Knowledge Base Augmentation using Tabular Data*

Yoones A. Sekhavat[1], Francesco di Paolo[2], Denilson Barbosa[1], Paolo Merialdo[2]
[1]University of Alberta [2]Roma Tre University
{yoones.a.s,denilson}@ualberta.ca {fradipi,merialdo}@dia.uniroma3.it

ABSTRACT

Large linked data repositories have been built by leveraging semi-structured data in Wikipedia (e.g., DBpedia) and through extracting information from natural language text (e.g., YAGO). However, the Web contains many other vast sources of linked data, such as structured HTML tables and spreadsheets. Often, the semantics in such tables is hidden, preventing one from extracting triples from them directly. This paper describes a probabilistic method that augments an existing knowledge base with facts from tabular data by leveraging a Web text corpus and natural language patterns associated with relations in the knowledge base. A preliminary evaluation shows high potential for this technique in augmenting linked data repositories.

1. INTRODUCTION

The Web is bursting with valuable tabular data that could be leveraged in many applications such as knowledge base population and query answering. For instance, Cafarella et al. report 14.1 billion HTML tables in English documents in Googles's main index [4] and over 400,000 Excel spreadsheets in the Clueweb09[1] crawl [5]. Tapping into these semi-structured information sources is difficult, however, and often they end up being treated no differently than plain text documents. What is worse, state-of-the-art text-based information extraction systems fail on such tables, as they require full sentences to yield good results.

Tables have inherent semantics which are often implicit or only given in the pages that contain them. Consider the example in Figure 1, which is a snapshot of a table in Wikipedia. It may not obvious from the table that it is about winners of FIFA's World Player of the Year Award. Nevertheless, it should be noted that the fact that someone put those literals together in the same rows indicates that there are relationships between them. In this case, we know that Ronaldinho *was born in* Brazil and *played for* Barcelona. Moreover, the same relationships apply to all rows: Cannavaro *was born in* Italy and *played for* Juventus. In fact, we can say that the relations *was born in* and *played for* are defined over the columns (1st and 2nd, and 1st and 3rd, respectively).

Ronaldinho	Brazil	Barcelona FC
Fabio Cannavaro	Italy	Juventus
Kaka	Brazil	AC Milan
Lionel Messi	Argentina	Barcelona FC

Figure 1: Example tabular data.

A general approach for understanding such tables with the help of linked open data web [2] would be to (1) link the values in each cell to known entities in a linked data knowledge base (e.g., YAGO [14] and Freebase [3]), and (2) identify relations between the linked values. The best-case-scenario for this approach would be when all entities are linked to the same knowledge base, and a relation already exists between them (this is the approach in [8]). However, there are other situations. For example, it could be that the entities exist in different, unlinked, knowledge bases, or that some of the entities are not linked to anything yet. Tackling such cases would provide *new* triples, thus augmenting the linked open data web with new facts.

This paper focuses on the problem of identifying plausible relations between pairs of entities that appear in the same row of a table. Our main assumption is that if someone went to the trouble of juxtaposing these entities on the same file, then there must be a relation between them. Moreover, we seek to augment an *existing* repository with new instances of relations *already defined*. In other words, the list of possible relations is part of the input. We also assume the entities can be resolved and linked to a linked open data repository or knowledge base. However, unlike previous works (e.g., [8]), our method does not require that.

Overview. As an example, assume our input table has only columns 1 and 3 in Figure 1, and assume we have two candidate relations: *plays-for* and *lives-in*. We start from a list of textual patterns that are associated with each relation; such patterns are detected by automatic methods for

*All data used to build and test the models in our work can be found at `http://cs.ualberta.ca/~denilson/data/ldow14_ualberta_data.zip`

[1]ClueWeb, 2009, http://lemurproject.org/clueweb09.php

LDOW2014, April 8, 2014, Seoul, Korea.

