Triplifying Wikipedia's tables

Triplifying Wikipedia's Tables

Emir Muñoz, Aidan Hogan, and Alessandra Mileo

Digital Enterprise Research Institute, National University of Ireland, Galway
{emir.munoz, aidan.hogan, alessandra.mileo}@deri.org

Abstract. We are currently investigating methods to triplify the content of Wikipedia's tables. We propose that existing knowledge-bases can be leveraged to semi-automatically extract high-quality facts (in the form of RDF triples) from tables embedded in Wikipedia articles (henceforth called "Wikitables"). We present a survey of Wikitables and their content in a recent dump of Wikipedia. We then discuss some ongoing work on using DBpedia to mine novel RDF triples from these tables: we present methods that automatically extract 24.4 million raw triples from the Wikitables at an estimated precision of 52.2%. We believe this precision can be (greatly) improved through machine learning methods and sketch ideas for features that should help classify (in)correct triples.

1 Introduction

The "Web of Data" is a growing subset of the Web whose content is machine readable and interoperable, which offers the potential to directly query content integrated from arbitrary sources. However, the coverage of content on the Web of Data still pales in comparison with that available on the traditional HTML-centric Web. Mika et al. [13] called this problem "the Semantic Gap": an inequity in the supply of richly structured content and the typical demands of Web users.

To meaningfully address this Semantic Gap in the short-to-medium term, one potential route is to recover some of the latent structure and semantics embedded in traditional Web content. A promising target are HTML tables. Cafarella et al. [2] note that there are billions of tables on the Web, finding 14.1 billion HTML tables in a Google crawl from 2008, estimating that 154 million of these contain high-quality relational data. However, automatically recovering the semantics of such tables is exceptionally challenging [18].

Given the challenges of extracting high-quality RDF triples from generic HTML tables, herein, we rather focus on a *relatively* small but rich subset of Web tables: the tables of Wikipedia. As opposed to info-box tables, which have already been the target of various triplification efforts [1,8], we rather aim at "Wikitables", which are the relational tables embedded in Wikipedia articles. As opposed to generic Web tables, which have been explored before [9,20,19,7,16,2,10,4,18], Wikitables should contain a higher ratio of factual knowledge and should be syntactically cleaner (being generated from Wiki markup). Furthermore, cells in Wikitables often contain links to Wikipedia articles that directly disambiguate

the entities that are mentioned, and the table is embedded in an article that provides a clear context for the table (potentially a protagonist [3], aka., a subject entity). Last but not least, we propose that existing datasets—like DBpedia [1] and YAGO2 [8], which offer partial exports of Wikipedia's content as RDF—can be used as reference knowledge-bases to guide triplification.

By targeting Wikitables, we (mostly) avoid having to disambiguate entities and relations based on textual labels, where we can (often) directly map table cells to DBpedia entities and subsequently see what pre-existing relationships exist between them. We can thus often avoid the difficult problem of resolving entities (as tackled by, e.g., Limaye et al. [10], Syed et al. [17] or Mika et al. [12]). Similarly, by mining existing relations from DBpedia, we avoid having to identify or create new relations (i.e., pick an RDF predicate based on column headers), but can mine triples from the tables using a pre-existing (implicit) schema.

In this paper, we first provide a motivating example. We then discuss related works in the area of recovering the semantics of HTML tables and mapping relational content to RDF (§ 2). Next, we survey Wikitables available in a recent Wikipedia dump (§ 3). Thereafter, we discuss some of our ideas for using DBpedia as a reference knowledge-base to extract triples and sketch ideas for machine learning methods to further classify correct/incorrect triples (§ 4). We then wrap-up with discussion of future plans (§ 5).

Example 1. The table in Figure 1 is taken from the Colorado Wikipedia article (`http://en.wikipedia.org/wiki/Colorado`) and shows a list of the five executive officers of the state of Colorado. Some table cells are links to Wikipedia articles and others are plain string literals. The table contains a caption, header types **Office**, **Incumbent**, **Party** and **Term**; and instances for each type Governor, John Hickenlooper, Democrat, 2011–2015, Lieutenant Governor, Joseph Garcia, etc.

