Markov Logic Sets: Towards Lifted Information Retrieval using PageRank and Label Propagation

Marion Neumann and **Babak Ahmadi** and **Kristian Kersting**
Knowledge Dicovery Department, Fraunhofer IAIS
53754 Sankt Augustin, Germany
{firstname.lastname}@iais.fraunhofer.de

Abstract

Inspired by "Google$^{\text{TM}}$ Sets" and Bayesian sets, we consider the problem of retrieving complex objects and relations among them, i.e., ground atoms from a logical concept, given a query consisting of a few atoms from that concept. We formulate this as a within-network relational learning problem using few labels only and describe an algorithm that ranks atoms using a score based on random walks with restart (RWR): the probability that a random surfer hits an atom starting from the query atoms. Specifically, we compute an initial ranking using personalized PageRank. Then, we find paths of atoms that are connected via their arguments, variablize the ground atoms in each path, in order to create features for the query. These features are used to re-personalize the original RWR and to finally compute the set completion, based on Label Propagation. Moreover, we exploit that RWR techniques can naturally be lifted and show that lifted inference for label propagation is possible. We evaluate our algorithm on a real-world relational dataset by finding completions of sets of objects describing the Roman city of Pompeii. We compare to Bayesian sets and show that our approach gives very reasonable set completions.

Introduction

Imagine you catch parts of a conversation of two archaeologists: "... the room is a taberna ... City of Pompeii ..." What is the conversation about? Which concept is described by $\texttt{isa}(\texttt{taberna})$ and $\texttt{city}(\texttt{pompeii})$? "Taberna" is the Latin word for "shop", and Pompeii is the Roman city in southern Italy that was covered in volcanic ash during the eruption of Mt. Vesuvius in 79 AD. So, the two archaeologists might have talked about the daily life of Romans in Pompeii and where they went shopping.

This example encapsulates the very practical and interesting machine learning and information retrieval problem, namely to automatically create sets/clusters of items from a few examples only, and has originally been addressed by "Google$^{\text{TM}}$ Sets"[1]. Since it involves forming a cluster once some elements of that cluster have been revealed and in turn can provide useful information as to the features which are relevant for forming that cluster, Ghahramani and Heller (2005) have called it *clustering on demand*. More formally, consider a universe of items $\mathcal{I}$. For instance, $\mathcal{I}$ may consist of publications, web pages, images, movies, places and artifacts found in Pompeii, or any other type of objects we may wish to form queries on. The user provides a query in the form of a very small subset of items $\mathcal{Q} \subset \mathcal{I}$ constituting examples of some concept in the data, say $\{\texttt{isa}(\texttt{taberna}), \texttt{city}(\texttt{pompeii})\}$. The algorithm then has to provide a completion to the set $\mathcal{Q}$, that is, some set $\mathcal{C} \subset \mathcal{I}$ that presumably includes all the elements in $\mathcal{Q}$ and other elements in $\mathcal{I}$, which are also elements of this concept, say $\{\texttt{house}(\texttt{h1}), \texttt{house}(\texttt{h2}), \texttt{house}(\texttt{h4}), \ldots\}$ and other objects that are related to shops in Pompeii.

"Google$^{\text{TM}}$ Sets" treats the *clustering on demand* problem as a large-scale clustering problem that involves many millions of data instances extracted from web data. Ghahramani and Heller (2005) have proposed a Bayesian approach, called *Bayesian sets* (BSets), for solving it. As in other retrieval problems, BSets computes a score measuring for each item how relevant it is to the query. Specifically, using a probabilistic model of the unknown concept, it computes the relevance score by comparing the posterior probability of an item, given the query, to the prior probability of that item. However, "Google$^{\text{TM}}$ Sets" and BSets have focused on propositional universes assuming that the items are representable as attribute-value vectors. Nowadays, the role of structure and relations in the data, becomes more and more important (Getoor and Taskar 2007): information about one item can help us to draw conclusions about other items. Such domains are hard to represent meaningfully using a set of propositional features. Thus, it is natural to ask the question "How do we solve the clustering on demand problem for relational domains?" An investigation of this question leads to our main contribution: *Markov logic sets* (MLS).

Defining the relevance score between two nodes is one of the fundamental building blocks in graph mining. One very successful technique is based on random walks with restart (RWR). Concretely, MLS employs the seminal PageRank (PR) algorithm (Brin and Page 2006), which is the (limit) stationary probability that a random walker is at some node, but personalized on the query. Following the idea underlying BSets, MLS actually computes the relevance score by

[1] `http://labs.google.com/sets`

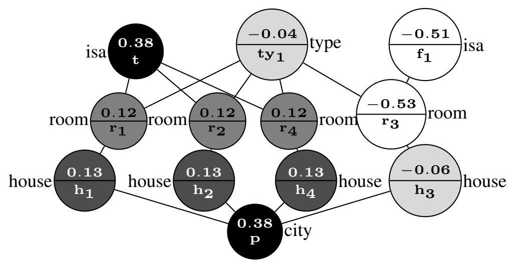


Figure 1: Markov logic sets' ranking (rounded to the second digit) for the query $\{\texttt{isa}(\texttt{taberna}), \texttt{city}(\texttt{pompeii})\}$ on the toy version of the Pompeii dataset, which is described in the main text. All houses with a shop are ranked high, whereas $\texttt{h}_3$, which has no shop, is ranked low.

