# Learning Multiple Hierarchical Relational Clusterings

**Aniruddh Nath** NATH@CS.WASHINGTON.EDU
**Pedro Domingos** PEDROD@CS.WASHINGTON.EDU
Department of Computer Science and Engineering, University of Washington, Seattle, WA 98195, U.S.A.

## Abstract

Three important generalizations of the basic clustering problem are relational, hierarchical, and multiple clustering. This paper proposes the first approach to clustering that unifies all three. We describe a general probabilistic model for relational clustering, and show that flat, hierarchical and multiple relational clustering models are special cases. This paper also describes an efficient search algorithm for learning multiple hierarchical clusterings. A preliminary empirical evaluation shows the promise of our approach.

## 1. Introduction

In recent years, relational clustering has been successfully used to learn simple models of complex domains, both for human interpretability and to make predictions about unobserved relations. Most existing approaches to relational clustering learn flat clusterings, grouping together similar objects in a relational database. However, in complex relational domains, a flat clustering is often an excessively simplistic model of the data. Different properties of the same objects may be best explained by different partitions of the objects. Some attributes and relations may be best modeled with a coarser or finer clustering than others. Multiple clustering and hierarchical clustering have been previously studied in isolation, by Kok & Domingos (2007) and Roy et al. (2007) respectively. In this paper, we present the first unified model for multiple hierarchical relational clusterings. Each atom in the database is predicted by the hierarchical clustering that best explains it. Within each clustering, each prediction takes into account information at multiple levels of the hierarchy, using shrinkage to learn more robust parameters.

This line of work was inspired by Kemp et al.'s (2006) *Infinite Relational Model (IRM)*, which jointly clusters objects and predicates in a relational database. Given the cluster assignments, the truth value of an atom can be predicted from the *cluster combination* the atom belongs to (i.e., the vector of cluster assignments of the predicate and objects associated with the atom).

Building on this idea, Kok & Domingos' (2007) *Multiple Relational Clustering (MRC)* learned several cross-cutting clusterings of a domain, rather than trying to explain a database from a single clustering. The rationale is that different clusterings may be better for predicting different subsets of the atoms; it may not be possible to learn a single clustering that satisfactorily explains the entire database. Kok & Domingos empirically demonstrated that learning multiple clusterings can greatly improve prediction quality.

One limitation of IRM and MRC is that both models learn flat clusterings rather than hierarchical models. Hierarchical models can be useful both for human interpretability and for improving predictive performance. Roy et al. (2007) proposed a generative model for *annotated hierarchies* of relational data, where objects are hierarchically clustered into a single tree, and each predicate is associated with the tree-consistent partition of objects that best explains it. We refer to their system as the *Annotated Hierarchy Model (AHM)*. The motivation for this system was twofold: (i) to learn compact, interpretable structures suitable for data analysis, and (ii) to provide a model of human learning. However, its capabilities as a predictive model were not investigated.

The rest of this paper is organized as follows. In section 2, we describe a unified model for multiple hierarchical relational clustering. In section 3, we describe an efficient algorithm for learning the weights and structure of the model. Section 4 describes a preliminary empirical evaluation of the single-hierarchy version of the algorithm. Finally, section 5 lists directions for future work.

## 2. Model

We start with a standard Naive Bayes mixture model, and then describe a series of extensions capturing relational, hierarchical and multiple clustering models.

Appearing in *ICML-12 Workshop on Statistical Relational Learning*, Edinburgh, Scotland, UK, 2012. 

### 2.1. Naive Bayes Mixture Model

A Naive Bayes mixture model assumes that an object's attributes $(X_1, \ldots, X_a)$ are independent given the object's class, $C$. The joint probability is simply:

$$P(C, X_1, \ldots, X_a) = P(C) \prod_{i=1}^{a} P(X_i|C) \tag{1}$$

### 2.2. Hierarchical Clustering Model

Hierachical models allow more robust predictions through the use of *shrinkage* (McCallum et al., 1998), a statistical technique that smoothes parameter estimates of data-sparse feature towards their more data-rich ancestors in the hierarchy.