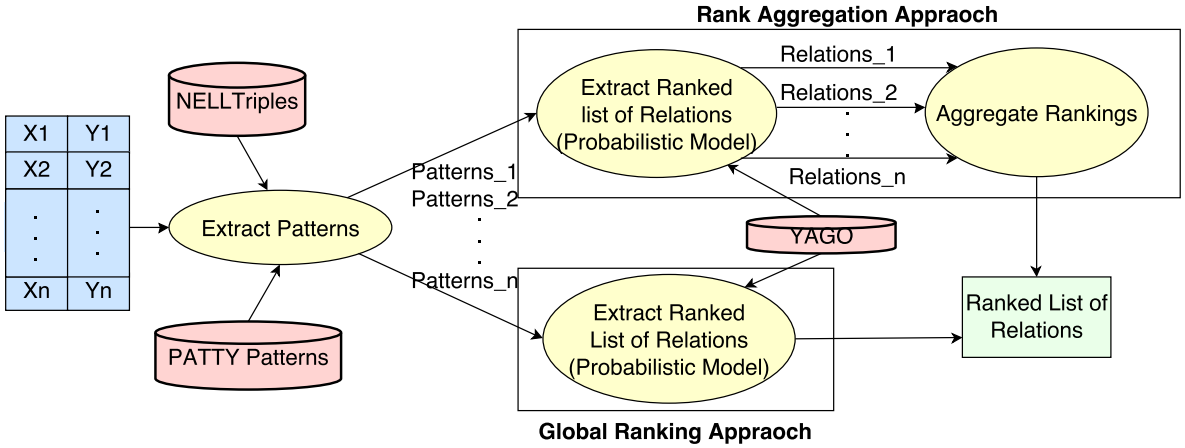

Figure 2: Rank Aggregation approach vs. Global Ranking.

building knowledge bases from natural language[2]. For example, patterns for the relation *plays-for* could be "scored for" and "signed contract with". In practice, thousands of patterns are associated with a single relation; conversely, the same pattern may be associated with multiple relations. Thus, we build a probabilistic model to estimate the posterior probability of a relation, *given* a set of text patterns observed. To make a prediction for a given pair of entities, we: (1) collect all sentences containing both entities from a large text corpus; (2) extract the text in between them; (3) match those texts against the list of patterns; and (4) estimate the posterior probability of all candidate relations.

Paper Outline. Section 2 illustrates our model to extract triples from tabular data. Section 3 discusses our first implementation of the model, and Section 4 reports the results of experiments on this first implementation. Section 5 presents related work, and Section 6 concludes the paper and presents ideas to improve our model.

2. TRIPLE EXTRACTION

In general terms, the problem we address is as follows:

Given a table T, with n rows and k columns, whose cells contain mentions to entities, and a set $R = \{r_1, \ldots, r_m\}$ of relations of interest, we aim to produce all triples $\langle t_i[x], r_j, t_i[y] \rangle$ $\forall t_i \in T$, $r_j \in R$, where $t_i[x]$ denotes the value of column x in row i.

Furthermore, we seek to assign relation r_j to *pairs of columns* in T, in the sense that the predicate r_j holds for all rows of T. The problem boils down to the case of an input table with just two columns x and y, and, without loss of generality, we address this case here.

Understanding semantic relations between entities using text corpora is a challenging task because a relation can be expressed with different textual (surface) forms. For example, the relation *plays-for* can be inferred from sentences with the patterns "scored for" and "signed contract with". On the other hand, a pattern may express more than one relation. For example, the pattern "played in" may represent relations *plays-for* and *performed-at* (e.g., "Pink Floyd played in Pompeii").

Another challenge, which has been the subject of substantial work recently, is entity linking. For example, "Barcelona" may represent the football club or the city. Evidently, the choice of entity linking approach will have an impact on the quality of the triples produced by a method like ours. To avoid this factor in our study of the relation assignment problem we use exact matching to "link" the entities. In effect, this is akin to assuming the entities are already linked.

2.1 Relation between a Pair of Entities

We start with determining whether a relation r applies to two entities. In the absence of further information, a reasonable approach is to search the (textual) Web for all sentences connecting the entities and determine whether r follows from those sentences. While textual entailment and other reasoning methods could be used for this purpose, we turn this into a probabilistic inference as follows. We start from a known set of textual patterns $p_1, \ldots, p_k$ that are commonly associated with relation r; thus giving us prior probabilities for each pattern expressing the relation. To determine if entities e_1 and e_2 belong in relation r, we search the corpus for all sentences containing both e_1 and e_2, extract all words in between them, and match those words against the list of patterns. We can use Bayesian inference to compute the posterior probability of r given the observed patterns.

