Automatically Generating Government Linked Data from Tables

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

Abstract

Most open government data is encoded and published in structured tables found in reports, on the Web, and in spreadsheets or databases. Current approaches to generating Semantic Web representations from such data requires human input to create schemas and often results in graphs that do not follow best practices for linked data. Evidence for a table's meaning can be found in its column headers, cell values, implicit relations between columns, caption and surrounding text but also requires general and domain-specific background knowledge. We describe techniques grounded in graphical models and probabilistic reasoning to infer meaning (semantics) associated with a table using background knowledge from the Linked Open Data cloud. We represent a table's meaning by mapping columns to classes in an appropriate ontology, linking cell values to literal constants, implied measurements, or entities in the linked data cloud (existing or new) and discovering or and identifying relations between columns.

Introduction

Most of the information found on the Web consists of text written in a conventional style, e.g. as news stories, blogs, reports, letters, advertisements, etc. There is also a significant amount of information encoded in structured forms like tables and spreadsheets, including stand-alone spreadsheets or table as well as tables embedded Web pages or other documents. Cafarella et al. (2008) estimated that the Web contains over 150 million high quality relational tables. In some ways, this information is easier to understand because of its structure but in other ways it is more difficult because it lacks the normal organization and context of narrative text. Both integrating or searching over this information will benefit from a better understanding of its intended meaning.

A wide variety of domains that are interesting both technically and from a business perspective have tabular data. These include medicine, healthcare, finance, e-science (e.g., biotechnology), and public policy. Key information in the literature of these domains, which can be very useful for informing public policy, is often encoded in tables. As a part of Open Data and transparency initiative, fourteen nations including the United States of America share data and information on websites such as www.data.gov in structured format like CSV, XML. As of May 2011, there are more than 390,000 raw and geospatial datasets available. This represents a large source of knowledge, yet we do not have systems that can understand and exploit this knowledge.

Many real world problems and applications can benefit from exploiting information stored in tables including evidence based medical research (Sackett et al. 1996). Its goal is to judge the efficacy of drug dosages and treatments by performing meta-analyses (i.e systematic reviews) over published literature and clinical trials. The process involves finding appropriate studies, extracting useful data from them and performing statistical analysis over the data to produce a evidence report.

Key information required to produce evidence reports include data such as patient demographics, drug dosage information, different types of drugs used, brands of the drugs used, number of patients cured with a particular dosage etc. Most of this information is encoded in tables, which are currently beyond the scope of regular text processing systems and search engines. This makes the process manual and cumbersome for medical researchers. By adding semantics to such tables, we can develop systems that can easily correlate, integrate and search over different tables from different studies to be combined for a single meta-analysis.

In this paper, we present techniques to automatically generate high quality linked data by jointly inferring the semantics of column headers, table cell values (e.g., strings and numbers), relations between columns, augmented with background knowledge from open data sources such as the Linked Open Data cloud (Bizer 2009). Our framework maps column headers to classes from an appropriate ontology, links cell values to literal constants or entities in the linked data cloud (existing or new) and discovers or and identifies relations between columns. The interpreted meaning is represented as machine understandable linked RDF assertions. We motivate our work, by its application on Open Government Data and how it can be used to generate high quality linked data from existing raw datasets.

Motivation

While there are more than 390,000 raw datasets on data.gov, only a fraction (279 or *(0.071 %)*) of

```
<rdf:Description rdf:about="#entry1">
<value>6444</value>
<label>Number of Farms</label>
<group>Farms with women principal operators</group>
<county_fips>000</county_fips>
<state_fips>01</state_fips>
<state>Alabama</state>
<rdf:type rdf:resource="http://data-gov.tw.rpi.edu/2009
/data-gov-twc.rdf#DataEntry"/>
</rdf:Description>
```

Figure 1: A part of RDF from dataset 1425 - Census of Agriculture Race, Ethnicity and Gender Profile Data from data.gov

the raw datasets are available in RDF format *(see http://www.data.gov/semantic/data/alpha).* Figure 1 shows a part of RDF representation of dataset 1425 - Census of Agriculture Race, Ethnicity and Gender Profile Data from data.gov (Dataset). The property names in the representation are column headers from the raw dataset and the values of the properties represent row values (in this case values from row one) for the respective columns.