comparing the posterior PR of an item given the query, i.e., the personalized PageRank, to the prior PR of that item, i.e., the uniform PageRank. This differential PR (DPR) has been proven to be successful for ranking folksonomies (Hotho et al. 2006) but has two major drawbacks for our setting:

1. It does not make use of the rich structure provided by relational domains. It treats items, i.e., ground atoms monolithically. To generalize across items, we find paths of ground atoms that are connected via their arguments and variablize the ground atoms in each path, in order to create logical features, and use the features to compute a personalization vector for (D)PR.
2. It ranks items high that are generally surprising but unrelated to the query. To overcome this, we only estimate negative (positive) labels for items ranked lowest (highest) by DPR if they conform the features. Then, the labels are spread across the graph using label propagation (LP), see e.g. (Zhou et al. 2003), another RWR technique.

We evaluate MLS[2] by finding completions of sets of real-world objects describing the Roman city of Pompeii. For instance, MLS discovers that the two archaeologists talked about "*shopping places in Pompeii*" as shown in Fig. 1. A comparison to BSets using our logical features shows that MLS give very reasonable set completions.

However, this paper makes another important and novel contribution. We note that the core computations in both PR and LP, namely solving systems of linear equations can be done distributively and efficiently using Gaussian belief propagation (GaBP), see e.g. (Shental et al. 2008). There are, however, many improvements that can be made to (Ga)BP, especially when applied to inference in graphical models containing structural symmetries. Such symmetries are commonly found in relational domains. To exploit these symmetries, lifted BP (LBP) approaches have been recently developed (Singla and Domingos 2008; Kersting, Ahmadi, and Natarajan 2009) to automatically group nodes and potentials together into supernodes and superpotentials if they send and receive identical messages. LBP then runs a modified BP on this lifted network. LBP has been proven to yield significant efficiency gains compared to BP on important AI tasks such as link prediction and entity resolution. Here, we demonstrate its potential for another important AI and novel lifted inference task: information retrieval using LP.

We start off by touching upon related work. After reviewing relational logic, pathfinding, and RWR, we develop MLS. Before concluding, we present our experiments.

Related Work

Similar to (Silva et al. 2010), MLS considers the clustering on demand problem as within-network relational learning problem: the input is a single data graph in which some of the instances are labeled and the task is to infer the labels of the remaining instances, see e.g. (Neville and Gallagher 2009) and references in there. Existing within-network relational learning methods indeed exploit the statistical dependencies among instances in order to improve classification performance. However, they assume that positive and negative labels are given for some nodes, and inference is generally still at the level of propositional logic. The latter also holds for the relational variants of BSets (Silva et al. 2010) and it is difficult — if not impossible — to employ existing lifted inference for them. Moreover they assume a single, finite feature space for all items; MLS does not since it employs RWR and learns query specific features automatically. Lao and Cohen (2010) present an approach to relational retrieval based on path-constrained RWR; MLS, however, constructs positive and negative labels from logical features and apllies LP. Muggleton (1996) has proposed inductive logic programming (ILP) techniques for single-predicate learning from positive examples only; MLS solves a multi-predicate learning problem in a transductive setting. Probabilistic Explanation Based Learning (Kimmig, De Raedt, and Toivonen 2007) produces generalized explanations from examples; MLS can also provide set completions, even if no concept/explanation is present.

Relational pathfinding has been proven successful for learning the structure of Markov logic networks (MLN) (Richardson and Domingos 2006). For instance, Mihalkova and Mooney (2007) as well as Kok and Domingos (2009) proposed to use it to find a path of ground atoms in the training data. They also variablize ground atoms in the path but then they construct a MLN whose formulas are the paths viewed as Boolean variables (conjunctions of atoms). In contrast, MLS uses the features to compute a PR that generalizes well across similar items.

Finally, MLS is related to recent lifted probabilistic inference approaches traditionally designed for discrete domains, e.g. (Milch et al. 2008). LP, however, requires to deal with continuous domains. Only very recently, Choi and Amir (2010) have presented the first continuous lifted variable elimination. Arguably, the simplest and most efficient approximate lifted inference algorithms are based on the already mentioned LBP approaches. They have been developed for discrete domains only. Only recently, we have developed a lifted GaBP (Ahmadi, Kersting, and Sanner 2011) and present its first application to clustering here.

To summarize, lifted clustering on demand of arbitrary relational domains has not been addressed so far.

[2]The use of *Markov* random walks as well as *logical* features to compute *set* completions also explains the name *Markov logic sets*.

