Pre and Post Counting for Scalable Statistical-Relational Model Discovery

Richard Mar and Oliver Schulte

School of Computing Science
Simon Fraser University, Vancouver, Canada

Code Repository: https://github.com/sfu-cl-lab/FactorBase

First-Order Bayesian Networks

- Type of Bayesian network (BN) that can be learned from a relational dataset.
- Scoring metrics can be used to determine the best structure of the BN for a given dataset.

BDeu Scoring Metric

\[
BDeu(B, D) = \log \{P(B)\} + \sum_{i=1}^{n} \sum_{j=1}^{k} \left( \log \left( \frac{\Gamma \left( \frac{N_{ij}}{6} \right)}{\Gamma \left( \frac{N_{ij}}{6} + \frac{N_{ik}}{6} \right)} \right) + \sum_{l=1}^{m} \log \left( \frac{\Gamma \left( \frac{N_{ijkl}}{6} \right)}{\Gamma \left( \frac{N_{ijkl}}{6} + \frac{N_{ijk}}{6} \right)} \right) \right)
\]

Contingency Tables

- Allow for efficient computation of the \(N_{ij}\) and \(N_{ijkl}\) terms of the BDeu scoring metric.
- Applicable for other Bayesian network scoring metrics as well.
- Generating this data structure is a challenging problem!

References


Precount Counts Caching Method

Algorithm 1 The PRECOUNT method: pre-compute ct-tables for each lattice point.
1: for each latticePoint LP \( \in \) relationshipLattice do
2:    \( ct_+ (LP) \leftarrow \text{INNERJOIN(TABLES(LP))} \)
3:    \( ct (LP) \leftarrow \text{MOBIUSJOIN}(ct_+ (LP)) \)
4: end for
5: for each family \( \in \) structureLearning do
6:    \( ct(family) \leftarrow \text{PROJECT}(ct(LP), family) \)
7:    score \( \leftarrow \text{BDEU}(ct(family)) \)
8: end for

Ondemand Counts Caching Method

Algorithm 2 The ONDEMAND method: compute ct-tables for each family during structure search.
1: for each family \( \in \) structureLearning do
2:    \( ct_+(family) \leftarrow \text{INNERJOIN(TABLES(family))} \)
3:    \( ct(family) \leftarrow \text{MOBIUSJOIN}(ct_+(family)) \)
4: score \( \leftarrow \text{BDEU}(ct(family)) \)
5: end for

Hybrid Counts Caching Method

Algorithm 3 The HYBRID method: pre-compute positive ct-tables for each lattice point and compute ct-tables for each family during structure search.
1: for each latticePoint LP \( \in \) relationshipLattice do
2:    \( ct_+ (LP) \leftarrow \text{INNERJOIN(TABLES(LP))} \)
3: end for
4: for each family \( \in \) structureLearning do
5:    \( ct_+(family) \leftarrow \text{PROJECT}(ct_+(LP), family) \)
6:    \( ct(family) \leftarrow \text{MOBIUSJOIN}(ct_+(family)) \)
7: score \( \leftarrow \text{BDEU}(ct(family)) \)
8: end for

Results

- Table: DB, Estimated ct(family) Count, Total Row Count, ct(defalase) Total Row Count

Conclusion

- Generating instantiation counts for relational data is a non-trivial task where pure pre and post count counting methods do not scale well.
- A hybrid approach where pre-counting is used to generate the counts for True relationships and post-counting is used to generate the counts for False relationships is best.
- Hybrid mitigates the numerous expensive operations of post-counting by using the pre-counting approach.