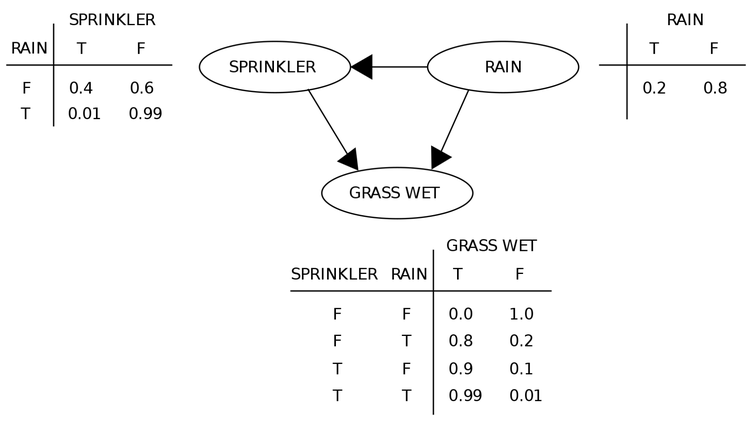
Learning Bayesian Network Structures using Curriculum Learning

Tristan Klintworth
Mentors: Mohammad Ali Javidian, Pooyan Jamshidi
Department of Computer Science and Engineering
University of South Carolina

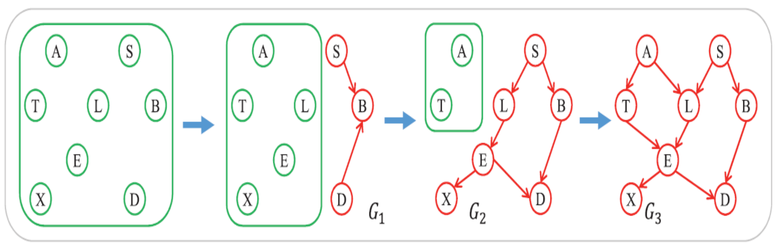
Introduction

Bayesian Networks (BNs):
Bayesian networks are a type of probabilistic graphical model built from probability distributions. BNs can be used for many different tasks requiring the laws of probability such as prediction, reasoning, and decision making under uncertainty. Bayesian networks are typically represented as a directed acyclic graph (DAG), where each node is a variable and each edge represents conditional dependencies.



Curriculum Learning (CL):
Curriculum learning is an intuitive approach based on the idea that people tend to learn more effectively when learning information is ordered from less to more complex [1].

Learning Bayesian Network Structure:



This shows the process of curriculum learning of a Bayesian network structure. In this example, a step size of 3 is used. The figure shows three stages, the first subnet learned being G_1, the second being G_2, and the third being G_3. In each case, the newly learned subnet is of size 3 and each larger subnet is the start point of the next search.

Main purpose:
Most approaches look at all the variables at once. Using curriculum learning, the BN is constructed by stages, with each stage being a subnet that is learned over a number of variables (defined by the "step size") and conditioned on fixed values for every other variable. The algorithm iterates through these subnets until all variables are included in the final stage and the full BN is constructed.

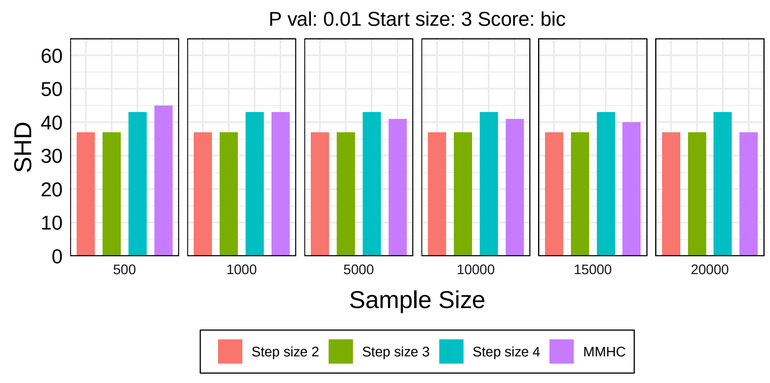
Conclusion

By measuring the Structural Hamming Distance, we generally achieved better results than state-of-the-art algorithms for learning the structure of Bayesian networks such as the MMHC algorithm.