Relational Logic and Pathfinding

In relational logic, see e.g. (De Raedt 2008), formulas are constructed using three types of symbols: constants, variables, and predicates. Constants represent objects in a domain of discourse (e.g., houses such as $\mathtt{h_1}$, $\mathtt{h_2}$, etc., cities such as pompeii, and rooms such as $\mathtt{r_1}$, $\mathtt{r_2}$, etc. Variables (e.g., uppercase X, Y) range over the objects. Predicates represent relations among objects (e.g., $\mathtt{in(h_1, p)}$ and $\mathtt{in(r_1, h_1)}$), or attributes of objects (e.g., $\mathtt{isa(r_1, t)}$ where t denotes taberna). An atom is a predicate symbol applied to a list of arguments, which may be variables or constants (e.g., $\mathtt{in(X, p)}$). A ground atom is an atom all of whose arguments are constants (e.g., $\mathtt{in(h_1, p)}$). A universe is a set of ground atoms. As an example, consider the following universe over objects in the city of p, houses $\mathtt{h_1}$,$\mathtt{h_2}$, $\mathtt{h_3}$,$\mathtt{h_4}$, rooms $\mathtt{r_1}$,$\mathtt{r_2}$,$\mathtt{r_3}$,$\mathtt{r_4}$, functions of rooms t, $\mathtt{f_1}$, and room types $\mathtt{ty_1}$: $\{\mathtt{in(h_1, p)}$, $\mathtt{in(h_2, p)}$, $\mathtt{in(h_3, p)}$, $\mathtt{in(h_4, p)}$,$\mathtt{in(r_1, h_1)}$, $\mathtt{in(r_2, h_2)}$, $\mathtt{in(r_3, h_3)}$, $\mathtt{in(r_4, h_4)}$, $\mathtt{isa(r_1, t)}$,$\mathtt{isa(r_2, t)}$, $\mathtt{isa(r_4, t)}$, $\mathtt{isa(r_3, f_1)}$, $\mathtt{typeof(r_1, ty_1)}$,$\mathtt{typeof(r_2, ty_1)}$, $\mathtt{typeof(r_3, ty_1)}$, $\mathtt{typeof(r_4, ty_1)}\}$, which is motivated by a real-world database[3] of objects found in Pompeii (Allison 2001).

A universe can be viewed as a hypergraph. A hypergraph is a straightforward generalization of a graph, in which an edge can link any number of nodes, rather than just two. More formally, a hypergraph is a pair (V, E) where V is a set of nodes, and E is a multiset of labeled non-empty subsets of V, called hyperedges. Now, the constants appearing in a universe are the nodes and ground atoms are the hyperedges. Each hyperedge is labeled with the predicate symbol of the corresponding ground atoms. Nodes (constants) are linked by a hyperedge if and only if they appear as arguments in the hyperedge. Examples are shown in Fig. 2 where we have omitted the edge labels but have associated the type of nodes for the sake of readability.

In MLS, we find paths in a hypergraph corresponding to a universe. A path is defined as a set of hyperedges such that for any two hyperedges e_0 and e_n in the set, there exists an ordering of (a subset of) hyperedges in the set $e_0, e_1, \ldots, e_n$ such that e_i and e_{i+1} share at least one node. A path of hyperedges can be generalized into path features, i.e., conjunctions of relational atoms by variablizing their arguments.

(Lifted) Random Walk with Restarts

Recall the RWR approach to ranking nodes in a graph touched upon in the introduction. In a nutshell, a Markov chain transition matrix $\mathbf{M}$ is constructed out of a given graph G. Let $\mathbf{A}$ be the $m \times m$ adjacency matrix of G, that is $\mathbf{A}_{ij} = 1$ if there is an edge from vertex j to vertex i and zero otherwise. PageRank (PR) now constructs the probability transition matrix $\mathbf{M}$ by adding $\mathbf{1}^T$ to all $\mathbf{0}^T$ rows and renormalizes each row of A to sum up to 1. Furthermore, a prior probability $\mathbf{v}$ can be taken to weight the results. The personalized PR (PPR) can then be computed by solving the following system of linear equations $(\mathbf{I} - \alpha\mathbf{M}^T)\mathbf{x} = \mathbf{v}$ where α trades off speed of convergence with the accuracy of the solution and $\mathbf{I}$ is the identity matrix. If we additionally have positive and negative labels for some of the nodes of the graph, we can employ another RWR technique, called label propagation (LP), to propagate the information through the graph in order to label all unlabeled nodes, see e.g. (Zhou et al. 2003). We compute the diagonal degree matrix $\mathbf{D}$ with $\mathbf{D}_{ii} = \sum_j \mathbf{A}_{ij}$. Then, we compute the normalized graph Laplacian $\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$, and initialize a column vector $\mathbf{y}$ according to the labels we have, that is $\mathbf{y}_i = 1$ if node i is labeled positive, $\mathbf{y}_i = -1$ if labeled negative, and $\mathbf{y}_i = 0$ otherwise. The final labeling $\mathbf{x}$ can then be computed by solving the following system of linear equations $(\mathbf{I} - \alpha\mathbf{L})\mathbf{x} = (1 - \alpha)\mathbf{y}$. We note that the core computations in both PR and LP can be solved by combining Shental et al.'s (2008) GaBP approach to solving linear systems together with Kersting, Ahmadi, and Natarajan's (2009) color passing approach for LBP, yielding Lifted GaBP (LGaBP).

