STUDY ON TYPES OF RELATIONAL MODELS AND PREVIOUS PERFORMANCE COMPARISONS

ADIL ZAMIL, #DR. RAJESH PATHAK

#HOD Deptt. Of Computer Science, & Engg.

GNIT Institute of Technology,

Gautam Budh Nagar, U.P.

ABSTRACT

The learning task was to predict gene localization in the cell. There are 15 locations ranging from mitochondria to plasma membrane, with a default error rate of 0.57. In addition to gene location, each gene has 13 boolean attributes indicating gene function. Each gene may have as many as six functions. For non-collective models, we used relational Bayesian classifiers (RBCs) [13] to predict gene location given the function attributes. The Intrinsic model considered the 13 function attributes on the genes themselves. The R1 model added another 13 function attributes for genes one link away (through interactions) for a total of 26 attributes. The R2 model then added another 13 attributes for genes two links away, for a total of 39 attributes. For collective models, we used relational dependency networks (RDNs) with RBCs to represent the component conditional probability distributions. The CI model considered the location attribute of genes one link away, in addition to the 13 function attributes of the genes in isolation. The RCI model added in the 13 function attributes for genes one link away for a total of 27 attributes. The RDNs used 250 Gibbs iterations and all models used Laplace correction for zero-values.

To compare the five approaches, we evaluated zero-one loss over ten-fold cross validation trials. We report average error over the ten folds and use two-tailed, paired t-tests to assess the significance of the results.

Key words: *localization, attributes, dependency, isolation, significance.*

INTRODUCTION

The process of rollout is sufficiently general that it can be applied to a wide variety of graphical models for relational data. Each type of model imposes different constraints on the Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

For example, consider a data graph with a regular structure of n2 objects arranged in an n × n lattice. Each object in the lattice links to each of its immediate neighbors. With the exception of objects along the outer boundary, each object links to four others positioned above, below, left, and right. All links are undirected. Each object is characterized by a set of variables that includes a single probabilistic variable C (a class label) and several other variables Ai (one or more attributes) whose values are known with certainty. The task is to construct a joint model of the probability distribution over all the values of the class labels.

REVIEW OF LITERATURE

Intrinsic — For a given object, the Intrinsic model estimates the joint distribution of the class label and attributes on that object. It assumes that objects are i.i.d., and thus corresponds to traditional models used in many knowledge discovery applications. The model is depicted graphically in figure 1a using the plate notation common in the graphical modeling community. The inner box, along with the edge connecting A and C, indicates that m different versions of node A (corresponding to m attributes Ai) each depend on C. The outer box indicates that the model creates N different versions of the network, each containing a single node C. For example, this model would indicate that the words on a web page (the attributes Ai) depend only on the topic of that page (C) and are independent of the topic and words on any other page.

