Learning Relational Probability Trees

Jennifer Neville David Jensen Lisa Friedland Michael Hay

Computer Science Department
University of Massachusetts
Amherst, MA 01003 USA
[jneville | jensen | lfriedl | mhay]@cs.umass.edu

ABSTRACT

Classification trees are widely used in the machine learning and data mining communities for modeling propositional data. Recent work has extended this basic paradigm to probability estimation trees. Traditional tree learning algorithms assume that instances in the training data are homogenous and independently distributed. Relational probability trees (RPTs) extend standard probability estimation trees to a relational setting in which data instances are heterogeneous and interdependent. Our algorithm for learning the structure and parameters of an RPT searches over a space of relational features that use aggregation functions (e.g. AVERAGE, MODE, COUNT) to dynamically propositionalize relational data and create binary splits within the RPT. Previous work has identified a number of statistical biases due to characteristics of relational data such as autocorrelation and degree disparity. The RPT algorithm uses a novel form of randomization test to adjust for these biases. On a variety of relational learning tasks, RPTs built using randomization tests are significantly smaller than other models and achieve equivalent, or better, performance.

1. INTRODUCTION

Classification trees are used widely in the machine learning and data mining community for propositional data. Due to their selectivity and intuitive representation of knowledge, tree models are often easily interpretable. This makes classification trees an attractive modeling approach for the knowledge discovery community. Conventional tree learning algorithms were designed for data sets where the instances are homogeneous and statistically independent. In this paper we present an algorithm for learning classification trees over relational data that are heterogeneous and interdependent.

Classification tree popularity has resulted in a large body of research detailing the results of various algorithm design choices. For example, it has been shown that cross-validation can be used to avoid attribute selection biases [6] and that split criteria are generally insensitive to misclassification costs [14]. Recent work has extended the basic classification tree paradigm to probability estimation trees and has focused on improving probability estimates in leaves [14]. We can leverage this body of research to construct a new algorithm for relational data where many sources of potential variance have been addressed, allowing us to focus on effects due to the characteristics of relational data.

Our recent work has concentrated on the challenges of learning probabilistic models in relational data, where the traditional assumption of instance independence is violated [7, 8]. We have identified three characteristics of relational data—concentrated linkage, degree disparity, and relational autocorrelation—and have shown how they can complicate efforts to construct good statistical models. They can lead to feature selection bias and discovery of spurious correlations, resulting in overly complex models with excess structure.

Excess structure in models is harmful for several reasons. First, such models are factually incorrect, indicating that some variables are related when they are not. Second, such models require more space and computational resources than models that do not contain unnecessary components. Third, using a model with excess structure can require the collection of unnecessary features for each instance, increasing the cost and complexity of making predictions. Fourth, large models are more difficult to understand. The unnecessary components complicate attempts to integrate models with knowledge derived from other sources.

Many techniques common to machine learning, data mining, and statistical modeling rely on the underlying assumption that data instances are independent. Techniques for learning statistical models of relational data need to address the problems associated with violating these assumptions. We have developed a promising class of techniques, based on randomization tests and resampling methods, to adjust for the characteristics of a given relational data set and make accurate parameter estimates and hypothesis tests. We have incorporated these approaches into our algorithm for constructing relational probability trees (RPTs). To our knowledge, RPTs are the only statistical models for relational data that adjust for potential biases due to the characteristics of relational data.

We describe an example analysis task and present an abbreviated version of an RPT model learned for this task. We outline the details of the RPT algorithm and finish with an experimental section that evaluates RPTs against C4.5 [15] and relational Bayes classifiers [13].

2. EXAMPLE TASK

Recent research has examined methods for constructing statistical models of complex relational data [4]. Examples of such data include social networks, genomic data, and data on interrelated people, places, things, and events extracted from text documents. The data set collected by the WebKB Project [2]

SIGKDD '03, August 24-27, 2003, Washington, DC, USA.

consists of a set of web pages from four computer science departments. The web pages have been manually classified into categories: course, faculty, staff, student, research project, or other. The category "other" denotes a page that is not a home page (e.g. a curriculum vitae linked from a faculty page). The collection contains approximately 4,000 web pages and 8,000 hyperlinks among those pages. The database contains attributes associated with each page and hyperlink, including the path and domain of urls, and the directionality (e.g. lateral, up, down) of the link in the department hierarchy.