The problem with a representation like this is it does not using existing vocabulary to annotate the raw data. Most of the column headers are mapped to properties that are local to the RDF file. Mapping column headers to classes and properties say for example from the linked data cloud, will provide more richer description as compared to the local properties. Such a representation often uses string identifiers for table cell values instead of linking them to existing entities in the linked data cloud. Linking the string cell values can further enrich the semantic representation of the data.

In the domain of open government data, Ding et al.(2010) present techniques to convert raw data (CSV,spreadsheets) to RDF. However the generated RDF does not use existing classes or properties for column headers, nor does it link cell values to entities from the linked data cloud. To generate a richer, enhanced mappings, users will need to manually specify a configuration file. Their focus has been on generating massive quantity linked government data rather quality linked government data.

Several systems have been implemented to generate Semantic Web data from databases (Sahoo et al. 2009) and spreadsheets (Han et al. 2008; Langegger and Wob 2009). Virtually all are manual or semi-automated and none have focused on automatically generating *linked* RDF data. Current systems on the Semantic Web either require users to specify the mapping to translate relational data to RDF or systems that do it automatically focus only a part of the table (like column header strings). These systems have mainly focused on relational databases or simple spreadsheets.

The key shortcoming in such systems is that they rely heavily on users and their knowledge of the Semantic Web. Most systems on the Semantic Web also do not automatically link classes and entities generated from their mapping to existing resources on the Semantic Web. The output of such systems turns out to be just "raw string data" represented as RDF, instead of generating high quality linked RDF.

Wang et al.(2011) present a table understanding system which identifies a concept to be associated with the table based on the evidence provided by the column header and strings in the "entity column" of the table. The concepts come from their knowledge base Probase (Wu et al. 2011) created from the text on the World Wide Web which can be noisy and "semantically poor" as compared to concepts from the Linked Open Data cloud.

Venetis et al.(2011) identify concepts to be associated with the column headers in a table based on the evidence provided by strings in a given column. They also identify relations between the "subject column" and other columns in the table. However they also rely on a isA database they create from the text on the Web which can be noisy as well as "semantically poor".

Limaye, Sarawagi, and Chakrabarti(2010) present a probabilistic graphical model based framework that identifies concepts to be associated with the column headers, links table cell values to entities and identifies relations between columns with Yago as a background knowledge base.

None of the current table understanding systems propose or generate any form of linked data from the inferred meaning. A key missing component in current systems is tackling literal constants. The work mentioned above will work well with string based tables. To the best of our knowledge, no work has tackled the problem on interpreting literals in tables and using them as evidence in the table interpretation framework. The framework we present is complete automated interpretation of a table that focuses on all aspects of a table - column headers, row values, relations between columns. Our framework will tackle strings as well as literals.

Approach

Consider the table shown in Figure 2. Its presents a subset of rows and column header from the raw format of dataset 1425, the *2007 Census of Agriculture Race, Ethnicity, and Gender Profiles* which contains county level census data presented by the race, ethnicity, and gender of farm operators. The column headers suggest the type of information in the columns: *State* and *County* might match classes in a target ontology such as DBpedia (Bizer et al. 2009); *State FIPS* and *County FIPS* could match properties in the same or related ontologies.

Examining just the column header may not be enough to disambiguate it correctly. Consider column five - *Group*. The string can disambiguate to groups of musicians, blood group, a group of companies or group of persons bounded by common social interests. Examining the data values in the column provides further evidence to what it is representing. The values African American, American Indian, Alaska Native, women suggest that the column is representing a set of census and ethnic groups in the United States.

The evidence that the strings are representing ethnic groups can be obtained by linking the strings to entities from

State	*State FIPS*	*County*	*County FIPS*	*Group*	*Label*	*Value*
Alabama	1	Macon	87	Farms with Black or African American operators	Value of sales of grains, oil seeds, dry beans, and dry peas (farms)	5
Arizona	4	Navajo	17	Farms with Spanish, Hispanic, or Latino operators	Value of sales of vegetables, melons, potatoes and sweet potatoes (farms)	8
Arkansas	5	Union	139	Farms with women principal operators	Total value of agricultural products sold (farms)	56
California	6	Humboldt	23	Farms with American Indian or Alaska Native operators	Days worked off farm - none	19

Figure 2: Subset of rows from the raw dataset 1425, which is from the 2007 Census of Agriculture Race, Ethnicity, and Gender Profiles and contains county level census data presented by the race, ethnicity, and gender of farm operators.

the linked open data cloud. African American would map to *dbpedia:African_American*, Alaska native would map to *dbpedia:Alaska_Natives*. Once the entities are linked, we can easily infer from DBpedia that all the strings are types of US ethnic and census groups and hence the column is representing US ethnic and census groups.