This table constitutes quite a complex example. First of all, there are obvious relationships between the politicians listed as **Incumbent** and the elements of the **Party** column. Also, there are some implicit relationships that hold between many of the politicians and the subject of the article (Colorado), such as `residence`, etc. Furthermore, as per an attribute–value table, there are explicit relationships between the incumbents and Colorado, where the attributes are given in the **Office** column (e.g., Walker Stapleton is State Treasurer of Colorado) and a temporal context is provided in the **Term** column.

In this paper, we survey the corpus of all such tables in Wikipedia. We furthermore propose some initial ideas on how to triplify *some* of the content of the table using DBpedia as a reference knowledge-base. The core idea is to map the elements of the cells with wiki-links to their respective DBpedia entities and to then look for existing relationships in DBpedia between entities on the same row: for example, we can find that `dbr:John_Hickenlooper` has the relation `dbo:party` to `dbr:Democratic_Party_(United_States)`, where we can suggest that such a relation might hold between entities in the respective columns on other rows. Furthermore, we can look for DBpedia relations from entities in a given column to the article body, where we find, e.g., the relation `dbp:residence` from `dbr:Joseph_Garcia_(United_States_politician)` to `dbr:Colorado`.

The Five Executive Officers of the State of Colorado

Office	Incumbent	Party	Term
Governor	John Hickenlooper	Democrat	2011–2015
Lieutenant Governor	Joseph Garcia	Democrat	2011–2015
Secretary of State	Scott Gessler	Republican	2011–2015
State Treasurer	Walker Stapleton	Republican	2011–2015
Attorney General	John Suthers	Republican	2005–2015

Fig. 1: An example table from the Colorado Wikipedia article.

We can also use features to classify triples extracted thusly as (in)correct; for example, we could consider that the more rows for which a given relation is found, or the closer the predicate label matches the column header text (e.g. `dbo:party`), the higher the confidence in the associated triples. Similarly, we could use the co-occurrence of certain properties for entities in different rows of the same column to adjust the confidence of a candidate relation. Later, we sketch some features that we plan to investigate (in future work) for classification. □

2 Related Work

Processing Web Tables. Many works have looked at identifying, parsing, normalising, categorising and interpreting Web tables (e.g., [9,20,19,7,16,2,10,4,18]). Many of the challenges that these authors have tackled are partly solved by the nature of Wikipedia. For example, Crestan and Pantel [4] identify the "*protagonist problem*", which refers to identifying the subject or context of the table; in Wikipedia, the article in which the table is embedded offers a direct notion of context (as per Colorado in Example 1). Other authors, such as Yoshida et al. [20], have looked at categorising tables; we rather rely on the HTML `class` attribute associated with Wikipedia tables for classification, targeting tables with the value `wikitable`. Other more recent works have tried to resolve entities or relationships in tables with mixed results [10,18]; we can use wiki-links to resolve entities and propose to use a reference knowledge-base to resolve relations.

Triplifying Wikipedia. Systems such as DBpedia [1] and YAGO2 [8] already extract RDF from Wikipedia. Both DBpedia and YAGO2 define a number of extractors that mine RDF triples from Wikipedia. However, both efforts rely heavily on info-boxes (attribute–value tables that appear in the top right of Wikipedia articles). The uniform structure of info-boxes, and the use of common templates, makes triplification much more straightforward than for generic tables. YAGO focuses on highly accurate triplication based on manually specified rules. DBpedia create high-quality RDF in an "ontology" namespace based on manual mappings and in a "property" namespace based on automatically

generating predicates from attribute labels. Our work is complementary since neither work proposes concrete methods to triplify generic Wikipedia tables.

Extracting RDF from Tables. Various proposals have been made to extract RDF from relational database tables, where the W3C has recently recommended the Direct Mapping and the R2RML language for mapping relational content to RDF [5]. Ding et al. [6] propose a structural triplification of tables, considering tables rows as subjects, column headers as predicates, and cell values as objects. Mulwad et al. [14,15,17] propose extracting the content of tables as RDF, performing entity-resolution and relationship discovery using reference knowledge-bases. They use what we would call a vertical, column-centric approach (matching columns to relations). Based on evaluation over 15 relational tables they report 66.12% of accuracy for linking table cell strings and 25% for identifying relations. Versus these works, our proposals are (semi-)automatic and extraction of triples from tables is on a row-centric, horizontal basis.

