Practical Markov Logic Containing First-Order Quantifiers with Application to Identity Uncertainty

Aron Culotta and **Andrew McCallum**
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
{culotta, mccallum}@cs.umass.edu

Abstract

Markov logic is a highly expressive language recently introduced to specify the connectivity of a Markov network using first-order logic. While Markov logic is capable of constructing arbitrary first-order formulae over the data, the complexity of these formulae is often limited in practice because of the size and connectivity of the resulting network. In this paper, we present approximate inference and estimation methods that incrementally instantiate portions of the network as needed to enable first-order existential and universal quantifiers in *Markov logic networks*. When applied to the problem of identity uncertainty, this approach results in a conditional probabilistic model that can reason about objects, combining the expressivity of recently introduced BLOG models with the predictive power of conditional training. We validate our algorithms on the tasks of citation matching and author disambiguation.

1 Introduction

Markov logic networks (MLNs) combine the probabilistic semantics of graphical models with the expressivity of first-order logic to model relational dependencies (Richardson and Domingos, 2004). They provide a method to instantiate Markov networks from a set of constants and first-order formulae.

While MLNs have the power to specify Markov networks with complex, finely-tuned dependencies, the difficulty of instantiating these networks grows with the complexity of the formulae. In particular, expressions with first-order quantifiers can lead to networks that are large and densely connected, making exact probabilistic inference intractable. Because of this, existing applications of MLNs have not exploited the full richness of expressions available in first-order logic.

For example, consider the database of researchers described in Richardson and Domingos (2004), where predicates include Professor(person), Student(person), AdvisedBy(person, person), and Published(author, paper). First-order formulae include statements such as "students are not professors" and "each student has at most one advisor." Consider instead statements such as "all the students of an advisor publish papers with similar words in the title" or "this subset of students belong to the same lab." To instantiate an MLN with such predicates requires existential and universal quantifiers, resulting in either a densely connected network, or a network with prohibitively many nodes. (In the latter example, it may be necessary to ground the predicate for each element of the power set of students.)

However, as discussed in Section 2, there may be cases where these *aggregate predicates* increase predictive power. For example, in predicting the value of HaveSameAdvisor($a_i \dots a_{i+k}$), it may be useful to know the values of aggregate evidence predicates such as CoauthoredAtLeastTwoPapers($a_i \dots a_{i+k}$), which indicates whether there are at least two papers that some combination of authors from $a_i \dots a_{i+k}$ have co-authored. Additionally, we can construct predicates such as NumberOfStudents(a_i) to model the number of students a researcher is likely to advise simultaneously.

These aggregate predicates are examples of universal and existentially quantified predicates over observed and unobserved values. To enable these sorts

of predicates while limiting the complexity of the ground Markov network, we present an algorithm that incrementally expands the set of aggregate predicates during the inference procedure. In this paper, we describe a general algorithm for incremental expansion of predicates in MLNs, then present an implementation of the algorithm applied to the problem of identity uncertainty.

2 Related Work

MLNs were designed to subsume various previously proposed statistical relational models. *Probabilistic relational models* (Friedman et al., 1999) combine descriptive logic with directed graphical models, but are restricted to acyclic graphs. *Relational Markov networks* (Taskar et al., 2002) use SQL queries to specify the structure of undirected graphical models. Since first-order logic subsumes SQL, MLNs can be viewed as more expressive than relational Markov networks, although existing applications of MLNs have not fully utilized this increased expressivity. Other approaches combining logic programming and log-linear models include *stochastic logic programs* (Cussens, 2003) and MACCENT(Dehaspe, 1997), although MLNs can be shown to represent both of these.

Viewed as a method to avoid grounding an intractable number of predicates, this paper has similar motivations to recent work in *lifted inference* (Poole, 2003; de Salvo Braz et al., 2005), which performs inference directly at the first-order level to avoid instantiating all predicates. Although our model is not an instance of lifted inference, it does attempt to reduce the number of predicates by instantiating them incrementally.

Identity uncertainty (also known as record linkage, deduplication, object identification, and co-reference resolution) is the problem of determining whether a set of constants (*mentions*) refer to the same object (*entity*). Successful identity resolution enables vision systems to track objects, database systems to deduplicate redundant records, and text processing systems to resolve disparate mentions of people, organizations, and locations.

