Generating Linked Data by Inferring the Semantics of 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

Vast amounts of information is encoded in structured tables found in documents, on the Web, and in spreadsheets or databases. Integrating or searching over this information benefits from understanding its intended meaning. 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 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. 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 jointly infer the semantics of column headers, table cell values (e.g.,strings and numbers) and relations between columns and represent the inferred meaning as graph of RDF triples. We motivate the value of this approach using tables from the medical domain, discussing some of the challenges presented by these tables and describing techniques to tackle them.

Keywords

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

1. 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. [4] 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 like www.data.gov in structured format like CSV, XML. As of May 2011, there are nearly 390,000 raw 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 [11]. 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.

Presently medical researchers perform keyword based search on systems such as PubMed's MEDLINE which end up producing many irrelevant studies, requiring researchers to manually evaluate all of the studies to select the relevant ones. Figure 1 obtained from [5] clearly shows the huge difference in number of meta-analysis and number of clinical trials published every year. 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 com-

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

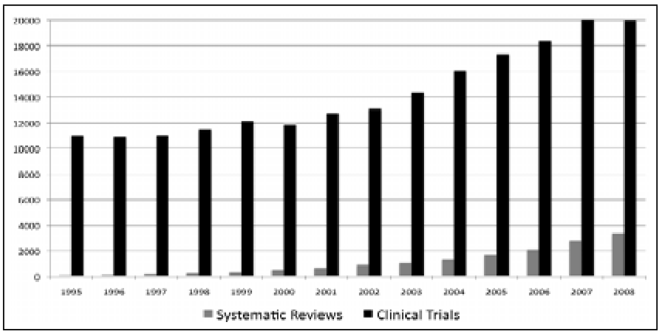

Figure 1: The number of papers reporting on systematic reviews and meta-analyses is small compared to those reporting on individual clinical trials, as shown in this data from MEDLINE.

bined for a single meta-analysis.

In this paper, we present techniques to infer the intended meaning of tables 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 [1]. Our framework maps columns 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.

2. RELATED WORK

Early work in table understanding focused on extracting tables from documents and web pages [7, 6]. While progress has been made in identifying the structure of the table, relatively little work has been focused on understanding the semantics and meaning associated with tables. Recently, three groups have focused on understanding the meaning associated with tables.

Wang et al. [16] use an approach that begins by identifying a single ‘entity column’ in a table and, based on its values and rest of the column headers, associate a concept with the table. Their work focuses only on identifying the concept to be associated with the table (i.e., with the “entity column”). The concepts come from the Probase [17] knowledge base created from the text on the World Wide Web. Such concepts may not be semantically rich as compared to concepts from DBpedia or the Linked Open Data cloud. Their work does not attempt to link the table cell values or identify relations between columns.

Ventis et al. [15] use framework associating multiple class labels (or concepts) with columns in a table. They identify relations between the ‘subject’ column and the rest of the columns in the table. Both the concept identification for columns and relation identification is based on maximum likelihood hypothesis, i.e., the best class label (or relation) is one that maximizes the probability of the values given the class label (or relation) for the column. They also rely on a *isA* database they create from the text on the Web which may not be semantically rich. Their work also does not attempt to link the table cell values.

Limaye et al. [9] describe a system based on a graphical model which maps every column header to a class from a known ontology, links table cell values to entities from a knowledge-base and identifies relations between columns. They rely on Yago [12] for background knowledge.

Current systems for interpreting tables rely on semantically poor and possibly noisy knowledge-bases. Neither do they focus on a “complete interpretation” of a table. None of the current 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. To interpret tables from specialized domains such as medical research will require incorporating modules that can understand literals.

Several systems have been implemented to generate Semantic Web data from databases and spreadsheets. Virtually all are manual or semi-automated and none has focused on automatically generating *linked* RDF data. None of the systems or methods proposed above focus on a truly complete automated interpretation of a table. 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.

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 not only tackle strings but also handle literals and work across multiple domains - web tables, medical and open government data.

3. INTERPRETING A TABLE

One might be tempted to think that regular text processing might work with tables as well. After all tables also store text. However that is not the case. It is said that tables store information in a “structured form”. It is this very structure used to represent the data, that hinders systems from understanding the intended meaning of a table. To differentiate between text processing and table processing consider the the text “Barack Hussein Obama II (born August 4, 1961) is the 44th and current President of the United States. He is the first African American to hold the office.” The overall meaning can be understood from the meaning of words in the sentence. The meaning of each word can be can be recovered from the word itself or by using context of the surrounding words.

