Graph Summarization in Annotated Data Using Probabilistic Soft Logic

Alex Memory[1], Angelika Kimmig[1,2], Stephen H. Bach[1], Louiqa Raschid[1] and Lise Getoor[1]

[1] University of Maryland
[2] KU Leuven

Abstract. Annotation graphs, made available through the Linked Data initiative and Semantic Web, have significant scientific value. However, their increasing complexity makes it difficult to fully exploit this value. Graph summaries, which group similar entities and relations for a more abstract view on the data, can help alleviate this problem, but new methods for graph summarization are needed that handle uncertainty present within and across these sources. Here, we propose the use of probabilistic soft logic (PSL) [1] as a general framework for reasoning about annotation graphs, similarities, and the possibly confounding evidence arising from these. We show preliminary results using two simple graph summarization heuristics in PSL for a plant biology domain.

1 Introduction

The Linked Data initiative and Semantic Web technologies have been very successful in providing access to a diversity of data collections. Of particular interest are *annotation graphs*, where scientific concepts are tagged with controlled vocabulary terms from ontologies or thesauri. As these collections grow, tools and techniques to analyze, explore and inspect such data become ever more important. In this paper we consider the problem of mining the richly curated annotation graphs, in conjunction with the wealth of semantic knowledge encoded within ontologies, to create graph summaries. Graph summaries group entities and relations based on similarity as well as local graph structure, thus creating a graph at a higher level of abstraction that can be easier to analyze. This can help the scientist to understand the underlying evidence, to find patterns, and to make predictions.

Linked Data can provide multiple rich and possibly confounding sources of evidence about concepts. As a motivating example, we consider an annotation graph from the domain of plant biology. The nodes in this graph are genes from the model organism Arabidopsis thaliana (these are the concepts) as well as terms from both the Gene Ontology (GO) and the Plant Ontology (PO) (these are the annotations). Edges represent annotations of genes with such terms. Other sources of information of interest include sequence-based similarity between pairs of genes, co-occurrence frequencies of pairs of GO terms, taxonomic

distances between pairs of PO or pairs of GO terms, etc. This evidence may be confounding; for example, genes can have high sequence based similarity to other genes in their same family. However, more useful evidence may be that they share high GO functional similarity with genes in unrelated families (with or without high sequence similarity).

We propose the use of probabilistic soft logic (PSL) [1] as a general framework for reasoning about annotation graphs, similarities, and the possibly confounding evidence arising from these. PSL is a framework for collective, probabilistic reasoning in relational domains that directly exploits available similarities. It uses rules to capture the dependency structure of the domain, based on which it builds a joint probabilistic model over the data. This allows us to easily encode the annotation graph, similarity information for nodes, and a number of graph summarization heuristics, and to explore the effect of these heuristics on the resulting graph summaries. In this work, we show preliminary results from two simple heuristics.

2 Motivating Example and Problem Setting

We use an example annotation graph from the plant biology domain to present our goals for graph summarization. We also use this domain for our experimental evaluation in Section 6. The graph represents gene annotation data for the model organism Arabidopsis thaliana, which originates in The Arabidopsis Information Resource (TAIR).[3] Each gene in TAIR is *annotated* with terms from the Plant Ontology (PO) and from the Gene Ontology (GO). A fragment of the resulting annotation graph is illustrated in Figure 1, with PO terms on the left, genes in the center, and GO terms on the right.

For a scientist exploring a set of genes of interest within a biological context, e.g., genes related to light-mediated development, finding regularities in such a graph can provide useful information. Our goal is to facilitate this process by providing *summaries* of the graph, that is, by grouping together nodes (and edges). The grouping can exploit multiple sources of evidence including explicit similarity between pairs of nodes, or shared annotations. For ease of illustration, we drastically simplify the graph to the topmost part of Figure 1, shown on the left in Figure 2. On the right, Figure 2 shows a possible graph summary, where three pairs of nodes have been grouped into three supernodes or clusters, and sets of edges between all pairs of nodes in adjacent clusters are represented by single edges between clusters. However, for real-world graphs, many clusterings are possible, and so different heuristics and combinations of heuristics may be appropriate for different graphs. In this work, we show how two such heuristics can be easily incorporated into a probabilistic framework, but others are certainly possible. Future work can extend this approach by incorporating additional heuristics and adapting heuristics to different graph-summarization tasks.