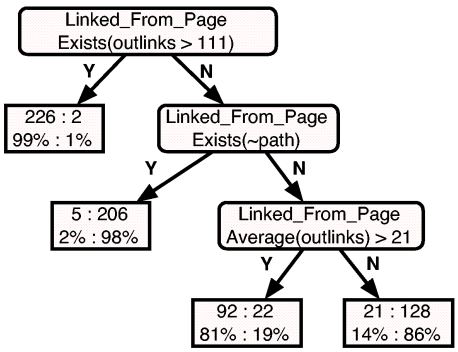


Figure 1: Example probability estimation tree.

Figure 1 shows an RPT constructed by our learning algorithm for the task of predicting whether a web page is a student home page (P(+)=0.51). The tree represents a series of questions to ask about a web page and the pages in its relational neighborhood. The leaf nodes contain probability distributions over the values of the *isStudent* target label. They show the number of training set pages that reached the leaf (partitioned by class) and the resulting class probabilities. In this tree, the root node asks whether the page is linked to from a page with more than 111 out-links (e.g. a directory page). If so, the page travels down the left-hand branch and returns a 99% probability of being a student page. If not, the page travels down the right-hand branch. The next node asks whether the page is linked to from a page without a path appended to the url (e.g. a department home page). If so, it is unlikely to belong to a student. If not, the next node asks whether the page has links from pages with high out degree on average (e.g. directory pages, research group pages). If so, the page is likely to be a student page.

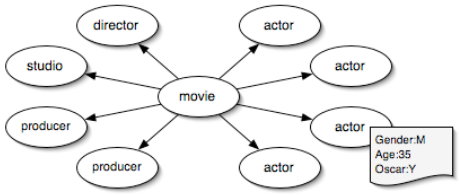
This tree is a condensed version of an RPT learned on data from three departments. The full tree had seven nodes but we aggregated the lower levels for readability and space considerations. When tested on the fourth held-out department, the full tree had perfect area under the ROC curve. The tree construction algorithm is described in the next section, and the details of this experiment are reported in section 4.

3. RELATIONAL PROBABILITY TREES

RPT models estimate probability distributions over possible attribute values. The task of estimating probability distributions over the values of a given attribute would appear to differ little from traditional propositional learning. However, algorithms for relational learning typically look beyond the item for which the attribute is defined to consider the effect of related objects. For example, in order to predict the box-office success of a movie, a relational model might consider not only the attributes of the movie, but also attributes of the actors in the movie and the director, producer, and studio that made the movie (figure 2a). A model might go even further and consider attributes of more "distant" objects (in the sense of graph neighborhood), such as other movies made by the director.

Relational data violate two assumptions of conventional classification techniques. First, algorithms designed for propositional data assume the data are independent and identically distributed (i.i.d.). Relational data, on the other hand, have dependencies both as a result of direct relations (e.g. hyperlinked pages) and through chaining multiple relations together (e.g. pages linked to by the same directory page). Second, algorithms for propositional data assume that the data instances are recorded in homogeneous structures (a fixed set of fields for each object), but relational data "instances" are usually more varied and complex. For example, some movies may have 10 actors and others may have 1000. A relational classification technique needs to contend with dependent, heterogeneous data instances for both learning and inference.

Aggregation is widely used as a means to "propositionalize" relational data for modeling, applied either as a pre-processing step (e.g. [11]) or dynamically during the learning process (e.g. [5]). Heterogeneous data instances are transformed into homogenous records by aggregating multiple values into a single value (e.g. average actor age). Conventional machine learning techniques are then easily applied to the transformed data. Figure 2b contains a portion of an aggregated relational data set where multiple values have been averaged (mode is used for discrete attributes).



(a)

Receipts >$2mil	Mode Actor Gender	Avg Actor Age	Mode Actor Oscar	Mode Studio Loc	...
+	F	28.50	N	USA	...
+	M	33.34	Y	USA	...
-	F	51.00	N	USA	...
+	M	22.79	N	Canada	...
...	...	...	...	...	...

(b)

Figure 2: (a) example relational instance, (b) portion of a propositionalized relational data set.