[3] http://www.stoa.org/projects/ph/home

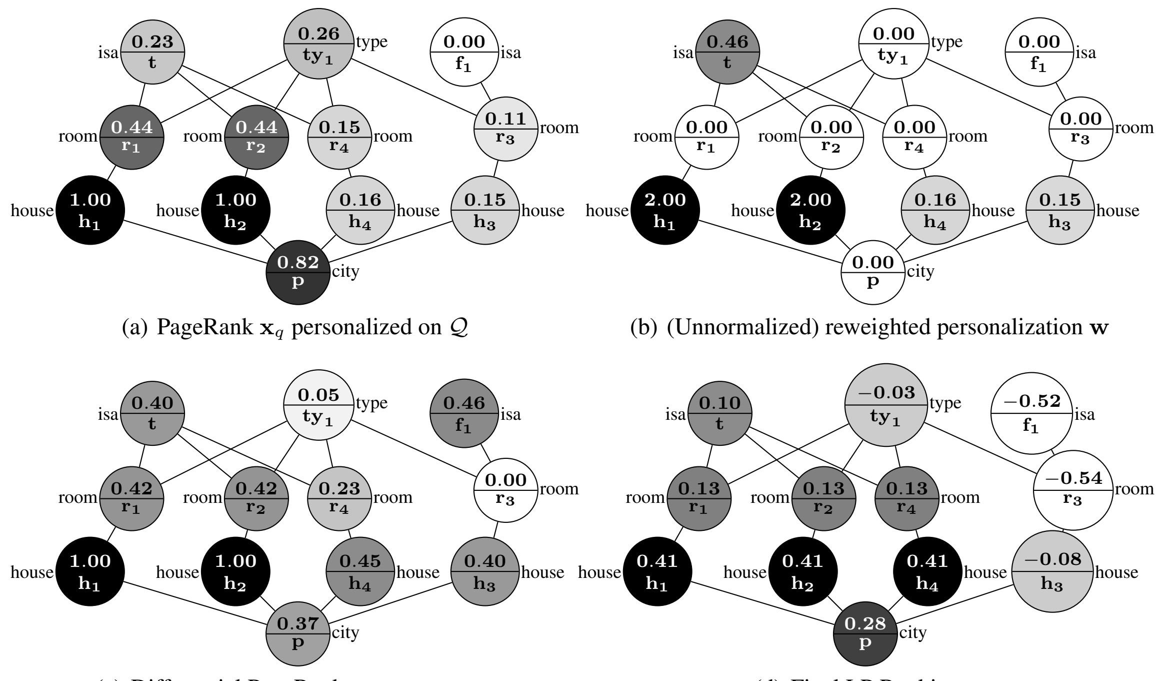
***Algorithm 1**: Markov Logic Sets*

```
   Input: Query set $\mathcal{Q}$, Graph $G = (V, E)$
1  Set $V_+ = \mathcal{Q}$ and $V_- = \emptyset$ /* Initialize labels */;
2  Set $\mathcal{L} = \emptyset$ /* Initialize feature set */;
3  $\mathbf{x}_q = \mathbf{PPR}(G, \mathcal{Q})$ /* PPR on $\mathcal{Q}$*/ ;
4  Compute paths $\mathcal{P}$ in $G$ (up to depth $k$) that start in one
   of the query atoms in $\mathcal{Q}$;
5  foreach $p \in \mathcal{P}$ do /* Pathfeatures */
6      Generalize $p$ into pathfeatures $\mathcal{F}$ by variablizing
       some of the arguments /* see main text */;
7  foreach $f \in \mathcal{F}$ do /* Logical features */
8      if $f$ is discriminative then /* see main text */
9          Include $f$ in $\mathcal{L}$;
10
11 foreach $v \in V$ do /* Reweighted Personalization */
12     $\mathbf{w}(v) = 0$ /* weight of item $v$ */ ;
13     foreach matching of $l \in \mathcal{L}$ in $G$ ending in $v$ do
14         Let $\mathbf{x}_q(f, v)$ be the PPR of the item matching
           the end atom of $l$;
15         $\mathbf{w}(v) = \mathbf{w}(v) + \mathbf{x}_q(f, v)$;
16
17 $\mathbf{x}_p = \mathbf{PPR}(G, \mathbf{w})$ /* reweighted PPR */ ;
18 $\mathbf{x}_u = \mathbf{PR}(G)$ /* uniform PR */ ;
19 $\mathbf{x}_d = \mathbf{x}_p - \mathbf{x}_u$ /* differential PR */ ;
20 foreach $v \in V$ do /* Label Set */
21     Add $v$ to $V_-$ if $|\mathbf{x}_i(v) - \min \mathbf{x}_i| < \epsilon$, $i \in \{p, d\}$ and
       no ground feature fired for $v$;
22     Add $v$ to $V_+$ if $|\mathbf{x}_i(v) - \max\{\mathbf{x}_i(u) | u \notin \mathcal{Q}\}| < \epsilon$,
       $i \in \{p, d\}$ and a feature fired for $v$;
23 Run LP on $G$ using $V_+$ and $V_-$ as pos resp. neg labels;
24 Return the ranking $\mathbf{r}$ produced by LP;
```

Markov Logic Sets

Alg. 1 summarizes MLS, and Fig. 2 provides an example run with query $\{\mathtt{house(h_1)}, \mathtt{house(h_2)}\}$ for the universe of objects in the city of Pompeii described previously.