Consider column two - *State FIPS*. The data values in the column are literals. These literals can map to values of a property and the property can be associated with some other entity in the table. In this case the column maps to *dbpedia:Federal_Information_Processing_Standard_state_code*. The string along with the expanded abbreviation can help infer what the property could be. Further evidence would be provided by identifying relation between column one *State* and column two *State FIPS*. The knowledge that column one represent states (*dbpedia:AdminstrativeRegion*) further confirms that column two maps to *dbpedia:Federal_Information_Processing_Standard_state_code*. The discovered relation also further helps to associate the property with the entities in column one.

Producing an overall rich and correct semantic representation from tabular structures is a complex task that requires developing an overall understanding 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 down the process into following tasks: a) assign every column (or row header) a class label from an appropriate ontology b) link table cell values to appropriate LD entities, if possible c) discover relationships between the table columns and link them to linked data properties d) generate a linked data representation of the inferred data.

Heuristic Baseline System

We first describe a baseline system (Mulwad et al. 2010b) that we developed to evaluate the feasibility in tackling the problem. The baseline system is a sequential, multi-step framework that first maps every column header to a class from an appropriate ontology. Using the predicted class as additional evidence, it then links table cell values to entities from the Linked Data Cloud. The final step in the framework is discovering relations between table columns and generating a linked data representation of the table's meaning.

Mapping column header to class. In a typical well formed table, each column contains data of a single syntactic type (e.g., strings) that represent entities or values of a common semantic type (e.g., people, places, yearly salary in USD). The column's header, if present, may name or describe the semantic type or perhaps a relation in which the column values participate. The algorithm determines the class for a table column based on the class of the individual strings in the column. For all the cell values in every column of the table, the algorithm submits a complex query to the Wikitology (Syed and Finin 2011) knowledge base to determine the type of each cell value in the column. For every query, the KB returns a set of entities; each entity has a set of classes associated with it. Combining the classes of all the entities, produces a set of candidate classes for a column. Each class label from the set of candidate class labels is scored. The class label with the highest score is chosen as the class label to be associated with the column. We predict class labels from four vocabularies: DBpedia Ontology, Freebase, WordNet, and Yago.

Linking table cells to entities. Using the predicted class labels as additional evidence, for every table cell, the algorithm for linking table cell to entities, re-queries our KB. For every table cell, the KB returns the top *N* possible entities. For each of the top *N* entities, the algorithm generates a feature vector consisting of the entity's KB score, entity's Wikipedia page length, entity's page rank, the Levenshtein distance between the entity and the string in the query and the Dice score between the entity and the string. The set of feature vectors for each table cell are ranked using a SVM-Rank classifier. To the highest rank feature vector from SVM rank, two more features are added - the SVM rank score of the feature vector and the difference in SVM-Rank scores between the top two feature vectors. A second SVM classifier decides whether to link the table cell to this top ranked entity or not. If the evidence is not strong enough, it is likely that the table cell is a new entity not present in the KB; this step is useful in discovery of new entities in a given table. If the evidence is strong enough, the table cell is linked to the top ranked entity returned by SVM-Rank.

```
@prefix dbpedia: <http://dbpedia.org/resource/>.
@prefix dbpedia-owl: <http://dbpedia.org/ontology/>.
@prefix dbpprop: <http://dbpedia.org/property/>.
@prefix dgtwc: <http://data-gov.tw.rpi.edu/2009/data-gov-twc.rdf#>.
"State"@en is rdfs:label of dbpedia-owl:AdminstrativeRegion.
[ a dgtwc:DataEntry;
dbpedia-owl:state dbpedia:Alabama;
dbpedia:FIPS_county_code 000;
dbpedia:Federal_Information_Processing_Standard_state_code 001;
dbpedia-owl:ethnicGroup "Farm with women principal operators"@en;
dbpedia-owl:number 6444].
```

Figure 3: This N3 encoded linked data shows the representation used for a row of the table shown in Figure 2.