Many probabilistic models of object identification have been proposed in the past 40 years in databases (Fellegi and Sunter, 1969; Winkler, 1993) and natural language processing (McCarthy and Lehnert, 1995; Soon et al., 2001). With the introduction of statistical relational learning, more sophisticated models of identity uncertainty have been developed that consider the dependencies between related consolidation decisions.

Most relevant to this work are the recent relational models of identity uncertainty (Milch et al., 2005; McCallum and Wellner, 2003; Parag and Domingos, 2004). McCallum and Wellner (2003) present experiments using a conditional random field that factorizes into a product of pairwise decisions about mention pairs (Model 3). These pairwise decisions are made collectively using relational inference; however, as pointed out in Milch et al. (2004), there are shortcomings to this model that stem from the fact that it does not capture features of *objects*, only of mention pairs. For example, aggregate features such as "a researcher is unlikely to publish in more than 2 different fields" or "a person is unlikely to be referred to by three different names" cannot be captured by solely examining pairs of mentions. Additionally, decomposing an object into a set of mention pairs results in "double-counting" of attributes, which can skew reasoning about a single object (Milch et al., 2004). Similar problems apply to the model in Parag and Domingos (2004).

Milch et al. (2005) address these issues by constructing a generative probabilistic model over possible worlds called BLOG, where realizations of objects are typically sampled from a generative process. While BLOG model provides attractive semantics for reasoning about unknown objects, the transition to generatively trained models sacrifices some of the attractive properties of the discriminative model in McCallum and Wellner (2003) and Parag and Domingos (2004), such as the ability to easily incorporate many overlapping features of the observed mentions. In contrast, generative models are constrained either to assume the independence of these features or to explicitly model their interactions.

Object identification can also be seen as an instance of *supervised clustering*. Daumé III and Marcu (2004) and Carbonetto et al. (2005) present similar Bayesian supervised clustering algorithms that use a Dirichlet process to model the number of clusters. As a generative model, it has similar advantages and disadvantages as Milch et al. (2005), with the added capability of integrating out the uncertainty in the true number of objects.

In this paper, we present of identity uncertainty that incorporates the attractive properties of McCallum and Wellner (2003) and Milch et al. (2005), resulting in a discriminative model to reason about objects.

3 Markov logic networks

Let $F = \{F_i\}$ be a set of first order formulae with corresponding real-valued weights $w = \{w_i\}$. Given a set of constants $C = \{c_i\}$, define $n_i(x)$ to be the number of true *groundings* of F_i realized in a setting

of the world given by atomic formulae x. A Markov logic network (MLN) (Richardson and Domingos, 2004) defines a joint probability distribution over possible worlds x. In this paper, we will work with discriminative MLNs (Singla and Domingos, 2005), which define the conditional distribution over a set of query atoms y given a set of evidence atoms x. Using the normalizing constant Z_x, the conditional distribution is given by

$$P(Y = y|X = x) = \frac{1}{Z_x} \exp\left(\sum_{i=1}^{|F_y|} w_i n_i(x, y)\right) \quad (1)$$

where $F_y \subseteq F$ is the set of clauses for which at least one grounding contains a query atom, and $n_i(x, y)$ is the number of true groundings of the ith clause containing evidence atom x and query atom y.

3.1 Inference Complexity in Ground Markov Networks

The set of predicates and constants in Markov logic define the structure of a Markov network, called a *ground Markov network*. In discriminative Markov logic networks, this resulting network is a conditional Markov network (also known as a *conditional random field* (Lafferty et al., 2001)).

From Equation 1, the formulae F_y specify the structure of the corresponding Markov network as follows: Each grounding of a predicate specified in F_y has a corresponding node in the Markov network; and an edge connects two nodes in the network if and only if their corresponding predicates co-occur in a grounding of a formula F_y. Thus, the complexity of the formulae in F_y will determine the complexity of the resulting Markov network, and therefore the complexity of inference. When F_y contains complex first-order quantifiers, the resulting Markov network may contain a prohibitively large number of nodes.