In this model, relation r is a categorical class variable whose domain is R, and the patterns $p_1, ..., p_n$ are binary evidence variables, representing patterns observed between entities in the text corpus. The model is thus:

$$Pr(r|p_1, ..., p_k) = \frac{Pr(r)Pr(p_1, ..., p_k|r)}{Pr(p_1, ..., p_k)} \quad (1)$$

Relations maximizing $Pr(r|p_1, ..., p_k)$ are the most probable relations for the entities (given the corpus used). A last component in the model is a threshold to filter low-confidence predictions.

We assume evidence variables $\{p_1, ..., p_n\}$ are conditionally

[2]We use the patterns in the PATTY system [12].

independent, which is reasonable since the probability that pattern p_i represents a relation does not depend on the probability that another pattern p_j also represents that relation. Given this assumption and using the Chain rule, we have:

$$Pr(r|p_1, ..., p_k) = \frac{Pr(r) \prod_{i=1}^{k} Pr(p_i|r)}{Pr(p_1, ..., p_k)} \quad (2)$$

2.2 Relation between Two Lists of Entities

Consider now the case of a table with n rows (i.e. a list of n entity pairs). There are two main approaches, illustrated in Figure 2: we can find one relation assignment for each row and compute an *aggregate* from all these assignments, or we can observe all text patterns for all rows at once and obtain a *global* assignment. We discuss each next.

Rank Aggregation. In this approach, we first obtain a ranking of all relations in R sorted in decreasing likelihood according to the model above, for each pair of entities in the table. Next, we combine these ranked lists to find a single best relation for all entity pairs.

Rank aggregation [7] is a well-studied problem: given n ranked lists $l_1, ..., l_n$, and a distance measure d, the problem is to find a list l such that $\sum_{i=1}^{n} d(l, l_i)$ is minimized. Among different distance measures to aggregate ranking lists, we use Spearman's Footrule (SP) and Kendall's tau (KE) which were shown to outperform other approaches [7]. Kendall's tau distance between two permutations of a list is the sum of the number of pairs from the list which are not in the same order in these two permutations. This distance is also known as the number of exchanges needed in a bubble sort to convert one permutation to another. On the other hand, Spearman's Footrule distance is the sum of the distances between positions of each item in two different permutations. We also employ a simple average score method to aggregate ranking lists called Mean Ranking (MR), which works as follows: given a sorted (descending) list of probabilities $Pr(r|p_1, ..., p_k)$ generated for each pair $\langle x_i, y_i \rangle$, and m possible relations in the knowledge base, we assign a score $sc_i(r) = m - pos(r)$, where $pos(r)$ is the position of relation r in this sorted list.

Global Ranking. Another approach is to feed all patterns for all entity pairs in the relation as evidence to the probabilistic model. In this Global Ranking (GR) approach, all observed patterns for all entity pairs simultaneously contribute to selection of the most probable relation.

3. IMPLEMENTATION

This section describes one instantiation of the probabilistic model developed in the previous section, using off-the-shelf, real-world knowledge bases and text corpora.

Data. We use YAGO [14] and PATTY [12] to obtain relations and patterns. YAGO is a popular knowledge base with about 10 million entities and 120 million facts, and PATTY, developed by the same research group, associates 22,779 patterns mined from New York Times articles and Wikipedia with 25 relations. Of those, 24 exist in YAGO and were used here. Our text corpus is the NELL *Subject-Verb-Object* triple corpus [15], with about 604 million triples extracted via dependency parsing on the ClueWeb09 dataset (Clueweb09 is a Web crawl with about 1 billion web pages in ten different languages).