Classification tree algorithms are easily modified for relational data—the examples simply consist of subgraphs instead of independent objects. Feature specifications need to be enhanced to consider (object, attribute) pairs and aggregation functions, but the basic algorithm structure remains relatively unchanged. Several other decision tree algorithms for relational data have already been developed including TILDE [1], Multi Relational Decision Trees [9] and Structural Regression Trees (SRTs) [10]. These systems focus on extending decision tree algorithms to work in the first-order logic framework used by inductive logic programming systems (ILP). Although these systems can be used to build classification trees with

relational data, the space of feature classes they consider is restricted to a subset of first-order logic. We chose to implement a classification tree algorithm with the capacity to explore a wider range of feature families. In return for this flexibility we gave up some of the expressiveness of the ILP systems mentioned above. RPTs are not able to refer to a particular object throughout a series of conjunctions.

3.1 Algorithm Overview

The RPT algorithm takes a collection of subgraphs as input. Each subgraph contains a single target object to be classified; the other objects and links in the subgraph form its relational neighborhood (figure 2a). For reasons of efficiency, in these experiments we use subgraphs consisting of the target object and everything one link away, but the RPT algorithm can reason with subgraphs of arbitrary complexity. The RPT algorithm constructs a probability estimation tree to predict the target class label given (1) the attributes of the target objects, (2) the attributes of other objects and links in the relational neighborhood, and (3) degree attributes counting objects and links in the relational neighborhood.

The algorithm searches over a space of binary relational features to split the data. For example, MODE(actor.gender)=female tests whether a movie's actors are predominantly female or not. The algorithm constructs features from the attributes of different object/link types in the subgraphs and multiple methods of aggregating the values of those attributes. Feature scores are calculated from the class counts after splitting the data, using chi-square to measure correlation. To choose a feature the algorithm looks at each possible feature, calculating its score and the p-value associated with the score. If the p-value is not significant the feature is dropped from consideration. Among the features that are significantly correlated with the class, the feature with max score is selected for inclusion in the tree.

The algorithm recursively partitions the subgraphs by choosing features and binary splits greedily until further partitions no longer change the class distributions significantly. The current implementation does not use post-pruning, but rather pre-pruning in the form of a p-value cutoff. If the p-value associated with the maximum chi-square score exceeds the cutoff, the method returns without splitting the node and growing stops. Because the features considered are not necessarily independent we use a Bonferroni adjustment on the p-value cutoff to account for any dependence. All experiments reported in this paper used an adjusted alpha of 0.05/|attributes|. This threshold may be too conservative. Future work will explore using cross-validation to determine the correct p-value threshold. Once the tree-growing phase is halted the algorithm calculates the class distribution of the examples at each leaf and stores it in the leaf node. Laplace correction is applied to the distribution to improve the probability estimates [14].

Given an RPT model learned from a set of training examples, the model can be applied to unseen subgraphs for prediction. The chosen feature tests are applied to each subgraph and the example travels down the tree to a leaf node. The model then uses the probability distribution estimated for that leaf node to make a prediction about the class label of the example.

3.2 Relational Features

Features in propositional data typically combine an attribute, an operator, and a value. For example, a feature for a movie might be *genre=comedy*. Relational features are similar in that they identify both an attribute and a way of testing the values of the attribute. However, relational features may also identify a particular relation (e.g. ActedIn(x,y)) that links a single object x (e.g. movie) to a set of other objects Y (e.g. actors). If this is the case, the attribute referenced by the feature may belong to the related objects Y (e.g. actor age), and the test is conducted on the set of attribute values of the objects in Y. For example, the relational feature:

$$Movie(x), Y = \{y \mid ActedIn(x, y)\} : Max(Age(Y)) > 65$$

determines whether the oldest actor in movie x is over 65.

When the relation is non-deterministic, relational features must consider *sets* of attribute values on the objects Y. In this situation, standard database aggregation functions can be used to map sets of values into single values. The RPT algorithm considers the following aggregation functions: MODE/AVERAGE, COUNT, PROPORTION, DEGREE. MODE is used for discrete attributes, AVERAGE for continuous. MINIMUM, MAXIMUM, and EXISTS are special cases of these aggregation functions. COUNT, PROPORTION and DEGREE features consider a number of different thresholds (e.g. PROPORTION>10%). Features of continuous attributes search for the best binary discretization of the attribute (e.g. COUNT(actor.age>15)).