We define $(\mathcal{D}^{(1)}, \ldots, \mathcal{D}^{(h)})$, where $\mathcal{D}^{(i)}$ is the set of cluster symbols $\{c_1^{(i)}, \ldots, c_k^{(i)}\}$ in the $i$th level of the hierarchy. Each cluster $c_j^{(i)}$ with $1 \leq i < h$ has a *parent* cluster $\rho(c_j^{(i)})$ in layer $\mathcal{D}^{(i+1)}$.

The model contains one cluster assignment variable $C^{(i)}$ for each level $i$ in the hierarchy. $\mathcal{D}^{(i)}$ is the set of possible values for $C^{(i)}$. If $C^{(i)} = c$, then $C^{(i+1)} = \rho(c)$; this ensures that the cluster assignments are consistent with the hierarchy structure.

The cluster assignments are generated top-down; $C^{(h)}$ is generated from a multinomial distribution over the symbols in $\mathcal{D}^{(h)}$. For the finer layers, cluster memberships are generated from a multinomial distribution over a set of siblings, i.e. clusters that share the same parent. $P(c)$ is the multinomial parameter corresponding to cluster $c$.

The attributes are generated conditioned on the vector of cluster assignments for the object. The distribution for each attribute is a log-linear model, with one feature for each level in the hierarchy:

$$P(X_j|C^{(1)}, \ldots, C^{(h)}) = \frac{1}{1 + \exp\left(-\sum_{i=1}^{h} w_{C^{(i)}}^{(j)}\right)}$$

The full joint probability distribution is:

$$P(\mathbf{C}, \mathbf{X}) = \prod_{i=1}^{h} P(C^{(i)}) \prod_{X_j \in \mathbf{X}^+} \frac{1}{1 + \exp\left(-\sum_{i=1}^{h} w_{C^{(i)}}^{(j)}\right)} \prod_{X_j \in \mathbf{X}^-} \frac{1}{1 + \exp\left(\sum_{i=1}^{h} w_{C^{(i)}}^{(j)}\right)} \tag{2}$$

Where $\mathbf{C} = (C^{(1)}, \ldots, C^{(h)})$; $\mathbf{X} = (X_1, \ldots, X_a)$; $\mathbf{X}^+$ and $\mathbf{X}^-$ are respectively the sets of true and false attributes in $\mathbf{X}$.

### 2.3. Relational Clustering Model

Relational models such as the *stochastic blockmodel* (Holland et al., 1983; Nowicki & Snijders, 2001) jointly cluster a set of object and predicate symbols $\{o_1, \ldots, o_n\}$ based on their relations. The traditional blockmodel formulation draws the cluster memberships from a multinomial distribution with a fixed number of components, $k$. Let $\{c_1, \ldots, c_k\}$ be the cluster symbols. Let $\mathbf{C} = (C_1, \ldots, C_n)$ be the vector of cluster membership variables for the $n$ object and predicate symbols (i.e., $C_i = c_j$ means that item $o_i$ is assigned to cluster $c_j$). The probability of cluster assignment $\mathbf{C}$ is:

$$P(\mathbf{C}) = \prod_{i=1}^{n} P(C_i) = \prod_{j=1}^{k} P(c_k)^{|c_k|}$$

(Here, $P(c_k)$ is the probability of component $c_k$ in the multinomial, and $|c_k|$ is the number of items assigned to component $c_k$.)

Each atom is then generated conditioned on the vector of cluster assignments of the items in the relation:

$$X = o_r(o_1, \ldots, o_p) \sim \text{Bernoulli}(\eta(C_r, C_1, \ldots, C_p))$$

We refer to a vector of cluster assignments as a *cluster combination*. Let $M_X$ be the cluster combination that explains atom $X$ as described above, and let $\eta_{M_X}$ be the parameter of the corresponding Bernoulli distribution. $\mathbf{X}$ is the set of atoms in the domain; $\mathbf{X}^+$ and $\mathbf{X}^-$ are respectively the sets of true and false atoms in $\mathbf{X}$. The full joint probability distribution is:

$$P(\mathbf{C}, \mathbf{X}) = \prod_{j=1}^{k} P(c_k)^{|c_k|} \prod_{X \in \mathbf{X}^+} \eta_{M_X} \prod_{X \in \mathbf{X}^-} (1 - \eta_{M_X}) \tag{3}$$

Note that the number of components need not be fixed in advance; IRM generates the clusters using a Chinese Restaurant Process (Pitman, 2002), allowing the data to dictate the number of components.