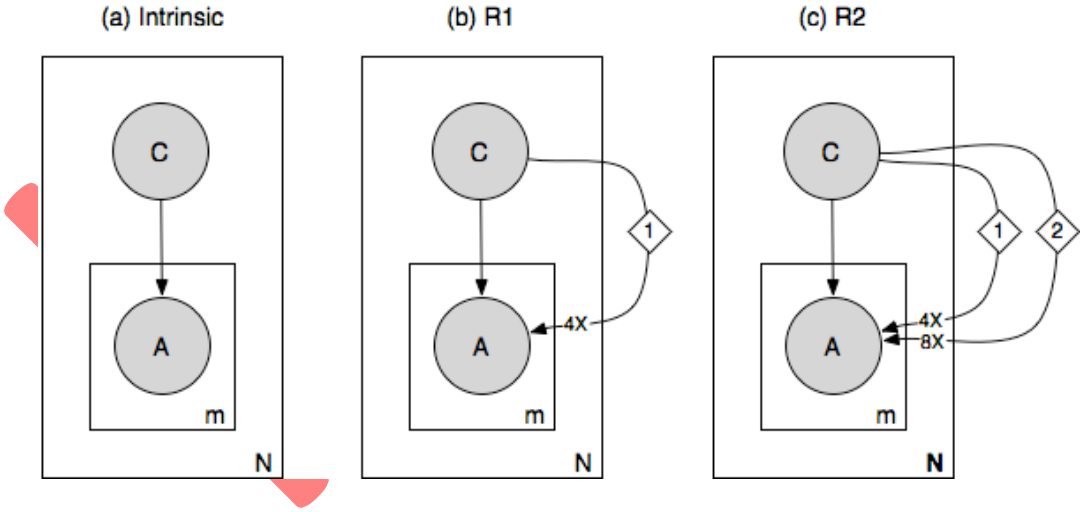

Figure 3.16: Relational models

Relational 1 (R1) — The model R1 is a simple relational model indicating that the attributes of an object depend on the class label of that object as well as the class labels of objects one link away. Figure 1b shows this model using a modified plate notation in which the integer within the diamond-shaped annotation (“1”) indicates the graph distance of neighboring objects and the multiplier on the edge (“4x”) indicates the number of such neighboring objects. The path of the annotated edge outside the outer box emphasizes the dependence on the class labels of adjoining

objects. Here, the value of each Ai depends on five different parents C, four of which are from neighboring objects. For example, this model would indicate that the words on a web page depend on the topic of that page and the topics of four adjoining pages.

Relational 2 (R2) — A somewhat more complex relational model R2 indicates that the attributes of an object depend on the class label of that object and the class labels of objects up to two links away (Figure 1c).

None of these three models allows interdependence among class labels, which is a prerequisite for collective inference. We examine two additional models that do allow for such dependence:

Collective Inference (CI) — The model CI, shown in figure 2a, provides the same type of dependence as Intrinsic, but adds dependence between the class label of an object and the class label of adjoining objects. This is equivalent to specifying that the topics of web pages depend on those of adjoining pages (and also determine the words on the page).

Relational Collective Inference (RCI) — The model RCI, extends the R1 model by adding dependence among class labels of neighboring objects one link away.

These models are relatively simple because the example data are highly regular and contain only a single object and link type. More heterogeneous data might require models with longer and more complex paths among objects. For example, paths connecting auto correlated objects might pass through one or more intervening objects of specified types. However, the simplicity of this example allows us to focus on the critical aspects of learning and inference in relational data. Collective inference has been a small but active area of research in relational learning for at least six years, since the publication of Chakrabarti, Dom, and Indyk's detailed study of hypertext categorization strategies. Several more recent studies of collective inference have extended and broadened this work . Finally, some work has extended the basic paradigm of collective inference to incorporate selecting among a range of possible actions. For example, Domingos and Richardson's work on mining the network value of customers incorporates collective inference into a larger approach to "viral marketing". Table 1 summarizes the types of models evaluated in seven key papers.

Many studies of collective inference have reported large reductions in error when the method is applied. For example, Chakrabarti et al report large reductions in classification error, including one drop in error of over 70% (from 68% to 21%). In previous work, two of the authors reported significant accuracy gains from a relatively simple technique for collective inference . Macskassy and Provost show how models that consider only autocorrelation in class labels (equivalent to CI without attributes) can perform very well when only a small fraction of the class labels are known .

Several studies have also pointed out that collective inference of various types can also reduce accuracy. For example, Chakrabarti et al. [1] discuss an experiment where including relational information about web pages actually reduces accuracy. They hypothesize that the additional features meant that the learning and inference scheme was "overwhelmed by the signal to noise ratio".

Based on these results, it appears clear that collective inference is capable of significantly improving probabilistic inferences in relational data. Important questions remain, however: why and under what circumstances does collective inference improve the accuracy of relational models?

One reasonable explanation is that the power of collective inference lies merely in the larger feature-space provided by models such as CI. These models consider features that their less expressive cousins (e.g., R1) do not. In experiments below, we will show that this explanation is inadequate to explain the power of collective inference.

Instead, we show that methods for collective inference benefit from a clever factoring of the space of dependencies. The models CI and RCI have substantially smaller parameter spaces than the model R2, yet they can benefit from information propagated from outside of their local neighborhood. Predictions about the class label C on other objects essentially "bundle information" about the graph beyond the immediate neighborhood. In addition, collective models can make use of known class labels (e.g., known topics of web pages) to improve inferences about unknown labels. This provides a new, and often highly reliable, additional feature for learning and inference.

This increased representational power is purchased with only an incremental increase in the parameter space. In this way, CI and RCI emulate other robust techniques such as simple Bayesian classifiers and linear regression models. Even when their assumptions are violated, CI and RCI often perform well.

MATERIAL AND METHOD

To evaluate different models and inference methods, we conducted experiments with both real and synthetic data.

