Compiling Relational Database Schemata into Probabilistic Graphical Models

Sameer Singh
University of Massachusetts,
Amherst MA, USA
sameer@cs.umass.edu

Thore Graepel
Microsoft Research,
Cambridge, United Kingdom
thoreg@microsoft.com

1 Introduction

A majority of scientific and commercial data is stored in relational databases. Probabilistic models over such datasets would allow probabilistic queries, error checking, and inference of missing values, but to this day machine learning expertise is required to construct accurate models. Fortunately, current probabilistic programming tools ease the task of constructing such models [1, 2, 3, 4, 5, 6] and work in statistical relational learning has focused on making it even easier to define models specific to relational data [7, 8, 9, 10]. However, within these frameworks the user still needs to specify all the probabilistic dependencies in the data, requiring a level of expertise in probability and statistics that domain experts often do not have, thus severely restricting the practical applications of such techniques. On the other hand, domain experts do spend considerable effort and expertise in designing the database schemata used to represent their data, providing type information for table columns and foreign key relations to specify dependencies.

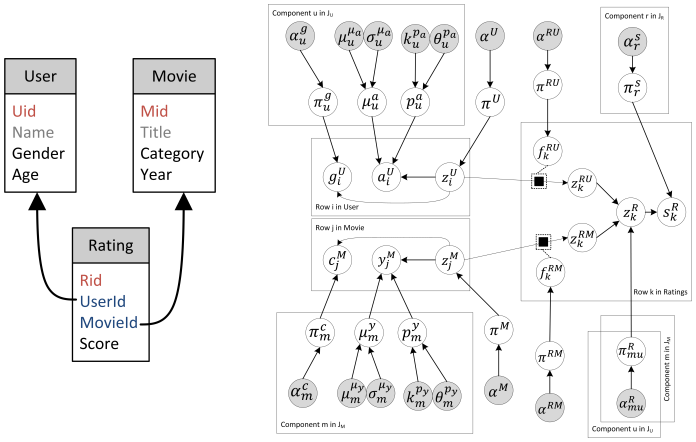
In this work, we view relational database schemata as *programs* that describe probabilistic dependencies that exist in the data. The goal is to simplify the task of model construction for the domain expert and to be able to construct probabilistic models automatically for a large number of existing databases without manual intervention. Using a given schema, a customized fully-Bayesian, generative graphical model is generated. Each table is modeled with a mixture model, along with edges that model dependencies between these table models according to their foreign key relationships. This underlying model is similar to relational latent variable models [11, 12], but extends them by incorporating referential uncertainty (foreign key prediction) and using a parametric approach for real-world tractability. We use variational message passing inference to learn the parameters of the model, allowing inference of missing values and probabilistic relational queries. Experiments demonstrate the accuracy and scalability of the approach using synthetic and real world data.

2 Compiling a Graphical Model from the Schema

In this section, we describe how, given a database schema, we create a Bayesian graphical model and perform inference with minimal manual intervention.

Single Table: We begin the description of the model by examining a schema that contains a single table A with attributes $\mathbf{x}^A$. We employ a *mixture model* for each table, wherein a mixture component is used to generate all the attributes $\mathbf{x}^A(i)$ of row i, and $z^A(i)$ is a latent variable that indicates which component to use for the row. The distribution used to generate each attribute $x_k^A(i)$ depends on the data type of the attribute; Gaussian for real-valued, Discrete for categorical-valued, and Bernoulli for Boolean-valued attributes, each distribution is latent and generated from its observed prior. The component indicators $\mathbf{z}^A$ are generated from a latent discrete distribution π^A, with its observed prior.

Foreign Component Link: Consider a table B that contains a single foreign key attribute to another table A. The data attributes of both tables A and B, $\mathbf{x}^A$ and $\mathbf{x}^B$ respectively, are modeled as described above. The foreign key attribute for each row i in table B is represented by f_i^B, which



(a) Schema (b) Generated Graphical Model

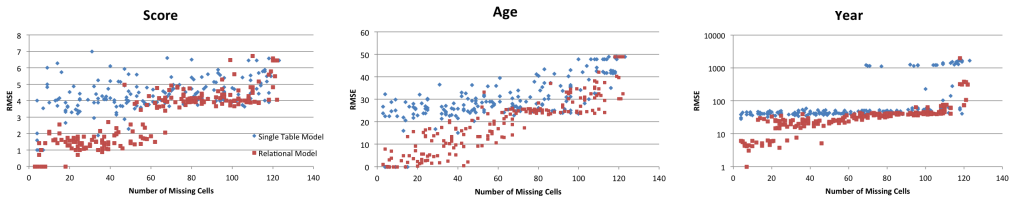
Figure 1: **User-Movie-Rating Schema:** Example schema (a) consisting of movie ratings by users. Attributes shown in gray and primary keys (red) are not modeled. The data attributes (black) are represented by variables g^U, a^U, c^M, y^M, s^R in the model (b). The foreign key relations (blue) are modeled using f_k^{RM} and f_k^{RU}.

indexes into a row in table A. Since we want the links between rows to reflect the dependencies between the tables, we make the component indicator $z^B(i)$ dependent on the component indicator of the foreign row it links to ($z^A(f_i^B)$). Specifically, instead of using a single distribution for z^B, we use as many discrete distributions as the number of components in table A (cardinality of z^A), and *select* the corresponding distribution using Gates [13]: $z^A(i) \leftarrow \pi^B[z^A(f_i^B)]$. We also model the uncertainty in foreign keys f^B as discrete distributions, which allows prediction of missing foreign links. This idea is easily generalized to tables with an arbitrary number of foreign keys by using additional number of discrete distributions π^B.