[3] `http://www.arabidopsis.org`

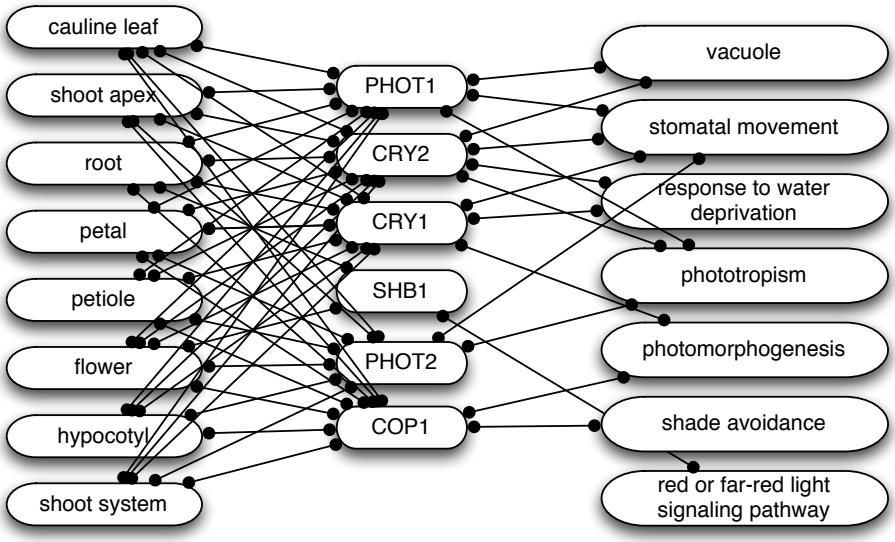

Fig. 1. Part of the annotation graph: PO terms (left), genes (middle), and GO terms (right).

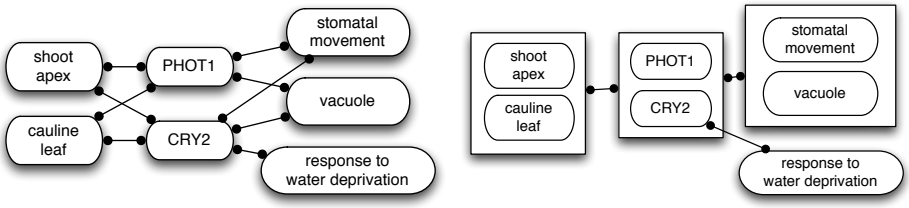

Fig. 2. Example subgraph (left) and a possible summary for it (right).

The first is an *annotation link heuristic*: We would like to cluster nodes that share a large fraction of their neighbors in the annotation graph. For instance, the two PO terms "cauline leaf" and "shoot apex" both annotate genes PHOT1 and CRY2 in our example, and there are no genes that are annotated with only one of these terms. The terms are thus similar in terms of the link structure they participate in, which supports clustering them. On the GO side, the same argument holds for "vacuole" and "stomatal movement", but not for "response to water deprivation", which only annotates CRY2. Clearly, the direct link structure alone does not provide sufficient evidence to decide whether the latter term should be added to the GO cluster or not. Choosing to include the term in the cluster would correspond to implicitly assuming that the term should actually annotate PHOT1 as well, and thus allow one to *predict* a new link, whereas the latter would tend more towards accepting the absence of such a link. Finally, we

observe that the two gene nodes share four out of their five neighbors, which can still be viewed as a relatively strong indication to cluster them.

Next, we consider explicit similarities between pairs of nodes. Such additional information could help deciding whether the third GO term should be included in the cluster. For this we use the sequence based similarity between pairs of genes and information retrieval based metrics between pairs of annotation terms. For instance, amongst the extensive statistics published by the GO Consortium, the *annotation co-occurrence* of pairs of GO terms has significant biological meaning and is a good predictor of new function. For the GO term `response to water deprivation`, `stomatal movement` is the 5th highest co-occurring term; for the reverse case the rank is 11. Incorporating a similarity measure between GO terms into the graph summarization process might thus provide additional evidence in favor of clustering all three terms in the example and further predicting the "response to water deprivation" annotation on PHOT1. This would help the biologist understand the new functional annotation of PHOT1 and also understand that there is a posssible interaction between CRY2 and PHOT1.