For example, let the set of constants C be the set of authors $\{a_i\}$, papers $\{p_i\}$, and conferences $\{c_i\}$ from a research publication database. Predicates may include AUTHOROF(a_i, p_j), ADVISOROF(a_i, a_j), and PROGRAMCOMMITTEE(a_i, c_j). Each grounding of a predicate corresponds to a random variable in the corresponding Markov network.

It is important to notice how *query* predicates and *evidence* predicates differ in their impact on inference complexity. Grounded evidence predicates result in observed random variables that can be highly connected without resulting in an increase in inference complexity. For example, consider the binary evidence predicate HAVESAMELASTNAME$(a_i \ldots a_{i+k})$. This *aggregate* predicate reflects information about a subset of $(k - i + 1)$ constants. The value of this predicate is dependent on the values of HAVESAMELASTNAME(a_i, a_{i+1}), HAVESAMELASTNAME(a_i, a_{i+2}), etc. However, since all of the corresponding variables are observed, inference does not need to ensure their consistency or model their interaction.

In contrast, complex *query* predicates can make inference more difficult. Consider the query predicate HAVESAMEADVISOR$(a_i \ldots a_{i+k})$. Here, the related predicates HAVESAMEADVISOR(a_i, a_{i+1}), HAVESAMEADVISOR(a_i, a_{i+2}), etc., all correspond to *unobserved* binary random variables that the model must predict. To ensure their consistency, the resulting Markov network must contain dependency edges between each of these variables, resulting in a densely connected network. Since inference in general in Markov networks scales exponentially with the size of the largest clique, inference in the grounded network quickly becomes intractable.

One solution is to limit the expressivity of the predicates. In the previous example, we can decompose the predicate HAVESAMEADVISOR$(a_i \ldots a_{i+k})$ into its $(k - i + 1)^2$ corresponding *pairwise* predicates, such as HAVESAMEADVISOR(a_i, a_{i+1}). Answering an aggregate query about the advisors of a group of students can be handled by a conjunction of these pairwise predicates.

However, as discussed in Sections 1 and 2, we would like to reason about *objects*, not just pairs of *mentions*, because this enables richer evidence predicates. For example, the evidence predicates ATLEASTTWOCOAUTHOREDPAPERS$(a_i \ldots a_{i+k})$ and NUMBEROFSTUDENTS(a_i) can be highly predictive of the query predicate HAVESAMEADVISOR$(a_i \ldots a_{i+k})$.

Below, we describe a discriminative MLN for identity uncertainty that is able to reason at the object level.

3.2 Identity uncertainty

Typically, MLNs make a *unique names* assumption, requiring that different constants refer to distinct objects. In the publications database example, each author constant a_i is a string representation of one author mention found in the text of a citation. The unique names assumption assumes that each a_i refers to a distinct author in the real-world. This simplifies the network structure at the risk of weak or fallacious predictions (e.g., ADVISOROF(a_i, a_j) is erroneous if a_i and a_j actually refer to the same author). The *identity uncertainty* problem is the task of removing the unique names assumption by determining which

constants refer to the same real-world objects.

Richardson and Domingos (2004) address this concern by creating the predicate $\textsc{Equals}(c_i, c_j)$ between each pair of constants. While this retains the coherence of the model, the restriction to pairwise predicates can be a drawback if there exist informative features over sets of constants. In particular, by only capturing features of pairs of constants, this solution cannot model the compatibility of object attributes, only of constant attributes (Section 2).

Instead, we desire a conditional model that allows predicates to be defined over a set of constants.

One approach is to introduce constants that represent objects, and connect them to their mentions by predicates such as $\textsc{IsMentionOf}(c_i, c_j)$. In addition to computational issues, this approach also somewhat problematically requires choosing the number of objects. (See Richardson and Domingos (2004) for a brief discussion.)

Instead, we propose instantiating *aggregate predicates* over sets of constants, such that a setting of these predicates implicitly determines the number of objects. This approach allows us to model attributes over entire objects, rather than only pairs of constants. In the following sections, we describe aggregate predicates in more detail, as well as the approximations necessary to implement them efficiently.