Discovering relation between columns. Once the table cells are linked, the framework identifies relations between table columns. For every pair of column, the algorithm generates a set of candidate relations from the relations that exist between the strings in each row of the two columns by querying DBpedia. The relation that gets majority vote is chosen as the relation between the columns.

Linked data representation. We have developed a template for annotating and representing tables as linked RDF. We choose the N3 serialization because it is compact and readable. Figure 3 shows an example of a N3 representation of a table. To associate the column header with its predicted class label, the *rdfs:label* property from RDF Schema (Brickley and Guha 2004) is used. The *rdfs:label* property is also used to associate the table cell string with its associated entity from DBpedia. To associate the table string with its type (i.e. class label of the column header), the *rdf:type* property is used. The properties and relations discovered can be used to represent every data entry row from the raw datasets. The example in Figure 3 shows an example of mapping column header to class and mapping a data entry row to linked RDF.

Evaluation of the baseline system. The baseline system was evaluated against 15 tables obtained from Google Squared, Wikipedia and from a collection of tables extracted from the Web. Excluding the columns with numbers, the 15 tables have 52 columns and 611 entities for evaluation of our algorithms. We used a subset of 23 columns for evaluation of relation identification between columns.

In the first evaluation of the algorithm for assigning class labels to columns, we compared the ranked list of possible class labels generated by the system against the list of possible class labels ranked by the evaluators. For *80.76%* of the columns the average precision between the system and evaluators list was greater than 0 which indicates that there was at least one relevant label in the top three of the system ranked list. The mean average precision for 52 columns was *0.411*.For *75%* of the columns, the recall of the algorithm was greater than or equal to 0.6. We also assessed whether our predicted class labels were reasonable based on the judgment of human subjects. *76.92 %* of the class labels predicted were considered correct by the evaluators. The accuracy in each of the four categories is shown in Figure 4.

66.12 % of the table cell strings were correctly linked by our algorithm for linking table cells. The breakdown of accuracy based on the categories is shown in Figure 4. Our dataset had 24 new entities and our algorithm was able to correctly predict for all the 24 entities as new entities not present in the KB. We did not get encouraging results for relationship identification with an accuracy of *25 %*.

Joint Inference

The baseline system makes local decision at each step of the framework. The disadvantage of such a system is that error percolates from the previous phase to the next phase which can lead to an overall poor interpretation of a table. To overcome this problem, we are developing a framework that performs joint inference over the evidence available in the table and jointly assign values to the column headers, table cell values and relations between columns.

Probabilistic graphical models (Koller and Friedman 2009) provide convenient framework for expressing a joint probability over a set of variables in a system and perform inferencing over them. Probabilistic graphical models use graph based representations to encode probability distribution over a set of variables for a given system. The nodes in such a graph represent the variables of the system and the edges represent the probabilistic interaction between the variables. Based on the graphical representation used to model the system, the graph needs to be parameterized and then an appropriate inference algorithm needs to be selected to perform inferencing over the graph.

Thus, constructing a graphical model involves the following steps: a)Identifying variables in the system b)Identifying interactions between variables and representing it as a graph c)Parametrizing the graphical structure d) Selecting an appropriate algorithm for inferencing. In this paper, we focus on modeling part of the problem: constructing and parameterizing the graphical model.

Variables in the system. The column headers, the table cell values and the relations between columns in the table represent the set of variables in the table interpretation framework. Each variable will be assigned an initial set of