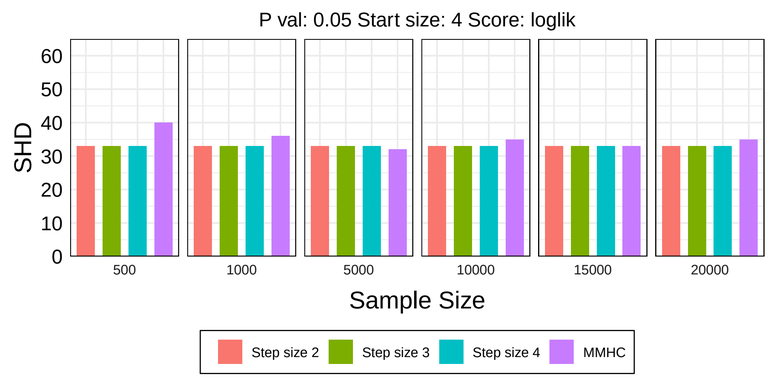
Results

We ran the CL algorithm over different data sets such as "Hailfinder", "Insurance", and "Alarm" to determine the Structural Hamming Distance (SHD) between the true graph and the learned graph. This measure determines how close the learned network is to accurately representing the actual network, where a lower number is better. These values are compared across different step sizes and compared to the MMHC algorithm.

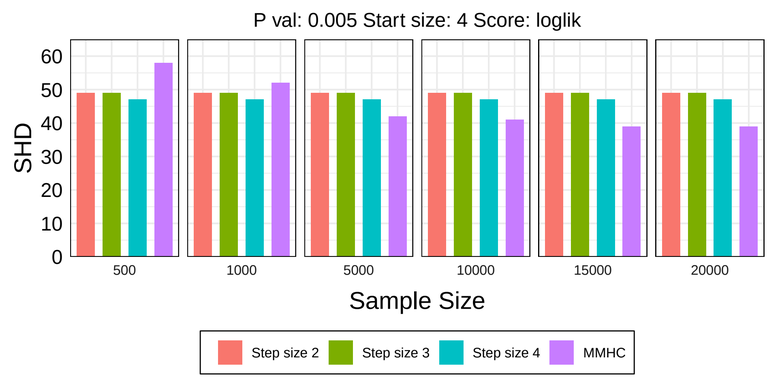
Insurance:



Alarm:



Hailfinder:



Discussion:
- Our implementation outperforms MMHC across many different sample sizes, especially when the size of observational dataset is small which is the case in many real-world problems.
- Variation between step sizes is minimal although it appears the smaller step sizes perform better on average.

Methods

Our implementation uses an algorithm from [2] that builds a Bayesian network of the data by using Algorithm 1.

Algorithm 1: Curriculum Learning of Bayesian Network Structures

input: variable set $\mathbf{X}$, training data D, size of $\mathbf{X}_{(1)}$ s, a step size t
$(\mathbf{X}_{(1)},...,\mathbf{X}_{(m)}) \longleftarrow ConstructCurriculum(\mathbf{X}, D, s, t)$
for $i \in \{1, ..., n\}$ **do**
  $S_i \longleftarrow MMPC(X_i, D)$
end
$\mathbf{S} \longleftarrow (S_1, ..., S_n)$
Initialize G_0 to a network containing variables in $\mathbf{X}_{(1)}$ with no edge.
i$\longleftarrow 1$
for $i \leq m$ **do**
  Generate the set of data segments $D_i = \{D_{i,1}, ..., D_{i,q}\}$ based on the values of $\mathbf{X} \setminus \mathbf{X}_{(i)}$
  $G_i \longleftarrow search(D_i, \mathbf{X}_{(i)}, \mathbf{S}, G_{i-1})$
  $i \longleftarrow i + 1$
end
return G_m

Algorithm 1 is implemented across several different R source files. The call to ConstructCurriculum builds the appropriate curriculum (the list of variable sets to learn subnets for). From there, the learned net is built iteratively using the search function which in this case we use a greedy hill-climbing algorithm.

Future Work

A limitation of the software presented here is that it is not a complete replication of the original algorithm in [2]. The results presented here and the results presented in [2] differ because we use a different score function than the original one.

Their score function is as follows:

$$score(G_i : X_i) = \sum_{j=1}^{q} score_{BDe}(G_i, X_{i,j}) - \left(\frac{1000}{SS} + \frac{V(G_i)}{100}\right) E(G_i). \quad (1)$$

To achieve better results, especially for very large BNs, this score function will be patched into the existing R curriculum learning code.

References

[1] Bengio. Curriculum learning. *Journal of the American Podiatry Association*, 2009.

[2] Zhao. Learning bayesian network structures under incremental construction curricula. *Neurocomputing*, 258:30–40, 2002.

[3] Marco Scutari. Learning bayesian networks with the bnlearn R package. *Journal of Statistical Software*, 35(3):1–22, 2010.

Acknowledgements

This project was partially supported by the McNair Junior Fellows program through the College of Engineering and Computing.