Fusing Time-Dependent Web Table Data

Yaser Oulabi, Robert Meusel, Christian Bizer
Data and Web Science Group, University of Mannheim, Germany
{yaser, robert, chris}@informatik.uni-mannheim.de

ABSTRACT
A subset of the HTML tables on the Web contains relational data. The data in these tables covers a multitude of topics and is thus very useful for complementing or validating cross-domain knowledge bases, such as DBpedia, YAGO, or the Google Knowledge Graph. A large fraction of the data in these knowledge bases is time-dependent, meaning that the correctness of an attribute value depends on a point in time. Fusing data from web tables in order to determine correct values for time-dependent attributes is challenging as most web tables do not contain timestamp information. A possibility to deal with this sparsity is to exploit timestamps which appear in different locations on the web page around the table. But as these timestamps might not apply to the web table value in question, this approach introduces noise. This paper investigates the extent to which the performance of data fusion strategies that rely on voting, PageRank, and Knowledge-Based-Trust can be improved by incorporating noisy and sparse timestamp information. For this, we present a machine-learning-based approach which considers different types of noisy timestamps in the data fusion process, and experiment with propagating timestamp information between web tables in order to overcome sparsity. We evaluate the data fusion strategies using a large public corpus of web tables and a public gold standard of time-dependent attribute values. We find that our methods effectively choose and weigh timestamp information per attribute and reduce sparsity using propagation. By incorporating timestamp information into data fusion strategies that previously did not exploit temporal meta information, we are able to increase F1-measure on average by 5%.

1. INTRODUCTION
The Web contains large amounts of valuable information covering a wide range of different topics. Besides being represented as text, information on the Web might also be represented in the form of relational HTML tables, referred to as web tables [3]. This relational data is potentially very useful to extend or validate multi-domain knowledge bases (KBs), such as DBpedia, YAGO, or the Google Knowledge Graph. Such KBs are being employed for an increasing number of application domains including natural language processing, web search, and question answering [11, 13]. The process of consolidating information from a large number of web tables into a single KB requires two basic steps: First, the web tables are matched to the knowledge base [12]. Second, a data fusion strategy is applied in order to select the web table values that are most likely correct [6, 13, 15, 16]. A specific challenge in the data fusion step is that a large fraction of the web table data is time-dependent, e.g., the population of a city or the team of an athlete, and thus only valid given a certain temporal scope.

Two major challenges arise for the fusion of time-dependent web table data: First, web tables provide temporal information for specific attribute values only in few cases and therefore this information are rather sparse. Second, the other cells of the web table as well as the page around the table might contain more temporal information, but it is challenging to determine whether this information applies to a specific web table value or not. Therefore, data fusion strategies for time-dependent web table data should be able to deal with the sparsity and noisiness of temporal meta information.

Existing fusion strategies for web data utilize weighted voting [15], link structure [16], or ground truth [6, 13] as quality indicators, but in most cases do not explicitly consider temporal meta information as well as the sparsity and noisiness of such information. We propose data fusion strategies that are capable of utilizing noisy and sparse timestamp information for fusing time-dependent data from multiple web tables. Our approach employs timestamp propagation in order to reduce timestamp sparsity, and we use a machine-learning-based approach to assign weights to possibly noisy timestamps extracted from a web table and its context. The two mechanisms are then combined with established fusion strategies such as voting, PageRank (PR) [2], and Knowledge-based Trust (KBT) [6], in order to determine to which extent the performance of the existing methods can be improved by considering temporal meta-information.

The contributions of this work are: (1) insights to which extent the consideration of the temporal dimension of attributes and the usage of timestamp information during fusion improves the performance for time-dependent attributes, (2) an approach that successfully propagates timestamps to reduce timestamp sparsity, (3) an effective approach for selecting and weighting different types of possibly noisy time-
3. METHODOLOGY

The fusion strategies presented in this paper assume as input a set of web table values which are matched to a single triple of a KB. We define a triple as combination of an entity, an attribute, and a value. The fusion strategy should return the value from all the given matches for this particular triple of the KB that is most likely correct. In addition, whenever possible, the fusion strategy should be able to find the most correct value given a specific temporal information. We use HeidelTime [14] to detect temporal information within web tables and also, in contrast to other works (e.g. [18]), in the HTML pages of the web tables. We extract the year from each detected temporal information. The different locations of timestamps in the web tables and the HTML pages of the table are described in Subsection 3.2.