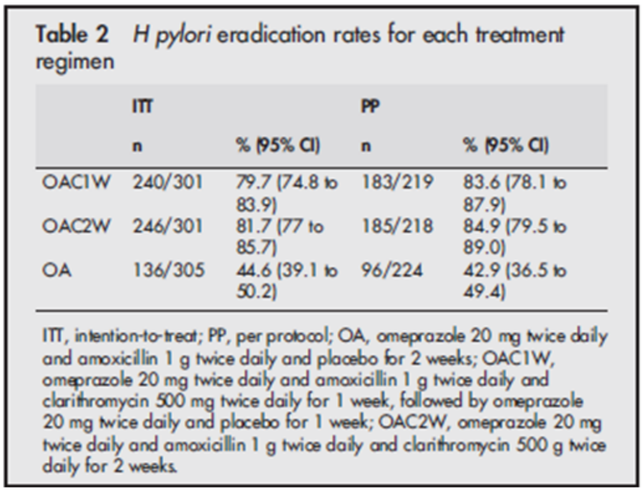

Table 2 *H pylori* eradication rates for each treatment regimen

	ITT		PP	
	n	% (95% CI)	n	% (95% CI)
OAC1W	240/301	79.7 (74.8 to 83.9)	183/219	83.6 (78.1 to 87.9)
OAC2W	246/301	81.7 (77 to 85.7)	185/218	84.9 (79.5 to 89.0)
OA	136/305	44.6 (39.1 to 50.2)	96/224	42.9 (36.5 to 49.4)

ITT, intention-to-treat; PP, per protocol; OA, omeprazole 20 mg twice daily and amoxicillin 1 g twice daily and placebo for 2 weeks; OAC1W, omeprazole 20 mg twice daily and amoxicillin 1 g twice daily and clarithromycin 500 mg twice daily for 1 week, followed by omeprazole 20 mg twice daily and placebo for 1 week; OAC2W, omeprazole 20 mg twice daily and amoxicillin 1 g twice daily and clarithromycin 500 g twice daily for 2 weeks.

Figure 2: Tables in clinical trials literature have characteristics that differ from typical, generic Web tables. They often have row headers well as column headers, most of the cell values are numeric, cell values are often structured and captions can contain detailed metadata. (From [18])

Now consider the table in Figure 2 which has data on the eradication rates for different treatment regimens for a disease, in this case *H-pylori*. The abbreviations in the row header of the table represent the different treatment regimens and the abbreviations in the column headers represent the different types of analyses used in the clinical trial. The data values in the table indicate the number of patients cured for a particular regimen and under a particular analysis. There is often additional information encoded in the table which is not directly evident, for example in this table the drugs used in the treatment are some combination of a *Proton pump inhibitor* drug and *antibiotics*.

It is clear from this example that true meaning associated with a table is often encoded in its structure, column (and row) headers of the table, the relations implicit between the various columns and the values (string or literal) in the table. Evidence to what a table means may also come from the caption associated with it as well as the free text surrounding the table.

How does one interpret what the column (or row) headers, data values intend to convey? Expanding the abbreviations in the row headers will produce strings that map to existing entities from a knowledge base. For example *OA* will map to *dbpedia:Omeprazole* and *dbpedia:Amoxicillin*. A combination of drugs in the given string indicates that the string is a type of dosage or treatment regimens. Once all the row headers are disambiguated, using information from the Linked Open Data cloud, we can infer additional information encoded in the table that all the drugs are combination of a *Proton pump inhibitor* and *antibiotics*.

The numbers in the first column of the table in Figure 2 and the way they are represented indicate that it is some form of a count/total. Using this evidence along with the evidence provided by the row headers and the expanded form of the abbreviation *ITT*, it can be inferred that the column header maps to *dbpedia:Intention_to_treat_analysis* which is a type of *yago:ClinicalTrials*. Once it is known that the row headers represent dosages and column one represents a type of analysis in a clinical trials, it can be inferred that the data values in column one represent eradication rate for some disease for a given dosage. That *some disease* in our example would be known from the caption of the table.

City	*State*	*Mayor*	*Population*
Baltimore	MD	S.C.Rawlings-Blake	640,000
Philadelphia	PA	M.Nutter	1,500,000
New York	NY	M.Bloomberg	8,400,000
Boston	MA	T.Menino	610,000

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix dbpedia: <http://dbpedia.org/resource/>.
@prefix dbpedia-owl: <http://dbpedia.org/ontology/>.
@prefix dbpprop: <http://dbpedia.org/property/>.

