Ontology Augmentation Through Matching with Web Tables

Oliver Lehmberg[1] and Oktie Hassanzadeh[2]

[1] University of Mannheim, B6 26, 68159 Mannheim, Germany
[2] IBM Research, Yorktown Heights, New York, U.S.A.

Abstract. In this paper, we examine the possibility of using data collected from millions of tables on the Web to extend an ontology with new attributes. There are two major challenges in using such a large number of potentially noisy tables for this task. First, table columns need to be matched to create groups of columns that represent a new (or existing) attribute for a particular class in the ontology. Second, the column groups need to be ranked according to their "usefulness" in augmenting the ontology. We show several approaches to addressing these challenges and report on the results of our extensive experiments using Web Tables from the Web Data Commons corpus, and using the DBpedia Ontology as our target ontology.

1 Introduction

The Web is a vast source of valuable knowledge that can be used to extend or augment a given ontology. Knowledge extraction from the Web is a well-studied problem and an active area of research [5, 7, 11]. While such knowledge is often extracted from textual (or semi-structured) contents using information extraction and wrapper induction techniques, there have also been attempts in using the structured data that is exposed on web pages as HTML tables [5, 14, 13].

In this paper, we examine the possibility of using Web tables to augment a given ontology with a new set of attributes. Our hypothesis is that for each class in the given ontology, there are tables on the Web describing instances of the class and their various attributes. Further, not only a large number of these attributes are not already captured in the ontology, but many are not considered "useful", i.e., may be irrelevant, inaccurate, or redundant.

The approach we take in this work is a two-step process. First, tables are matched among each other and to the target ontology, to group columns that refer to the same attribute and align them with classes and existing attributes in the target ontology. The second step ranks the column groups based on a measure of *quality* or usefulness of the group in augmenting the existing attributes in the target ontology. We perform an empirical study of the performance of this approach in using Web Tables extracted from the Common Crawl[3] to augment the properties in DBpedia ontology.

[3] http://commoncrawl.org/

2 Related Work

The pioneering work using Web tables to discover new attributes was done by Cafarella et al. in 2008 [3]. They create the so-called "attribute correlation statistics database (AcsDb)" which contains attribute counts based on the column headers in a large corpus of Web tables. From these counts, they estimate attribute occurrence probabilities. Applications for this database are a schema auto-complete function, synonym generation and a tool enabling easy join graph traversal for end-users. We extend their approach as we use clusters derived from matched columns instead of columns headers as basic unit for the statistics.

Das Sarma et al. [4] use label- and value-based schema matching methods to map Web tables to a given query table. For their "Schema Complement" operation they consider all unmapped columns and rank them using the AcsDb and the entity coverage of the input table provided by the user. Their goal is to rank complete tables by their usefulness for the complement task. While they use a matching of Web table columns to the query table to rule out existing attributes, when it comes to finding new attributes, they fall back to the AcsDb approach. In contrast to that, we calculate attribute statistics based on matched column clusters.

Lee et al. [9] extract attributes from Probase [15], Web documents, a search engine query log and DBpedia [1] and estimate their typicality using frequencies of class/attribute and instance/attribute occurrences. The extraction process is completely label-based. For the merging of attributes, they use synonyms derived from Wikipedia.

Several systems have been proposed to extend a user-specified query table with content from a corpus of Web tables [2, 16, 10]. For the task of finding new attributes, the user can specify a keyword query which describes the new attribute, so no ranking is required. Alternatively, the InfoGather system [16] and the Mannheim SearchJoin Engine [10] can generate additional attributes based on a schema matching, but both systems do not rank the resulting attributes based on a relevance score.

3 Approach

Our goal is to design an ontology augmentation solution to find new attributes for an ontology using an external source of structured data, such as a corpus of web tables. The general idea followed by existing approaches is to count attribute occurrences in the table corpus and use them to estimate probabilities for encountering these attributes. Based on these probabilities, several different metrics can be defined to assess the value of adding an attribute to the ontology (see Section 3.3). These metrics measure how likely a new attribute is to co-occur with existing attributes (in the ontology) or how consistent the resulting schema would be if the new attribute is added to the ontology.