3.3 Aggregate predicates

Aggregate predicates are predicates that take as arguments an arbitrary number of constants. For example, the $\textsc{HaveSameAdvisor}(a_i \dots a_{i+k})$ predicate in the previous section is an example of an aggregate predicate over $k - i + 1$ constants.

Let $I_C = \{1 \dots N\}$ be the set of indices into the set of constants C, with power set $\mathcal{P}(I_C)$. For any subset $\mathbf{d} \in \mathcal{P}(I_C)$, an aggregate predicate $\mathrm{A}(\mathbf{d})$ defines a property over the subset of constants $\mathbf{d}$.

Note that aggregate predicates can be translated into first-order formulae. For example, $\textsc{HaveSameAdvisor}(a_i \dots a_{i+k})$ can be re-written as $\forall(a_x, a_y) \in \{a_i \dots a_{i+k}\}$ $\textsc{SameAdvisor}(a_x, a_y)$. By using aggregate predicates we make explicit the fact that we are modeling the attributes at the object level.

We distinguish between *aggregate query predicates*, which represent unobserved aggregate variables, and *aggregate evidence predicates*, which represent observed aggregate variables. Note that using aggregate *query* predicates can complicate inference, since they represent a collection of fully connected hidden variables. The main point of this paper is that although these aggregate query predicates are specifiable in MLNs, they have not been utilized because of the resulting inference complexity. We show that the gains made possible by these predicates often outweigh the approximations required for inference.

As discussed in Section 3.1, for each aggregate query predicates $\mathrm{A}(\mathbf{d})$, it is critical that the model predict consistent values for every related subset of $\mathbf{d}$. Enforcing this consistency requires introducing dependency edges between aggregate query predicates that share arguments. In general, this can be a difficult problem. Here, we focus on the special case for identity uncertainty where the main query predicate under consideration is $\textsc{AreEqual}(\mathbf{d})$.

The aggregate query predicate $\textsc{AreEqual}(\mathbf{d})$ is true if and only if all constants $d_i \in \mathbf{d}$ refer to the same object. Since each subset of constants corresponds to a candidate object, a (consistent) setting of all the AreEqual predicates results in a solution to the object identification problem. The number of objects is chosen based on the optimal grounding of each of these aggregate predicates, and therefore does not require a prior over the number of objects. That is, once all the AreEqual predicates are set, they determine a clustering with a fixed number of objects. The number of objects is not modeled or set directly, but is implied by the result of MAP inference. (However, a posterior over the number of objects could be modeled discriminatively in an MLN (Richardson and Domingos, 2004).)

This formulation also allows us to compute aggregate evidence predicates over objects to help predict the values of each AreEqual predicate. For example, $\textsc{NumberFirstNames}(\mathbf{d})$ returns the number of different first names used to refer to the object referenced by constants $\mathbf{d}$. In this way, we can model aggregate features of an object, capturing the compatibility among its attributes.

For a given C, there are $|\mathcal{P}(I_C)|$ possible groundings of the AreEqual query predicates. Naively implemented, such an approach would require enumerating all subsets of constants, ultimately resulting in an unwieldy network.

An equivalent way to state the problem is that using N-ary predicates results in a Markov network with one node for each grounding of the predicate. Since in the general case there is one grounding for each subset of C, the size of the corresponding Markov network will be exponential in $|C|$. See Figure 1 for an example instantiation of an MLN with three constants (a, b, c) and one AreEqual predicate.

In this paper, we provide algorithms to perform approximate inference and parameter estimation by incrementally instantiating these predicates

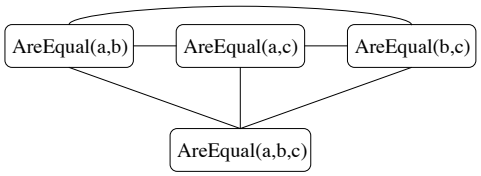

Figure 1: An example of the network instantiated by an MLN with three constants and the aggregate predicate AreEqual, instantiated for all possible subsets with size ≥ 2.

as needed.