Yeast Protein Experiments

Our empirical experiments considered relational data about the yeast genome, containing information about 1,243 genes and 1,734 interactions among their associated proteins (http://www.cs.wisc.edu/~dpage/kddcup2001/). Both gene location and function are auto-

correlated in this dataset [11] so we expect it to be a good test bed for investigating the relative performance of the various relational models.

The learning task was to predict gene localization in the cell. There are 15 locations ranging from mitochondria to plasma membrane, with a default error rate of 0.57. In addition to gene location, each gene has 13 boolean attributes indicating gene function. Each gene may have as many as six functions. For non-collective models, we used relational Bayesian classifiers (RBCs) [13] to predict gene location given the function attributes. The Intrinsic model considered the 13 function attributes on the genes themselves. The R1 model added another 13 function attributes for genes one link away (through interactions) for a total of 26 attributes. The R2 model then added another 13 attributes for genes two links away, for a total of 39 attributes. For collective models, we used relational dependency networks (RDNs) with RBCs to represent the component conditional probability distributions. The CI model considered the location attribute of genes one link away, in addition to the 13 function attributes of the genes in isolation. The RCI model added in the 13 function attributes for genes one link away for a total of 27 attributes. The RDNs used 250 Gibbs iterations and all models used Laplace correction for zero-values.

To compare the five approaches, we evaluated zero-one loss over ten-fold cross validation trials. We report average error over the ten folds and use two-tailed, paired t-tests to assess the significance of the results.

Synthetic Experiments

To generate synthetic data, we extended the example presented in section 2.1. We generated data with a regular two dimensional lattice structure. The first and last two rows and columns make up the "frame" of the lattice. Objects in the frame are not used to train models, and objects in the frame are not used for loss estimates, although inference is performed over all objects in the lattice, including the frame. Thus, training or test sets of size S2 correspond to a lattice of (S+4) × (S+4) objects, and models are trained or evaluated on the S2 objects in the core of the lattice.

Each object in a given dataset contains the same set of attributes. In every dataset, objects contain a class label C and a single attribute A1, that is correlated with C. Depending on dataset generation parameters, objects may also contain up to 14 additional attributes, none of which are correlated with C. We generated the values of attributes and class labels in two ways, which we label "relational" and "collective". Both use parameters given in Table 2. For collective data generation, we begin by assigning each object in the lattice an initial class label with P(C=1) = 0.5. We then perform Gibbs sampling over the entire lattice. The class labels assigned to each object after 200 iterations are used as the final labels. To assign class labels during Gibbs sampling, we use a manually specified model that assigns class labels to each object based on the class values of neighboring objects one link away. The parameters of this model are varied to

produce different levels of autocorrelation among neighboring class labels. Once class labels are assigned, a value for the A1 attribute is randomly drawn from a distribution conditioned on the class label of the object — derived from the P(C|A1) and P(A1) data generation parameters. Finally, random values are assigned to all other attributes with P(Ai) = 0.5. Once a dataset is generated, we measure the proportion of objects with positive class labels, and any dataset with a value outside the range [0.4, 0.6] is discarded and replaced with a new dataset. This ensures consistency in P(C) across datasets and reduces variance in estimated model performance.

Table 1 Data Generation Parameters

Bold numbers in the value column indicate default values.

Name	Description	Values
TrainSize	Number of core objects in training set.	25, 49, 100, 225, 484, **1024**, 5041
TestSize	Number of core objects in test set.	**484**
NumAttrs	Number of Attributes	1,**3**,5,10,15
PropLabel	Proportion of objects with known class labels	**0.0**, 0.1, 0.3, 0.5, 0.7, 0.9
Autocorr	Autocorrelation of class labels for neighboring object *(see text)*.	0.01, 0.27, **0.49**, 0.75, 0.98
$P(A_1)$	Prior probability of $P(A_1=1)$.	0.1, **0.3**, 0.5
$P(C\|A_1)$.	Conditional probability of $P(C=1\|A_1=1)$.	0.6, 0.75, **0.9**

For relational data generation, we begin by training the parameters of an R2 model on a large dataset consisting of 100 lattices of 1000 objects each. The attributes and class labels on the objects of each lattice are determined by the collective data generation method described above. We also train a univariate model of P(Ai) for each attribute Ai. We create a lattice of objects in the usual way, assign attribute values randomly to each object based on the learned model of P(Ai), and assign class labels to each object based on the attribute values of itself and all neighboring objects up to two links away using the learned R2 model.