Table 1: Number of PATTY patterns and ranked obtained by each strategy, for each relation. A – indicates the relation has been incorrectly ranked 4th or lower by the method

Relation	Patterns	SP	KE	MR	GR
ismarriedto	1274	1	1	1	1
created	1148	1	1	1	1
haschild	1090	–	–	–	3
influences	694	1	1	2	–
actedin	624	2	2	1	1
graduatedfrom	472	1	1	1	1
isknownfor	452	–	–	–	–
worksat	447	–	–	1	1
holdspoliticalposition	417	–	3	1	1
directed	400	2	–	2	2
playsfor	354	1	1	1	1
diedin	335	3	–	1	2
wasbornin	273	–	–	1	1
islocatedin	249	–	3	–	–
livesin	200	–	–	–	–
isleaderof	156	–	–	–	–
iscitizenof	121	1	1	–	–
haswonprize	81	–	–	3	1
dealswith	59	–	–	–	1
ispoliticianof	49	1	–	–	1
participatedin	33	1	1	2	2
happenedin	12	2	1	–	1
hascapital	1	2	1	–	–

As mentioned before, we link entities mentioned in the tables to those in the text corpus with exact string matching. That is, given entities e_1 and e_2 in a table row, we find all triples in the NELL corpus which have e_1 as the *subject* and e_2 as the *object*. Similarly, we match the *verbs* of the resulting triples against the PATTY patterns (to obtain the observations). In effect, our system uses the "intersection" of PATTY patterns and NELL triples, consisting of 4,357 unique patterns and 108,699,400 triples. The YAGO relation *has-academic-advisor* was discarded as none of its patterns are found in the NELL corpus. Table 1 shows the relations used and the number of patterns in each of them. Note the wide variation in the number of patterns per relation.

Estimating Prior Probabilities. Recall Equation 2. We estimate the prior probability of each of the 24 relations as $Pr(r)=|r|/\sum_{r_i \in R}(|r_i|)$, where $|r|$ is the number of instances of relation r, and R is the set of all relations in YAGO. As for $Pr(p|r)$, the prior probability that pattern p occurs among instances of relation r, we use the associations between YAGO relations and textual patterns in PATTY: $Pr(p|r) = |p|/\sum_{p_i \in PT(r)} |p_i|$, where $PT(r)$ is the set of patterns associated with r. To avoid zero probabilities, we use the add-one Laplace smoothing technique.

4. EXPERIMENTAL EVALUATION

Since we do not query YAGO to make predictions, we use some of its facts to build a ground-truth to test the accuracy of our model. We extracted facts from YAGO relations where both entities can be matched exactly in the NELL

Table 2: Results of rankings on 23 YAGO relations

	Rank Aggregation			Global Ranking
	SP	**KE**	**MR**	**GR**
Ranked First	8	9	9	12
Ranked Second	4	1	3	3
Ranked Third	1	2	1	1
Ranked > 3	10	11	10	7

Table 3: Results of rankings on filtered relations

	Rank Aggregation			Global Ranking
	SP	**KE**	**MR**	**GR**
Ranked First	9	7	9	8
Ranked Second	1	3	2	2
Ranked Third	2	1	2	3
Ranked > 3	2	3	1	1

corpus. The number of facts from YAGO that can be found from NELL triples using exact matching varies across relations. The lower bound was 25 facts for relation *is-known-for*. For the other relations, we picked several random samples with 25 pairs and tested each separately. The difference in accuracy was negligible, so we used 25 facts per relation.

Effectiveness. We performed experiments to evaluate the effectiveness of three different rank aggregation techniques (i.e., Spearman's Footrule (SP), Kendall's tau (KE), and Mean Ranking (MR)) as well as the Global Ranking approach (GR). The ranking of the correct relation generated by the system is considered as a measure of success to annotate that relation. The best result is achieved when the correct relation appears at the top of the ranked list for the facts in that relation in the ground-truth. Table 2 shows the number of relations ranked the top 3 positions, as well as anywhere above the 3rd place.

As one can see, GR outperforms the rank aggregation techniques in identifying correct relations. Among the rank aggregation techniques, MR works slightly better than SP and KE in this test. However, there is no statistical significance in the differences between the results of rank aggregation techniques based on analysis of variance. Another observation is that the number of PATTY patterns has an effect on the accuracy, and this effect is more pronounced for the ranking aggregation methods. As shown in Table 1, relations associated with fewer patterns are less likely to be identified correctly by rank aggregation techniques. We argue this is due to lack of sufficient evidence (patterns) for each pair. It follows that rank aggregation techniques require more evidence in order to infer correct relations. On the other hand, the GR performs better for relations associated with fewer patterns. This happens because it is more likely that many relevant patterns appear in the union of patterns of all pairs compared to individual entity pairs.