3.4 MAP Inference

Maximum a posteriori (MAP) inference seeks the solution to

$$y^* = \underset{y}{\operatorname{argmax}}\, P(Y = y | X = x)$$

where y^* is the setting of all the query predicates F_y (e.g. AreEqual) with the maximal conditional density.

In large, densely connected Markov networks, a common approximate inference technique is loopy belief propagation (i.e. the *max-product* algorithm applied to a cyclic graph). However, the use of aggregate predicates makes it intractable even to instantiate the entire network, making max-product an inappropriate solution.

Instead, we employ an incremental inference technique that grounds aggregate query predicates in an agglomerative fashion based on the model's current MAP estimates. This algorithm can be viewed as a greedy agglomerative search for a local optimum of $P(Y|X)$, and has connections to recent work on *correlational clustering* (Bansal et al., 2004) and graph partitioning for MAP estimation (Boykov et al., 2001).

First, note that finding the MAP estimate does not require computing Z_x, since we are only interested in the *relative* values of each configuration, and Z_x is fixed for a given x. Thus, at iteration t, we compute an unnormalized score for y^t (the current setting of the query predicates) given the evidence predicates x as follows:

$$S(y^t, x) = \exp\left(\sum_{i=0}^{|F^t|} w_i n_i(x, y^t)\right)$$

where $F^t \subseteq F_y$ is the set of aggregate predicates representing a partial solution to the object identification task for constants C, specified by y^t.

Algorithm 1 Approximate MAP Inference Algorithm

```
1: Given initial predicates F^0
2: while ScoreIsIncreased do
3:     F_i^* ⇐ FindMostLikelyPredicate(F^t)
4:     F_i^* ⇐ true
5:     F^t ⇐ ExpandPredicates(F_i^*, F^t)
6: end while
```

Algorithm 1 outlines a high-level description of the approximate MAP inference algorithm. The algorithm first initializes the set of query predicated F^0 such that all AreEqual predicates are restricted to pairs of constants, i.e. AreEqual(c_i, c_j) $\forall(i, j)$. This is equivalent to a Markov network containing one unobserved random variable for each pair of constants, where each variable indicates whether a pair of constants refer to the same object.

Initially, each AreEqual predicate is assumed false. In line 3, the procedure FindMostLikelyPredicate iterates through each query predicate in F^t, setting each to true in turn and calculating its impact on the scoring function. The procedure returns the predicate F_i^* such that setting F_i^* to True results in the greatest increase in the scoring function $S(y^t, x)$.

Let $(c_i^* \ldots c_j^*)$ be the set of constants "merged" by setting their AreEqual predicate to true. The ExpandPredicates procedure creates new predicates AreEqual$(c_i^* \ldots c_j^*, c_k \ldots c_l)$ corresponding to all the potential predicates created by merging the constants $c_i^* \ldots c_j^*$ with any of the other previously merged constants. For example, after the first iteration, a pair of constants (c_i^*, c_j^*) are merged. The set of predicates are expanded to include AreEqual(c_i^*, c_j^*, c_k) $\forall c_k$, reflecting all possible additional references to the proposed object referenced by c_i^*, c_j^*.

This algorithm continues until there is no predicate that can be set to true that increases the score function.

In this way, the final setting of F_y is a local maximum of the score function. As in other search algorithms, we can employ look-ahead to reduce the greediness of the search (i.e., consider multiple merges simultaneously), although we do not include look-ahead in experiments reported here.

It is important to note that each expansion of the aggregate query predicates F_y has a corresponding set of aggregate *evidence* predicates. These evidence predicates characterize the compatibility of the attributes of each hypothesized object.

3.4.1 Pruning

The space required for the above algorithm scales $\Omega(|C|^2)$, since in the initialization step we must ground a predicate for each pair of constants. We use the *canopy method* of McCallum et al. (2000), which thresholds a "cheap" similarity metric to prune unnecessary comparisons. This pruning can be done at subsequent stages of inference to restrict which predicates variables will be introduced.

Additionally, we must ensure that predicate settings at time t do not contradict settings at $t-1$ (e.g. if $F^t(a,b,c) = 1$, then $F^{t+1}(a,b) = 1$). By greedily setting unobserved nodes to their MAP estimates, the inference algorithm ignores inconsistent settings and removes them from the search space.