Database Schema: An input database schema consists of a number of tables and their attributes, and the foreign key relations that form a directed, acyclic graph. We can use the building blocks above to iteratively construct a model over a schema by applying the single-table model for the tables without any foreign keys, and using the foreign links to define the dependencies between the component indicators for tables with foreign keys. For example, consider a simple schema consisting of three tables shown in Figure 1a. Figure 1b shows the generated model, where the model for User and Movie tables is similar to a regular mixture model, while the Rating table consists of additional variables and edges for foreign links, and dependencies of the component indicators across tables.

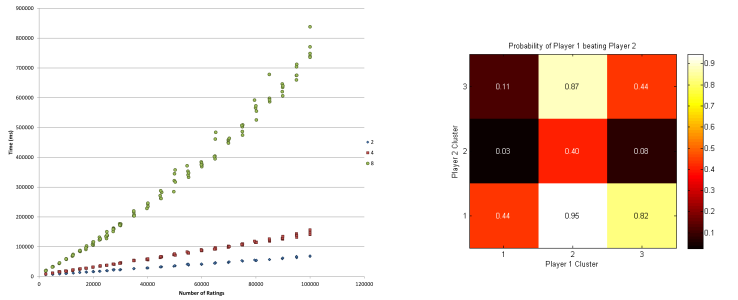
Model Assumptions: As described, a number of priors in the models need to be specified. Most hyper parameters can be set to be *uninformative*, however specifying the number of components in each table is crucial. Too many components result in slower inference, while too few components produce inaccurate models. Non-parametric approaches such as [11, 12] are much slower in practice, however recent work suggests that exploiting conditional exchangeable properties of our data may be useful [14]. Another assumption in the generated model is that the attributes of the row are independently generated given the component, which often does not hold in practice. An alternative is to explore the range of independent to fully-correlated attributes, using cross-cutting models [15].

Inference: Inference on the resulting model is performed using variational message passing [16], as implemented in Infer.NET [6]. Since the model contains strong dependencies and deterministic factors (*gates*), inference approaches such as Gibbs sampling are not practical when applied directly. During training, in which we learn the parameters of the model and use it to predict missing values in the database, the complexity of message passing is linear in the number of rows (when all the foreign keys are observed). The approach also supports *probabilistic queries* over the trained model; queries take the form of a small set of records with missing entries. Inference is used to predict marginal posterior belief distributions over these entries. The inference for querying is also efficient; linear in the size of the query if the foreign keys are observed.



(a) Rating Score of a Movie (b) Age of a User (c) Release year of a Movie

Figure 2: **Results on Synthetic Data:** Comparison of the relational model (*in red*) with a single table model generated using a join over the foreign keys (*in blue*) using RMSE on three real-valued attributes.



(a) Varying the number of rows (b) Clustering of Xbox Players

Figure 3: **Experiments on Real-World Data:** (a) MovieLens 100k, and (b) Xbox Head-on-Head data

3 Experiments

In this section, we present preliminary experiments that evaluate the accuracy, clustering quality, and the scalability of the schema-based probabilistic models.

Synthetic User-Movies-Ratings Data: One typical approach to modeling values in a relational database is to perform a join over all the tables, and to use a single-table mixture model on the resulting table. Unlike in our relational model, the dependencies across rows are lost in the join operation. To evaluate this effect on accuracy, we compare the two models by treating a proportion of cells as missing (before performing the join). We create synthetic data for the schema in Figure 1a, and perform inference to predict the values of the missing cells. The error of the predictions for the real-valued attributes is shown in Figure 2, demonstrating that the schema-based probabilistic model is consistently more accurate and more robust in the presence of missing cells. In particular, the rating scores are accurate even when half of the values are missing.

MovieLens dataset: We evaluate scalability on the MovieLens dataset. The schema of the data is similar to User-Movie-Rating database, but includes a few more attributes. The data consists of 943 users, $1,682$ movies, and $100,000$ ratings. Since the number of rows in the *leaf* table is usually much higher than in other tables, we examine the scalability in terms of its size. We run a fixed number of iterations of inference as we vary the number of ratings, and examine the running time. The results, shown in Figure 3a, show a linear trend for the running time. Further, the figure also shows the increase in running time as the number of components in each table is increased.