### 2.4. Hierarchical Relational Clustering Model

In this section, we describe a hierarchical relational clustering model that generalizes stochastic blockmodels much like the model in section 2.2 extends Naive Bayes mixture models. The hierarchical model contains a vector of cluster assignments for each level $i$ in the hierarchy: $\mathbf{C}^{(i)} = (C_1^{(i)}, \ldots, C_n^{(i)})$. The full hierarchical clustering is defined by the matrix $\mathbf{C} = (\mathbf{C}^{(1)}, \ldots, \mathbf{C}^{(h)})$, each row of which captures the cluster assignments in one level of the hierarchy.

As in section 2.2, each cluster $c_j^{(i)}$ has a *parent* in the layer above, and each cluster membership is generated from a multinomial distribution over a set of siblings.

As in section 2.3, the atoms are generated conditioned on the cluster memberships of their predicate and object symbols. For each atom $X = o_r(o_1, \ldots, o_p)$, let cluster combination $M_X^{(i)} = (C_r^{(i)}, C_1^{(i)}, \ldots, C_p^{(i)})$ be the vector of cluster assignments of the symbols in $X$, in level $i$ of the hierarchy. $X$ is generated from a log-linear model with one feature for each cluster combination that explains it:

$$P(X|\mathbf{C}) = \frac{1}{1 + \exp\left(-\sum_{i=1}^{h} w_{M_X^{(i)}}\right)}$$

$w_M$ is the weight of the feature corresponding to cluster combination $M$. The model can be straightforwardly extended to include cross-level features, but we assume in this paper that all clusters in a cluster combination come from the same level in the hierarchy.

The full joint probability distribution is:

$$P(\mathbf{C}, \mathbf{X}) = \left(\prod_{i=1}^{h} \prod_{c_k \in \mathcal{D}^{(i)}} P(c_k)^{|c_k|}\right) \prod_{X \in \mathbf{X}^+} \frac{1}{1 + \exp\left(-\sum_{i=1}^{h} w_{M_X^{(i)}}\right)} \prod_{X \in \mathbf{X}^-} \frac{1}{1 + \exp\left(\sum_{i=1}^{h} w_{M_X^{(i)}}\right)} \quad (4)$$

There are several differences between our hierarchical model (HRC) and Roy et al.'s AHM. Their system does not incorporate any form of predicate clustering, which is important in domains with a large number of predicates. AHM learns the single tree-consistent partition of objects that best explains each predicate, while our model makes the prediction jointly over the entire hierarchy. A form of shrinkage can be achieved within the AHM framework, by marginalizing over tree-consistent partitions. However, HRC achieves shrinkage even when we just use the MAP hypothesis, which is advantageous both for computational reasons and for human interpretability. Another difference is that HRC allows a node to have an arbitrary number of children, allowing us to represent more compact, interpretable trees (sometimes referred to as *rose trees*, Blundell et al., 2011).

### 2.5. Multiple Hierarchical Relational Clustering Model

There are two alternative approaches to unifying hierarchical clustering with multiple clustering: the model could consist of multiple hierarchical clusterings, or a hierarchy of multiple-clustering models. The model we describe in this section takes the former approach.

A *hierarchical clustering* $\mathbf{C}^{(\cdot,j)} = (\mathbf{C}^{(1,j)}, \ldots \mathbf{C}^{(h_j,j)})$ is a matrix of cluster assignment variables (as in section 2.4). $\mathbf{C}^{(i,j)} = (C_1^{(i,j)}, \ldots, C_n^{(i,j)})$ represents the cluster assignments of the $n$ items in level $i$ of the $j$th hierarchical clustering. $h_j$ is the number of levels in the $j$th clustering. The complete set of cluster assignments in the model is defined by the 3-dimensional tensor $\mathbf{C} = \{\mathbf{C}^{(\cdot,1)}, \ldots, \mathbf{C}^{(\cdot,m)}\}$, which represents $m$ hierarchical clusterings.