3.1 Overall Fusion Methodology

All the fusion strategies presented in Section 5 are based upon a single underlying fusion methodology, consisting of four steps:

**Scoring:** All fusion strategies calculate scores for each matched value extracted from the web tables. The individual strategies are scoring functions that influence the fusion process solely by scoring the individual matched values.

**Filtering:** Based on their scores, values are filtered using a learned threshold. The filtering is an important step as it strongly influences a possible precision/recall trade-off. The trade-off is characterized by the proportion of triples filtered out by a strategy. A filter with a higher threshold increases precision at a possible cost of recall and vice-versa. To address this trade-off, we optimize our thresholds for the F1-measure. We learn the threshold by successively increasing the threshold in 0.05 steps from 0.0 to 1.0. We also divide the data used for learning into four parts and learn four different thresholds, which we then average to prevent overfitting. Thresholds are learned per fusion strategy and per attribute-class combination.

**Grouping:** For a given triple, we group all remaining matched values that are considered to be equal and sum the scores for each group. We employ datatype-specific similarity measures\(^1\) to compare values and decide if two values are equal using an equivalence threshold. We group string values using a combination of tokenization, overlap similarity, and Levenshtein similarity. We learn the threshold for values of types string upfront using an independent data sample. Reference types need to refer to the exact same entity to be equivalent, while for numeric values, we require a normalized similarity of at least 0.98. Date types are compared using a boolean similarity, where we only compare the parts of the date (e.g. month or year) which are available for both date values.

**Selection:** We select the group with the highest summed score and determine a value that represents that group. As values in a group can differ slightly, while still determined as equivalent, we use simple datatype fusers to select one value that represents this group. For string values, we choose the value that occurs most often, for date and numeric values we use the median. All values in groups of reference types refer to the same entity.

\(^1\)Values can be of one of four datatypes: string, date, numeric, and reference, where a reference is a link to another entity.
3.2 Timestamp Strategies

To calculate scores for web table values with timestamps, we compare the temporal meta information (year) of the web table value \((t_{\text{w}})\) with the timestamp of the corresponding triple in the KB \((t_{\text{KB}})\). The final score is calculated by

\[
\text{max}(1 - \frac{|t_{\text{w}} - t_{\text{KB}}|}{4}, 0)
\]

, meaning that it is 1.0 if the years are equal, and discounted by \(\frac{1}{4}\) for each year of difference, where the lowest score is 0.0. With this score timestamps with the exact correct year are scored the highest, while timestamps with very close years are scored higher than values with a timestamp that is very distant to the one in the KB. If no timestamp is found, a score of 0.0 is returned.

The described scoring method requires one single year value for each value. As discussed above, temporal information can be extracted from multiple locations within and around the web table, e.g., in the page title or the cell next to the value. This makes the selection of the most likely correct timestamp difficult. In order to overcome this, we generate scores for different types of timestamps. We then combine these using weights learned through linear regression. The web table value is then assigned the combined score. The different timestamp types are:

- **ColumnHeader**: Year is extracted from the column header. Most recent year is chosen, if multiple exist.
- **RowCell**: Year is extracted from cells in the same row of the matched value. Most recent year is chosen, if multiple exist.
- **PageTitle**: Year is extracted from the page title of website. Most recent year is chosen, if multiple exist.
- **TableTitle**: Year is extracted from the title of the table. Most recent year is chosen, if multiple exist.
- **TableContext**: Year is extracted from text that surrounds the table. Most frequent year is chosen, if multiple exist.
- **Hierarchy**: A hierarchy type with the order ColumnHeader, RowCell, TableTitle and PageTitle.
- **FullHierarchy**: A hierarchy type with the order Hierarchy and TableContext.
- **TableHierarchy**: A hierarchy type with the order TableTitle and TableContext.

