Learning Discrete Directed Acyclic Graphs via Backpropagation

Andrew J. Wren[♈] Pasquale Minervini[♈Ω]
Luca Franceschi[Π] Valentina Zantedeschi[✓♈♏]

andrew.wren@ntlworld.com

arXiv:2210.15353

Andrew J. Wren is very grateful to his co-authors who supervised him in the University College London MSc Machine Learning thesis project from which this paper derives.

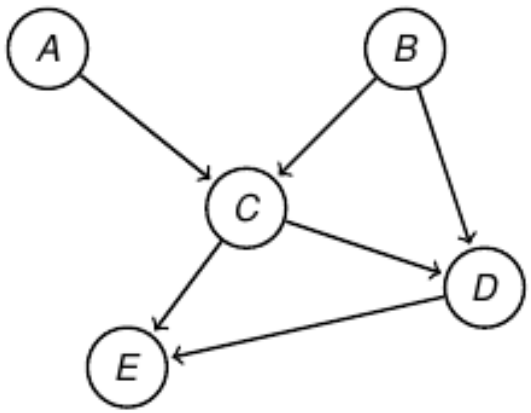
[♈] University College London [Ω] University of Edinburgh [♏] Inria London
[Π] Amazon Web Services (work done while at UCL, and prior to joining Amazon) [✓] ServiceNow

Overview

- Problem: learning a Bayesian network's DAG from data.
- Question: can a probabilistic discrete backpropagation approach be used?
- Backprop methods.
- Architecture: DAG-DB.
- Experiments and results.
- Conclusion and future work.

Problem: learning a DAG from data (1)

- Recall the problem of structure identification: learning a Bayesian network's directed acyclic graph (DAG) from data it has generated.



$$P(A, B, C, D, E) = P(A)P(B)P(C|A, B)P(D|B, C)P(E|C, D)$$

Figure. Example of Bayesian network, with $d = 5$ nodes.[1]

- Given data $\boldsymbol{X} = (\boldsymbol{x}_{n,\mathrm{A}}, ..., \boldsymbol{x}_{n,\mathrm{E}})_{n=1}^{N} \in \mathbb{R}^{N \times d}$ from a hidden d-node DAG D, make prediction $\mathrm{D}_{\mathrm{pred}}$.

[1] Adapted from Sahani, UCL (2021).

Problem: learning a DAG from data (2)

- Existing approaches to the problem can mainly be divided between combinatoric, continuous and Bayesian.
- Differences include representation of DAG edges in core calculations, and whether gradient descent (GD) is used.

Approach	DAG edges	GD?	Examples
Combinatoric	Binary	No	PC-Stable[1], FGES[2]
Continuous	Float	Yes	NOTEARS[3], GOLEM[4]
Bayesian	Binary	Yes	DiBS[5], DAG-GFlowNet[6]
Probabilistic	Binary	Yes	DAG-DB *presented today*

[1] Colombo & Maathuis. *Order-independent constraint-based causal structure learning.* (2014)

[2] Ramsey, Glymour, Sanchez-Romero & Glymour. *A million variables and more... .* (2017)

[3] Zheng, Aragam, Ravikumar & Xing. *DAGs with NO TEARS....* (2018)

[4] Ng, Ghassami & Zhang. *On the Role of Sparsity and DAG Constraints for Learning Linear DAGs.* (2020)

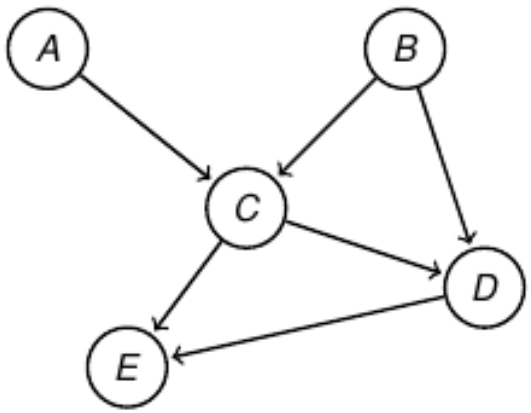
[5] Lorch, Rothfuss, Schölkopf & Krause. *DiBS: Differentiable Bayesian Structure Learning.* (2021)

[6] Dele, Góis, Emezue, Rankawat, Lacoste-Julien, Bauer & Bengio. *Bayesian Structure Learning with Generative Flow Networks.* (2022)

Question: probabilistic approach? (1)

Underlying approach:

- For data point $\boldsymbol{x}$, predict value at a 'child' node from its 'parents'.



$$P(A, B, C, D, E) = P(A)P(B)P(C|A, B)P(D|B, C)P(E|C, D)$$

Figure. Example of Bayesian network.[1]

- E.g, predict $\boldsymbol{x}_D$ from $\boldsymbol{x}_B$ and $\boldsymbol{x}_C$.

