Here a possible solution is to create a unique property for the column, link it to the table key, and represent the cell values a appropriate literals.

Many problems remain. Of these we mention two that are shared with other Web information use cases: temporal qualification and handling inconsistencies. The first has to do with interpreting entities and relations in a table that might refer to one time period by using a collection of knowledge describing facts and situations from essentially many time periods. The second problem arises when the data – both in the table to be interpreted and the reference knowledge bases and sources – are inconsistent due to mistakes, noise or differing opinions.

8. 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 described techniques 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 representation as linked data. The techniques have been prototyped for a subset of linked data that covers the core of Wikipedia. Initial evaluation shows that the techniques are promising and can be employed for automated extraction of linked data from structured data in the form of tables.

9. ACKNOWLEDGMENTS

The research described in this paper was supported in part by a Fulbright fellowship, a gift from Microsoft Research, NSF award IIS-0326460 and the Human Language Technology Center of Excellence.

10. REFERENCES

[1] Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. (2007). DBpedia: A Nucleus for a Web of Open Data. Proc. 6th Int'l Semantic Web Conf., Springer.

[2] Berners-Lee, T. (1989). Information management: A Proposal. Euro. Particle Physics Lab., unpublished report.

[3] Bizer, C. (2009). The Emerging Web of Linked Data. IEEE Intelligent Systems, v24, n5, pp. 87-92.

[4] Bizer, C. 2003. D2R Map - A Database to RDF Mapping Language. WWW (Posters) 2003, WWW'03.

[5] Cafarella, M. J., Halevy, A., Wang, D. Z., and Zhang, Y. (2008). Web tables: Exploring the power of tables on the web, Proceedings of the VLDB, pp 538--549, 2008.

[6] Ding, L., Finin, T., Joshi, A., Pan, R., Cost, S., Sachs, J., Doshi, V., Reddivari, P. and Peng, Y. (2004). Swoogle: A Search and Metadata Engine for the Semantic Web, Thirteenth ACM Conf. on Information and Knowledge Management (CIKM'04), Washington DC, November 2004.

[7] Etzioni, O., Banko, M., Soderland, S., and Weld, D.S. (2008). "Open Information Extraction from the Web", Communications of the ACM, 51(12): 68-74.

[8] Finin, T., Syed, Z., Mayfield, J., McNamee, P. and Piatko, C. (2009). Using Wikitology for cross-document entity coreference resolution, AAAI Spring Symposium on Learning by Reading and Learning to Read, AAAI Press.

[9] Finin, T. and Syed, Z. (2010). Creating and Exploiting a Web of Semantic Data, Proc. 2nd Int. Conf. on Agents and Artificial Intel-ligence, Valencia.

[10] Finin, T., Syed, Z., Mayfield, J., McNamee, P. and Piatko, C. (2009). 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.

[11] Google Squared – http://www.google.com/squared

[12] Halevy, A., Norvig, P., and Pereira, F. (2009). The Unreasonable Effectiveness of Data, IEEE Intelligent Systems, 24(2), pp. 8-12, March/April 2009.

[13] Han, L., Finin, T., Parr, C., Sachs, J., and Joshi, A. (2008). RDF123: from Spreadsheets to RDF, Proceedings of the Seventh International Semantic Web Conference, Springer, October 2008.

[14] Han, L., Finin, T. and Yesha, Y. (2009) Finding Semantic Web Ontology Terms from Words, Proc. 8th Int. Semantic Web Conf., Washington DC, Springer.

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

[16] Hu, M., Lim, E., Sun, A., Lauw, H. and Vuong, B. (2007). Measuring Article Quality in Wikipedia: Models and Evaluation. Proc. 16th ACM Conf. on Information and Knowledge Manage-ment, pp. 243-252, ACM Press.

[17] Hu, W., Qu, Y. (2007). Discovering Simple Mappings Between Relational Database Schemas and Ontologies. ISWC/ASWC 2007: 225-238

[18] Lenat, D. and Guha, R. (1990). Building Large Knowledge Bases. Addison-Wesley, Reading MA.

[19] McNamee, P., Dredze, M., Gerber, A., Garera, N., Finin, T., Mayfield, J., Piatko, C., Rao, D., Yarowsky, D., Dreyer, M. (2009). HLTCOE Approaches to Knowledge Base Population at TAC 2009. In proc. of Text Analysis Conference 2009.

[20] McNamee, P. and Dang, H.T. (2009). Overview of the TAC 2009 knowledge base population track. In Proceedings of the 2009 Text Analysis Conference. National Institute of Standards and Technology, November.

[21] Parr, C., Sachs, J., Han, L., Wang, T., RDF123 and Spotter: Tools for generating OWL and RDF for biodiversity data in spreadsheets and unstructured text, Proceedings of Biodiversity Information Standards Annual Conference (TDWG 2007), October 2007,

[22] Suchanek, F., Kasneci, G. and Weikum, G. (2008). Yago: A Large Ontology from Wikipedia and WordNet. Web Semantics: Science, Services and Agents on the World Wide Web, v6, n3, pp. 203-217, Elsevier.

[23] Syed, Z., Finin, T. and Joshi, A. (2008). Wikipedia as an Ontology for Describing Documents, Proc. 2nd Int. Conf. on Web-logs and Social Media, AAAI Press.

[24] Vanoirbeek, C. (1992). Formatting structured tables. In Proc. of Electronic Publishing'92, pp. 291{309. Cambridge University Press, Apr. 1992.

[25] Wu, F. and Weld, D. (2008). Automatically Refining the Wikipedia Infobox Ontology. Proc. 17th Int. Conf. on the World Wide Web, pp. 635-644, ACM, New York.

[26] Ziegler, P. and Dittrich, K., Three Decades of Data Integration - All Problems Solved?, Proceedings of the 18th IFIP World Computing Congress, pp. 3–12, 2004.