For all of these types, except for TableContext, we expect them to contain only one year value. In case of multiple values (e.g. within the header of the table), we choose the most recent value. For the TableContext, where we expect multiple year values, we choose the most frequent one.

In addition, we generated three hierarchy timestamp types, which we derive from the simple types. This is done by using a hierarchy, where the year of one of the simple timestamp types is returned in a predetermined order. If the year highest in the hierarchy does not exist, the next year is returned. The intuition behind these derived types is that a certain location for a timestamp is more likely to be relevant than another. For example a timestamp found in a column header is more likely to be relevant to a timestamp found in the text after the table, e.g. in the document copyright statement. Through the hierarchy, we are able to create a score that could return the year that is in our opinion the most relevant to the matched value.

3.3 Timestamp Propagation

Given that timestamp information is very sparse, we implemented a method of timestamp propagation, which is able to induce temporal information for web table values where timestamp information is absent.

Imagine a web table describing countries and their population without giving any timestamp information. Such a table might not be considered by a data fusion strategy for time-dependent attributes as no temporal information is present. Imagine a second table containing very similar population numbers for the same countries as well as explicit timestamps. The central idea of timestamp propagation is to add the existing timestamp information to all other tables that provide very similar population numbers for the same countries as these numbers likely refer to the same point in time as the numbers in the table containing the timestamps.

For deciding whether or not to propagate timestamp information, we use the equivalence functions described in the previous subsection. Given a certain triple for which we matched a set of values from the web tables, we cluster all matches using the equivalence measures. For each timestamp type, we detect values in each cluster where we could find a timestamp and identify the timestamp that appears most frequently. We then propagate this timestamp to all values in that cluster that do not have their own timestamp.

3.4 Weighting of Temporal Information

As already mentioned in Subsection 3.2, for each value, we are possibly able to extract more than one timestamp containing contradicting information. Furthermore, depending on the attribute, one or another timestamp might be more likely correct. For example, in the case of the country population number, a timestamp within an adjacent cell might contain the official founding year, where the year of the census is stated in the PageTitle. In contrast, in the case of the NFL athlete number, the cell next to the value might contain the season information and therefore be the most relevant for this attribute.

A manual determination of the importance of the different timestamp types for each class-attribute combination for potentially thousands of web tables is not applicable. Therefore, we train weighted-multiple-linear regression models\(^2\) to determine the importance (weights) of each timestamp type per class-attribute combination.

4. EXPERIMENTAL SETUP

In this section, we introduce the data as well as the ground truth that is used to evaluate the different fusion strategies. All methods presented in this paper are implemented as part of the T2K Framework.\(^3\)

4.1 Data Corpus

For our experiments, we use the WDC Web Table Corpus, Version 2015\(^4\), which was extracted from the July 2015 Common Crawl web corpus. The original corpus contains 1.78 billion HTML pages. The extracted web table corpus consists of 90 million relational HTML tables [7]. In addi-

\(^2\)We use WEKA (http://www.cs.waikato.ac.nz/ml/weka/) to train the regression models using 5-fold cross-validation.

\(^3\)Code and description available at http://dws.informatik.uni-mannheim.de/en/research/T2K

\(^4\)http://webdatacommons.org/webtables/\#toc2
tion the corpus also contains contextual information from which we extracted temporal information.

We selected four time-dependent attributes to extract from the corpus and evaluate the different fusion strategies on. We use the T2K Framework [12] in order to match the classes and attributes to the KB. Table 1 provides an overview of the four-time dependent attributes including their extracted timestamp information. The column with the number of matched values shows the overall number of individual values present for that specific attributes. The remaining columns contain the number of values which are accompanied by a timestamp correspondence of the given type. The first four types of timestamps are extracted from the context and the page of the table. Timestamps before table and after table are timestamps extracted from text close to the table. This includes 200 characters before and after the table. On page are timestamps found anywhere on the page, which we do not use in our timestamp types. We also extracted timestamps from the title of the page, the column header of the matched value as well as other cells of the same row.