3 Survey of Wikitables

3.1 Classifying Wikipedia's Tables

Wikipedia editors can choose from three classes of tables to add to an article: (1) toc: table of contents; (2) infobox: attribute-value tables embedded in the top-right of the article; and (3) wikitable: relational tables embedded in the article's body (as per Figure 1). Each class of table is directly identifiable by the HTML attribute class for the associated table tag in the Wikipedia page. Since toc tables represent article layout and extracting RDF from infobox tables has already been studied [1,8], we focus on the wikitable class of tables.

3.2 Normalising Wikipedia's Tables

The structure and content of Wikitables can be quite complex: there is no standard to define/design/publish Web tables. Instead, users are informally guided by look-and-feel. Instead of duplicating the content of contiguous cells, tables often contain spans: cells that span multiple positions, including column-spans (colspans) and row-spans (rowspans). To avoid ugly narrow long tables, or tables wider than the standard display width, users often employ split tables. Oftentimes, cells may also contain (i) multiple values, (ii) prose-text alongside the primary value giving justification or context for that value, (iii) superscript references to sources, (iv) images or other embedded content, (v) empty cells in "optional" columns. To simplify matters, we first normalise Wikitables by "squaring" them such that they can be represented as a matrix of cells.

Table Model. We leverage the TARTAR extraction process and logical representation proposed by Pivk et al. [16], where the table is divided into logical regions and frame logic is used as a semantic representation. We model a table T as a matrix $M_T(n, m)$, where n and m represents the number of rows and

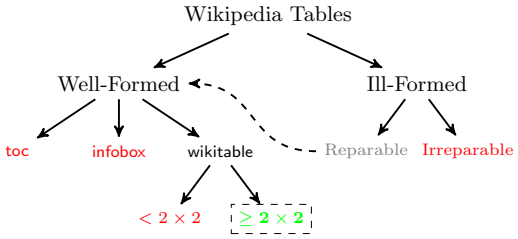

Fig. 2: Structural classification of all of Wikipedia's tables.

Table 1: Statistical overview of tables in January 2012 dump.

Total articles:	10,854,204	100.0%
Articles w/tables:	1,875,415	17.3%
All tables:	3,923,427	100.0%
Ill-formed:	163,105	4.2%
Well-formed:	2,955,714	95.8%
TOC:	1,648,786	43.9%
Infobox:	815,350	21.7%
$< 2 \times 2$:	158,940	4.2%
$\geq \mathbf{2 \times 2}$ (*useful*)	**1,137,246**	**30.2%**

columns, respectively. We extract Wikitables from Wikipedia's HTML content, where an individual Wikipedia article represents a source $\mathcal{T}$ (a set) of tables.

Creating Matrices. Table headers are very common in tables, and can be simple (one row) or hierarchical (multiple rows). We check for two levels of headers. If T contains table headers (<TH>), we say that $M_T(h, .)$ represents those headers, where h is the bottom row with TH tags. It is also common that tables contain their caption embedded in the first row: we discard that row when generating the matrix. For each table, we must also deal with colspans and rowspans, where we divide such cells and copy the original content into each division. For instance, if the table contains a cell <TD colspan='2'>180</TD> with a colspan of 2, the cell is replicated as <TD>180</TD><TD>180</TD>.

Well-formed/Ill-formed. If the result of the pre-processing of a table $T \in \mathcal{T}$ can be represented as a matrix $M_T(n, m)$ (i.e., is dense and rectangular), we call the table *well-formed*. Otherwise we call it *ill-formed*. The most common cause of ill-formed tables are jagged rows, which we found to be prevalent in Wikipedia due to idiosyncrasies of the wiki-markup used by editors. However, many jagged tables contain rich content. Thus, we extended TARTAR to 'repair' jagged tables by simply completing empty cells with empty strings. Finally, we do not consider tables $M_T(n, m)$ where $m = 1$ or $n = 1$; tables must be larger than 2×2. This leads us to the final structural taxonomy of tables that we extract from Wikipedia, as illustrated in Figure 2.