Existing methods often consider the use-case of extending a user-provided data source in an ad-hoc setting. In the case of extending an ontology, however,

a variety of matching methods can be used to align the schema of the Web tables with the ontology. We propose to incorporate the mapping created by such methods by calculating all co-occurrence frequencies based on the mapping.

The relevance of new attributes is measured based on how frequently they co-occur with known attributes. Using exact string matching, these frequencies can be obtained from a corpus of web tables by counting, as shown by the approaches using the AcsDb [3]. When using fuzzy matching methods, however, the attributes must first be mapped among each other and then be partitioned according to their similarity values. This results in attribute clusters whose frequency can be determined by adding up the frequencies of all attributes in the cluster.

3.1 Identifying Equal Attributes

We compare several different approaches of defining attribute similarity, which will be introduced in the following.

Equality of Known Attributes. For the attributes that already exist in the ontology, we create a mapping from the web tables to the ontology. For the results in this paper, we use T2K Match [12] to map Web Tables to DBpedia ontology. This mapping defines which columns in the web tables correspond to which property in the ontology. By transitivity, all attributes which correspond to the same property are equal.

Equality of Unknown Attributes. Based on the mapping produced by T2K Match, we can group the web tables by their class in the knowledge base (blocking step) and then match all un-mapped attributes among each other. For attributes which do not exist in the ontology, we compare the following schema matching approaches:

Label-based Matching. Using the column headers of web tables as features, we evaluate using exact column header equality to find matching columns. We refer to this approach as "Exact" in our experiments. We further evaluate "String Similarity", which calculates the similarity of the column headers using the Generalised Jaccard Similarity with Edit Distance as inner similarity function.

Instance-based Equality. We further evaluate similarities which are created by the instance-based schema matcher of the Helix System [6]. We refer to the configuration using cosine similarity as "Helix Cosine" and to the configuration using containment similarity as "Helix Containment".

Key/Value-based Equality. The Key/Value-based equality "Key/Value Matching" compares only those values of two columns, which are mapped to the same instance in the ontology. This means, two columns are equivalent only if they contain similar values for the same instances. To obtain the similarity values, we use the value-based matching component of T2K Match.

3.2 Similarity Graph Partitioning

After the calculation of the similarity values, we must decide which set of columns refers to the same attribute. For attributes that already exist in the ontology, all columns with a similarity value which is above a threshold are considered to be equal to the existing attribute. However, for attributes which do not exist in the ontology, there is no such central attribute. We hence evaluate different partitioning strategies [8] for the graph that is defined by the similarities among the columns of the web tables.

Connected Components. We calculate the connected components on the similarity graph. Each resulting component is a cluster.

Center. The Center algorithm uses the list of similarities sorted in descending order to create star-shaped clusters. The first time a node is encountered in the sorted list, it becomes the center of a cluster. Any other node appearing in a similarity pair with this node is then assigned to the cluster having the former node as center.

MergeCenter. The MergeCenter algorithm is similar to the Center algorithm, but has one extension. This extension is that if a node is similar to the centers of two different clusters, these clusters are merged together.

3.3 Attribute Ranking

After defining attribute equality, we can now specify how the relevance of new attributes is determined. All compared ranking methods are defined based on attribute cooccurrence probabilities, which we define according to Cafarella et al. [3].

Let a schema $s \in S$ be a set of attributes and S be the set of all schemata. A table has this schema if its columns correspond to the attributes (based on the schema mapping), regardless of their order and column header. Let $freq(s)$ be the number of tables with schema s in the corpus and $schema\_freq(a)$ be the number of tables that contain attribute a:

$$schema\_freq(a) = \sum_{\{s|s \in S \wedge a \in s\}} freq(s) \tag{1}$$

Then the probability of encountering a in any table in the corpus is

$$p(a) = \frac{schema\_freq(a)}{\sum_{s \in S} freq(s)} \tag{2}$$

The number of tables that contain two attributes a_1, a_2 is defined analogously as $schema\_freq(a_1, a_2)$. The conditional probability of seeing attribute a_1 given a_2 is