Overall, we observe that most timestamps are found in text around the extracted web table. The second most frequent source is the page title. Few timestamps are found in the table title and for most attributes also in the column header. For NFL athletes, we also detect a lot of timestamps in the cells of the same row of the matched value. In summary, we find that already the distribution of timestamps over the different types depends on the class-attribute combination, which underlines the need of a different weighting function for each of the combinations.

The last column of the table shows the percentual increase of timestamps we could generate by using the described timestamp propagation approach. Except for the NFL Athlete-Number, we could create at least 12% additional temporal information.

4.2 Ground Truth

As ground truth (gold standard) for the evaluation of the performance of the different fusion strategies, we use facts about countries and NFL athletes from the Worldbank\(^5\) and footballldb.com.\(^6\)

We selected 212 different countries in the year 2008 and 2014 with evolving values for population and population density between the years. For NFL athletes, we found 737 players, which switched at least once their number or team in the years 2010, 2012, and 2014. From those extracted facts, we selected one value for each entity and each class-attribute combination, meaning that each country and NFL athlete is only present once per attribute.\(^6\)

We use this gold standard to act as a KB. It is therefore used along the web tables to test our fusion methods. For this, we require correspondences between the gold standard and the web tables. These correspondences were computed by matching both the web tables and the gold standard to a comprehensive KB, in our case DBpedia, using the matching component of the T2K Framework [13]. The resulting number of matches between the web tables and the facts contained in the gold standard can be found in Table 2. Except for the Number of NFL athletes, we could on average found more than one match per fact.

5. FUSION RESULTS

In this section, we present the results of the evaluation of the different fusion strategies. We report precision (P), recall (R) and F1-measure (F1) based on our gold standard described in Subsection 4.2. In order to measure R, it is necessary to calculate the number of triples for which the gold standard contains a correct value. This number is equal to the number of triples where at least one corresponding web table value is similar to the correct value for this particular triple. For those cases, a perfect fusion method should be able to identify this correct value.

First, we report the results of the three baseline fusion strategies Voting, PageRank, and Knowledge-Based-Trust. In addition, we evaluate a fusion strategy which only takes timestamp information into account. In Subsection 5.2, we further report the results of the baseline strategies in combination with the timestamp information.

5.1 Baseline Strategies

5.1.1 Voting

Voting is a common baseline fusion strategy [5]. Voting assigns all values matched to a triple a score of 1.0 and is therefore effectively a simple count of the number of web tables that contain a specific value. The performance of this strategy (Voting) is presented in Table 3.

As all values have the same score, filtering with a threshold is not applicable, which means that in any case at least one match for a given triple was found a value is fused. In the case of only incorrect matched values, this necessarily leads to an incorrect fused value. Improving on this baseline method can therefore be achieved in two ways: (1) by scoring the values to allow the selection of the correct group, and (2) by providing scores for thresholds that remove triples for which no correct value can be fused.

5.1.2 PageRank

PageRank [2] uses the link structure of the Web to rank websites. Therefore, PR can indirectly also be used to gen-

\(^5\)http://data.worldbank.org/indicator/EN.POP.DNST
http://data.worldbank.org/indicator/SP.POP.TOTL
\(^6\)The gold standard data can be downloaded from http://dws.informatik.uni-mannheim.de/en/research/T2K.