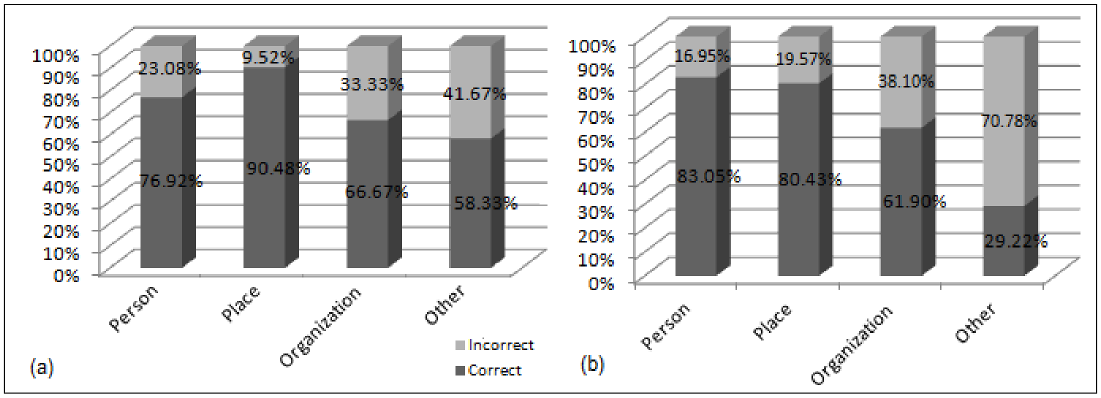

Figure 4: Category wise accuracy for (a) "column correctness" and (b) entity linking.

values that it can take on. For example the initial set of values for each of the table cell values can be determined by querying the Wikitology knowledge base.

Graphical Representation. We choose a Markov network based graphical representation to represent the interaction between the variables (see Figure 6), since the interaction between the column headers, table cell values and relation between table columns are symmetrical. The interaction between a column header and cell values in the column is captured by inserting an edge between the column header and each of the values in the column in the graph. To correctly disambiguate what a table cell value is, evidence from the rest of the values in the same row can be used. This is captured by inserting edges between every pair of cell values in a given row. Similar interaction exists between the column headers and is captured by the edges between every pair of table column headers. For simplicity of the figure, we show only one example of interaction between values in a given row and interaction between a column header and values in that column.

A parameterized Markov network. To represent the distribution associated with the graph structure, we need to parameterize the structure. One way to parameterize a Markov network is representing the graph as a factor graph. A factor graph is an undirected graph containing two types of nodes: variable nodes and factor nodes. The graph has edges only between the factor nodes and variable nodes. A factor node captures and computes the *affinity* between the variables interacting at that factor node. Variable nodes can also have associated "node potentials". Our parameterized graph (Figure 5) consists of two node potentials (associated with each of the column headers and table cell values) and three factor nodes.

The node potential for column header variable computes the affinity between the string in the column header and the class its being mapped to. The node potential for table cell variable computes the affinity between the string in the table cell and the entity its being linked to. The function of the three factor node is as follows: the first factor node computes affinity between the class being assigned to the column header and the entities linked to the cell values in the column; the second factor node computes the affinity between the classes that have been assigned to all the column headers; and the third factor node computes the affinity between the entities linked to the cell values in a given row. We are presently working on defining the functions in the factor node that will compute the affinity between the values assigned to the various variables in the system.

Discussion

The results of the baseline system show feasibility in tackling the problem. An evaluation of the system, also presented issues in the baseline system which we are address by working on framework grounded in graphical models and probabilistic reasoning. In the following sections, we present issues that will help us build a better framework to deal with open government data.

Literals. Many datasets in the domain of open government data have literal constants like numerical data. Literals are not entities that can be linked to entities from a knowledge base; but rather represent values of properties. The properties themselves can be associated with other entities in the table. The numbers in a given column also can be used as evidence in the table interpretation framework. They can be used as evidence to disambiguate the property to which the column will map to. For example if all the numbers are in the range of 0 - 100 then they could be representing age or percentage; if they are all ten digit numbers, then they could be telephone numbers and so on. We could refer to a set of rules or a knowledge base to extract this kind of information. For complete table interpretation, it will be important to take literals into consideration.

Meta data. Tables and spreadsheets often have meta data providing additional information about its meaning. This additional meta data is in the form of captions and text surrounding the table. In the case of open government data, it is in the form of dataset landing page (see (Dataset) for an example). Such meta data can prove to be useful source of