To be able to exploit this similarity information, we introduce a *similarity heuristic*: we prefer to cluster nodes that are similar according to some available similarity measure. Recall however that this may also introduce conflicting evidence. For instance, the two genes in our example belong to different groups of blue light receptors and are therefore dissimilar in terms of sequence similarity, but similar in terms of their annotations in the graph.

We integrate the multiple types of evidence from the annotation links, the various similarity metrics, and the two graph summarization heuristics within a probabilistic model using PSL. We discuss this model in more detail in Section 5, after an introduction to PSL in Section 4.

3 Related Work

Graph summarization as broadly considered in this paper is a form of multi-relational clustering that exploits attributes of the nodes or objects to be clustered, as well as additional relational features or properties in which these nodes participate [2–4]. Multi-relational clustering aims at grouping nodes in heterogeneous, multi-relational networks, i.e., networks with both multiple types of nodes and multiple types of relationships between nodes. The clusters group nodes based on their similarities, where the value is either given explicitly, or derived from node attributes or relations between nodes of the same or different type(s). There is a large body of work on multi-relational clustering, and methods include matrix factorization approaches, generative models, and other optimization methods. Other work on graph summarization explores summarization techniques that can be tailored to user needs and which scale to large graphs with minimal loss [5–8]. Our proposed approach makes use of the notion of *examplars*, used in methods such as affinity propagation [9], to denote the elements which are chosen as the canonical representation for nodes in each cluster. Besides multi-relational clustering and graph summarization, there is a

broad range of other mining and analysis techniques for heterogeneous information networks, cf. for instance [10].

Probabilistic soft logic (PSL) [1] combines ideas from fuzzy logic [11] and graphical models. Similar to Markov Logic [12], it uses first order logic as a template language for a graphical model. However, its use of soft truth values turns inference from a discrete into a continuous optimization task, which can be solved efficiently.

4 Probabilistic Soft Logic

Probabilistic soft logic (PSL) [1] is a framework for collective, probabilistic reasoning in relational domains. PSL uses rules to capture the dependency structure of the domain, based on which it builds a joint probabilistic model over all atoms. Each rule has an associated non-negative weight that captures the rule's relative importance. Furthermore, PSL uses soft truth values in the interval $[0, 1]$, which allows one to directly incorporate similarity functions into the logical model. We refer to Broecheler et al. [1] for full technical details and instead illustrate the key concepts in the context of the following example program:

$$w_1 : exemplar(A, B) \rightarrow similar(A, B) \quad (1)$$

$$w_2 : link(A, B) \wedge exemplar(A, C) \wedge exemplar(D, C) \rightarrow link(D, B) \quad (2)$$

Here, for simplicity of presentation, we assume $w_1 = w_2 = 1$. Consider any concrete nodes a, b, c, and d instantiating logical variables A, B, C, and D respectively. The first rule states that if a is in the cluster exemplified by b, they should be similar (similarity heuristic), whereas the second states that if a and d are both in the cluster exemplified by c, and a has a link to b, then d should also have a link to b (link heuristic). While PSL shares the syntax of its rules with first order logic, PSL uses *soft truth values* from the interval $[0, 1]$ instead of its extremes 0 (false) and 1 (true) only. Given a set of atoms $\ell = \{\ell_1, \ldots, \ell_n\}$, we call the mapping $I : \ell \rightarrow [0, 1]^n$ from atoms to soft truth values an *interpretation*. PSL defines a probability distribution over interpretations that makes those satisfying more ground rule instances more probable.

To determine the degree to which a ground rule is satisfied, PSL uses the *Lukasiewicz t-norm* and its corresponding *co-norm* as the relaxation of the logical AND and OR, respectively. These relaxations are exact at the extremes, but provide a consistent mapping for values in-between. Given an interpretation I, the formulas for the relaxation of the logical conjunction ($\wedge$), disjunction ($\vee$), and negation ($\neg$) are as follows:

$$\begin{aligned} \ell_1 \tilde{\wedge} \ell_2 &= \max\{0, I(\ell_1) + I(\ell_2) - 1\}, \\ \ell_1 \tilde{\vee} \ell_2 &= \min\{I(\ell_1) + I(\ell_2), 1\}, \\ \tilde{\neg} l_1 &= 1 - I(\ell_1), \end{aligned}$$

where we use $\tilde{}$ to indicate the relaxation from the Boolean domain. For a ground rule $r \equiv r_{body} \rightarrow r_{head} \equiv \tilde{\neg} r_{body} \tilde{\vee} r_{head}$, where r_{body} and r_{head} are logical formulas composed of atoms and the logical operators defined above, an interpretation