$$p(a_1|a_2) = \frac{schema\_freq(a_1, a_2)}{schema\_freq(a_2)} \tag{3}$$

And the joint probability is

$$p(a_1, a_2) = \frac{schema\_freq(a_1, a_2)}{\sum_{s \in S} freq(s)} \tag{4}$$

Attribute Ranking Methods. We now define the methods that are used to calculate a score for each attribute, which is then used to rank all unknown attributes. A higher score indicates a higher relevance of the attribute for the schema extension task.

Conditional Probability based on Class. Given the class C in the ontology, how likely is it to encounter the attribute a [9]. If each schema is mapped to a class C and $schema\_freq(a, C)$ is the number of tables mapped to C that contain a, we can define the conditional probability of encountering an attribute based on the class as in Equation 5, where S_C is the schema of class C. This measure only considers the class mapping of the web tables, irrespective of the presence of known attributes in the same web table.

$$p(a|C) = \frac{schema\_freq(a, C)}{\sum_{a_2 \in S_C} schema\_freq(a_2, C)} \tag{5}$$

Schema Consistency. This measure reflects the likelihood of seeing a new attribute together with the existing attributes [4]. It is based on the conditional probability derived from the cooccurrence statistics. This measure considers all known attributes which co-occur with the new attribute a, i.e., the more known attributes co-occur, the higher the score.

$$SchemaConsistency(a, s) = \frac{1}{|s|} \cdot \sum_{a_2 \in s} p(a|a_2) \tag{6}$$

Schema Coherency. Based on Point-wise Mutual Information (PMI), schema coherency is the average of the PMI scores of all possible attribute combinations [3]. The PMI score of two attributes is positive if the attributes are correlated, zero if they are independent, and negative if they are negatively correlated.

$$SchemaCoherency(a, s) = \frac{1}{|s|} \cdot \sum_{a_1 \in s} npmi(a_1, a) \tag{7}$$

$$npmi(a_1, a_2) = -\frac{1}{\log p(a_1, a_2)} \cdot \log \frac{p(a_1, a_2)}{p(a_1) \cdot p(a_2)} \tag{8}$$

4 Experiments

4.1 Experiments on T2D Gold Standard

Our first set of experiments are performed on the T2D Gold Standard [12], which was originally developed to evaluate systems for the web table to knowledge base matching task (using DBpedia as the knowledge base).

Identifying equal Attributes We evaluate the different matching and partitioning approaches introduced in Section 3.1. The gold standard contains mappings from the web table columns to properties in the ontology. As we are interested in finding partitions of columns which represent the same attribute, we create one partition for each property in the ontology, which contains all columns which are mapped to this property. We then apply the different methods to all columns of the web tables in the gold standard and measure the degree to which we can reconstruct these partitions.

Figure 1 shows a comparison of the different partitioning approaches. The x-axis depicts the similarity threshold and the y-axis shows the resulting F1-score. We can see that the best performance is achieve with rather low thresholds and the Center algorithm.

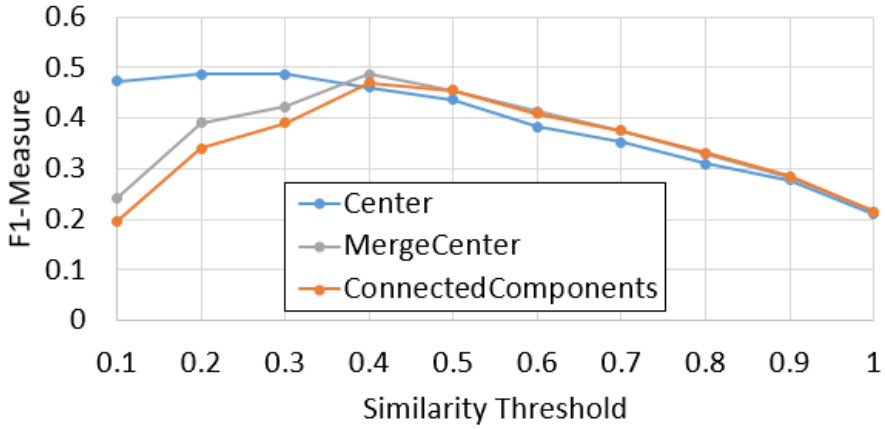

Fig. 1. Evaluation of similarity graph partitioning methods.