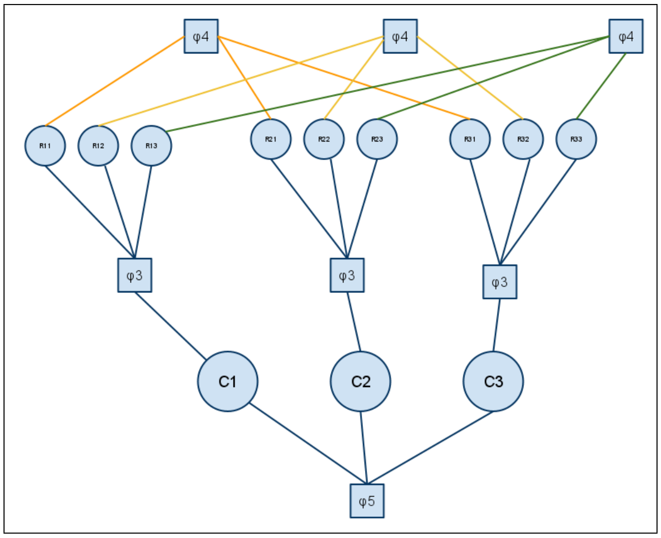

Figure 5: This parameterized Markov network consists of two node potentials (associated with each of the column headers and table cell values) and three factor nodes (represented as squares).

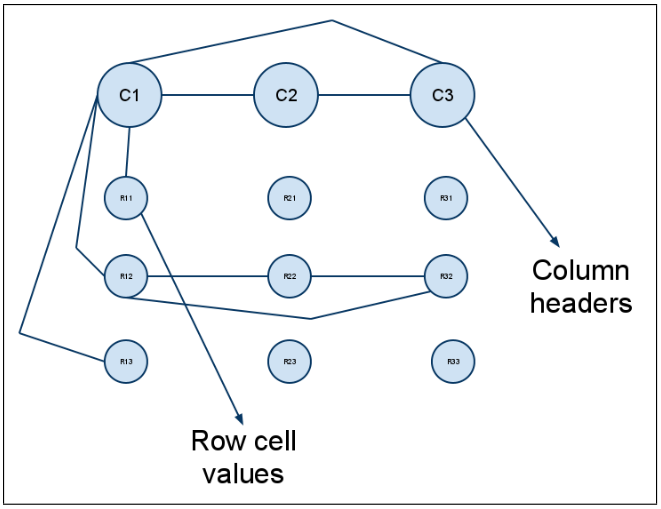

Figure 6: Interaction between variables in a table

evidence in a table interpretation framework. Table captions often encode the context in which tables need to be interpreted as well as the expanded form of abbreviations used in tables. In certain cases, captions encode units information associated with literal values in the table. Such information also can be found in the text surrounding the table.

In the case of open government data, the dataset landing page provides valuable information related to the dataset. Dataset name, description, tags, owner (agency) provide important context information for interpreting the dataset. The dataset landing page also point to additional documents which provide a detailed description of the dataset. Such documents often describe complex abbreviations used in the dataset. Parsing such documents will be useful in understanding and expanding abbreviations commonly used in government data.

We need to extend our framework to incorporate this additional evidence and context. One possible approach would incorporating it in our Wikitology query module. Using additional evidence, Wikitology can serve better results, which can lead to an improvement in the performance of the algorithms.

Human in the Loop. While we have proposed an automatic framework for interpreting and representing tabular data as linked data, it may be helpful to develop a framework with human in the loop to make the linked data more useful and customized for certain applications.

Depending upon the application wants to exploit the generated linked data, users may wish the modify the generated interpretation. Some applications may wish to have more general classes and properties describing the data as compared to specific classes. If our framework chooses a more general class, user may be able to correct it by selecting a more specific class.

The intended meaning of the table maybe encoded in the structure of the table. The reasoning behind structuring the table in one form over the other is because the creator of the table intends to convey the context in which the table should be interpreted. Presently our framework does not take structure into consideration and works well with simple tables (m columns and n rows). A human in the loop will allow modifications to any incorrect interpretation generated by the framework.

Sampling and Interpreting. Datasets from open government data tend have large number of data rows. Dataset 1425, for example, has more than 425,000 data rows. Performing either the heuristic techniques or our joint inference process over such a large table is not feasible. However, in most cases it will not be necessary since a good model of the table's meaning can be developed by looking at a modest-sized sample. This model can then be used in conjunction with separate techniques to process the rows that were not part of the sample. The details of how large a sample is needed, how to select it, and how to subsequently process the rows outside the sample remain to be determined.

Conclusion

Open government data must be converted from its native form to linked data before we can obtain the many benefits of a semantic representation. Doing so even for simple tables is a complex process that currently requires human input. Major tasks include discovering the table's implicit relations and properties, identifying the semantic classes associated with columns, and mapping cell values into appropriate literals, class instances and existing entity objects. We described preliminary work on automating these steps to produce a linked data representation of the table relative to a background linked data knowledge base. The results are promising and can be used as a component in an interactive environment that would allow a person to offer advice to improve the accuracy of the translation.