To allow different clusterings to model different subsets of the objects and predicates, the model contains a matrix of indicator variables $O$; when $O_i^{(\cdot,j)} = 1$, we say that clustering $j$ *contains* item $o_i$. For notational convenience, we also define $N_X^{(\cdot,j)}$; this indicator variable has a value of 1 iff $O_i^{(\cdot,j)} = 1$ for all of $X$'s arguments $o_i$. When this is the case, we say that clustering $j$ *contains* atom $X$.

The cluster memberships in hierarchical clusterings are generated from multinomials, as in sections 2.2 and 2.4. As in the previous section, the atoms are generated conditioned on the cluster assignments using a log-linear model with one feature per cluster combination. The only difference is that we now need to include features from all of the hierarchical clusterings that contain each atom:

$$P(\mathbf{C}, \mathbf{X}) = \left(\prod_{j=1}^{m} \prod_{i=1}^{h_j} \prod_{c_k \in \mathcal{D}^{(i,j)}} P(c_k)^{|c_k|}\right) \prod_{X \in \mathbf{X}^+} \frac{1}{1 + \exp\left(-\sum_{j=1}^{m} N_X^{(\cdot,j)} \sum_{i=1}^{h_j} w_{M_X^{(i,j)}}\right)} \prod_{X \in \mathbf{X}^-} \frac{1}{1 + \exp\left(\sum_{j=1}^{m} N_X^{(\cdot,j)} \sum_{i=1}^{h_j} w_{M_X^{(i,j)}}\right)} \quad (5)$$

Here, $\mathcal{D}^{(i,j)}$ is the set of cluster symbols in level $i$ of clustering $j$. Cluster combination $M_X^{(i,j)}$ is the vector of cluster assignments of the arguments of atom $X$ in level $i$ of clustering $j$.

## 3. Learning

Given this model and an evidence database, the learning problem is to find the MAP (*maximum a posteriori*) cluster assignment and the optimal parameter values. The resulting model can be used directly to predict missing data, or it may be used as input for subsequent processing, for instance using coarse-to-fine algorithms (Kiddon & Domingos, 2011).

The MAP inference problem has two subtasks, each of which introduces some new challenges in our setting: **weight learning** (learning optimal values of the parameters, given a candidate structure) and **structure search** (finding the highest-scoring cluster assignment tensor, $\mathbf{C}$). We describe our approach to each of these problems below.

### 3.1. Weight Learning

Let us first assume that cluster assignments are known. The learning problem then reduces to finding the optimal parameters for the model, i.e., the cluster weights ($P(c)$) and the weights of the predictive features ($w_{M_X}^{(i,j)}$). The $P(c)$'s are simply multinomial parameters, and can be trivially computed in closed form.

Computing the atom prediction weights is more challenging. In other relational clustering models such as IRM, MRC and AHM, each atom is predicted by a single cluster combination, making the weight-learning problem trivial. For instance, in MRC (a special case of our model), the weight of a cluster combination is simply the log-odds of a random atom being true. In our setting, however, a single atom may be influenced by features at multiple levels in the hierarchy.

A further complication is that we regularize the predictive weights to prevent overfitting; we use both $\ell_0$ and $\ell_1$ penalties. The $\ell_0$ term is simply a penalty in log-space for each non-zero predictive feature we introduce; this has the effect of discouraging unnecessarily fine-grained predictive features. The $\ell_1$ term penalizes the sum of the absolute values of the weights. Weight learning for log-linear models with $\ell_1$ regularization typically requires the use of iterative optimization algorithms (e.g. Andrew & Gao, 2007). Since the weights need to be relearned for each structure evaluated during the search, these algorithms may be prohibitively expensive.

Instead of running a full $\ell_1$ optimizer at each step of the search, we learn the weights approximately, in a coarse-to-fine manner that takes advantage of the hierarchical structure. This can be done if each atom is explained by exactly one hierarchical clustering (as is the case for the search algorithm discussed in the next section). Note that each cluster combination $M$ on level $i$ is a subset of some cluster combination $M' = \rho(M)$ on level $i+1$; we refer to the latter as the *parent* cluster combination. Notice that if $M = (c_1, \ldots, c_p)$, then $\rho(M) = (\rho(c_1), \ldots, \rho(c_p))$. We compute the weights one level at a time, starting at the coarsest level of the hierarchy. The model learned at each level is a refinement of the level above, adding new, more specific features if the improvement to likelihood outweighs the cost incurred by the $\ell_0$ and $\ell_1$ penalties.