3.3 Corpus Composition

We extract our corpus of Wikitables from the January 2013 Wikipedia dump. We first apply the Bliki engine parser[1] to convert wiki-markup to the HTML pages for articles. Each HTML page is then cleaned and canonicalized (fixing syntax mistakes) using CyberNeko[2] before all HTML tables (including nested tables) are extracted. The size of the resulting corpus is summarised in Table 1.

In total, 17.3% of Wikipedia articles contain at least one table (vs. 75% reported in [4] for general Web documents). We consider 30.2% of the total

[1] http://code.google.com/p/gwtwiki/
[2] http://nekohtml.sourceforge.net/

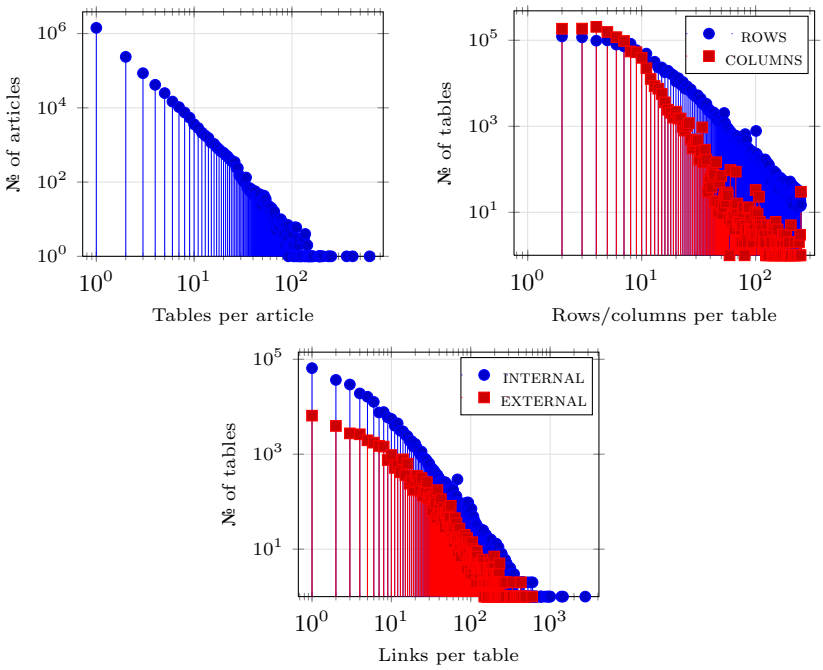

Fig. 3: Various distributions for the Wikitables considered.

tables in Wikipedia for our extraction: just over one million tables. Figure 3 plots three distributions in *log/log* scale, including the number of articles with a given number of tables (excluding $x = 0$, $y = 8,978,789$ due to the logarithmic x-axis), the number of tables with a given number of rows and columns, and the number of tables with a given number of internal (Wikipedia) and external links. We see that the plots generally follow long-tailed distributions. Furthermore, we see that tables tend to have more rows than columns, as could be expected. Also, internal links in are much more prevalent in tables than external links. We make the following additional high-level observations:

- The article[3] with the highest number of well-formed tables contains 623 (most of them nested inner-tables).
- Considering articles with at least one table, there are 1.66 tables per article.
- The maximum number of detected rows in a table is 250, commonly found in List_of_* articles (likely due to a Wikipedia formatting constraint). The average number of rows per table is 12.44.

[3] http://en.wikipedia.org/wiki/Winners_and_runners-up_in_ the_legislative_elections_of_Nepal_1994_and_1999

– The maximum number of columns in a table[4] is 250 (due to an erroneous colspan for a caption in a table with 3 columns). The average number of columns per table is 5.55.
– The highest number of internal links in a single table is 2,774, and 594 for external links. The averages are 1.93 and 0.35 for internal and external links, respectively. Internal links are important for us since they ensure that we can map cell entries to Wikipedia articles (and thus to DBpedia entities).
– 19.4% of tables do not contain column headers. 79.9% contain headers only in the first row, 7.4% contain headers in the second row, and the remaining tables contain captions in further rows.
– Only 5.5% of tables contain non-empty captions.