3.5 Parameter estimation

Given a fully labeled training set $\mathcal{D}$ of constants annotated with their referent objects, we would like to estimate the value of w that maximizes the likelihood of $\mathcal{D}$. That is, $w^* = \text{argmax}_w P_w(y|x)$.

When the data are few, we can explicitly instantiate all AREEQUAL($\mathbf{d}$) predicates, setting their corresponding nodes to the values implied by $\mathcal{D}$. The likelihood is given by Equation 1, where the normalizer is $Z_{\mathbf{x}} = \sum_{y'} \exp\left(\sum_{i=1}^{|F'_y|} w_i n_i(x, y')\right)$.

Although this sum over y' to calculate $Z_{\mathbf{x}}$ is exponential in $|\mathbf{y}|$, many inconsistent settings can be pruned as discussed in Section 3.4.1.

In general, however, instantiating the entire set of predicates denoted by y and calculating $Z_{\mathbf{x}}$ is intractable. Existing methods for MLN parameter estimation include pseudo-likelihood and voted perceptron (Richardson and Domingos, 2004; Singla and Domingos, 2005). We instead follow the recent success in *piecewise training* for complex undirected graphical models (Sutton and McCallum, 2005) by making the following two approximations. First, we avoid calculating the global normalizer $Z_{\mathbf{x}}$ by calculating local normalizers, which sum only over the two values for each aggregate query predicate *grounded in the training data*. We therefore maximize the sum of *local* probabilities for each query predicate given the evidence predicates.

This approximation can be viewed as constructing a log-linear binary classifier to predict whether an isolated set of constants refer to the same object. Input features include arbitrary first-order features over the input constants, and the output is a binary variable. The parameters of this classifier correspond to the w weights in the MLN. This simplification results in a convex optimization problem, which we solve using gradient descent with L-BFGS, a second-order optimization method (Liu and Nocedal, 1989).

The second approximation addresses the fact that all query predicates from the training set cannot be instantiated. We instead sample a subset $F_S \in F_y$ and maximize the likelihood of this subset. The sampling is not strictly uniform, but is instead obtained by collecting the predicates created while performing object identification using a weak method (e.g. string comparisons). More explicitly, predicates are sampled from the training data by performing greedy agglomerative clustering on the training mentions, using a scoring function that computes the similarity between two nodes by string edit distance. The goal of this clustering is not to exactly reproduce the training clusters, but to generate correct and incorrect clusters that have similar characteristics (size, homogeneity) to what will be present in the testing data.

4 Experiments

We perform experiments on two object identification tasks: *citation matching* and *author disambiguation*. *Citation matching* is the task of determining whether two research paper citation strings refer to the same paper. We use the Citeseer corpus (Lawrence et al., 1999), containing approximately 1500 citations, 900 of which are unique. The citations are manually labeled with cluster identifiers, and the strings are segmented into fields such as author, title, etc. The citation data is split into four disjoint categories by topic, and the results presented are obtained by training on three categories and testing on the fourth.

Using first-order logic, we create a number of aggregate predicates such as ALLTITLESMATCH, ALLAUTHORSMATCH, ALLJOURNALSMATCH, etc., as well as their existential counterparts, THEREEXISTSTITLEMATCH, etc. We also include *count* predicates, which indicate the number of these matches in a set of constants.

Additionally, we add edit distance predicates, which calculate approximate matches[1] between title fields, etc., for each pair of citations in a set of citations. Aggregate features are used for these, such as "there exists a pair of citations in this cluster which have titles that are less than 30% similar" and "the minimum edit distance between titles in a cluster is greater than 50%."

We evaluate using pairwise precision, recall, and F1, which measure the system's ability to predict whether each pair of constants refer to the same object or not. Table 1 shows the advantage of our

[1] We use the Secondstring package, found at http://secondstring.sourceforge.net

Table 1: Precision, recall, and F1 performance for citation matching task, where OBJECTS is an MLN using aggregate predicates, and PAIRS is an MLN using only pairwise predicates. OBJECTS outperforms PAIRS on three of the four testing sets.