MLS begins with computing the PPR on the query $\mathcal{Q}$ (line 3). The result, see Fig 2(a), illustrates that this PPR is not enough to solve the clustering on demand problem.



(a) PageRank $\mathbf{x}_q$ personalized on $\mathcal{Q}$

(b) (Unnormalized) reweighted personalization $\mathbf{w}$

(c) Differential PageRank $\mathbf{x}_d = \mathbf{x}_p - \mathbf{x}_u$

(d) Final LP Ranking $\mathbf{r}$

Figure 2: The major steps of running Markov logic sets for the query $\{\mathtt{house}(\mathtt{h}_1), \mathtt{house}(\mathtt{h}_2)\}$ on the toy version of the Pompeii dataset, which is described in the main text. The final ranking is shown in (d). Using 0.0 as threshold would result in $\{\mathtt{h}_1, \mathtt{h}_2, \mathtt{h}_4, \mathtt{p}, \mathtt{r}_1, \mathtt{r}_2, \mathtt{r}_4, \mathtt{t}\}$ as set completion.

House $\mathtt{h}_4$ should be in the set completion as it has a shop. Including also $\mathtt{r}_4$, the room of $\mathtt{h}_4$, also includes $\mathtt{h}_3$, the only house without a shop. Excluding $\mathtt{r}_4$, however, is suboptimal as it is the only room and the shop of $\mathtt{h}_4$.

To improve upon this, MLS generalizes the personalization values across the network. Hence, MLS finds paths in the hypergraph (line 4), say $p =$ "$\mathtt{house}(\mathtt{h}_1) - \mathtt{in}(\mathtt{r}_1, \mathtt{h}_1) - \mathtt{room}(\mathtt{r}_1) - \mathtt{typeof}(\mathtt{r}_1, \mathtt{t}) - \mathtt{isa}(\mathtt{t})$", and variablizes all its arguments except the atoms in the start and end nodes. To create candiate logical features per path the constants in the start and end nodes are variablized in all possible combinations (lines 5-6), for the example path above, one of the four possible pathfeatures is: $f =$ "$\mathtt{house}(\mathtt{X}) - \mathtt{in}(\mathtt{Y}, \mathtt{X}) - \mathtt{room}(\mathtt{Y}) - \mathtt{typeof}(\mathtt{Y}, \mathtt{t}) - \mathtt{isa}(\mathtt{t})$", here only $\mathtt{h}_1$ is variablized. However, MLS selects only discriminative features (lines 7-9). It checks whether (1) the variance of the personalized PageRank scores of query items covered by a feature (PR presonalized on end atoms) is small and (2) their mean is significantly different from the mean of PageRank scores of items over the same predicate that are not in the query (line 7). Only if (1) and (2) succeed using a χ^2-test, we include the feature (line 9), this applies for example to f.

Afterwards, MLS recomputes the personalization of each node in G using the resulting logical features, again based on the PPR on $\mathcal{Q}$ (lines 11-15). For our running example, this yields the unnormalized personalization vector shown in Fig 2(b). As one can see, the weight of item `taberna` is increased from 0.0 (it is not a query member, consequently its initial personalization was 0.0) to 0.46. One is tempted to just recompute the PPR using this new personalization, and indeed items with minimal rank are definitely unrelated to the query — therefore we give them a negative label (line 21) but only if no feature fired for the item — and `taberna` gets ranked higher now. However, $\mathtt{h}_3$ and $\mathtt{h}_4$ still get similar, low relevance score: the concept "houses with shops" has not been identified yet. Why is this the case?

Ranking items simply by the (reweighted) personalized PR is not sensible since some items may be more probable than others, regardless of the query. MLS removes this effect by computing the differential PR (DPR) w.r.t. to the uniform PR (lines 17-19). This DPR is shown in Fig 2(c). Now, it is sensible to assign positive (negative) labels to the items with maximal (minimal) DPR (lines 20-22) if features have (not) fired. Finally, MLS runs LP using V_+ and V_- to create the final ranking (line 23) as shown in Fig. 2(d). As one can see, $\mathtt{r}_4$ and $\mathtt{h}_3$ can clearly be separated.

Experimental Evaluation

Our intention here is to investigate the following questions: **(Q1)** Is MLS effective in ranking atoms w.r.t. a given query? **(Q2)** How does MLS perform compared to BSets and MLS's intermediate rankings $\mathbf{x}_p$ and $\mathbf{x}_d$? **(Q3)** Are there symmetries in the LP matrix that can be employed by LGaBP?