We also filtered out relations with 200 or less patterns in our corpus, recomputed their prior probabilities, and re-executed the experiments on the same dataset for them. Table 3 shows the results of this experiment. Although MR performed slightly better than the other techniques, a statistical test reveals that the differences are not significant. What is important to note is that, as expected, filtering out less popular relations leads to higher overall accuracy. Applications using our technique can exploit this trade-off to set the appropriate threshold.

Performance. For efficiency reasons, we index patterns using a suffix tree in memory. The average execution times in milliseconds for processing a pair of entities (taken over 20 executions) are: 1688 for SP, 1868 for KE; 1729.4 for MR; and 1719 for GR. As one can see, there are no considerable differences among the methods. In fact, our observations are that the majority of the time is spent on matching the entities against the NELL corpus.

4.1 Towards Knowledge Augmentation

The ultimate goal of our technique is knowledge augmentation by generating new instances of relations from tabular data that are not already in the knowledge base. We performed preliminary experiments to assess if our system could accomplish this goal.

Our first test was with a spreadsheet including song data available at `www.aardvarkdjservices.co.uk` (a website specialized in music services). We looked at 48 *singer*, *song* pairs from 2 albums, with 24 songs from Elvis Presley, and 24 songs from Frank Sinatra. Every approach returned *created* as the best relation between those entities. We manually verified the 48 facts in this case, and found that only 31 were already in YAGO. In another experiment, we used a spreadsheet with data about NBA players extracted from `wwww.espn.go.com`, and tested our system with 100 *player*, *team* pairs. YAGO had 92 of these pairs in the *is-affiliated-to* relation. Every configuration of our system identified all 100 pairs as instances of the *plays-for* relation, which, one can argue, is a suitable and more specific relation for these entities.

5. RELATED WORK

A lot of work has been done towards understanding tables within text or online. Some have attempted to exploit column headers to identify relations between two columns (e.g., [6]), which is akin to schema-based data integration. We make no assumption about the existence of this information in our approach as this information may not be available for all tables, making our method more similar to an instance-based data integration approach.

A probabilistic model is proposed in [16] to associate class labels with columns and identify relations between entity columns and the rest of columns. Recently, the joint inference technique is used to simultaneously annotate table cells, table columns and relations between columns. In [8, 10], graphical models are employed to annotate column headers, table cells, and relations between columns. Our work is similar, to some extent, to [11] in which Wikipedia's tables are used to generate new triples using DBpedia as a knowledge base. Unlike our method, these techniques require linking entities to one or more linked open data repositories.

The problem has also been considered in terms of extracting schema for tabular data. In [5], an extraction system is proposed to convert data stored in spreadsheets into relational tuples. In [1], a set of row classes representing common features of individual rows in a table is identified. Then,

Conditional Random Fields techniques are used to generate a sequence of row labels.

Our work is also related to relation extraction techniques from text corpora. In supervised learning (e.g., [18]), manually labelled relations are used to train a model for labeling relations. On the other hand, in unsupervised approaches (e.g., [13]), strings between entities in a text corpus are clustered and then simplified to generate relations. In [9], the classifier is trained using textual features of sentences between known entities in Freebase. This technique generates instances of new relations, while our technique generates instances of relations linked to exiting relations in linked open data repositories.

6. CONCLUSION

In this paper, we described a probabilistic approach for augmenting linked open data repositories using tabular data, thus tapping into these under-explored sources of valuable information. Unlike prior methods that focus on natural language understanding to determine whether two named entities are even related, we start from the (reasonable) assumption that all entities in the same row of a table are related by construction. Unlike previous methods that attempt to understand tabular data, we take a more pragmatic and effective stance: we label pairs of columns in the table with relations coming from an established knowledge base. By doing so, all facts we extract can be interpreted in the same way as those in the knowledge base.