I over the atoms in r determines whether r is satisfied, and, if not, its distance to satisfaction. Abusing notation, we can expand the usage of I to also denote the truth assignments to logical formulas induced by assignments to atoms and applying the definitions of the logical operators in the formula, i.e., $I(r)$ is the truth value that results from applying the logical operators in r to the truth values of atoms in r given by I. Then, given I, r is satisfied, i.e., $I(r) = 1$, if and only if $I(r_{body}) \leq I(r_{head})$, that is, the head has at least the same truth value as the body. Again, this coincides with the usual definition of satisfaction of a rule when truth values are restricted to 0 and 1. The rule's *distance to satisfaction* under interpretation I then measures the degree to which this condition is violated:

$$d_r(I) = \max\{0, I(r_{body}) - I(r_{head})\} \tag{3}$$

For instance, consider the interpretation $I = \{link(a, b) \mapsto 1, exemplar(a, c) \mapsto 0.9, exemplar(d, c) \mapsto 0.8, link(d, b) \mapsto 0\}$ and let r be the corresponding ground instance of Rule (2) above. We get $I(r_{body}) = \max\{0, 1 + 0.8 + 0.9 - 2\} = 0.7$ and thus $d_r(I) = \max\{0, 0.7 - 0\} = 0.7$, whereas the distance would be 0 if the head had truth value 0.7 or greater.

Given a set of atoms ℓ of interest, a PSL program induces a distribution over possible interpretations I. ℓ first induces a set of ground rules R, which contains every possible ground rule r such that r can be obtained by performing variable substitution on one of the rules in the program and each atom mentioned in r is in ℓ. The probability density function f over I is:

$$f(I) = \frac{1}{Z}\exp[-\sum_{r \in R} \lambda_r (d_r(I))^p] \quad ; \quad Z = \int_I \exp[-\sum_{r \in R} \lambda_r (d_r(I))^p] \tag{4}$$

where λ_r is the weight of the rule r, Z is the continuous version of the normalization constant used in discrete Markov random fields, and $p \in \{1, 2\}$ determines the loss function for minimizing the distance from satisfaction. If $p = 2$ the loss function is quadratic and the distance from satisfaction for each ground rule is squared. Constraints can be imposed on interpretations and the domain updated accordingly, for instance, requiring a predicate to be functional. Also, the density function can be conditioned on a partial interpretation and the domain and definitions of distances to satisfaction updated accordingly.

Finding the most probable interpretation in PSL is an instance of MPE inference. Maximizing the density function $f(I)$ is equivalent to minimizing the summation in the exponent. This optimization problem, if subject only to linear equality and inequality constraints on the interpretation, can be solved efficiently by casting it as a second-order cone program [1].

5 A PSL Model for Graph Summarization

Figure 3 lists the set of PSL rules used in this work for graph summarization in annotation data. Different subsets of these rules are experimentally evaluated and compared in Section 6. We model similarity of pairs of nodes of the same type

$$exemplar(A, B) \rightarrow similar(A, B) \quad (5)$$
$$exemplar(A, B) \rightarrow exemplar(B, B) \quad (6)$$
$$link(A, B) \wedge link(C, B) \wedge exemplar(A, D) \rightarrow exemplar(C, D) \quad (7)$$
$$link(A, B) \wedge exemplar(A, C) \wedge exemplar(D, C) \rightarrow link(D, B) \quad (8)$$

Fig. 3. PSL rules for graph summarization as discussed in Section 5 and experimentally evaluated in Section 6. Labels refer to the introduction of rules in the text.

with predicate $similar/2$ and relations between pairs of nodes of different types with predicate $link/2$. Both predicates are symmetric. Note that while these predicates allow us to easily write general rules for all types of links and nodes appearing in the data, the inference engine takes into account the node types during grounding and thus ensures that clustering respects the types. Given truth values for all relevant atoms of these two predicates, the task of inference is to infer truth values of the remaining predicate $exemplar/2$, which encodes clusters. More specifically, the truth value of an atom $exemplar(a, b)$ indicates whether node a is a member of the cluster that has node b as its exemplar. We constrain $exemplar/2$ to be a functional predicate, that is, the truth values of all its groundings using a given node a as first argument have to sum to one. We also set a small prior on $exemplar/2$, further limiting its groundings.