“City”@en is rdfs:label of dbpedia-owl:City.
“State”@en is rdfs:label of dbpedia-owl:AdminstrativeRegion.
“Baltimore”@en is rdfs:label of dbpedia:Baltimore.
dbpedia:Baltimore a dbpedia-owl:City.
“MD”@en is rdfs:label of dbpedia:Maryland.
dbpedia:Maryland a dbpedia-owl:AdministrativeRegion.
```

Figure 3: This example shows a simple table about cities in the United States and some output of the prototype system that represents the extracted information as linked data annotated with additional metadata. We use the N3 serialization of RDF for readability.

Consider the table shown in Figure 3. The column headers suggest the type of information in the columns: *city* and *state* might match classes in a target ontology such as DBpedia [2]; *mayor* and *population* could match properties in the same or related ontologies. Examining the data values, which are initially 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 that are instances of the *dbpedia-owl:Place* class. Additional analysis can automatically generate a narrower description such as major cities located in the United States(*yago:IndependentCitiesInTheUnitedStates*).

Consider the strings in column three. The string by themselves suggest that they are politicians. The column header provides additional evidence and better interpretation that the strings in column three are actually mayors. Discovering relations between columns is important as well. By identifying relation between column one and column three, we can infer that the strings in column three are mayors of cities presented in column one. Linking the table cell values to known entities enriches the table further. Linking S.C.Rawlings-Blake to *dbpedia:Stephanie_C._Rawlings-Blake*, T.Menino to *dbpedia:Thomas_Menino* , M.Nutter to *dbpedia:Michael_Nutter* we can automatically infer additional information that all three belong to the Democratic party, since the information will be associated with the linked entities.

Column four in this table presents literal values. The numbers in the column are values of the property *dbpedia-owl:populationTotal* and this property can be associated with the cities in column one. All the values in the column are in the range of 100,000. They provide evidence that the column may be representing the property population. Once relation between column one and column four is discovered, we can also look up on DBpedia, where the linked cities in column one will further confirm that the numbers represent population of the respective cities.

Producing an overall interpretation of a table 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.

4. APPROACH

In the following sections, we first present a baseline system that we developed to evaluate the feasibility in tackling the table interpretation problem. Later we present techniques for building a framework which overcomes the shortcomings in the baseline system and a framework grounded in the principles of graphical models and probabilistic reasoning. Finally we discuss challenges posed by tables in medical literature and present some techniques for dealing with them.

4.1 The Baseline System

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 [13] 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.

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. The second part of 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 [3] 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.

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 judgement 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.

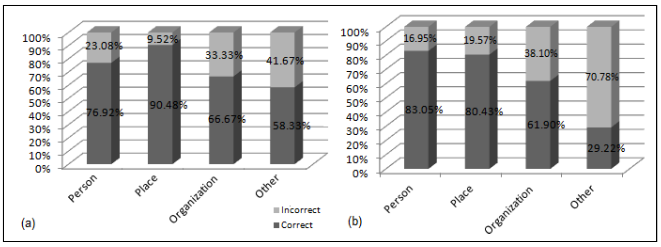

Figure 4: Category wise accuracy for (a) "column correctness" and (b) entity linking.

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* %.

4.2 Joint Inference over a table

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 [8] provide convenient framework for expressing a joint probability over a set of variables in a system and perform inferencing over them. 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 present the first three steps in constructing a graphical model for interpreting tables.

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.

Graphical Representation. We choose a Markov network based graphical representation,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.

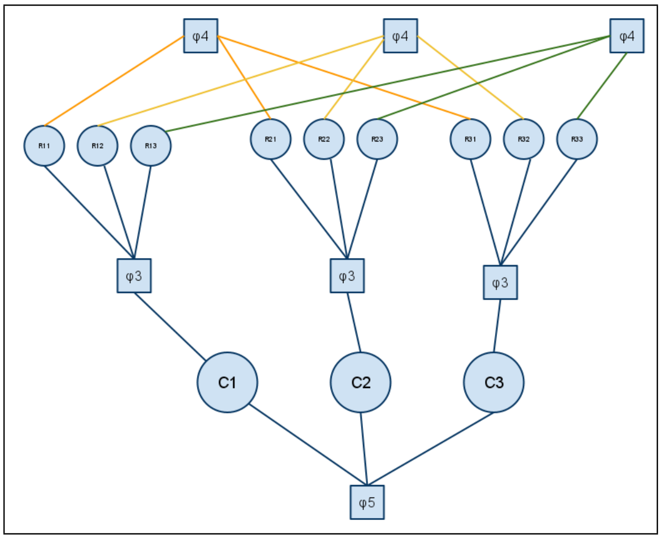

Figure 5: Parametrized Markov network. The square nodes are the factor nodes in the graph

A parametrized Markov network. To represent the distribution associated with the graph structure, we need to parametrize the structure. One way to parametrize 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 parametrized 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 value 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.

4.3 Challenges

Results of our baseline system demonstrated feasibility in interpreting tables as proposed above. In the following section we present techniques for dealing with challenges posed by tables in the medical literature and how such tables can be adapted to be processed using our existing techniques.