We measured bias and variance for each model using the decomposition defined for squared-loss by Domingos . Loss is decomposed into three factors: bias, variance and noise.

CONCLUSION

Although calculation of variance is straightforward for relational data, calculation of bias is not. Fortunately for the synthetic data experiments, we know the probabilities from the generative model and can use these as the optimal predictions. Bias and variance estimates are

calculated for each test example using 10 different training sets and averaged over the entire test set. This was repeated for 20 test sets to calculate average test set bias and variance.

REFERENCES

- *Abdul-Rahman and S. Hailes, "A distributed trust model," in Proceedings of New Security Paradigms Workshop, 1997, pp. 48–60.*
- *R. Agrawal, S. Dar, and H. V. Jagadish, "Direct transitive closure algorithms: Design and performance evaluation." ACM Transactions on Database Systems, vol. 15, no. 3, pp. 427– 458, 1990.*
- *R. Agrawal and H. V. Jagadish, "Multiprocessor transitive closure algorithms." in Proceedings of the International Symposium on Databases in Parallel and Distributed Systems, Austin, TX, 1988, pp. 56–66.*
- *Agresti, Categorical Data Analysis. New York, NY: Wiley, 1990.*
- *V. Aho, J. E. Hopcroft, and J. D. Ullman, The Design and Analysis of Computer Algorithms. Reading, MA: Addison-Wesley, 1974.*
- *Anderson, P. Domingos, and D. Weld, "Relational Markov models and their application to adaptive Web navigation," in Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Edmonton, Canada: ACM Press, 2002, pp. 143–152.*
- *F. Bacchus, Representing and Reasoning with Probabilistic Knowledge. Cambridge, MA: MIT Press, 1990.*
- *F. Bancilhon, "Naive evaluation of recursively defined relations." in On Knowledge Base Management Systems (Islamorada), 1985, pp. 165–178.*
- *R. Bellman and M. Giertz, "On the analytic formalism of the theory of fuzzy sets." Information Sciences, vol. 5, pp. 149–156, 1973.*
- *T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," Scientific American, vol. 284, no. 5, pp. 34–43, 2001.*

- *J. Besag, "Statistical analysis of non-lattice data," The Statistician, vol. 24, pp. 179–195, 1975.*
- *M. Blaze, J. Feigenbaum, and J. Lacy, "Decentralized trust management," in Proceedings of the 1996 IEEE Symposium on Security and Privacy, Oakland, CA, 1996, pp. 164–173.*
- *L. Breiman, "Bagging predictors," Machine Learning, vol. 24, pp. 93–140, 1996.*
- *S. Brin and L. Page, "The anatomy of a large-scale hypertextual Web search engine," in Proceedings of the Seventh International World Wide Web Conference. Brisbane, Australia: Elsevier, 1998.*
- *Carre, Graphs and Networks. Oxford: Claredon Press, 1978.*
- *S. Chakrabarti, B. Dom, D. Gibson, J. Kleinberg, P. Raghavan, and S. Rajagopalan, "Automatic resource compilation by analyzing hyperlink structure and associated text," in Proceedings of the Seventh International World Wide Web Conference. Brisbane, Australia: Elsevier, 1998, pp. 65–74.*
-
- *M. Chickering and D. Heckerman, "Efficient approximations for the marginal likelihood of Bayesian networks with hidden variables," Machine Learning, vol. 29, pp. 181–29, 1997.*
- *K. Chow and C. N. Liu, "Approximating discrete probability distributions with dependence trees," IEEE Transactions on Information Theory, vol. 14, pp. 462–467, 1968.*
- *S. A. Cook, "The complexity of theorem-proving procedures," in Proceedings of the Third Annual ACM Symposium on Theory of COmputing, 1971, pp. 151–158. [Online]. Available: http://theory.lcs.mit.edu/ dmjones/STOC/stoc71.html*
- *G. F. Cooper, "The computational complexity of probabilistic inference using Bayesian belief networks." Artificial Intelligence, vol. 42, no. 2-3, pp. 393–405, 1990.*
- *V. S. Costa, D. Page, M. Qazi, , and J. Cussens, "CLP(BN): Constraint logic programming for probabilistic knowledge," in Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence. Acapulco, Mexico: Morgan Kaufmann, 2003, pp. 517–524.*