Consider a cluster combination $M$, containing true atoms $\mathbf{X}_M^+$ and false atoms $\mathbf{X}_M^-$. When we compute $w_M$, we have already computed the weights of $M$'s ancestors in the hierarchy (i.e., the cluster combinations of which $M$ is a subset). Let $s_M$ be the sum of the weights of $M$'s ancestors. At this point, all atoms in $M$ occur in the same set of non-zero features (i.e., $M$ and its ancestors); this may change as we learn the weights of finer features in the lower levels of the tree. At this stage of the computation, the likelihood of the atoms in $M$ can be written as follows:

$$L_M = \left(\frac{1}{1+e^{-(s_M+w_M)}}\right)^{|\mathbf{x}_M^+|} \left(\frac{1}{1+e^{(s_M+w_M)}}\right)^{|\mathbf{x}_M^-|}$$

$s_M$ is fixed, having been previously computed. The goal is to learn the optimal value of $w_M$:

$$w_M^* = \underset{w_M}{\textbf{arg max}}\ \log L_M - \alpha_0 \mathbb{1}[w_M \neq 0] - \alpha_1 |w_M|$$

$\alpha_0$ and $\alpha_1$ are respectively the $\ell_0$ and $\ell_1$ penalty parameters. We also add smoothing parameters $\beta^+$ and $\beta^-$ to the true and false atom counts; we omit them from the above equations for simplicity.

The above computation can be done in closed form, despite the discontinuity introduced by the $\ell_1$ term. The expression $(s_M + w_M)$ can be interpreted as a 'smoothed' value for $M$'s feature, shrunk towards a coarser feature in the hierarchy. Note that without the regularization terms, the optimal value of $w_M$ would simply be $(\log({}^t/_f) - s_M)$, effectively ignoring the coarser features and learning a flat model. The higher we set the regularization terms, the less $M$ changes the predictions of its ancestors.

### 3.2. Structure Search

The search for the optimal cluster assignments proceeds in two stages. First, we search for the best flat multiple clustering. This initial clustering can be done with any multiple clustering algorithm; MRC is a natural choice, since its model is a special case of ours. Like MRC, we require for tractability that each atom occur in exactly one clustering (though objects may still occur in multiple clusterings). This partitions the database into several smaller databases, each of which poses a separate hierarchical clustering problem. Algorithm 1 provides an overview of this procedure. MULTICLUSTER$(\mathbf{X}, ml)$ may be any search algorithm that returns a set of non-overlapping flat clusterings; we refer the reader to Kok & Domingos (2007) for an example of how this can be done. The novel part of the algorithm is HIERARCHICALCLUSTER$(\mathbf{X}, ml)$, which replaces each of the flat clusterings with a hierarchical clustering. We devote the rest of this section to the discussion of this subroutine.

We construct the hierarchy layer by layer, starting with each object and predicate in a singleton cluster, and building progressively coarser clusters one level at a time. This allows us to treat the hierarchical clustering problem as a series of flat clustering problems, where the individual items being clustered at each layer are the clusters from the level below.

Algorithm 2 describes this procedure in more detail. FLATCLUSTER$(\mathbf{C}^{(i)})$ is a subroutine that takes a set of

**Algorithm 1** MHRC-LEARN($\mathbf{X}$, $ml$)

**input:** $\mathbf{X}$, a relational database
$ml$, max number of levels in the hierarchy
**output:** $\mathbf{C}$, a multiple hierarchical clustering
$\mathbf{C} \leftarrow \emptyset$
$\mathbf{D} \leftarrow$MULTICLUSTER($\mathbf{X}$)
**for all** flat clusterings $\mathbf{D}^{(\cdot,j)} \in \mathbf{D}$ **do**
  $\mathbf{X}_{D_j} \leftarrow$ atoms contained by $\mathbf{D}^{(\cdot,j)}$
  $\mathbf{C}^{(\cdot,j)} \leftarrow$HIERARCHICALCLUSTER($\mathbf{D}^{(1,j)}, ml$)
**end for**
**return** $\mathbf{C}$