We thus view this corpus of tables as a rich source of structured data that can be exploited for information extraction and ultimately for triplification, with many tables containing a high number of internal wiki-links (as per the bottom plot in Figure 3), which can be used for entity disambiguation.

4 Towards Triplifying Wikitables

4.1 Reference Knowledge-base

To extract RDF from these Wikitables, we propose to use a reference knowledge-base to mine relations, where we use English-language data from DBpedia v3.8, describing 9.4 million entities. The overall corpus consists of 465 million unique RDF triples, and contained 57,985 unique RDF relations (i.e., RDF predicates), 48,293 of which were in the DBpedia property namespace, 1,397 of which were in the curated DBpedia ontology namespace, and 28 of which were from external vocabularies.[5] As per Figure 4 (*log*/*log*), we can see that the number of triples in which different DBpedia relationships appear follows a long-tailed distribution, where many relationships appear in few triples and few relationships appear in many triples.

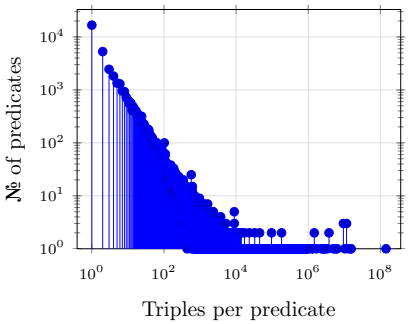

Fig. 4: Distribution of number of triples per predicate (relationship) in DBpedia knowledge-base.

4.2 Entity/Resource Extraction

In Section 3, we described how we extract a set of tables from Wikipedia, where each table is represented as a matrix and each element (cell) of the matrix

[4] `http://en.wikipedia.org/wiki/2007--08_QMJHL_season`
[5] We also found examples of noise where 8,267 DBpedia resource URIs referring to Wikipedia templates appeared as predicates.

$M_T(i,j)$ can contain a variety of content. However, we are primarily interested in internal links (`class=internal`) present in the cells since they can be directly mapped to DBpedia entities. We represent the set of internal links in a cell as $R(i,j)$. Each link $r \in R(i,j)$ is further filtered by removing superscript footnote links, media-links with prefix Image: or File:, and by pruning fragment identifiers.

Then, for each $r \in R(i,j)$ after filtering, we map the corresponding Wikipedia URL to a DBpedia entity URI by following redirects and replacing the namespace http://en.wikipedia.org/wiki/ of the URL with http://dbpedia.org/resource/. For optimisation purposes, we cache the set of redirects involved. We additionally filter entities corresponding to DBpedia categories and list pages. We denote by $E(i,j)$ the set of DBpedia entities extracted for table cell $M_T(i,j)$. Furthermore, column headers in tables contain plain strings that identify types of data, which we do not map to DBpedia resources; header cells are not mapped to resources, but could potentially be matched with DBpedia relationships at a later phase.

4.3 Discovering Relations

For discovering relations within tables, we query the DBpedia knowledge base for relationships that exist between entities in different cells of the same row. For each row, we look for relations between all pairs $\{(e_{i,j}, e_{i,k}) \mid e_{i,j} \in E(i,j), e_{i,k} \in E(i,k), 1 \leq i \leq m, 1 \leq j < k \leq n\}$. Any relations found for $(e_{i,j}, e_{i,k})$ are suggested as candidates for relating $(e_{h,j}, e_{h,k})$ for $1 \leq h \leq m$, $h \neq i$.

For discovering relations from entities in the table and p the protagonist entity of the article, we query the knowledge-base for pairs $\{(e_{i,j}, p) \mid e_{i,j} \in E(i,j), 1 \leq i \leq m, 1 \leq j \leq n\}$. Any relations found for $(e_{i,j}, p)$ are suggested as candidates for relating $(e_{h,j}, p)$ for $1 \leq h \leq m$, $h \neq i$.