The graph structure of relational data may also be used in features. For example, a feature could count the number of actors associated with a movie, instead of counting a particular attribute value on actors. We will refer to such features as DEGREE features throughout the rest of this paper. DEGREE features can count the degree of links (e.g. number of phone calls between two people), the degree of objects (e.g. number of children for a given parent), or even the degree of attribute values (e.g. number of aliases for a given person). In certain restricted types of relational data, such as spatial or temporal data, it is not as important to be able to represent degree because all instances have similar structure. However, we have analyzed a number of relational data sets that have degree disparity, where degree features are highly correlated with the class label [8].

3.3 Feature Selection Biases

Accurate feature selection is a crucial component of tree learning algorithms. We have shown that feature selection will be biased when relational learning algorithms ignore common characteristics of relational data. Concentrated linkage and relational autocorrelation can cause learning algorithms to be strongly biased toward certain features, irrespective of their predictive power [7], and degree disparity can lead relational learning algorithms to discover misleading correlations [8].

Concentrated linkage occurs when many objects are linked to a common neighbor, and relational autocorrelation occurs when values of a given attribute are highly uniform among objects that share a common neighbor. Linkage and autocorrelation bias feature selection in a two-step chain of causality. First, linkage and autocorrelation combine to reduce the *effective sample size* of a data set, thus increasing the variance of scores estimated using that set. Just as small data samples can lead to inaccurate estimates of the scores used to select features, concentrated linkage and autocorrelation can cause the scores of some features to have high variance. Second, increased variance of score distributions increases the probability that features formed from objects with high linkage and autocorrelation will be selected as the best feature, even when these features are random.

Degree disparity, another common characteristic of relational data, occurs when the frequency (non-determinism) of a rela-

tion is correlated with the values of the target variable. When aggregation is used on data with degree disparity it can lead data mining algorithms to include spurious elements in their models and to miss useful elements. These errors arise because many aggregation functions (e.g. MAX) will produce apparent correlation between the aggregated values (e.g. maximum movie receipts) and a class label (e.g. studio location) whenever degree disparity occurs, regardless of whether the attribute values have any correlation with class label.

3.4 Hypothesis Tests

Statistical hypothesis tests can be used to adjust for feature selection biases. Hypothesis tests compare the value of a statistic (e.g. the correlation of a given feature with the class label) to a *sampling distribution*. A sampling distribution represents the values of the statistic that would be expected under a given *null hypothesis*. A typical null hypothesis is that a feature and a class label are statistically independent, though other null hypotheses are also common. If the value of the statistic exceeds a large percentage of the values in the sampling distribution, the null hypothesis is rejected, and some alternative hypothesis (e.g. that the given feature is correlated with the class label) is accepted.

Widely known and computationally simple procedures exist for hypothesis tests in propositional data. One classical hypothesis test scores the bivariate association between an attribute and the class label using a chi-square statistic. Exact calculations and approximations of the sampling distribution for chi-square are known (parameterized by the number of values in each variable). If the observed chi-square value exceeds the vast majority of the values in the sampling distribution, a learning algorithm may reject the null hypothesis with high confidence and include the variable in an induced model. However, conventional sampling distributions are derived from i.i.d. data. If the instances are drawn from a *relational* data set, the data instances may not be independent so the known sampling distributions may not be appropriate.

3.5 Randomization Tests

Randomization tests provide a method for accurate hypothesis testing in relational data. Randomization tests have been widely applied to propositional data over the past two decades. We have modified these techniques for relational data. The RPT algorithm use randomization tests to account for bias and variance in feature scores due to linkage, autocorrelation and degree disparity.

A randomization test is a type of computationally intensive statistical test [3], which involves generating many replicates of an actual data set—typically called pseudosamples—and to estimate a sampling distribution. Pseudosamples are generated by randomly reordering (permuting) the values of one or more variables in an actual data set. Each unique permutation of the values corresponds to a unique pseudosample. A score is calculated for each pseudosample and the randomized scores are used to estimate a sampling distribution. This sampling distribution is used to calculate an empirical p-value for the score calculated from the actual data.

To construct pseudosamples in relational data, we randomize the attribute values prior to aggregation. We retain the relational structure of the data and randomize attribute vectors associated with each object type. For example, in the case of movies, actors, and studios, this approach randomizes the attribute vectors of movies (e.g. genre and length), actors (e.g. age and gender), and of studios (e.g. location)—preserving each intrinsic attribute vector but moving it to a new object.