**Algorithm 2** HIERARCHICALCLUSTER($\mathbf{X}$, $ml$)

**input:** $\mathbf{X}$, a relational database
$ml$, max number of levels in the hierarchy
**output:** $\mathbf{C}$, a hierarchical clustering
**for all** object and predicate symbols $o_k$ in $\mathbf{X}$ **do**
  $C_k \leftarrow \{o_k\}$
**end for**
$curLevel \leftarrow (C_1, \ldots, C_n)$
**for** $i \leftarrow 1$ to $ml$ **do**
  $topLevel \leftarrow$ FLATCLUSTER($curLevel$)
  **if** $topLevel$ is identical to $curLevel$ **then**
    **break**
  **end if**
  $\mathbf{C}^{(i)} \leftarrow topLevel$
  $curLevel \leftarrow topLevel$
**end for**
**return** $(\mathbf{C}^{(1)}, \ldots, \mathbf{C}^{(h)})$

clusters as input and partitions them, returning a coarser set of clusters. We can use any clustering procedure that exactly or approximately optimizes the posterior probability. In our experiments, we use a simple greedy search that individually moves each item to the best cluster, iterating until convergence. Items can also be moved to newly created singleton clusters. (The $\ell_0$ penalty described in the previous section discourages the creation of new clusters.) It is straightforward to incorporate other search operations, such as greedy merges and splits, random moves, etc. We also restrict the algorithm to only merge predicate symbols with other predicates of the same arity.

The coarse-to-fine weight learning scheme in section 3.1 runs in time linear in the number of features. However, since we have one feature per cluster combination, this can still be too expensive to run at each step of the search. The weight learning can be further sped up by only updating the weights significantly affected by the change in structure. With the search procedure described in Algorithm 2, the only features affected by a change are the descendents of the altered top-level cluster combinations. Furthermore, we only update the *immediate* descendents of the altered features. This is an approximation; in principle, the weights of the entire subtree could be affected. In practice, however, the weights of non-immediate descendents are not significantly affected by changes in the top-level structure.

For example, consider a top-level cluster combination $M_0$, its child $M_1$, and $M_1$'s child $M_2$. If one of $M_0$'s clusters is altered, then $M_0$'s true and false atom counts may change, necessitating an update for $M_0$'s weight, $w_0$. This changes $s_1$, the sum of the weights of $M_1$'s ancestors ($s_1 = w_0$). We update $w_1$ accordingly.

Now, note that $s_2 = w_0 + w_1 = s_1 + w_1$. Recall that $s_1 + w_1$ is a smoothed version of $M_1$'s parameter. $M_1$'s counts are not affected by the changes to $M_0$; the feature explains the same set of atoms before and after the change. Therefore, unless the regularization parameters are extremely large, changes to $M_0$ will not cause big changes to $s_2 = w_1 + s_1$. Consequently, $w_2$ is relatively unaffected.

## 4. Preliminary Results

Note that HIERARCHICALCLUSTERING($\mathbf{X}, ml$) can be run as a standalone algorithm on the full database, optimizing the model in section 2.4. For our initial experiments, we implemented this version of the algorithm, learning a single hierarchy. We compared two versions of the algorithm: HRC1, which constructs a single level clustering ($ml = 1$); HRC10 learns a hierarchy with up to 10 levels ($ml = 10$). We evaluated these algorithms on the Animals (ANML), Nations (NATS), Kinship (KINS) and UMLS datasets from Kemp et al. (2006). Table 1 describes the size and composition of these domains. We performed 10-fold cross validation, and evaluated the algorithms in a transductive setting: the test atoms were set to 'unknown', and their values were predicted from the known atoms.

We used two evaluation criteria: **AUC** (area under the precision-recall curve of the query atoms) and **CLL** (average conditional log-likelihood of query atoms).

We compared our algorithm to IRM and MRC. IRM and MRC's results are taken directly from (Kok & Domingos, 2007). For HRC, we use $\alpha_0 = 0.01$. $\alpha_1 = 0.1$ on ANIMALS, and 0.01 on the other domains. We set $\beta^+ + \beta^- = 0.1$, with the ratio between them reflecting the ratio of true to false evidence atoms for the corresponding predicate (this smoothing scheme is similar to the one used by Kok & Domingos, 2008).