To query a pair, we use a SPARQL query as follows over the DBpedia knowledge-base, looking for relationships that hold in either direction:

```
SELECT DISTINCT ?p1 ?p2
       { { <e1> ?p1 <e2> } UNION { <e2> ?p2 <e1> } }
```

Example 2. With respect to relations within tables, if we consider table T in Figure 1, we have:

$E(2,1) = \{$`dbr:Governor_of_Colorado`$\}$,
$E(2,2) = \{$`dbr:John_Hickenlooper`$\}$,
$E(2,3) = \{$`dbr:Democratic_Party_(United_States)`$\}$, and
$E(2,4) = \emptyset$ (in the example $M_T(2,4)$ is a literal).

If we query DBpedia indexes looking for relations in this row, between columns 1 and 2, or sets $E(2,1)$ and $E(2,2)$, we find two relations in DBpedia: dbp:incumbent from 1 to 2 and dbp:title from 2 to 1. We suggest this relation may hold across entities in, e.g., $E(3,1)$, $E(3,2)$ and so forth. The principle is similar for the protagonist and elements of the table. For example, we find the relation `dbp:residence` from `dbr:Joseph_Garcia_(United_States_politician)` to `dbr:Colorado` and suggest that relation from all entities in column **Incumbent** ($\bigcup_{\forall x} E(x,2)$) to `dbr:Colorado`. Of course, this method is approximate and may produce incorrect triples (as discussed later in Sections 4.5–4.6). □

4.4 Initial Triplification

We extract all candidate relationships found for all Wikitables as RDF triples:

Example 3. Consider applying the `dbp:title` relation found in Example 2 to row 3 (from $E(3,2)$ to $E(3,1)$). The resulting RDF triple is then:

```
dbr:Joesph_Garcia_(United_States_politician) dbp:title dbr:Lieutentant_Governor .
```

□

We use local indexes of the DBpedia knowledge-base for answering queries, and for each pair, we perform two atomic on-disk lookups for relations in either direction. We also use in-memory LRU caching to catch repeat queries (as caused by spans). Still, given that tables can potentially contain hundreds of entities, and that the number of entity-pairs to consider is quadratic for a given row, each table may require a large amount of lookups for relations. However, the extraction can be effectively partitioned on a per-table basis. For example, given a cluster of shared-nothing commodity servers, if we make the full index for the reference knowledge-base available to each machine (in our scenario, a 7.7GB file), individual tables can then be assigned arbitrarily to each machine, and the extraction of triples run in an embarrassingly parallel manner.

We adopt this parallel approach and using six machines (bought ca. 2005) with 4GB of RAM, 160GB SATA hard-drives, 2.2GHz single-core processors, assigning an even work-load of input Wikipedia articles to each, the full process of extracting and normalising Wikitables from the articles and computing the candidate triples took approximately 16 days. We extracted a total of 27.9 million candidate RDF triples. However, from this set, we further filter triples that, from initial inspection, we found to be often incorrect: we filter 3.4 million reflexive RDF triples (with the same subject and object resource), and a further 12 thousand triples with the predicate `dbo:wikiPageDisambiguates`. Afterwards, 24.4 million candidate triples remain.

4.5 Gold Standard

Many of the extracted triples are incorrect. Aside from incorrect source data (be it the table or the relations in DBpedia), imprecision can be due to a number of reasons. Multiple entities or additional text in table cells can cause problems: e.g., in many tables referring to international sports results, the sports-person and their nationality appear in the same cell and are not currently distinguished by our approach; thus, we might say that countries have participated in sports events. Oftentimes a relation that holds between two entities in a given row (or from an entity to the protagonist) do not apply for analogous entities on a different row; referring to the running example, although Walker Stapleton was born in Colorado (the table protagonist), many of the other politicians were not and extracted triples suggesting otherwise would be incorrect.

To investigate this, we created an initial gold standard. We randomly selected 250 candidate triples from the total set of 24.4 million. Each of the three authors

manually labelled each triple as: *correct*, *incorrect* or *unknown*. The information given was the triple, a link to the original Wikipedia article containing the source table and a number to identify the particular table. We did not pre-agree on any judging strategies, where interpretation of the validity of triples was left to the discretion of individual judges. After judging, we found few *unknown* verdicts, which we mapped to *incorrect*. The results are given in Table 2.