In the following, we discuss the individual rules in more detail, showing how they encode the clustering heuristics introduced in Section 2 as probabilistic dependencies.

5.1 Similarity Heuristic

We start with the similarity heuristic, which indicates that pairs of similar nodes of the same type should probably be clustered. It is modeled by the first PSL rule:

$$exemplar(A, B) \rightarrow similar(A, B) \quad (5)$$

This rule connects truth values of $similar/2$, which are given, to those of $exemplar/2$, which are inferred. For a pair of nodes (a, b) with low similarity, the rule is only satisfied for low truth values of $exemplar(a, b)$. In other words, it encourages node a to choose a different, more similar exemplar. If a and b are highly similar, on the other hand, a wider range of truth values for $exemplar(a, b)$ will satisfy the rule, making it possible for a to choose b or another node as its exemplar without penalty.

We further encourage clusters with a single exemplar, which is modeled by the second PSL rule:

$$exemplar(A, B) \rightarrow exemplar(B, B) \quad (6)$$

This rule breaks chains of exemplar choices by penalizing situations where a node that is chosen as exemplar by some node in the cluster does not choose itself as exemplar. As truth values of $exemplar/2$ atoms are inferred during clustering, this rule can propagate information in both directions. If the truth value of $exemplar(b, b)$ is low for a given node b, it will encourage low truth values for all atoms $exemplar(a, b)$ with other nodes a as first argument. Conversely, each atom $exemplar(a, b)$ with high truth value encourages a high truth value for $exemplar(b, b)$.

5.2 Annotation Link Heuristic

The following two PSL rules model the annotation link heuristic:

$$link(A, B) \wedge link(C, B) \wedge exemplar(A, D) \rightarrow exemplar(C, D) \quad (7)$$

$$link(A, B) \wedge exemplar(A, C) \wedge exemplar(D, C) \rightarrow link(D, B) \quad (8)$$

Rule (7) states that a shared neighbor is an indication that two nodes should be clustered. Consider a pair of candidate nodes a and c for clustering, and keep the exemplar d and the node b on the other side fixed. Due to symmetry, we get two groundings of the rule, one replacing A with a and C with c, the other replacing A with c and C with a:

$$link(a, b) \wedge link(c, b) \wedge exemplar(a, d) \rightarrow exemplar(c, d) \quad (9)$$

$$link(c, b) \wedge link(a, b) \wedge exemplar(c, d) \rightarrow exemplar(a, d) \quad (10)$$

During clustering, the truth values of $link/2$ atoms are fixed to either 1 (link exists) or 0 (link does not exist). If one of the two links does not exist, both groundings are trivially satisfied, as their bodies will have the minimal truth value 0. If they both exist, the rules simplify to

$$exemplar(a, d) \rightarrow exemplar(c, d) \quad (11)$$

$$exemplar(c, d) \rightarrow exemplar(a, d) \quad (12)$$

and thus encourage the truth value of $exemplar(a, d)$ to be at most and at least that of $exemplar(c, d)$, respectively. In other words, the two nodes should agree in the degree to which they choose that specific exemplar and its corresponding cluster. Note that the influence of this rule grows with the number of joint neighbors the two candidates for clustering share, as those will produce individual groundings.

While Rule (8) again involves a pair of nodes that are candidates for clustering (a neighboring node and an exemplar), due to its different form, it encodes a different dependency. Consider the grounding

$$link(a, b) \wedge exemplar(a, c) \wedge exemplar(d, c) \rightarrow link(d, b) \quad (13)$$

As truth values of $link/2$ are fixed to either 0 or 1, this grounding is trivially satisfied if there is no link between a and b (in which case the truth value of the

body is minimal) or if there is a link between d and b (in which case the truth value of the head is maximal). The interesting case is thus the one where there is a link between a and b, but no link between d and b.[4] In this case, the rule increases the probability that

$$exemplar(a, c) \tilde{\wedge} exemplar(d, c) \leq 0 \tag{14}$$

In words, the rule will be satisfied in this case if and only if the truth values of the two $exemplar/2$ atoms sum to at most 1, thus encouraging the two nodes not to strongly agree on a joint exemplar. The influence of this rule grows with the number of neighbors on which a and d disagree. Together, the two rules thus allow one to take into account both shared and unshared neighbors during clustering.