Acknowledgments

This research was supported in part by NSF awards 0326460 and 0910838, AFOSR MURI award FA9550-08-1-0265, and a gift from Microsoft Research.

References

Bizer, C.; Lehmann, J.; Kobilarov, G.; Auer, S.; Becker, C.; Cyganiak, R.; and Hellmann, S. 2009. Dbpedia - a crystallization point for the web of data. *Journal of Web Semantics* 7(3):154–165.

Bizer, C. 2009. The emerging web of linked data. *IEEE Intelligent Systems* 24(5):87–92.

Brickley, D., and Guha, R. 2004. RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation, World Wide Web Consortium.

Cafarella, M. J.; Halevy, A. Y.; Wang, Z. D.; Wu, E.; and Zhang, Y. 2008. Webtables: exploring the power of tables on the web. *PVLDB* 1(1):538–549.

Dataset 1425 - Census of Agriculture Race, Ethnicity and Gender Profile Data. http://explore.data.gov/Agriculture/Census-of-Agriculture-Race-Ethnicity-and-Gender-Pr/yd4n-fk45. 2009.

Ding, L.; DiFranzo, D.; Graves, A.; Michaelis, J. R.; Li, X.; McGuinness, D. L.; and Hendler, J. A. 2010. Twc data-gov corpus: incrementally generating linked government data from data.gov. In *Proceedings of the 19th international conference on World wide web*, WWW '10, 1383–1386. New York, NY, USA: ACM.

Han, L.; Finin, T.; Parr, C.; Sachs, J.; and Joshi, A. 2008. RDF123: from Spreadsheets to RDF. In *Proc. 7th Int. Semantic Web Conf.* Springer.

Koller, D., and Friedman, N. 2009. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press.

Langegger, A., and Wob, W. 2009. Xlwrap - querying and integrating arbitrary spreadsheets with sparql. In *8th International Semantic Web Conference (ISWC2009)*.

Limaye, G.; Sarawagi, S.; and Chakrabarti, S. 2010. Annotating and searching web tables using entities, types and relationships. In *Proc. 36th Int'l Conference on Very Large Databases*.

Mulwad, V.; Finin, T.; Syed, Z.; and Joshi, A. 2010a. T2LD: Interpreting and Representing Tables as Linked Data . In *Proceedings of the Poster and Demonstration Session at the 9th International Semantic Web Conference, CEUR Workshop Proceedings*.

Mulwad, V.; Finin, T.; Syed, Z.; and Joshi, A. 2010b. Using linked data to interpret tables. In *Proc. 1st Int. Workshop on Consuming Linked Data*.

Sackett, D.; Rosenberg, W.; Gray, J.; Haynes, R.; and Richardson, W. 1996. Evidence based medicine: what it is and what it isn't. *Bmj* 312(7023):71.

Sahoo, S. S.; Halb, W.; Hellmann, S.; Idehen, K.; Thibodeau Jr, T.; Auer, S.; Sequeda, J.; and Ezzat, A. 2009. A survey of current approaches for mapping of relational databases to rdf. Technical report, W3C.

Syed, Z., and Finin, T. 2011. *Creating and Exploiting a Hybrid Knowledge Base for Linked Data*. Revised Selected Papers Series: Communications in Computer and Information Science. Springer.

Syed, Z.; Finin, T.; Mulwad, V.; and Joshi, A. 2010. Exploiting a Web of Semantic Data for Interpreting Tables. In *Proc. 2nd Web Science Conf.*

Venetis, P.; Halevy, A.; Madhavan, J.; Pasca, M.; Shen, W.; Wu, F.; Miao, G.; and Wu, C. 2011. Recovering semantics of tables on the web. In *Proc. 37th Int. Conf, on Very Large Databases*.

Wang, J.; Shao, B.; Wang, H.; and Zhu, K. Q. 2011. Understanding tables on the web. Technical report, Microsoft Research Asia.

Wu, W.; Li, H.; Wang, H.; and Zhu, K. 2011. Towards a probabilistic taxonomy of many concepts. Technical report, Microsoft Research Asia.