With this approach, pseudosamples retain the linkage present in the original sample and the autocorrelation among the class labels. In addition, any degree disparity present in the data is preserved. Randomizing attribute vectors destroys the correlation between the attributes and the class label in pseudosamples, thus making them appropriately conform to the null hypothesis—that there is no correlation between the attribute values and the class label. In addition to adjusting for the effects of autocorrelation and degree disparity, randomization tests adjust for the effects of attribute selection errors due to multiple comparisons [6].

Randomization tests provide one method of hypothesis testing to adjust for the effects of degree disparity, linkage and autocorrelation on parameter estimates. An alternative is to estimate the parameters (feature scores) more accurately. We have explored adjustments to chi-square calculations that "factor out" the bias introduced by degree disparity [8]. However, adjustments are hard to calculate for some combinations of aggregation function and attribute distributions. Also, we do not yet know how to adjust for the high variance associated with objects having high linkage and autocorrelation. For now, we use randomization tests to make accurate assessments of significance when choosing among features with differing levels of bias and variance. This approach facilitates unbiased feature selection and prevents excessive tree structure.

4. EVALUATION

The first data set is drawn from the Internet Movie Database (IMDb) (www.imdb.com). We gathered a sample of all movies released in the United States from 1996 to 2001, with opening weekend receipt information. Our randomization procedure is limited to data sets with non-zero degree so we selected the set of 1364 movies that had at least one actor, director, studio and producer. In addition to these movies, the collection contains all associated actors, directors, producers, and studios. We discretized movie receipts so that a positive class label indicates a movie earned more than $2 million in its opening-weekend (P(+)=0.45).

Our first task used a modified version of the IMDb data set where the only features correlated with the class label are DEGREE features. Movies with a positive class label have higher degree with respect to actors and producers, though no significant difference in director degree or studio degree. On each actor, director, producer, and studio object we added 6 random attributes (3 discrete and 3 continuous). Discrete attributes were drawn from a uniform distribution of ten values; continuous attribute values were drawn from a uniform distribution in the range [0,10]. The model could consider four degree features, one for each type of object linked to the movie.

The second task also used the IMDb dataset, but used both the structure and the attributes in the original data. The models had eight attributes available for classification, such as the genre of the movie and the year of a producer's first film.

The third task used a data set drawn from Cora, a database of computer science research papers extracted automatically from the web using machine learning techniques [12]. We selected the set of 1,511 machine-learning papers that had at least one author, reference and journal. In addition to these papers, the collection contains all associated authors, references, and journals. The class label indicates whether a particular paper

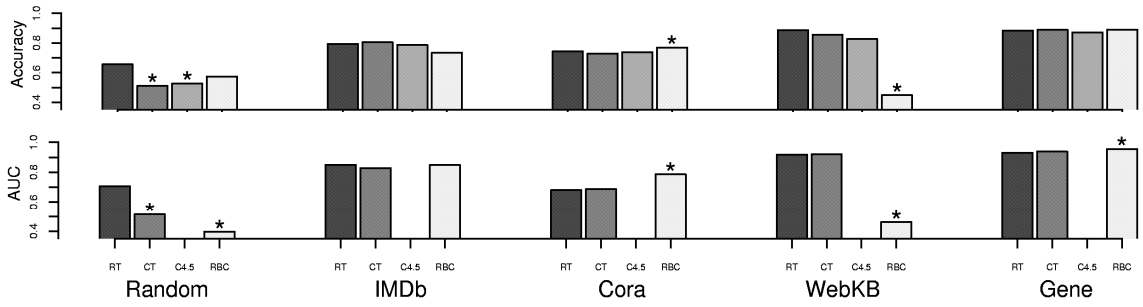


Figure 3: Accuracy, AUC for the four models across the various classification tasks.

was assigned the topic of "neural networks" (P(+)=0.32). The models had 15 attributes available for classification, including a reference's high-level topic (e.g. artificial intelligence) and an author's number of publications.

The fourth task used a relational data set containing information about the yeast genome at the gene and the protein level (www.cs.wisc.edu/~dpage/kddcup2001/). The data set contains information about 1,243 genes and 1,734 interactions among their associated proteins. The class label indicates whether or not a gene is located in the nucleus (P(+)=0.44). The models used seven attributes for prediction, including gene phenotype, complex, and interaction type.