Table 2: Results of evaluation of 250 triples by three judges.

Three *correct*	69	27.6%
Two *correct*, one *incorrect*	61	24.4%
One *correct*, two *incorrect*	46	18.4%
Three *incorrect*	74	29.6%
Full agreement	**143**	**57.2%**
Some disagreement	**107**	**42.8%**

To measure the inter-rater agreement, we computed a Fleiss' κ coefficient of 0.43 (0 indicates no agreement, 1 indicates perfect agreement), which by convention is considered as "moderate agreement" between judges. Looking at cases of disagreement, predicates of DBpedia are often not well-defined and their meaning generally has a subjective dimension. Temporality also caused disagreement: for example, would stating that `dbr:Bill_Clinton` has the relation `dbp:president` to `dbr:United_States` be correct or incorrect? The answer would seem open to interpretation. If we define consensus by a majority vote, we see that in the shared labelling, 52% of the raw extracted triples are considered correct and 48% considered incorrect without any further classification. If instead we only consider those triples with unamimous correct/incorrect verdicts, then 48% of the extracted triples are deemed correct while 52% are deemed incorrect.

4.6 Classifying Correct/Incorrect Triples

We are currently investigating machine learning methods to classify correct/incorrect triples and to improve upon the 52% precision of our raw extraction process. For this, we are currently investigating the following main features to use for binary classification (amongst a variety of others):

Extraction type: Whether the triple is a table-row or a protagonist relation.
Ratio of rows held: The ratio of rows for which we could find the extracted relation in the original table (we assume that higher ratios are better).
Predicate label: The string similarity between the predicate label and the subject/object column header (we assume that more similar is better).
Predicate multiplicity: For the triple predicate, we measure the ratio of unique subjects and unique objects to unique DBpedia triples with that

predicate to indicate whether or not the triple is $1-1$, $1-*$, $*-1$, $*-*$, etc. For example, if `dbp:governor` is deemed $1-1$ (high ratio), we should not extract multiple such relations to a common protagonist (e.g. Colorado).

Cell content: Some cells contain multiple internal links, additional text content, bullets, and so forth (we assume such noise lowers confidence).

Such features (and many more besides) can be automatically associated with candidate triples during the extraction process and we are currently investigating machine learning methods—such as SVM, Naïve Bayes, Decision Trees, etc.—to train on labelled examples and to build classifiers that boost precision by filtering incorrect triples. Evaluation of such methods is subject to future work.

5 Conclusions and Future Directions

In this paper, we have discussed ongoing work in the area of extracting triples from the relational tables of Wikipedia. We have discussed related works, given a survey of Wikitables in a recent Wikipedia corpus, and discussed methods we are exploring that leverage existing knowledge-bases to guide triplification. We have applied our methods to one million Wikipedia tables and extracted 24.4 million raw triples with an estimated precision of 52%. To boost precision further, we are currently evaluating machine-learning methods that take a feature-set and classify raw triples as correct/incorrect. In the short term, we also hope to compare other knowledge-bases, such as YAGO2 and Freebase, for extraction.

In the longer term, since our automatic extraction is often incomplete (e.g. does not consider text/numeric cells), we are also considering a method to detect common table structures in Wikipedia that are candidates for manual mapping, where, for example, we note that climate tables for cities are often copy/pasted). If we could cluster structurally similar tables (that contain similar content in an "isomorphic" schema) with high accuracy, we could write a single mapping to triplify all such tables in the cluster. This would be similar to DBpedia, where common info-box templates are manually mapped to the core ontology terms: we would similarly have a higher-quality mapping for common table structures, and lower-quality automatic extraction (as presented) for the "long tail". Integrating our methods with DBpedia's extractors could then be possible.

An even more ambitious direction would be to generalise our methods to enrich existing knowledge-bases from generic HTML tables. For this, we could investigate use of existing entity recognition tools (e.g., [11,12]), removing our reliance on wiki-links and opening our methods up for the broader Web. However, even aside from entity recognition, there would be many open challenges with respect to identifying factual (relational) tables, parsing, cleaning, and so forth. Although we feel—by extending our current work with machine learning classifiers—that high-quality (semi-)automatic triplification of Wikipedia tables is feasible, realistically, we would have lower expectations for precision when considering arbitrary Web tables.