We described a first implementation of our model using linked open data resources–YAGO, PATTY and NELL–and showed experimentally that the approach is effective, despite the limitations in the way we match entities. We also showed that it rather easy to find new knowledge with our model. Yet, we have only sketched a research direction rich in opportunities to improve knowledge building and linking in our opinion. There are some limitations that we aim to address in future work. Instead of a limited number of YAGO relations, we aim to use a wide range of relations (e.g., those in Freebase). Moreover, we can increase recall by using proper entity linking techniques such as those in [17].

Another interesting line of future work would be to estimate how many new triples could be extracted from tabular data on the entire Web, and how accurate they could be. To do so, one needs a systematic approach and some machinery to automatically check if the new facts already exists in the knowledge base, as well as whether or not these facts are accurate. Different notions of accuracy apply here. For example, it may be that the new facts contradict existing knowledge, or it could be that they are expressed at a different granularity, as was the case for our experiment with NBA players. One could also use both quantitative and qualitative metrics to chart which websites provide the best data.

Acknowledgements. This work was supported in part by grants from the Natural Sciences and Egineering Reseach Council of Canada and the IBM Alberta Centre for Advanced Studies.

7. REFERENCES

[1] M. D. Adelfio and H. Samet. Schema extraction for tabular data on the web. *Proc. VLDB Endow.*, 6(6):421–432, 2013.

[2] C. Bizer, T. Heath, and T. Berners-Lee. Linked data-the story so far. *Int. J. Sem. Web Inf. Sys.*, 5(3), 2009.

[3] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *SIGMOD'08 Conf. Proc.*, 2008.

[4] M. J. Cafarella, A. Halevy, D. Z. Wang, E. Wu, and Y. Zhang. Webtables: Exploring the power of tables on the web. *Proc. VLDB Endow.*, 1(1), 2008.

[5] Z. Chen and M. Cafarella. Automatic web spreadsheet data extraction. In *SSW'13 Conf. Proc.*, 2013.

[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 Symp.: Linked data meets AI*, 2010.

[7] R. Fagin, R. Kumar, and D. Sivakumar. Comparing top k lists. In *SODA'03 Conf. Proc.*, 2003.

[8] G. Limaye, S. Sarawagi, and S. Chakrabarti. Annotating and searching web tables using entities, types and relationships. *Proc. VLDB Endow.*, 3(1-2):1338–1347, 2010.

[9] M. Mintz, S. Bills, R. Snow, and D. Jurafsky. Distant supervision for relation extraction without labeled data. In *ACL'09 Conf. Proc.*, 2009.

[10] V. Mulwad, T. Finin, and A. Joshi. Semantic Message Passing for Generating Linked Data from Tables. In *ISWC'13 Conf. Proc.*, 2013.

[11] E. Munoz, A. Hogan, and A. Mileo. Triplifying wikipedia's tables. In *LD4IE'13 Workshop Proceedings, ISWC. CEUR*, 2013.

[12] N. Nakashole, G. Weikum, and F. Suchanek. Patty: A taxonomy of relational patterns with semantic types. In *EMNLP-CoNLL'12 Conf. Proc.*, 2012.

[13] Y. Shinyama and S. Sekine. Preemptive information extraction using unrestricted relation discovery. In *HLT-NAACL'06 Conf. Proc.*, 2006.

[14] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A core of semantic knowledge. In *WWW'07 Conf. Proc.*, 2007.

[15] P. P. Talukdar, D. Wijaya, and T. Mitchell. Acquiring temporal constraints between relations. In *CIKM'12 Conf. Proc.*, 2012.

[16] P. Venetis, A. Halevy, J. Madhavan, M. Paşca, W. Shen, F. Wu, G. Miao, and C. Wu. Recovering semantics of tables on the web. *Proc. VLDB Endow.*, 4(9):528–538, 2011.

[17] M. A. Yosef, J. Hoffart, I. Bordino, M. Spaniol, and G. Weikum. Aida: An online tool for accurate disambiguation of named entities in text and tables. *Proc. VLDB Endow.*, 4(12):1450–1453, 2011.

[18] G. Zhou, M. Zhang, D. Ji, and Q. Zhu. Tree Kernel-Based relation extraction with Context-Sensitive structured parse tree information. In *EMNLP-CoNLL'07 Conf. Proc.*, 2007.