[1] Adapted from Sahani, UCL (2021).

Question: probabilistic approach? (2)

Framework:

- **Directed graph.** With $\boldsymbol{X} \in \mathbb{R}^{N \times d}$, looking for graph on d nodes. Let $\boldsymbol{Z}$ be $d \times d$ *binary* adjacency matrix for a directed graph.

- **Distribution.** A *float* matrix $\boldsymbol{\Theta}$ (same shape as $\boldsymbol{Z}$) parametrizes an ExpFam distribution,

$$p(\boldsymbol{Z}; \boldsymbol{\Theta}) = \exp\left(\langle \boldsymbol{Z}, \boldsymbol{\Theta} \rangle - \mathrm{A}(\boldsymbol{\Theta})\right).$$

- For **sampling**, use 'Perturb-&-MAP'[1] to get S samples. For each sample $\boldsymbol{Z}^{(s)}$, take the most probable value of $\boldsymbol{Z}$ given parameter $\boldsymbol{\Theta}$ plus noise:

$$\boldsymbol{Z}^{(s)} = \mathrm{MAP}(\boldsymbol{\Theta} + \epsilon^{(s)}) \text{ with } \epsilon^{(s)} \sim \mathrm{Logistic}(0, 1), \quad s = 1, ..., S.$$

[1] Papandreou & Yuille. *Perturb-and-MAP random fields: Using discrete optimization to learn and sample from energy models.* (2011)

Backprop methods (1)

To learn $\boldsymbol{\Theta}$, need a way to backpropagate from $\boldsymbol{Z}$ to $\boldsymbol{\Theta}$.
Backprop methods which keep $\boldsymbol{Z}$ binary include:

- **Straight-Through Estimation (STE).**[1] Very simple.
- **Score-Function Estimation (SFE).** Aka REINFORCE.
- **Blackbox Estimation (BBE).** Not probabilistic.
- **Implicit Maximum Likelihood Estimation (I-MLE).**[2] Taking advantage $p(\boldsymbol{Z}; \boldsymbol{\Theta}) \in$ ExpFam.

[1] Hinton, Srivastava & Swersky. *Neural networks for machine learning.* (2012); Bengio, Léonard & Courville. *Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation.* (2013)

[2] Niepert, Minervini & Franceschi. *Implicit MLE: Backpropagating Through Discrete Exponential Family Distributions.* (2021)

Backprop methods (2, STE)

- For backward pass, with average loss (plus regularizers) L, Straight-Through Estimation (STE), approximates gradient of L w.r.t. $\boldsymbol{\Theta}$ as

$$\frac{\partial L}{\partial \boldsymbol{\Theta}} \propto \sum_{s=1}^{S} \frac{\partial L}{\partial \boldsymbol{Z}^{(s)}}.$$

Backprop methods (3, I-MLE)

- Or, for backward pass with Implicit Maximum Likelihood (I-MLE), perturb $\boldsymbol{\Theta}$ using each sample $\boldsymbol{Z}^{(s)}$,

$$\widetilde{\boldsymbol{\Theta}}^{(s)} = \boldsymbol{\Theta} - \lambda \frac{\partial L}{\partial \boldsymbol{Z}^{(s)}}.$$

- Use noise values from sampling $\epsilon^{(s)}$ to set

$$\widetilde{\boldsymbol{Z}}^{(s)} = \mathrm{MAP}(\widetilde{\boldsymbol{\Theta}}^{(s)} + \epsilon^{(s)}).$$

- Approximate gradient of L w.r.t. $\boldsymbol{\Theta}$ as

$$\frac{\partial L}{\partial \boldsymbol{\Theta}} \approx \frac{1}{\lambda S} \sum_{s=1}^{S} \left[\boldsymbol{Z}^{(s)} - \widetilde{\boldsymbol{Z}}^{(s)} \right].$$