Acknowledgements: This paper was funded in part by Science Foundation Ireland under Grant No. SFI/08/CE/I1380 (Lion-2).

References

1. C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann. DBpedia – a crystallization point for the Web of Data. *J. Web Sem.*, 7(3):154–165, 2009.
2. M. J. Cafarella, A. Y. Halevy, D. Z. Wang, E. Wu, and Y. Zhang. WebTables: exploring the power of tables on the Web. *PVLDB*, 1(1):538–549, 2008.
3. E. Crestan and P. Pantel. A fine-grained taxonomy of tables on the Web. In J. Huang, N. Koudas, G. J. F. Jones, X. Wu, K. Collins-Thompson, and A. An, editors, *CIKM*, pages 1405–1408. ACM, 2010.
4. E. Crestan and P. Pantel. Web-scale table census and classification. In I. King, W. Nejdl, and H. Li, editors, *WSDM*, pages 545–554. ACM, 2011.
5. S. Das, S. Sundara, and R. Cyganiak. R2RML: RDB to RDF Mapping Language. W3C Recommendation, September 2012.
6. L. Ding, D. DiFranzo, A. Graves, J. Michaelis, X. Li, D. L. McGuinness, and J. Hendler. Data-gov Wiki: Towards Linking Government Data. In *AAAI Spring Symposium: Linked Data Meets Artificial Intelligence*. AAAI, 2010.
7. W. Gatterbauer, P. Bohunsky, M. Herzog, B. Krüpl, and B. Pollak. Towards domain-independent information extraction from Web tables. In *WWW*, pages 71–80, New York, NY, USA, 2007. ACM.
8. J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum. YAGO2: A spatially and temporally enhanced knowledge base from wikipedia. *Artif. Intell.*, 194:28–61, 2013.
9. M. Hurst. Layout and language: Challenges for table understanding on the web. In *Workshop on Web Document Analysis (WDA)*, pages 27–30, 2001.
10. G. Limaye, S. Sarawagi, and S. Chakrabarti. Annotating and searching web tables using entities, types and relationships. In *PVLDB*, 2010.
11. P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer. DBpedia spotlight: shedding light on the web of documents. In *I-SEMANTICS*, pages 1–8, 2011.
12. P. Mika, M. Ciaramita, H. Zaragoza, and J. Atserias. Learning to Tag and Tagging to Learn: A Case Study on Wikipedia. *IEEE Intelligent Systems*, 23(5):26–33, 2008.
13. P. Mika, E. Meij, and H. Zaragoza. Investigating the semantic gap through query log analysis. In *ISWC*, pages 441–455, 2009.
14. V. Mulwad, T. Finin, Z. Syed, and A. Joshi. T2LD: Interpreting and Representing Tables as Linked Data. In *ISWC Posters&Demos*, 2010.
15. V. Mulwad, T. Finin, Z. Syed, and A. Joshi. Using Linked Data to interpret tables. In *COLD*, November 2010.
16. A. Pivk, P. Cimiano, Y. Sure, M. Gams, V. Rajkovic, and R. Studer. Transforming arbitrary tables into logical form with TARTAR. *Data Knowl. Eng.*, 60(3):567–595, 2007.
17. Z. Syed, T. Finin, V. Mulwad, and A. Joshi. Exploiting a Web of Semantic Data for Interpreting Tables. In *WebSci10*, Raleigh NC, USA, April 26–27th 2010.
18. P. Venetis, A. Halevy, J. Madhavan, M. Paşca, W. Shen, F. Wu, G. Miao, and C. Wu. Recovering semantics of tables on the Web. *PVLDB*, 4(9):528–538, June 2011.
19. Y. Wang and J. Hu. A machine learning based approach for table detection on the web. In *WWW*, pages 242–250, New York, NY, USA, 2002. ACM.
20. M. Yoshida, K. Torisawa, and J. Tsujii. A method to integrate tables of the World Wide Web. In *Workshop on Web Document Analysis (WDA)*, pages 31–34, 2001.