Abbreviations. Tables from the medical literature tend to use abbreviations a lot, primarily to represent dosages, type of analyses used in the clinical trials, types of tests conducted and so on. Like in table 2, the meaning of the abbreviations are often encoded in the table caption. A pre-processing step would involve processing the table caption to generate abbreviations and their expansions and then replacing the abbreviations in the table.

Literals. Literals pose a unique challenge, especially for tables from medical literature. We demonstrated how strings in a column can be used as evidence in a table interpretation framework. But what about literals like numerical data values in a table? To begin with, the range of numbers in a given column can start providing evidence about what the column is. For example if the numbers are in the range of 100s' then the column could be percentages or ages. The row (or column) header may have additional clues. For example, in the case of percentages, the % sign maybe associated with the numbers in the table cell or it may be present in the row (or column) header in the table.

This brings us to next thing that needs to be extracted from such tables - units associated with numbers. The units associated with numerical data is either encoded in the row (or column) header of the table or caption of the table. An important step will be identifying the individual units to be associated with numerical data in the table.

Finally numerical data is often represented in pairs. Formats like *number/count, number(%), % (number), number,unit* are some examples of how numerical data is encountered in tables in medical literature. The meaning of this format is present again in the table caption or in the table header.

Table Interpretation. A useful interpretation of tables used in meta-analysis would be identifying and linking the drugs used in the treatment, identifying the type of analyses performed, success rate, identifying and linking to the disease(s) under consideration, adverse events in the treatments if any and generating a linked data representation of it. Once the "pre-processing steps" mentioned above, some of our existing techniques can be adapted to link the row and column headers to either classes or entities from a knowledgebase and then generating the requisite linked data interpretation of the table.

5. CONCLUSION

Generating an explicit representation of the meaning implicit in tabular data will support automatic integration and more accurate search. Clues for a table's intended meaning are present in column and row headers, cell values, implicit relations between columns, and any descriptive text. We described general techniques grounded in graphical models and probabilistic reasoning to infer a table's meaning relative to a knowledge base of general and domain-specific knowledge expressed in the Semantic Web language OWL. We represent a table's meaning as a graph of OWL triples where the columns have been mapped to classes, cell values to literals, measurements, or knowledge-base entities and relations to triples. One practical usecase we are studying is representing the meaning of tables found in papers from medical journals. We discussed some of the challenges presented by these tables and described techniques to tackle them.

6. REFERENCES

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

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

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

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

[5] A. Cohen, C. Adams, J. Davis, C. Yu, P. Yu, W. Meng, L. Duggan, M. McDonagh, and N. Smalheiser. Evidence-based medicine, the essential role of systematic reviews, and the need for automated text mining tools. In *Proc. 1st ACM Int. Health Informatics Symposium*, pages 376–380. ACM, 2010.

[6] D. W. Embley, D. P. Lopresti, and G. Nagy. Notes on contemporary table recognition. In *Document Analysis Systems*, pages 164–175, 2006.

[7] M. Hurst. Towards a theory of tables. *IJDAR*, 8(2-3):123–131, 2006.

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

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

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

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

[12] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A Core of Semantic Knowledge. In *16th Int. World Wide Web Conf.*, New York, 2007. ACM Press.

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

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

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

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

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

[18] R. Zagari, G. Bianchi-Porro, R. Fiocca, G. Gasbarrini, E. Roda, and F. Bazzoli. Comparison of 1 and 2 weeks of omeprazole, amoxicillin and clarithromycin treatment for helicobacter pylori eradication: the hyper study. *Gut*, 56(4):475, 2007.