<table>
<thead>
<tr>
<th>Class</th>
<th>Attribute</th>
<th>matched values</th>
<th>before table</th>
<th>after table</th>
<th>on page</th>
<th>in page</th>
<th>in table</th>
<th>in column header</th>
<th>in cells of same row</th>
<th>% Inc. by TsProp</th>
</tr>
</thead>
<tbody>
<tr>
<td>Country</td>
<td>Population</td>
<td>27,297</td>
<td>6,254</td>
<td>12,911</td>
<td>4,893</td>
<td>4,615</td>
<td>0</td>
<td>284</td>
<td>5</td>
<td>12.62</td>
</tr>
<tr>
<td>Country</td>
<td>Pop. Density</td>
<td>104,502</td>
<td>82,512</td>
<td>78,189</td>
<td>4,576</td>
<td>4,128</td>
<td>297</td>
<td>71,399</td>
<td>14</td>
<td>29.30</td>
</tr>
<tr>
<td>NFL Athlete</td>
<td>Team</td>
<td>35,035</td>
<td>27,413</td>
<td>16,240</td>
<td>4,224</td>
<td>17,425</td>
<td>8</td>
<td>42</td>
<td>11,289</td>
<td>22.19</td>
</tr>
<tr>
<td>NFL Athlete</td>
<td>Number</td>
<td>12,472</td>
<td>8,268</td>
<td>6,792</td>
<td>4,787</td>
<td>6,984</td>
<td>0</td>
<td>0</td>
<td>1,290</td>
<td>2.35</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Class</th>
<th>Attribute</th>
<th>Type</th>
<th>Number of Values</th>
<th>Matches</th>
</tr>
</thead>
<tbody>
<tr>
<td>Country</td>
<td>Population</td>
<td>Numeric</td>
<td>212</td>
<td>7,437</td>
</tr>
<tr>
<td>Country</td>
<td>Pop. Density</td>
<td>Numeric</td>
<td>212</td>
<td>80,598</td>
</tr>
<tr>
<td>NFL Athlete</td>
<td>Team</td>
<td>Reference</td>
<td>737</td>
<td>2,112</td>
</tr>
<tr>
<td>NFL Athlete</td>
<td>Number</td>
<td>Numeric</td>
<td>737</td>
<td>473</td>
</tr>
</tbody>
</table>
erate scores for the fusion of web table values.

For PR, we implement a scoring function that uses a full ranking of all hosts of the Common Crawl (2014) to calculate scores for the matched values, utilizing the WDC Hyperlink Graph\(^7\) [9]. The score of a given value is equal to the PR score of the host the value’s table was extracted from.

We normalize the PR scores by triples. Given a set of matched values for a single triple where each value can have a PR score, we normalize these scores between 0.0 and 1.0.

We find that using the PR scores directly leads to a decrease of performance in comparison to Voting. While the precision increases marginally, the recall drops. This might indicate that PR could potentially identify accurate data, but at the cost of a large decline in recall.

Furthermore, we expect that for some topical domains, popular websites might not contain the accurate values (especially for older values), therefore we learn a regression model\(^8\) using the PR scores for each class-attribute combination in order to predict a more realistic score for the value.

The learned model, containing a weight for the PR score and a constant base-score, reflects the influence of the PR score for the particular attribute. The results for this fusion strategy (wPr), which are presented in Table 3, show that we are able to outperform Voting in three out of four properties using this particular strategy. Relative to the base score, we find that the weight assigned to the PR score is the highest for the attributes Population, where the weight is six times larger than the fixed base score, followed by Population Density. The lowest relative weight was assigned to the attribute Number. The PR weight is about six times the base score, reflecting the influence of the PR score on the particular attribute.

Table 3 shows the results for this strategy (Kbt). In comparison to Voting and wPr when using Kbt the performance for all attributes increases consistently.

### 5.1.4 Baseline Timestamp Fusion Strategy

The major problem which needs to be faced when using timestamp information extracted from the web tables and its context is their potential noisiness. Some timestamp types might not be relevant at all such as the information extracted from the last updated information of the web page. Furthermore the relevance of a particular timestamp type might differ from attribute to attribute.

Based on this assumption, we made use of a machine-learning-based approach to learn the usefulness of the different timestamp types for each class-attribute combination as described in Section 3.4. The learned model uses as input all the available extracted and generated timestamp types and returns a score for the particular value.

The results of this strategy (Ts) can be found in Table 3. When compared to Voting, this strategy improves the performance for three of the four attributes. On the other hand, we find that wPr still outperforms this strategy. The highest weighted timestamp types for this strategy are shown in Table 4 and are further discussed in Subsection 3.4.