Figure 2 shows the quality of the Clusterings using different matching approaches. Again, the x-axis depicts the similarity threshold and the y-axis shows the resulting F1-score. We see that the label-based matching with string similarity outperforms the instance-based approaches. The reason for the good performance of the label-based approach is that the web tables are grouped by the class in the ontology to which they are mapped, and hence column headers are

in most cases not ambiguous. The rather bad performance of the instance-based approaches is explained by the fact that web tables usually only have very few rows and there might not be enough overlap among the columns from different tables.

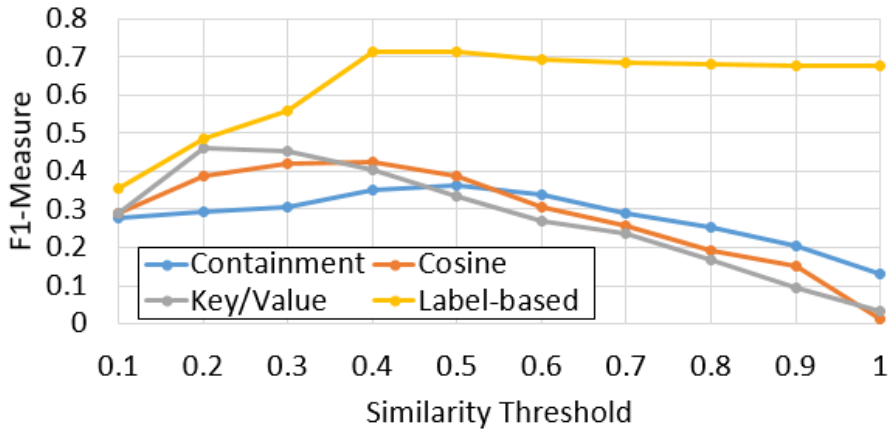

Fig. 2. Evaluation of matching methods.

Combining the instance-based and label-based approach into a hybrid matcher did not significantly improve the performance compared to the label-based approach. Closer inspection of the results showed that this is due to the used gold standard, which contains mostly tables with columns labels of high quality.

Attribute Ranking For a subset of the T2D tables, those mapped to the country class, we manually label all columns with either "useful" or "not useful". In total, this subset contains 207 columns, of which 86 are annotated as "useful". We then evaluate the performance of the different ranking methods. Figure 3 shows the precision@K and recall@K achieved by the different ranking approaches. In addition to the ranking methods described in Section 3.3, we further evaluate each of the ranking methods in a variant that is weighted by PageRank. The intuition is that web pages with a high PageRank likely contain useful content and hence the web tables on these pages also contain relevant attributes. The used PageRank values are obtained from the publicly available Common Crawl WWW Ranking.[4] For each partition of columns, we use the maximum PageRank of all source web pages and multiply it with the score that was calculated by the ranking method. Among the different ranking methods, schema consistency performs best, followed by schema coherency. The variations with PageRank

[4] http://wwwranking.webdatacommons.org/

perform worst, which might be caused by the rather small number of web sites in the gold standard.