	Objects			Pairs		
	pr	**re**	**f1**	**pr**	**re**	**f1**
constraint	85.8	79.1	**82.3**	63.0	98.0	76.7
reinforce	97.0	90.0	**93.4**	65.6	98.2	78.7
face	93.4	84.8	**88.9**	74.2	94.7	83.2
reason	97.4	69.3	81.0	76.4	95.5	**84.9**

Table 2: Performance on the author disambiguation task. OBJECTS outperforms PAIRS on two of the three testing sets.

	Objects			Pairs		
	pr	**re**	**f1**	**pr**	**re**	**f1**
miller d	73.9	29.3	41.9	44.6	1.0	**61.7**
li w	39.4	47.9	**43.2**	22.1	1.0	36.2
smith b	61.2	70.1	**65.4**	14.5	1.0	25.4

proposed model (OBJECTS) over a model that only considers pairwise predicates of the same features (PAIRS). Note that PAIRS is a strong baseline that performs collective inference of citation matching decisions, but is restricted to use only $\text{IsEqual}(c_i, c_j)$ predicates over pairs of citations. Thus, the performance difference is due to the ability to model first-order features of the data.

Author disambiguation is the task of deciding whether two strings refer to the same author. To increase the task complexity, we collect citations from the Web containing different authors with matching last names and first initials. Thus, simply performing a string match on the author's name would be insufficient in many cases. We searched for three common last name / first initial combinations (MILLER, D; LI, W; SMITH, B). From this set, we collected 400 citations referring to 56 unique authors. For these experiments, we train on two subsets and test on the third.

We generate aggregate predicates similar to those used for citation matching. Additionally, we include features indicating the overlap of tokens from the titles and indicating whether there exists a pair of authors in this cluster that have different middle names. This last feature exemplifies the sort of reasoning enabled by aggregate predicates: For example, consider a pairwise predicate that indicates whether two authors have the same middle name. Very often, middle name information is unavailable, so the name "Miller, A." may have high similarity to both "Miller, A. B." and "Miller, A. C.". However, it is unlikely that the same person has two different middle names, and our model learns a weight for this feature. Table 2 demonstrates the advantage of this method.

Overall, OBJECTS achieves $F1$ scores superior to PAIRS on 5 of the 7 datasets. These results indicate the potential advantages of using complex first-order quantifiers in MLNs. The cases in which PAIRS outperforms OBJECTS are likely due to the fact that the approximate inference used in OBJECTS is greedy. Increasing the robustness of inference is a topic of future research.

5 Conclusions and Future Work

We have presented an algorithm that enables practical inference in MLNs containing first-order existential and universal quantifiers, and have demonstrated the advantages of this approach on two real-world datasets. Future work will investigate efficient ways to improve the approximations made during inference, for example by reducing its greediness by revising the MAP estimates made at previous iterations.

Although the optimal number of objects is chosen implicitly by the inference algorithm, there may be reasons to explicitly model this number. For example, if there exist global features of the data that suggest there are many objects, then the inference algorithm should be less inclined to merge constants. Additionally, the data may exhibit "preferential attachment" such that the probability of a constant being added to an existing object is proportional to the number of constants that refer to that object. Future work will examine the feasibility of adding aggregate query predicates to represent these values.

More subtly, one may also want to directly model the size of the object population. For example, given a database of authors, we may want to estimate not only how many distinct authors exist in the database, but also how many distinct authors exist outside of the database, as discussed in Milch et al. (2005). Discriminatively-trained models cannot easily reason about objects for which they have no observations; so a generative/discriminative hybrid model may be required to properly estimate this value.

Finally, while the inference algorithm we describe is evaluated only on the object uncertainty task, we would like to extend it to perform inference over arbitrary query predicates.

6 Acknowledgments

We would like to thank the reviewers, and Pallika Kanani for helpful discussions. This work was supported in part by the Center for Intelligent Information Retrieval, in part by U.S. Government contract #NBCH040171 through a subcontract with BBNT Solutions LLC, in part by The Central Intelligence Agency, the National Security Agency and National Science Foundation under NSF grant #IIS-0326249, and in part by the Defense Advanced Research Projects Agency (DARPA), through the Department of the Interior, NBC, Acquisition Services Division, under contract number NBCHD030010. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s)' and do not necessarily reflect those of the sponsor.

