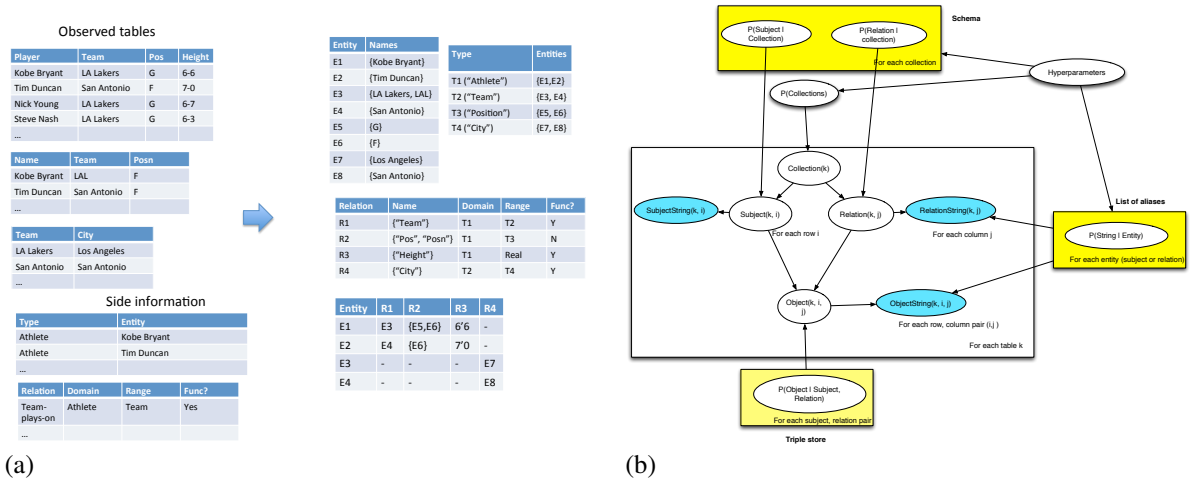
Extracting Entities and Relations from Web Tables Using a Non-parametric Generative Model

Jon Malmaud, MIT **Kevin Murphy, Google**

1 Introduction and related work

Knowledge Bases, such as Google's Knowledge Graph, contain information about entities and their relationships. Typically this information is stored in RDF format, which is a set of subject-predicate-object triples. In this paper, we discuss a way to extract such information from tables found on the web (Cafarella et al. 2008).



(a) (b)

Figure 1: (a) Sample input data (left) and sample output (right). (b) The graphical model. Blue nodes are observed. Yellow nodes represent the hidden schema and KB (model "parameters"). White nodes inside the plate are the table specific hidden variables.

Figure 1(a) illustrates the basic idea. We show excerpts from several tables containing information about various basketball players from four different teams (for simplicity of presentation, we assume the first column contains the subject and the other columns contain the attributes). In order to integrate this information into a KB, we need to solve two main problems. The first problem is entity linkage (aka record linkage or co-reference resolution), which means figuring out which entity is being referred to by each mention; for example, both "LA Lakers" and "LAL" refer to the same entity. The second problem is relation extraction (aka schema alignment), which is the task of figuring out which predicate is being referred to by each column header; for example, both the columns "Pos" and "Posn" refer to the same relation, representing the position of a player on a team.

There is a lot of prior work on entity linkage and schema alignment, which we do not have space to review here. However, it is worth mentioning (Limaye et al. 2010), which is closely related to our paper: they use a conditional random field (CRF) where each entry in a table is labeled by the corresponding entity, and each column is labeled by the corresponding relation. We adopt a similar approach in this paper, except we use a generative model, and we associate each row with a single latent subject.

A more important difference from this prior work is that we use a non-parametric generative model, which allows for an unbounded number of entities and relations. There have been previous papers, such as (Haghighi and Klein 2010; Wick et al. 2013), which have allowed for an unbounded number of entities, but we are not aware of methods that also allows for an unbounded number of relations. (The work on "open information extraction" (Etzioni et al. 2011) is not applicable here, since it does not attempt to align these relation mentions to a canonical latent schema.) In summary, the current paper is, as far as we know, the first to attempt to extract new entities and new relations from tabular data in a coherent probabilistic framework.

2 Method and results

Our probabilistic model is sketched in Figure 1(b). The corresponding generative story is as follows. For the k'th table, we pick a hidden type (e.g., the table could be about sports players) $t_k \sim p(t)$. For each row i, we pick a hidden subject from $s_{ki} \sim p(e|t_k)$, and for each column j, we pick a hidden relation $r_{kj} \sim p(r|t_k)$. Next, for the (i, j)'th cell in the table, we sample $o_{kij} \sim p(o|s_{ki}, r_{kj})$. Finally, for each hidden node (s_{ki} or r_{kj} or o_{kij}), we generate a corresponding observed string from $p(s|\cdot)$. All these distributions are represented by Chinese restaurant processes (CRPs), which allows for an unbounded number of values.