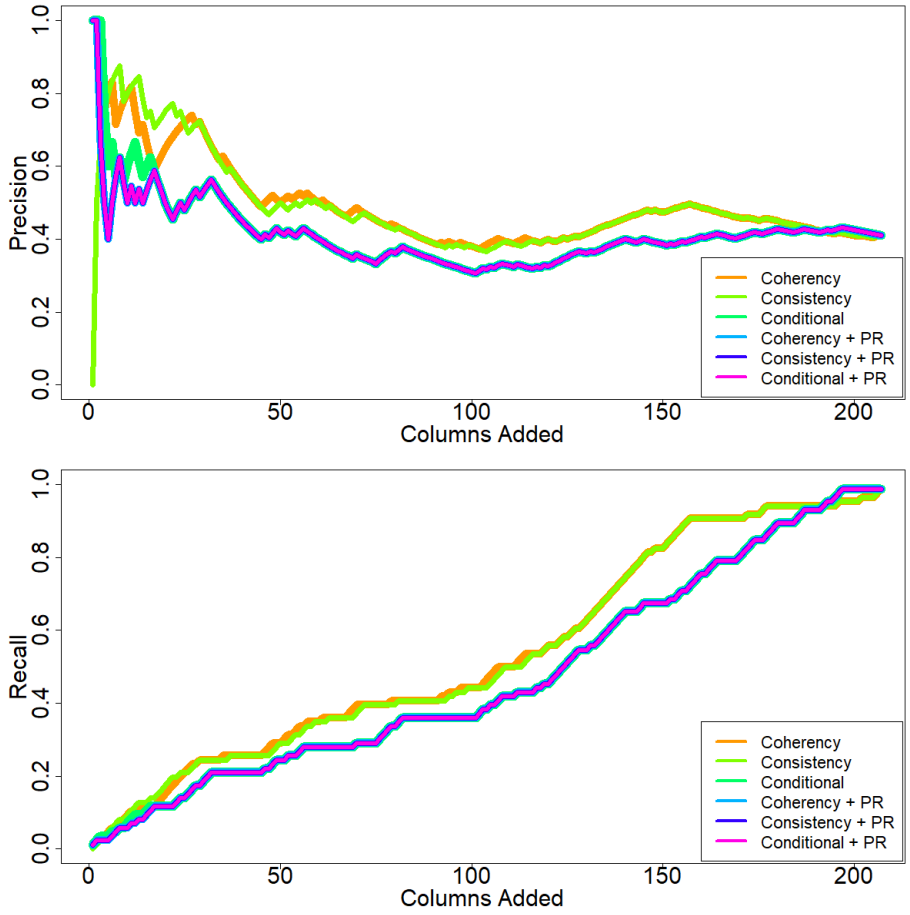

Fig. 3. Precision@K and Recall@K achieved by the different ranking methods using the Key/Value matcher.

Remove one Attribute Experiment The assessment of the usefulness of an attribute can be subjective. Hence we design another experiment, where we remove one existing attribute from the ontology for several classes. As this attribute was already existing, we can objectively say that it is useful. We then measure the quality of the first cluster that resembles this attribute and also the rank at which we can find it in the output. We use the following classes and attributes in this experiment: *Company* (*industry*), *Country* (*population*), *Film*

(*year*), *Mountain* (*height*), *Plant* (*family*), *VideoGame* (*genre*). The left chart in Figure 4 shows the average rank of the first attribute cluster which matches the removed attribute over all ranking methods by matcher. The bar "No Matching" shows the result of neither using correspondences to the ontology nor any of the matching approaches, i.e., attributes are equal only if their column headers match exactly. The results without prior mapping knowledge show the importance of matching attributes before calculating the ranking functions. Without mapping knowledge, attribute frequencies are under-estimated, and the respective attribute is ranked too low. The right chart in Figure 4 shows the average rank over all matching methods by ranking method. Again, the schema consistency ranking performs best and the variations including PageRank consistently perform worse.

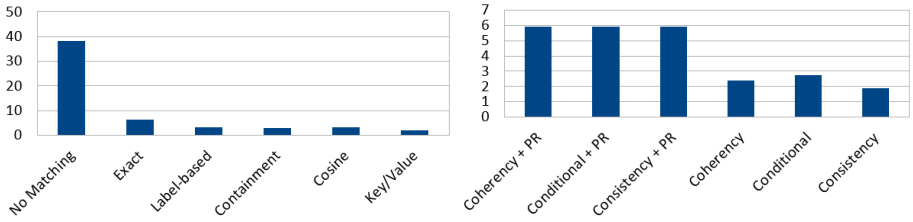

Fig. 4. Rank of the first cluster matching the removed attribute. Left: by matcher. Right: by ranking method.