We further run the proposed timestamp propagation approach (TsProp) to infuse additional temporal information to entity-attribute-value matches which originally did not have temporal information. Table 3 also shows the results of this approach. We can see a clear increase in the performance when using TsProp. Making use of the propagation strategy outperforms wPr in two of the attributes, unlike the timestamps strategy without propagation.

### 5.2 Timestamp extended Baseline Strategies

In the following, we report the results of the baseline fusion strategies enriched with temporal information gathered from the timestamps. In general, we use a regression model to combine the different scores (from the baseline strategies as well as from the timestamp types) which we train on an independent training set using 5-fold cross-validation.

#### 5.2.1 Extending PageRank with Timestamps

As a first combined fusion strategy we learned a regression model that includes the PR scores as well as the timestamp scores. In addition, we multiplied the PR scores with all timestamp scores and included the resulting scores them in the model. For example PR\_PageTitle in Table 4 is the PR score multiplied with PageTitle.

The intuition behind multiplying the timestamp indica-

<table>
<thead>
<tr>
<th>Table 3: Results for all fusion strategies.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Pop. Density</strong></td>
</tr>
<tr>
<td>**</td>
</tr>
<tr>
<td>Voting</td>
</tr>
<tr>
<td>Ts</td>
</tr>
<tr>
<td>TsProp</td>
</tr>
<tr>
<td>wPr</td>
</tr>
<tr>
<td>wPrTs</td>
</tr>
<tr>
<td>wPrTsProp</td>
</tr>
<tr>
<td>Kbt</td>
</tr>
<tr>
<td>KbtTs</td>
</tr>
<tr>
<td>KbtTsProp</td>
</tr>
<tr>
<td>AllTs</td>
</tr>
<tr>
<td>AllTsProp</td>
</tr>
</tbody>
</table>

\(^7\)http://webdatacommons.org/hyperlinkgraph/

\(^8\)For all learning, we use 5-fold cross-validation on an independent training set.
5.2.4 Utility of Weighting Temporal Information

Based on the results presented in Table 3, we observe that the weights of the timestamp types clearly differ per attribute, but less per strategy. Inspecting the learned weights, we discover patterns among the attributes that are similar across the fusion strategies. We observe that the attributes Population and Population Density often rank the type PageTitle high, whereas for Team the types RowCell and FullHierarchy have higher weights, and for Number the type FullHierarchy is frequently ranked highest. These patterns are especially visible when using timestamp propagation. Furthermore, we find that the hierarchies are an effective way of dealing with timestamps, as they are very often weighted highly. This applies especially to FullHierarchy, indicating that all timestamp types included in the hierarchy are actually considered.

In general, these observations confirm our assumption that the utility of some timestamps is higher for some attributes than for others, and that it is necessary to learn a separate regression model for each attribute.

6. SUMMARY

In this work, we presented a methodology for fusing data from a large corpus of web tables, that is able to deal with temporal information to fuse time-dependent attributes. We use regression and propagation to reduce noisiness and sparsity. We ensure that all timestamp information found on a web page are used, and we create various combinations of fusion strategies and test them using a publicly available gold standard. Based on the results shown in Table 3, we find that, on average, KBT outperforms other baseline strategies, and that we can further improve KBT by around 5% in F1-measure by incorporating temporal meta information and timestamp propagation. We have shown that the combination of different types of temporal meta information by using a machine-learning-based approach is suitable for this task. Inspecting the learned utilities of the different weights of types of meta information reveals that it is necessary to learn separate models for each class-attribute combination.

While the discussed weighted combination of individual scores of types of temporal meta information yields beneficial results, we further want to investigate models that, given the same input, find the most likely correct timestamp. This timestamp could be semantically more relevant than a numeric score. Such models would improve the interpretability of the results and could further facilitate the propagation of missing temporal meta information to values in web tables.

Furthermore, web tables also contain attributes which are updated more frequently than annually. Future work could therefore deal with timestamps that are not solely based on the year.
7. REFERENCES


[16] X. Yin, J. Han, and P. S. Yu. Truth discovery with multiple conflicting information providers on the web. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 2008.