The fifth classification task used the WebKB data set described in section 2. We selected the set of 910 course, faculty, project, staff, and student pages that have at least one in-link and out-link. The collection also includes all pages that link to/from these pages, a total of 3,877 web pages. The class label indicated whether a page is a student page (P(+)=0.51). The models used 10 attributes for prediction, including attributes such as the URL path and host, and attributes counting the number of in links and out links (directional degree) of each linked page.

For each of the tasks, we tested four models. The first model is an RPT that uses conventional chi-square significance tests (CTs) to evaluate feature splits. The second model is an RPT that uses randomization tests (RTs) to measure significance. In order to separate the effects of the randomization tests from the rest of the RPT learning algorithm we included a standard tree learner—we generated propositional data sets containing all the binary features considered by the RPT and applied the C4.5 algorithm [15]. As a baseline, we also applied a non-selective relational Bayes classifier (RBC) [13].

To evaluate the models, we measured accuracy and area under the ROC curve (AUC). Because the C4.5 models return classifications only (not probability estimates) we could not calculate AUC for the C4.5 models. The experiments all used ten-fold cross-validation, except in the case of WebKB, which used leave-one-department-out cross-validation. Due to the high degree of studio objects in the IMDb data set, we used stratified sampling by studios to create the test sets. We randomly sampled studios from three sets (studios with high, medium, and low degree) and all their associated movies, thus creating test sets of roughly equal proportion.

To examine the effects of feature selection we recorded the number of tree nodes that used features based only on relational structure (DEGREE features) and the overall number of nodes. We weighted each count based on the proportion of training instances that traveled through a given node.

Figure 3 shows accuracy and AUC results for each of the four models on the five classification tasks. We used two-tailed, paired t-tests to assess the significance of the results obtained from the cross-validation trials. The null hypothesis is that there is no difference between two approaches. We compared RTs to each of the other three approaches (CTs, C4.5, RBCs). An asterisk above the model indicates a significantly different performance from the RTs.

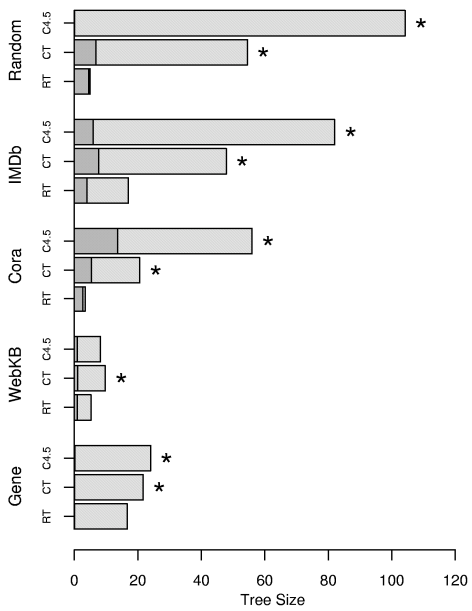


Figure 4: Tree size and weighted proportion of degree features.

The results from the IMDb data set with random attributes (Random in Figure 3 and 4) support three claims. First, RPTs using randomization tests (RTs) can adjust for linkage and autocorrelation and build more accurate models. RTs perform significantly better than the other three models. These results indicate the potential for biased models to select attributes that hinder performance. The lower performance is due to selection of random attributes on studio objects. Features involving these attributes have high variance because of the low effective sample size of studios. If objects with high linkage and autocorrelation, such as studios, are present in the data, attributes of those objects will be selected even if they have no correlation with the class. Second, aggregation functions can

cause misleading correlations in the presence of degree disparity. In the IMDb data set with random attributes, the only predictive structure is the degree disparity of "actor" and "producer" objects. Approximately 4/5 of the features in trees built with conventional tests (CTs) consisted of features derived from random attributes that served as surrogates for degree. Third, the results from the Random data set show that models that do not adjust for these biases can add unnecessary complexity. Trees built with conventional tests were, on average, an order of magnitude larger than the size of the trees built with randomization tests.

The results from experiments on real data show that RTs achieve comparable performance to CTs and C4.5 models, both in accuracy and AUC. However, the trees had radically different structure. Figure 4 summarizes the features used in RPT trees built with conventional tests and randomization tests, as well as trees built with C4.5. Note that in data sets where degree disparity is predictive (IMDb, and Cora), RTs give higher weight to degree features.