4.2 Experiments on WDC Table Corpus

We now repeat our experiments on the WDC Web Tables Corpus 2012[5], which contains 147 million relational web tables. To give an overall impression of the full corpus, Figure 5 shows the number of new columns and clusters that we can generate for selected classes. These numbers show the large amount of potentially new attributes that can be found in the corpus.

Attribute Ranking As we have no gold standard for the full corpus, we manually annotate the top 15 ranked clusters for each ranking method for several classes with either "useful" or "not useful". Figure 5 shows the performance of each method averaged over all classes in terms of precision@15. The results show again that the schema coherency and consistency measures outperform the conditional measure. This indicates that attribute co-occurrence is a stronger signal than pure frequency of attributes, even if conditioned with a class from the ontology.

[5] http://webdatacommons.org/webtables/index.html

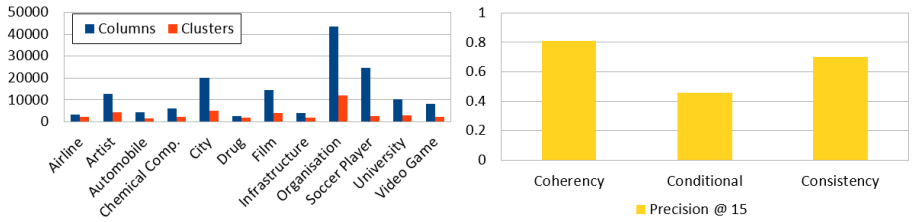

Fig. 5. Left: Number of attributes and clusters, that do not exist in the ontology. Right: Manual evaluation of the usefulness of new attributes.

Remove one Attribute Experiment Again, to have a more objective view on the results, we remove one attribute from DBpedia as before and find the top-ranked attribute cluster which matches the removed attribute. The left chart in Figure 6 shows the rank of these clusters by matcher and the right chart by ranking method. Concerning the matching approach, we now find that the label-based and key/value-based methods achieve comparable results. The difference here to the experiment on the gold standard is that we take into account a much larger number of tables and hence have more variety and a more realistic sample of the data quality. If we compare both of the matching approaches to a baseline approach ("No Matching"), which does not use the prior knowledge of the mappings to the ontology, we can again see that the ranking results are worse. Looking at the different ranking methods, we see a result that differs from the previous results. The Conditional Probability ranking now performs best. A possible explanation is that the attributes that we removed are quite common. Hence, many tables have such attributes and the ranking by frequency is sufficient. Another interesting fact is that now the PageRank makes a difference. Although it is still worse than without, we can presume that a reasonable evaluation of a ranking method incorporating the PageRank requires the use of a large corpus.

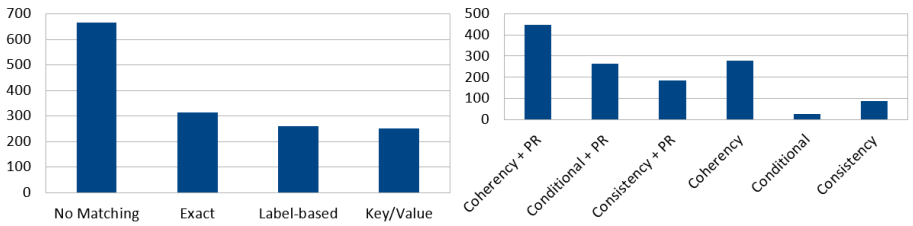

Fig. 6. Left: Rank of the first cluster matching the removed attribute by matcher. Right: Rank of the first cluster matching the removed attribute.

5 Conclusion & Future Work

In summary, the results of our experiments show that:

- It is feasible to use a large corpus of structured data from the Web to augment an ontology. In particular, we are able to augment a general-domain ontology such as DBpedia with millions of Web Tables extracted from the Web. We manually verified that a number of attributes ranked highly by our algorithms were strong candidates for augmenting the DBpedia ontology, and such augmentations would enable new applications of the ontology.
- Our results comparing different algorithms were mixed and without a clear winner across all the experiments. The size of the gold standard and the classes chosen for manual verification clearly affected the relative performance of the algorithms. This calls for larger benchmarks, more comprehensive evaluation, and hybrid/ensemble methods that effectively take advantage of the benefits of each of the algorithms.