To this aim, we implemented MLS and BSets in Python using LGaBP for the PR and LP computations. In all experiments, we set $\alpha = 0.5$ for the RWR computations. For (L)GaBP, we used the "flooding" message protocol where messages are passed in parallel each step. Messages were not damped, and the convergence threshold was 10^{-8}. We did not compare to Silva *et al.*'s (2010) relational BSets variant as they assume a single joint feature space. To accom-

	$\mathcal{Q}_1 = \{\mathbf{house(h_2), house(h_5), house(h_9), house(h_{12})}\}$
$\mathbf{x}_p$	4 query houses, 1 function, city(pompeii), 3 types, 2 functions, 13 rooms of h$_2$, 2 types, 1 function, 13 rooms of h$_5$
$\mathbf{x}_d$	4 query houses, 13 rooms of h$_2$, 20 rooms of h$_5$, 3 rooms of h$_{12}$, 4 rooms of h$_9$
BSets$_{\mathcal{F}}$	all 30 houses, 1 rooms of h$_2$, 4 rooms of h$_5$, 2 rooms of h$_{12}$, 3 types
BSets$_{\mathcal{L}}$	8 houses with shops, 11 rooms (shops), isa(taberna), typeof(ty$_{20}$), 19 other houses
MLS	typeof(ty$_{20}$), isa(taberna), 8 houses with shops, 22 other houses, city(pompeii), 7 rooms (shops)

	$\mathcal{Q}_2 = \{\mathbf{house(h_2), house(h_5), house(h_9)}\}$
$\mathbf{x}_p$	3 query houses, 1 function, city(pompeii), 3 types, 13 rooms of h$_2$, 1 function, 1 type, 17 rooms of h$_5$
$\mathbf{x}_d$	3 query houses, 13 rooms of h$_2$, 20 rooms of h$_5$, 4 rooms of h$_9$
BSets$_{\mathcal{F}}$	3 query houses, all remaining 27 houses, 8 rooms of h$_2$ resp. h$_5$, 2 types
BSets$_{\mathcal{L}}$	3 query houses, all remaining 27 houses, 10 rooms
MLS	3 types and 3 functions of rooms appearing in the query houses, all 30 houses, city(pompeii), 3 rooms

	$\mathcal{Q}_3 = \{\mathbf{room(r_{1e}), room(r_{1r}), room(r_{1h}), room(r_{1f}), room(r_{1o})}\}$
$\mathbf{x}_p$	house(h$_1$), 5 query rooms, 4 functions, 5 types, city(pompeii), 15 rooms of h$_1$, 6 houses, 2 types, 1 function
$\mathbf{x}_d$	house(h$_1$), 5 query rooms, 15 rooms of h$_1$, 1 type, 15 functions, 5 other rooms
BSets$_{\mathcal{F}}$	5 query rooms, 15 rooms of h$_1$, 20 other rooms
BSets$_{\mathcal{L}}$	20 rooms of h$_1$, city(pompeii), house(h$_1$), 18 other rooms
MLS	house(h$_1$), 5 query rooms, 15 rooms of h$_1$, 3 types resp. functions of rooms in $\mathcal{Q}_3$, city(pompeii), 12 other rooms

	$\mathcal{Q}_4 = \{\mathbf{p(a_1), p(a_3), p(a_4)}\}$
$\mathbf{x}_p$	p(a$_4$), p(a$_1$), p(a$_3$), p(a$_2$), p(a$_5$), p(b$_5$), p(d$_2$), c(a$_2$), s(a$_2$), s(a$_3$), p(d$_1$), p(d$_3$), p(b$_2$), p(c$_4$), p(d$_4$), p(d$_5$), p(b$_1$), p(b$_4$), p(b$_3$), p(c$_2$)
$\mathbf{x}_d$	p(a$_3$), p(a$_1$), s(a$_2$), s(a$_1$), s(a$_4$), c(a$_2$), c(a$_4$), p(a$_5$), s(a$_3$), p(a$_4$), s(b$_5$), c(b$_5$), s(d$_2$), s(b$_2$), c(d$_1$), c(b$_1$), s(b$_1$), s(b$_4$), p(b$_3$), p(d$_5$)
BSets$_{\mathcal{F}}$	p(a$_5$), p(a$_3$), s(a$_3$), s(a$_4$), s(a$_2$), s(a$_1$), s(b$_2$), s(b$_4$), s(b$_1$), s(b$_5$), s(d$_2$), p(a$_1$), p(a$_2$), c(a$_2$), c(a$_4$), c(b$_1$), c(b$_5$), c(d$_1$), p(a$_4$), p(d$_5$)
BSets$_{\mathcal{L}}$	p(a$_1$), p(a$_4$), p(a$_2$), p(a$_3$), p(a$_5$), s(a$_4$), s(a$_1$), s(a$_2$), s(a$_3$), c(a$_2$), c(a$_4$), c(b$_1$), s(b$_1$), s(b$_2$), s(b$_4$), s(b$_5$), c(b$_5$), s(d$_2$), c(d$_1$), p(b$_5$)
MLS	p(a$_5$), p(a$_3$), p(a$_4$), p(a$_1$), s(a$_2$), c(a$_2$), p(a$_2$), s(a$_3$), s(a$_1$), s(a$_4$), c(a$_4$), p(d$_2$), s(d$_2$), p(d$_4$), p(d$_5$), p(d$_1$), c(d$_1$), p(c$_4$), p(d$_3$), p(c$_3$)

