Towards Disambiguating Web Tables

Stefan Zwicklbauer, Christoph Einsiedler, Michael Granitzer, and Christin Seifert

University of Passau
Innstrae 33a, 94032 Passau, Germany
{stefan.zwicklbauer,christin.seifert,michael.granitzer}@uni-passau.de

Abstract. Web tables comprise a rich source of factual information. However, without semantic annotation of the tables' content the information is not usable for automatic integration and search. We propose a methodology to annotate table headers with semantic type information based on the content of column's cells. In our experiments on 50 tables we achieved an F_1 value of 0.55, where the accuracy greatly varies depending on the used ontology. Moreover, we found that for 94% of maximal F_1 score only 20 cells (37%) need to be considered on average. Results suggest that for table disambiguation the choice of the ontology needs to be considered and the data input size can be reduced.

Keywords: Disambiguation, Semantic Enrichment, Table Annotation

1 Introduction

Tables on web sites or in scientific papers represent a valuable source of information for the human reader. For machines the story is different: Although tables are a structural representation of knowledge, the information itself is meaningless to machines – unless it is enriched with semantic information. The Semantic Web, and specifically the Linked Open Data initiative provide means for representing any kind of knowledge semantically. If tables were enriched semantically a variety of new applications could evolve, as is the idea of Google Fusion Tables [1] where the annotation is done by humans. Tables from different sources could be automatically aggregated, compared and used to generate new insights.

In this paper we move one step towards fully-automatic semantic table annotation. We propose a simple algorithm to annotate table headers with semantic types based on the types of all cells in that column. Further, we investigate the influence of the number of cells on the accuracy of the header-type inference.

2 Related Work

Current approaches in table annotation pursue a collective approach, i.e., the annotation model encompasses entities, types and relation between types. The underlying assumption is that columns and the cells in a column have some relation in common which is modeled in the semantic knowledge base. Limaye et

al. [2] use a probabilistic graphical model to collectively annotate types (column headers), entities (cells), and semantic relations between types. They achieved an F_1 score of 0.56 on the Wiki table data set. The authors observed failures in the annotation if the corresponding correct links between entity types were not represented in the knowledge base. A recent paper employs a web search approach for entity classification [3] using the cell label as search query for a web search engine and applied text classification on the search results. Venetis et al. [4] annotate tables with class information crawled from the web based on a isA database mined with regular expressions. This web-crawled data base has a wider coverage than any modeled ontological database, but sufferes from more noise. The authors observe that using a hand-crafted ontology has a higher precision, but a high coverage is desired for their application of table search. Our approach differs in the following: First, we do not assume any relation of columns, or more specifically we do not assume that these relations are modeled in the knowledge base. Second, the only restriction we employ on the entity type is their availability in the knowledge base.

3 Approach

The general assumption behind our annotation algorithm is that the cells of a table column belong to one supertype, which we want to infer. We make no assumptions of interrelationships between columns, i.e. all columns are treated separately. Further, we assume that the tables do not have merged cells.

Let l_i $1 < i \leq n$ be the labels of non-header cells i and $E_i = \{e_i^k\}$ is the set of all possible semantic meanings of label l_i. The set T_i^k is the set of all type labels assigned to entity e_i^k. The annotation of table headers is performed in three steps (compare Figure 1):
Step 1 – Cell entity annotation: For each cell label l_i we derive a list of k possible entity candidates E_i using a search-based disambiguation method [5]. We set $k = 10$ in our experiments.
Step 2 – Entity-type resolution: For each entity candidate e_i^k in E_i a set of types is retrieved by following the `rdf:type` and `dcterms:subject` relations yielding the set of types T_i^k.
Step 3 – Type aggregation: The types assigned to the table header are the t types that occur most frequently in the set of all types of all cells $\bigcup_i \bigcup_k T_i^k$. We set $t = 1$ in our experiments, e.g. only use the most frequent type as result.

4 Experiments

Data Set. In our experiments we used dbpedia as knowledge base with type relations `rdf:type` and `dcterms:subject` from the Dublin Core Metadata Ontology[1]. We evaluated our approach on 50 tables extracted from Wikipedia pages

[1] `http://dublincore.org/documents/dcmi-terms/`

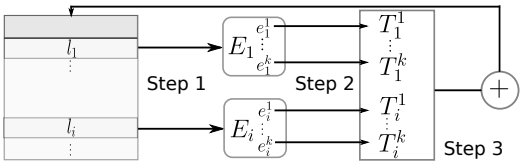

Fig. 1. Annotation process. 1) Cell labels are disambiguated to entity candidates. 2) Types of entity candidates are determined. 3) Type information is aggregated to determine the header type candidates.