Future work also includes: 1) extending our framework to include more advanced matching techniques particularly from recent work in ontology matching 2) evaluation on other sources of structured data (e.g., open data portals such as data.gov), and other ontologies 3) Extending the augmentation to relations and classes of the ontology 4) using the same quality metrics for ontology augmentation from textual and semi-structured sources and an evaluation of how well structured data on the Web can contribute to building and augmenting an ontology, comparing with the textual and semi-structured sources.

References

1. Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. *Dbpedia: A nucleus for a web of open data*. Springer, 2007.
2. Michael J Cafarella, Alon Halevy, and Nodira Khoussainova. Data integration for the relational web. *Proceedings of the VLDB Endowment*, 2(1):1090–1101, 2009.
3. Michael J Cafarella, Alon Halevy, Daisy Zhe Wang, Eugene Wu, and Yang Zhang. Webtables: exploring the power of tables on the web. *Proceedings of the VLDB Endowment*, 1(1):538–549, 2008.
4. Anish Das Sarma, Lujun Fang, Nitin Gupta, Alon Halevy, Hongrae Lee, Fei Wu, Reynold Xin, and Cong Yu. Finding related tables. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 817–828. ACM, 2012.
5. Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 601–610. ACM, 2014.
6. Jason Ellis, Achille Fokoue, Oktie Hassanzadeh, Anastasios Kementsietsidis, Kavitha Srinivas, and Michael J Ward. Exploring big data with helix: Finding needles in a big haystack. *ACM SIGMOD Record*, 43(4):43–54, 2015.

7. Rahul Gupta, Alon Halevy, Xuezhi Wang, Steven Euijong Whang, and Fei Wu. Biperpedia: An ontology for search applications. *Proceedings of the VLDB Endowment*, 7(7):505–516, 2014.
8. Oktie Hassanzadeh, Fei Chiang, Hyun Chul Lee, and Renée J Miller. Framework for evaluating clustering algorithms in duplicate detection. *Proceedings of the VLDB Endowment*, 2(1):1282–1293, 2009.
9. Taesung Lee, Zhongyuan Wang, Haixun Wang, and Seung-won Hwang. Attribute extraction and scoring: A probabilistic approach. In *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*, pages 194–205. IEEE, 2013.
10. Oliver Lehmberg, Dominique Ritze, Petar Ristoski, Robert Meusel, Heiko Paulheim, and Christian Bizer. The mannheim search join engine. *Web Semantics: Science, Services and Agents on the World Wide Web*, 2015.
11. Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2016.
12. Dominique Ritze, Oliver Lehmberg, and Christian Bizer. Matching html tables to dbpedia. In *Proceedings of the 5th International Conference on Web Intelligence, Mining and Semantics*, page 10. ACM, 2015.
13. Dominique Ritze, Oliver Lehmberg, Yaser Oulabi, and Christian Bizer. Profiling the potential of web tables for augmenting cross-domain knowledge bases. In *Proceedings of the 25th international conference on world wide web*, pages 251–261. International World Wide Web Conferences Steering Committee, 2016.
14. Petros Venetis, Alon Halevy, Jayant Madhavan, Marius Paşca, Warren Shen, Fei Wu, Gengxin Miao, and Chung Wu. Recovering semantics of tables on the web. *Proceedings of the VLDB Endowment*, 4(9):528–538, 2011.
15. Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Q Zhu. Probase: A probabilistic taxonomy for text understanding. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 481–492. ACM, 2012.
16. Mohamed Yakout, Kris Ganjam, Kaushik Chakrabarti, and Surajit Chaudhuri. Infogather: entity augmentation and attribute discovery by holistic matching with web tables. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 97–108. ACM, 2012.