Table 1: Rankings produced by MLS, intermediate rankings of MLS, and BSets: **($\mathcal{Q}_1$-$\mathcal{Q}_3$)** Top 40 ranked ground atoms for three different queries on the Pompeii dataset consisting of rooms in 30 Pompeian households. Due to space limitations, the rankings are shown using a compressed, almost natural language like representation. E.g. "4 query houses, 1 function, city(pompeii), ..." denotes a ranking in which the four houses in the query are ranked first, followed by a ground atom describing a function of a room, followed by the ground atom city(pompeii), and so on. **($\mathcal{Q}_4$)** Top 20 ranked ground atoms for a query consisting of 3 persons from clique a on the smoker-friends graph, where a$_\mathrm{i}$ denotes the i-th person in clique a. Predicates are person, cancer, smokes (denoted by p, c and s) and friends. Items of clique a are highlighted.

modate for this, we ran BSets (with its parameter $c = 2$) using our feature sets $\mathcal{F}$ resp. $\mathcal{L}$. More precisely, BSets$_{\mathcal{F}}$ uses all possible relational pathfeatures, whereas BSets$_{\mathcal{L}}$ includes only the discriminative features found by MLS. For evaluation purposes, we ran all approaches on two datasets: a relational dataset in the spirit of the smoker-friends Markov logic network and Allison's (2001) Pompeii dataset mentioned already earlier. Allison has used data about artifacts in their original context to challenge many common assumptions about the function of particular types of objects, the use of particular spaces, and the consistency of patterns of use across different houses. The dataset contains more than 6000 artifact records for finds in 30 architecturally similar "atrium-style" houses in Pompeii. Artifacts are annotated with one of 240 typological categories (coin, amphora, etc.) and in which of the rooms they were found. In total, there are 863 rooms in the 30 houses. For each room, we have its type (main garden, front hall, shop, etc.) and function (culina, taberna, etc.). We have focused on the rooms and houses providing us with a dataset consisting of 959 node potentials and 6063 edge potentials (viewed as a graphical model as used by (L)GaBP). The smoker-friends graph consists of one single connected component of 20 persons of which 9 smoke and 5 have cancer. That is, we have a total of 34 nodes and 150 edges. There are 4 cliques of friends {a, b, c, d} with few inter-clique links. Tab. 1 shows the results.

The first query consisted of 4 out of a total of 8 houses with shops. As one can see the rankings/completions of $\mathbf{x}_p$, $\mathbf{x}_d$, and BSets$_{\mathcal{F}}$ do not cover the set of houses with shops and the query relevant information. BSets$_{\mathcal{L}}$ and MLS, however, are able to retrieve the houses with shops, the respective rooms and their type and function information. In other words, they have discovered the concept "houses with shops". Other houses are also ranked high, since one of the discriminative features is house(X). By just removing one house from the query as done in $\mathcal{Q}_2$, the concept "houses with shops" is not clearly identified by any method. MLS, though, provides query relevant information such as common discriminative types and functions of the query houses (isa(taberna), among others). Apart from completing the set of all houses, it also retrieves city(pompeii). The intermediate MLS rankings fail to return reasonable information as for example other houses. The third query consisted of 5 out of a total of 20 rooms in house(h$_1$). The rankings illustrate that MLS is able to capture all relevant information, by ranking house(h$_1$) the highest followed by all its rooms, the types and functions of the query atoms and city(pompeii). BSets as well as $\mathbf{x}_p$ and $\mathbf{x}_d$ provide only parts of the information, but no complete summary. The results for query $\mathcal{Q}_4$ are qualitatively similar. In particular, all elements of clique a are ranked highest only by MLS and BSets$_{\mathcal{L}}$.

All together, this clearly shows that **(Q1)** MLS finds reasonable set completions and that **(Q2)** its performance is better than its intermediate rankings and as good as or even better than BSets using relational features. The difference in performances of the two BSets variants demonstrate that discriminative relational pathfeatures indeed carry useful information. The compression ratios of the LP matrices – ratio of (# nodes + # edges) lifted vs. ground – were 0.55 for $\mathcal{Q}_1$ - $\mathcal{Q}_3$ and 0.73 for $\mathcal{Q}_4$. This shows that **(Q3)** lifted inference for label propagation is possible and sensible; GaBP and LGaBP produced the same results.