Despite the simpler search strategy we use, even the flat version of HRC is generally competitive with IRM's more sophisticated search. Learning multiple levels does not prove to be useful on the propositional ANML domain, but slightly improves AUC on the three relational domains (at the cost of a slightly lower CLL on NATS). Although the

| | Objects | Features | Relations | Total atoms | True atoms | HRC1 AUC | HRC1 CLL | HRC10 AUC | HRC10 CLL | IRM AUC | IRM CLL | MRC AUC | MRC CLL |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ANML | 50 | 85 | 0 | 4250 | 1562 | 0.81 | -0.46 | 0.80 | -0.49 | 0.79 | -0.43 | 0.80 | -0.43 |
| NATS | 14 | 111 | 56 | 12,530 | 2565 | 0.68 | -0.41 | 0.69 | -0.43 | 0.75 | -0.31 | 0.75 | -0.31 |
| KINS | 104 | 0 | 26 | 281,216 | 10,686 | 0.74 | -0.06 | 0.75 | -0.06 | 0.68 | -0.06 | 0.85 | -0.05 |
| UMLS | 135 | 0 | 46 | 838,350 | 6529 | 0.70 | -0.014 | 0.70 | -0.014 | 0.80 | -0.011 | 0.97 | -0.004 |

*Table 1.* Experiments

improvements are small, the difference in AUC and CLL is statistically significant on KINS and UMLS, the two largest domains (as measured by a sign test with $p = 0.05$). We suspect that the hierarchy will provide greater benefit on larger, more complex domains with more missing data. MRC proves to be the most successful system overall, suggesting that the full multiple clustering version of the algorithm may perform better than using a single hierarchy.

## 5. Conclusions & Future Work

In this paper, we described a unified model for relational clustering, incorporating both hierarchies and multiple clustering, and we proposed an efficient learning algorithm for the model. The empirical usefulness of multiple clusterings has been previously demonstrated by Kok & Domingos (2007); in this work, we performed a preliminary evaluation of hierarchical relational clustering. Our initial experiments show that our approach is competitive with other relational clustering algorithms; however, more experiments are needed on larger, more complex domains.

The most immediate direction for future work is to evaluate the full version of the algorithm, learning multiple hierarchical clusterings. A further direction is to apply hierarchical relational clustering to learn the structure of Tractable Markov Logic Networks (Domingos & Webb, 2012).

**Acknowledgements:** This research was partly funded by ARO grant W911NF-08-1-0242, AFRL contract FA8750-09-C-0181, NSF grant IIS-0803481, and ONR grant N00014-08-1-0670. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ARO, AFRL, NSF, ONR, or the United States Government.

## References

Andrew, G. and Gao, J. Scalable training of L1-regularized log-linear models. In *Proc. ICML*, 2007.

Blundell, C., Teh, Y. W., and Heller, K. A. Discovering non-binary hierarchical structures with Bayesian rose trees. In *Mixture Estimation and Applications*. 2011.

Domingos, P. and Webb, A. A tractable first-order probabilistic logic. In *Proc. AAAI*, 2012.

Holland, P., Laskey, K. B., and Neinhardt, S. Stochastic blockmodels: Some first steps. *Social Networks*, 5, 1983.

Kemp, C., Tenenbaum, J. B., Griffiths, T. L., Yamada, T., and Ueda, N. Learning systems of concepts with an infinite relational model. In *Proc. AAAI*, 2006.

Kiddon, C. and Domingos, P. Coarse-to-fine inference and learning for first-order probabilistic models. In *Proc. AAAI*, 2011.

Kok, S. and Domingos, P. Statistical predicate invention. In *Proc. ICML*, 2007.

Kok, S. and Domingos, P. Extracting semantic networks from text via relational clustering. In *Proc. ECML*, 2008.

McCallum, A., Rosenfeld, R., Mitchell, T., and Ng, A. Y. Improving text classification by shrinkage in a hierarchy of classes. In *Proc. ICML*, 1998.

Nowicki, K. and Snijders, T. A. B. Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96, 2001.

Pitman, J. *Combinatorial Stochastic Processes*. Springer-Verlag, 2002.

Roy, D. M., Kemp, C., Mansinghka, V., and Tenenbaum, J. B. Learning annotated hierarchies from relational data. In *Proc. NIPS*, 2007.