TrueSkill Dataset: To perform a qualitative evaluation of the clustering of rows produced by our model, we use the Head-to-Head games data from Xbox matches, as used in Herbrich et al. [17]. The data consists of a table of player Ids (with no other attributes), and a table of match results that consists of foreign key attributes for two players, along with a Boolean result attribute that is true if the first player was the winner. The model generated for this data assigns each player row to one of three components, shown in Figure 3b. We also include the average result for each pair of clusters. Note that the three clusters correspond to bad, excellent, and good players respectively, demonstrating that the latent clustering can be used to predict the skills of players without making any further domain-specific modeling assumptions.

4 Conclusion and Future Work

We suggest automatically compiling probabilistic graphical models from database schemata. This approach allows us to make use of the domain knowledge that went into the design of the database schema and potentially makes probabilistic graphical models directly available for a large fraction of the world's data. Inference on the compiled Bayesian model allows the prediction of the values of missing cells in the database, detect outliers, visualize clustering of the data, and to answer basic probabilistic relational queries. We evaluated the accuracy, the clustering quality, and the scalability of our approach using a combination of synthetic and real world data, and found that the schema-based graphical models lead to interesting results.

This work is very much in progress, and there are a number of avenues for future directions. We would like to explore computationally efficient extensions to the model that are non-parametric, for example models similar to [11, 12], and using the ideas presented in [14]. We also want to investigate the utility of other inference techniques, such as Gibbs sampling and variational Bayes methods. Further work on evaluation of the approach on more real-world datasets is also of interest.

Acknowledgments

The authors would like to thank Lucas Bordeaux, Andy Gordon, Tom Minka, and John Guiver for valuable discussions and insights. We are also grateful for the feedback from the anonymous reviewers of the NIPS 2012 workshop on probabilistic programming.

References

[1] Avi Pfeffer. Ibal: A probabilistic rational programming language. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 733–740, 2001.

[2] Avi Pfeffer. Figaro: An Object-Oriented Probabilistic Programming Language. Technical report, Charles River Analytics, 2009.

[3] Brian Milch. *Probabilistic Models with Unknown Objects*. PhD thesis, University of California, Berkeley, 2006.

[4] Noah D. Goodman, Vikash K. Mansinghka, Daniel Roy, Keith Bonawitz, and Joshua B. Tenenbaum. Church: a language for generative models. In *Uncertainty in Artificial Intelligence (UAI)*, 2008.

[5] Andrew McCallum, Karl Schultz, and Sameer Singh. FACTORIE: Probabilistic programming via imperatively defined factor graphs. In *Neural Information Processing Systems (NIPS)*, 2009.

[6] Tom Minka, John M. Winn, John P. Guiver, and David A. Knowles. Infer.NET 2.4, 2010. Microsoft Research Cambridge. http://research.microsoft.com/infernet.

[7] Nir Friedman, Lise Getoor, Daphne Koller, and Avi Pfeffer. Learning relational data mining. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1300–09, 1999.

[8] Ben Taskar, Abbeel Pieter, and Daphne Koller. Discriminative probabilistic models for relational data. In *Uncertainty in Artificial Intelligence (UAI)*, 2002.

[9] David Heckerman, Christopher Meek, and Daphne Koller. Probabilistic models for relational data. Technical Report MSR-TR-2004-30, Microsoft Research, 2004.

[10] Jennifer Neville and David Jensen. Relational dependency networks. *Journal of Machine Learning Research*, 8:653–692, May 2007. ISSN 1532-4435.

[11] Zhao Xu, Volker Tresp, Kai Yu, and Hans-Peter Kriegel. Infinite hidden relational models. In *Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 544–551, Arlington, Virginia, 2006. AUAI Press.

[12] Charles Kemp, Joshua B. Tenenbaum, Thomas L. Griffiths, Takeshi Yamada, and Naonori Ueda. Learning systems of concepts with an infinite relational model. In *American Association of Artificial intelligence (AAAI)*, AAAI'06, pages 381–388. AAAI Press, 2006. ISBN 978-1-57735-281-5.

[13] Tom Minka and John M. Winn. Gates. In *Neural Information Processing Systems (NIPS)*, pages 1073–1080, 2008.

[14] James Robert Lloyd, Peter Orbanz, Zoubin Ghahramani, and Daniel M. Roy. Random function priors for exchangeable arrays with applications to graphs and relational data. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[15] Patrick Shafto, Charles Kemp, Vikash Mansinghka, Matthew Gordon, and Joshua B. Tenenbaum. Learning cross-cutting systems of categories. *Annual Conference of the Cognitive Science Society*, 2006.

[16] John M. Winn. *Variational Message Passing and its Applications*. PhD thesis, Department of Physics, University of Cambridge, 2003.

[17] Ralph Herbrich, Tom Minka, and Thore Graepel. Trueskill$^{\text{TM}}$: A bayesian skill rating system. In *Advances in Neural Information Processing Systems (NIPS)*, pages 569–576, 2007.