Conclusions

We have introduced *Markov logic sets* (MLS), the first, generally applicable, retrieval framework for relational data that can naturally be lifted[4]. It takes a query consisting of a small set of ground atoms, and returns additional ground atoms which belong in this set. Specifically, it computes a relevance score for each ground atom by comparing the personalized PageRank for it given the query, to the unpersonalized PageRank of that ground atom. Since similar items should intuitively get similar relevance scores, MLS uses relational pathfinding to find generalized personalization vectors. Based on the relevances, positive and negative labels are constructed that are turned into the final ranking using (lifted) label propagation. The experimental results demonstrate that lifted retrieval is not insurmountable: MLS performs effectively in retrieving relational set completions.

There are several attractive avenues for future work. We have not yet addressed in detail how to set the size of the completion. Since we use LP, using zero as threshold would be a natural choice but other thresholds are possible. It is also interesting to apply ILP techniques on the completion to extract a general description of the latent concept. Using dynamic PageRank techniques would pave the way to online clustering on demand and is definitely a growth path for lifted inference. Since lifting of LP itself is an advance, further experimentation and application of lifted inference for semi-supervised learning is another important direction.

Acknowledgments: The authors thank the anonymous reviewers for their comments. All authors were supported by the Fraunhofer ATTRACT fellowship STREAM and by the European Commission under contract number FP7-248258-First-MM.

References

Ahmadi, B.; Kersting, K.; and Sanner, S. 2011. Multi-Evidence Lifted Message Passing, with Application to PageRank and the Kalman Filter. In Walsh, T., ed., *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI–11).*

Allison, P. 2001. Pompeian Households: an Analysis of the Material Culture. Cotsen Institute of Archaeology.

Brin, S., and Page, L. 2006. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Computer Networks and ISDN Systems* 30:197–117.

Choi, J., and Amir, E. 2010. Lifted Inference for Relational Continuous Models. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence (UAI-10).*

De Raedt, L. 2008. *Logical and Relational Learning*. Springer.

Getoor, L., and Taskar, B., eds. 2007. *An Introduction to Statistical Relational Learning*. MIT Press.

Ghahramani, Z., and Heller, K. 2005. Bayesian Sets. In *Proc. Adv. in Neural Inform. Processing Systems (NIPS-05).*

Hotho, A.; Jäschke, R.; Schmitz, C.; and Stumme, G. 2006. Information Retrieval in Folksonomies: Search and Ranking. In *Proceedings of the 3rd European Semantic Web Conference (ESWC-06)*, 411–426.

Kersting, K.; Ahmadi, B.; and Natarajan, S. 2009. Counting Belief Propagation. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI–09).*

Kimmig, A.; De Raedt, L.; and Toivonen, H. 2007. Probabilistic Explanation Based Learning. In Kok, J. N.; Koronacki, J.; de Mántaras, R. L.; Matwin, S.; Mladenic, D.; and Skowron, A., eds., *ECML*, volume 4701 of *Lecture Notes in Computer Science*, 176–187. Springer.

Kok, S., and Domingos, P. 2009. Learning Markov Logic Network Structure via Hypergraph Lifting. In *Proc. of the International Conference on Machine Learning (ICML-09).*

Lao, N., and Cohen, W. W. 2010. Relational retrieval using a combination of path-constrained random walks. *Machine Learning* 81(1):53–67.

Mihalkova, L., and Mooney, R. 2007. Bottom-Up Learning of Markov Logic Network Structure. In *Proc. of the Intern. Conf. on Machine Learning (ICML-07).*

Milch, B.; Zettlemoyer, L.; Kersting, K.; Haimes, M.; and Pack Kaelbling, L. 2008. Lifted Probabilistic Inference with Counting Formulas. In *Proc. of the 23rd AAAI Conf. on Artificial Intelligence (AAAI-08).*

Muggleton, S. 1996. Learning from Positive Data. In *Proceedings of the Inductive Logic Programming Workshop (ILP-96)*, 358–376.

Neville, J., and Gallagher, B. 2009. Evaluating Statistical Tests for Within-Network Classifiers of Relational Data. In *Proc. of the Intern. Conf. on Data Mining (ICDM-09).*

Richardson, M., and Domingos, P. 2006. Markov Logic Networks. *Machine Learning* 62:107–136.

Shental, O.; Bickson, D.; Siegel, P. H.; Wolf, J. K.; and Dolev, D. 2008. Gaussian Belief Propagation Solver for Systems of Linear Equations. In *Proc. of the IEEE Int. Symp. on Inform. Theory (ISIT).*

Silva, R.; Heller, K.; Ghahramani, Z.; and Airoldi, E. 2010. Ranking Relations using Analogies in Biological and Information Networks. *Annals of Applied Statistics*.

Singla, P., and Domingos, P. 2008. Lifted First-Order Belief Propagation. In *Proc. of the 23rd AAAI Conf. on Artificial Intelligence (AAAI-08)*, 1094–1099.

Zhou, D.; Bousquet, O.; Lal, T. N.; Weston, J.; and Schölkopf, B. 2003. Learning with Local and Global Consistency. In Thrun, S.; Saul, L. K.; and Schölkopf, B., eds., *NIPS*. MIT Press.

[4]We attempt to develop a pragmatic and general framework that is liftable; at the moment, we make no claim that lifted techniques necessarily run faster than a specialized, one-off solution.