Table 1. Performance for different cell annotation methods and type vocabularies. Reporting macro-averaged precision π, recall ρ, F_1.

Vocabulary	π^M	ρ^M	F_1^M
Rdf-Type	0.24	0.22	0.23
DublinCore	0.59	0.51	0.55
Rdf-Type + DublinCore	0.64	0.27	0.38

including all tables mentioned in Limaye et al. [2]. We removed all columns containing numbers or complete sentences.[2] The number of columns amounts to 132, the number of rows varies from 10 to 232 (average 54.1). We manually annotated the tables' columns yielding a total of 329 type annotations, 169 `rdf:type` and 160 `dcterms:subject`, averaging to 2.49 annotations per column header.

Overall Performance. We assessed the overall performance on the complete data set. Table 1 shows the results for three different type vocabularies (using Rdf-Type relations only, using DublinCore subjects only, or using both). In terms of precision the combined vocabulary performs best (0.64), however only slightly better than using DublinCore subjects only (0.59), whereas Rdf-type annotations are worst (0.24). For the combined approach, F_1 is low due to the low recall, which is because we have more correct results in the ground-truth but consider only the best result in the evaluation.

Table Length. In a second experiment we assessed the influence of the number of cells on the accuracy of table header disambiguation. From all 192 columns we randomly selected k cells for the cell-entity annotation step and assessed the header disambiguation accuracy using the DublinCore vocabulary. We repeated the experiment 10 times with different randomly selected cells for each $k \in \{1, 2, ..7, 8, 10, 12, 15, 20\}$. Figure 2 shows precision, recall and F_1 measure averaged over all runs. As expected for small numbers of cells the performance increases significantly when adding one more cell (e.g. from 3 to 4 cells the F_1 measure increases from 0.27 to 0.35 a growth of 26%). For larger numbers of cells there is less information gain by adding one more cell resulting in smaller increases in performance (all below 10%). Using 20 cells results in F_1 of 0.514, which is 94% of the F_1 achieved with all cells (0.547).

[2] The data set is available at `https://github.com/quhfus/table-disambiguation`

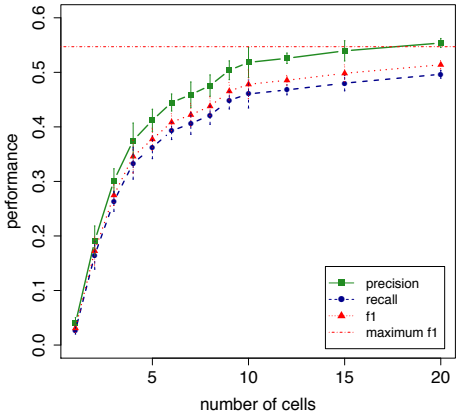

Fig. 2. Performance depending on number of cells. Showing mean and standard deviation over 10 runs, and maximum F_1 value (using all cells).

5 Conclusion and Future Work

We proposed an algorithm to annotate table headers with semantics based on the types of the column's cells. We achieved similar accuracy as previous work with more complex methods. We expect the reason for the comparable performance to be the knowledge base with more exhaustive and qualitative annotations. From our experiments it seems reasonable to use only a small number of cells for annotating the header (20 cells lead to 94% of the total achievable accuracy) if performance is an issue. We plan to exploit more relational knowledge (e.g. `same-as`) to further improve the annotations.

Acknowledgements. The presented work was developed within the CODE project funded by the EU Seventh Framework Programme, grant agreement number 296150.

References

1. Gonzalez, H., Halevy, A.Y., Jensen, C.S., Langen, A., Madhavan, J., Shapley, R., Shen, W., Goldberg-Kidon, J.: Google fusion tables: web-centered data management and collaboration. In: Proc. ACM SIGMOD, New York, ACM (2010) 1061–1066
2. Limaye, G., Sarawagi, S., Chakrabarti, S.: Annotating and searching web tables using entities, types and relationships. Proc. VLDB Endow. **3**(1-2) (September 2010) 1338–1347
3. Quercini, G., Reynaud, C.: Entity discovery and annotation in tables. In: Proc. EDBT, New York, NY, USA, ACM (2013) 693–704
4. Venetis, P., Halevy, A., Madhavan, J., Paşca, M., Shen, W., Wu, F., Miao, G., Wu, C.: Recovering semantics of tables on the web. Proc. VLDB Endow. **4**(9) (6 2011) 528–538
5. Zwicklbauer, S., Seifert, C., Granitzer, M.: Do we need entity-centric knowledge bases for entity disambiguation? In: Proc. I-KNOW. (2013)