For numeric nodes, we sample the observed value from a Gaussian centered on the hidden value. A more realistic model would represent the latent units (eg. feet or metres) and scale factor (eg. $\times 1000$) for each numeric column, and would transform the hidden value before "rendering" it, similar to the approach in (Zhang and Chakrabarti 2013). Note that a big virtue of generative models is their modularity; thus it is easy to plug in more flexible distributions for other kinds of data, such as dates.

To "learn" the model, we condition on all the data observed in all the tables and perform posterior inference; thus no labeled training data is required. (If we have missing data, we can impute it if desired; this can be thought of as a kind of model-based auto-completion function for tables.) To speed up inference, we can "seed" the model with known facts from the KB. We can also integrate triples extracted from text in the same way. Thus this model provides a convenient way to derive a "universal schema", combining prior knowledge from KBs with information from tables and text (cf. (Yao et al. 2012)).

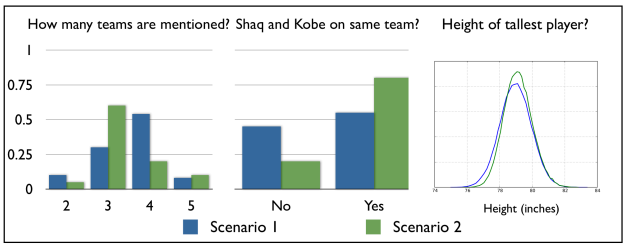


Figure 2: Posterior marginals for the 3 query random variables under 2 different sets of evidence.

To demonstrate the advantage of joint reasoning using our model, we created three tables, as shown in Figure 1(a) (for brevity, we only show two different teams, but in the actual data there are three). Note that these are based on actual tables found on the web. We consider using this data to compute probabilistic answers to the following three queries: "How many different sports teams are mentioned?", "Is Kobe Bryant on the same team as Shaquille O'Neal'?" and "What is the height of the tallest player on the Lakers?". The answers to these questions depend on what evidence we have seen. We consider two scenarios, one in which we only observe the tables, and another in which we additionally conditioned on side information from an auxiliary triple store.

The results are shown in Figure 2. In the first scenario, there is ambiguity whether 'LaL' and 'LA Lakers' refer to the same team or two different teams. In the second scenario, the model softly aligns the columns named 'Team' with the KB relation 'team-plays-on', based on the distribution of names within the 'Team' column and 'Player' column. The KB marks this relation as functional, which provides evidence that 'LaL' and 'LA Lakers' refer to the same team. Thus we now believe that it is more likely there are just 3 teams (the correct answer), and the posterior expected value of the height of the tallest Lakers player slightly increases (since Kobe is taller than Nick Young).

We have implemented a prototype of this model using Venture, which is a version of the Church probabilistic programming language (Goodman et al. 2008). We are currently in the process of writing custom MCMC code in Python/ C, so we can tackle inference on a larger number of real tables. However, to scale to the millions of tables found on the web, we will likely have to adopt blocking techniques, similar to those discussed in (Wick et al. 2013).

References

Cafarella, M., A. Halevy, Z. D. Wang, E. Wu, and Y. Zhang (2008). WebTables: Exploring the Power of Tables on the Web. *VLDB 1*(1), 538–549.

Etzioni, O., A. Fader, J. Christensen, S. Soderland, and Mausam (2011). Open Information Extraction: the Second Generation. In *Intl. Joint Conf. on AI*.

Goodman, N. D., V. K. Mansinghka, D. M. Roy, K. Bonawitz, and J. B. Tenenbaum (2008). Church: a language for generative models. In *UAI*.

Haghighi, A. and D. Klein (2010). Coreference resolution in a modular, entity-centered model. In *NAACL*.

Limaye, G., S. Sarawagi, and S. Chakrabarti (2010). Annotating and searching web tables using entities, types and relationships. *VLDB 3*(1).

Wick, M., S. Singh, H. Pandya, and A. McCallum (2013). A Joint Model for Discovering and Linking Entities. In *AKBC Workshop*.

Yao, L., S. Riedel, and A. McCallum (2012). Probabilistic databases of universal schema. In *AKBC Workshop*, pp. 116–121.

Zhang, M. and K. Chakrabarti (2013). InfoGather+: Semantic Matching and Annotation of Numeric and Time-Varying Attributes in Web Tables. In *Proc. SIGMOD*.