6 Evaluation

The goals of graph summarization include identifying patterns and making predictions in the annotation graph. We use prediction, specifically the task of predicting missing links, to explore the utility of the simple heuristics from Section 5. In our experimental setting, the missing links are links between genes and GO terms in the annotation graph for the model organism Arabidopsis thaliana, as described in Section 2. We begin by generating graph summaries using the rules described ealier. Next, we use this model to predict gene-GO annotations.

In addition to the GO, gene and PO annotation graph, we consider gene-gene sequence-based similarity,[5] as well as PO-PO path based distances from the PO ontology and GO-GO path based distances from the GO ontology. These are represented as $similar/2$ atoms between nodes of each of the three types within the graph: PO terms, genes and GO terms. In our model, all instances of $similar/2$ atoms are treated uniformly; however, they are computed using different similarity metrics. All other relations from the data are $link/2$ atoms between nodes of different types. Although the type of each node in the graph could be represented explicitly and used in the rules to control the graph summaries, e.g., to ensure that no cluster contains both PO and GO terms, in our implementation we only consider relations between nodes where a relation of that type might exist between nodes of those types. Further, in the grounding of the atoms in the data, we make each $similar/2$ atom symmetric by asserting its inverse, and we do the same for $link/2$ atoms.

Using each *graph summarization* program, we infer the $exemplar/2$ atoms forming clusters with soft membership, which is the input to our *link prediction* program. To then evaluate our link prediction using graph summaries, we perform leave-one-out evaluation. Specifically, for each link in the original annotation graph we first remove the link and compute the graph summary. We then

[4] Note that we get a symmetric grounding that affects the opposite case as well.

[5] We compute pair-wise sequence similarity between pairs of genes using the Nucleotide-Nucleotide BLAST 2.2.26+ package.

predict missing links. We sort the predicted links based on their truth values, and we interpret the truth values as the *confidence* in the prediction. We calculate the link prediction precision with recall of one, and we report on the mean average precision computed over all leave-one-out links.

Many combinations of heuristics could be explored for the summarization and prediction programs, so we chose the following four configurations to evaluate. The first configuration (**LINK1**) uses Rule (7) for graph summarization and link prediction. The second configuration (**LINK2**) is the same as **LINK1** but it uses Rule (8) in place of Rule (7). The third configuration (**SIM1**) uses only the similarity rules for graph summarization and adds Rule (7) for link prediction. The fourth configuration (**SIM2**) is the same as **SIM1** but uses Rule (8) in place of Rule (7). Each configuration also uses Rule (5) and Rule (6). All rules have weight one except Rule (6) which has a high weight, 1000, to encourage distinct clusters by breaking chains of $exemplar/2$ atoms. Learning individual weights for these rules may be beneficial, but we do not consider it in this work.

Finally, we choose the loss function for minimizing the distance from satisfaction, which is set by p in (4). In each configuration, we use the linear loss function for the graph summarization program and the quadratic loss function for the link prediction program. We use the quadratic loss function for the link prediction programs for two reasons. First, inference with quadratic loss is more expensive than with linear loss, so for the interest of time we only use it on link prediction, which is less expensive than graph summarization. Our link prediction programs are less expensive, in part, because inferring $link/2$ atoms involves a smaller number of rules than inferring $exemplar/2$ atoms. Second, quadratic loss tends to assign $link/2$ atom truth values between rather than at the extremes, 0 or 1, more often than linear loss, and this is helpful when ranking predicted links to calculate precision.

6.1 Results

Table 1 describes the TAIR annotation graph data sets we used for evaluation. The first two data sets (**DS1** and **DS2**) have fewer genes but more terms and annotations over all than the last data set (**DS3**). Table 2 reports mean average

Table 1. The evaluation data sets. The number of genes, Plant Ontology terms, Gene Ontology terms, PO-to-gene annotation links and GO-to-gene annotation links.

	DS1	**DS2**	**DS3**
Genes	10	10	18
PO Terms	53	48	40
GO Terms	44	31	19
PO-Gene	255	255	218
GO-Gene	157	157	92

precision (MAP) on the evaluation data sets for each PSL model configuration.

Considering the **LINK1** and **LINK2** configurations across all data sets, we see that Rule (8), used in the latter, consistently has higher precision than Rule (7), used in the former. Rule (8) is also used in **SIM2** where it similarly has higher precision than the other link rule in **SIM1** across all data sets.