References

Nikhil Bansal, Avrim Blum, and Shuchi Chawla. 2004. Correlation clustering. *Machine Learining*, 56:89–113.

Yuri Boykov, Olga Veksler, and Ramin Zabih. 2001. Fast approximate energy minimization via graph cuts. *In IEEE transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 23(11):1222–1239.

Peter Carbonetto, Jacek Kisynski, Nando de Freitas, and David Poole. 2005. Nonparametric bayesian logic. In *UAI*.

J. Cussens. 2003. Individuals, relations and structures in probabilistic models. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, pages 126–133, Acapulco, Mexico.

Hal Daumé III and Daniel Marcu. 2004. Supervised clustering with the dirichlet process. In *NIPS'04 Learning With Structured Outputs Workshop*, Whistler, Canada.

Rodrigo de Salvo Braz, Eyal Amir, and Dan Roth. 2005. Lifted first-order probabilistic inference. In *IJCAI*, pages 1319–1325.

L. Dehaspe. 1997. Maximum entropy modeling with clausal constraints. In *Proceedings of the Seventh International Workshop on Inductive Logic Programming*, pages 109–125, Prague, Czech Republic.

I. P. Fellegi and A. B. Sunter. 1969. A theory for record linkage. *Journal of the American Statistical Association*, 64:1183–1210.

Nir Friedman, Lise Getoor, Daphne Koller, and Avi Pfeffer. 1999. Learning probabilistic relational models. In *IJCAI*, pages 1300–1309.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA.

S. Lawrence, C. L. Giles, and K. Bollaker. 1999. Digital libraries and autonomous citation indexing. *IEEE Computer*, 32:67–71.

D. C. Liu and J. Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Math. Programming*, 45(3, (Ser. B)):503–528.

A. McCallum and B. Wellner. 2003. Toward conditional models of identity uncertainty with application to proper noun coreference. In *IJCAI Workshop on Information Integration on the Web*.

Andrew K. McCallum, Kamal Nigam, and Lyle Ungar. 2000. Efficient clustering of high-dimensional data sets with application to reference matching. In *Proceedings of the Sixth International Conference On Knowledge Discovery and Data Mining (KDD-2000)*, Boston, MA.

Joseph F. McCarthy and Wendy G. Lehnert. 1995. Using decision trees for coreference resolution. In *IJCAI*, pages 1050–1055.

Brian Milch, Bhaskara Marthi, and Stuart Russell. 2004. Blog: Relational modeling with unknown objects. In *ICML 2004 Workshop on Statistical Relational Learning and Its Connections to Other Fields*.

Brian Milch, Bhaskara Marthi, and Stuart Russell. 2005. BLOG: Probabilistic models with unknown objects. In *IJCAI*.

Parag and Pedro Domingos. 2004. Multi-relational record linkage. In *Proceedings of the KDD-2004 Workshop on Multi-Relational Data Mining*, pages 31–48, August.

D. Poole. 2003. First-order probabilistic inference. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, pages 985–991, Acapulco, Mexico. Morgan Kaufman.

M. Richardson and P. Domingos. 2004. Markov logic networks. Technical report, University of Washington, Seattle, WA.

Parag Singla and Pedro Domingos. 2005. Discriminative training of markov logic networks. In *Proceedings of the Twentieth National Conference of Artificial Intelligence*, Pittsburgh, PA.

Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Comput. Linguist.*, 27(4):521–544.

Charles Sutton and Andrew McCallum. 2005. Piecewise training of undirected models. In *Submitted to 21st Conference on Uncertainty in Artificial Intelligence*.

Ben Taskar, Abbeel Pieter, and Daphne Koller. 2002. Discriminative probabilistic models for relational data. In *Uncertainty in Artificial Intelligence: Proceedings of the Eighteenth Conference (UAI-2002)*, pages 485–492, San Francisco, CA. Morgan Kaufmann Publishers.

William E. Winkler. 1993. Improved decision rules in the fellegi-sunter model of record linkage. Technical report, Statistical Research Division, U.S. Census Bureau, Washington, DC.