5. CONCLUSIONS

We have shown that it is possible to extend conventional probability estimation tree algorithms to work with relational data. The RPT models built using randomization tests performs equivalently to RPT models using conventional hypothesis tests but the trees are significantly smaller. This supports our claim that common characteristics of relational can bias feature selection and result in excessively complex models. Randomization tests adjust for both the increased bias due to degree disparity and the increased variance due to linkage and autocorrelation. To our knowledge, no other relational learning algorithms adjust for these biases.

Models that are not selective (e.g. RBC) do not suffer from these biases. This can result in significantly better models, but we then lose the interpretability of the selective models. RBCs exhibited significantly lower performance on datasets where degree was the only feature correlated with the class (Random, WebKB). More work needs to be done to explore the situations in which RBC performance is distinct from RPT. We may be able to combine the strengths of RPT feature construction and selection methods with the low variance parameter estimates of RBC models.

Future work will investigate further enrichments to the RPT model and algorithm. Currently, the algorithm ranks by score to select features for inclusion in the tree but this may not be the best approach for biased feature scores. We are currently exploring alternate approaches to ranking and selection that standardize the observed scores with the sampling distributions obtained from randomization tests.

Also, extending the algorithm to consider multiway feature splits and alternative methods of modeling continuous attributes should improve performance. We will conduct a series of more focused ablation studies to determine which characteristics of the algorithm are most beneficial and to further understand the complexities of relational data and their effect on modeling choices.

6. ACKNOWLEDGMENTS

Helpful comments and assistance were provided by Amy McGovern and Hannah Blau. This research is supported by DARPA and NSF under contract numbers F30602-01-2-0566 and EIA9983215, respectively. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements either expressed or implied, of DARPA, NSF, or the U.S. Government.

7. REFERENCES

[1] A Blockeel, H., and De Raedt, L. Top-down induction of first-order logical decision trees. Artificial Intelligence, 101: 285-297, 1998.

[2] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam and S. Slattery. Learning to Extract Symbolic Knowledge from the World Wide Web. Proc. of 15th National Conference on Artificial Intelligence. 509-516, 1998.

[3] E. Edgington. *Randomization Tests*. New York: Marcel Dekker, 1980.

[4] S. Dzeroski and N. Lavrac, editors. *Relational Data Mining*. Springer-Verlag, 2001.

[5] L. Getoor, N. Friedman, D. Koller, and A. Pfeffer. Learning probabilistic relational models. Proc. of the 16th Intl Joint Conference on Artificial Intelligence. 1300-1309, 1999.

[6] D. Jensen and P. Cohen. Multiple comparisons in induction algorithms. Machine Learning 38(3):309-338, 2000.

[7] D. Jensen and J. Neville. Linkage and autocorrelation cause feature selection bias in relational learning. In Proc. of the 19th Intl Conference on Machine Learning. Morgan Kaufmann. 259-266, 2002.

[8] D. Jensen, J. Neville and M. Hay. Avoiding bias when aggregating relational data with degree disparity. Proc. of the 20th Intl Joint Conf. on Machine Learning, to appear.

[9] A. Knobbe, A. Siebes and D. Van der Wallen. Multi-relational decision tree induction. Proc. of the 3rd European Conference on Principles & Practice of KDD. 378-383, 1999.

[10] S. Kramer. Structural regression trees. Proc. of the 13th National Conference on Artificial Intelligence, 812-819, 1996.

[11] S. Kramer, B. Pfahringer, and C. Helma. Stochastic propositionalization of non-determinate background knowledge. Proc. of the 8th Intl Workshop on Inductive Logic Programming. Springer Verlag, 80-94, 1998.

[12] A. McCallum, K. Nigam, J. Rennie, & K. Seymore. A machine learning approach to building domain-specific search engines. Proc. of the 16th Intl Joint Conference on Artificial Intelligence, 662-667, 1999.

[13] J. Neville, D. Jensen, B. Gallagher and R. Fairgrieve. Simple estimators for relational Bayesian classifiers. University of Massachusetts Amherst, Tech Report 03-04, 2003.

[14] F. Provost and P. Domingos. Well-trained PETs: Improving probability estimation trees. CDER Working Paper #00-04-IS, Stern School of Business, NYU, 2000.

[15] J.R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.