Now considering precision across data sets, on **DS3** both **LINK2** and **SIM2** perform well compared to previous link prediction results on similar annotation data [5]. However, none of the configurations perform well on **DS1** and **DS2**. To interpret this result we convert the clusters to hard membership, calculate the average number of clusters of size greater than one produced by each configuration and normalize by the number of nodes in the data set. This is shown in the right side of Table 2.

Using this information, we see a small number of clusters formed in **DS1** and **DS2** and a larger number formed in **DS3**[6] except where Rule (7) is used. This suggests that Rule (8) is helpful for link prediction on this data and may be helpful for clustering; on the other hand, Rule (7) is not helpful for clustering or link prediction on this data, and may interfere with the use of similarity attributes in clustering. Finally, this also suggests that neither annotation link heuristic rule works well for prediction on graphs where we find few clusters of size greater than one. Since, for example in Rule (8) the truth value of inferred $link/2$ atoms is bounded only when there are multiple nodes A and D in the same cluster, this result is not surprising.

Table 2. Mean average precision of link prediction for evaluated PSL model configurations on each data set. Also, the average number of clusters of size greater than one, divided by the number of nodes in the data set.

Configuration	MAP			Clusters		
	DS1	**DS2**	**DS3**	**DS1**	**DS2**	**DS3**
LINK1	0.01	0.00	0.01	0.01	0.01	0.01
LINK2	0.06	0.04	0.30	0.03	0.03	0.10
SIM1	0.01	0.00	0.01	0.01	0.01	0.12
SIM2	0.07	0.12	0.28	0.01	0.01	0.12

7 Conclusions and Future Work

In this work, we demonstrated an exploratory use of graph summarization heuristics in probabilistic soft logic (PSL) on annotation graph data, combining relational and similarity evidence from multiple, heterogeneous sources. The power of the approach is the ease in which a variety of clustering criteria can be declaratively expressed. Our work, which is ongoing, will continue to explore the space of graph summarization rules for combining data from rich sources, such as the

[6] A similar pattern of cluster sizes emerges when a version of these graphs is clustered using a separate method similar to [5].

gene sequences, annotations and term ontologies used in this work and other sources now made available through the Linked Data initiative and Semantic Web.

Acknowledgements

Angelika Kimmig is a postdoctoral fellow of the Research Foundation Flanders (FWO Vlaanderen). This work is partially supported by NSF under grants IIS0746930 and CCF0937094.

References

1. Broecheler, M., Mihalkova, L., Getoor, L.: Probabilistic similarity logic. In: Conference on Uncertainty in Artificial Intelligence (UAI). (2010)
2. Taskar, B., Segal, E., Koller, D.: Probabilistic classification and clustering in relational data. In: International Joint Conference on Artificial Intelligence (IJCAI). (2001)
3. Kemp, C., Tenenbaum, J.B., Griffiths, T.L., Yamada, T., Ueda, N.: Learning systems of concepts with an infinite relational model. In: National Conference on Artificial Intelligence (AAAI). (2006)
4. Long, B., Zhang, Z.M., Yu, P.S.: A probabilistic framework for relational clustering. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD). (2007)
5. Thor, A., Anderson, P., Raschid, L., Navlakha, S., Saha, B., Khuller, S., Zhang, X.N.: Link prediction for annotation graphs using graph summarization. In: International Semantic Web Conference (ISWC). (2011)
6. Zhang, N., Tian, Y., Patel, J.M.: Discovery-driven graph summarization. In: International Conference on Data Engineering (ICDE). (2010)
7. Tian, Y., Hankins, R.A., Patel, J.M.: Efficient aggregation for graph summarization. In: ACM SIGMOD International Conference on Management of Data (SIGMOD). (2008)
8. Navlakha, S., Rastogi, R., Shrivastava, N.: Graph summarization with bounded error. In: ACM SIGMOD International Conference on Management of Data (SIGMOD). (2008)
9. Frey, B.J., Dueck, D.: Clustering by passing messages between data points. Science **315** (2007) 972–976
10. Berthold, M., ed.: Bisociative Knowledge Discovery. Volume 7250 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2012)
11. Lee, R.C.T.: Fuzzy logic and the resolution principle. J. ACM **19**(1) (1972) 109–119
12. Richardson, M., Domingos, P.: Markov logic networks. Machine Learning **62**(1-2) (2006) 107–136