Architecture

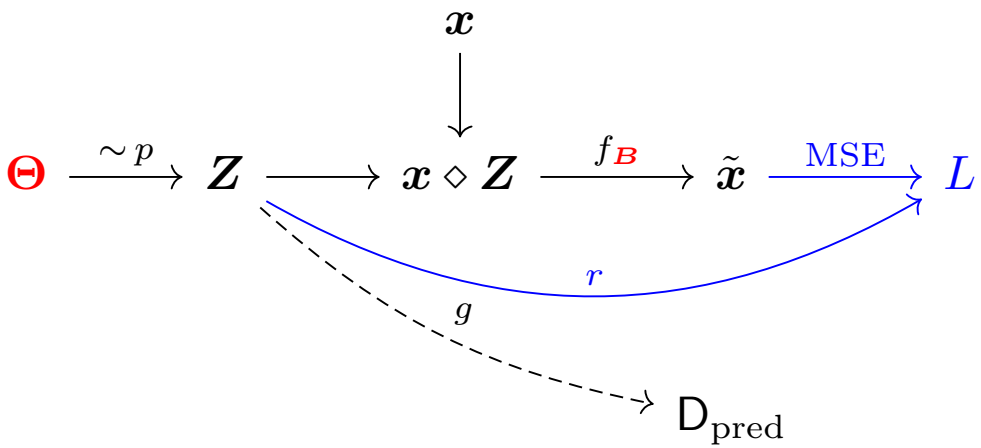
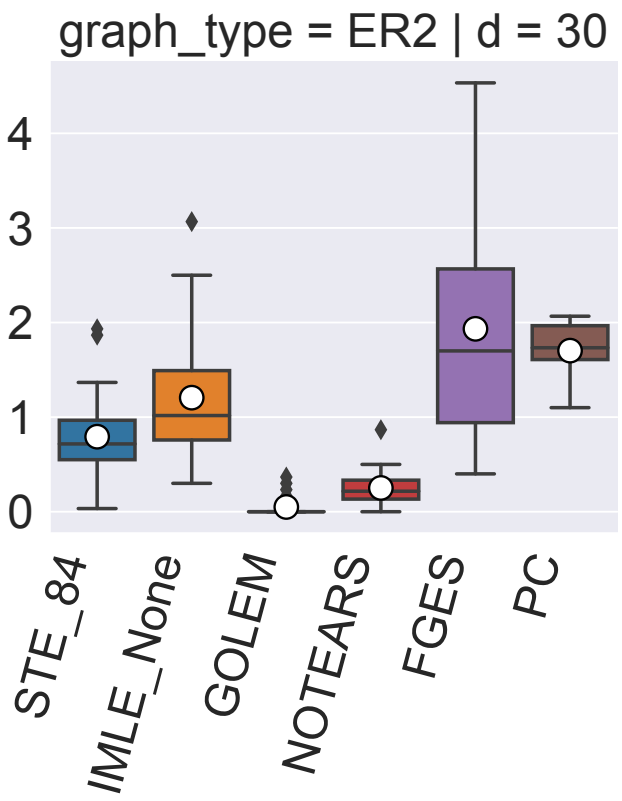


Figure. DAG-DB, including learnable parameters and loss calculation.

- $\boldsymbol{x} \diamond \boldsymbol{Z}$ ensures only 'parents' influence predictions for 'children'.
- $f_{\boldsymbol{B}}$ is a very simple neural net: linear no bias.
- r is regularizer: NOTEARS-like regularizer pushing $\boldsymbol{Z}$ towards being a DAG, plus sparsity regularizer.
- Depending on compute and size of problem, $\boldsymbol{Z}$ is transformed into a DAG $\mathrm{D}_{\mathrm{pred}}$ by a maximum DAG algorithm g in training or only at evaluation.

Experiments (1, synthetic data)



ER2: linear Gaussian additive noise model on random (Erdös-Rényi) graphs with d nodes, and expected number of edges is $2d$. In this example, $d = 30$ nodes.

Results on synthetic data showing a normalised class **structural Hamming distance** $\mathrm{nSHD_c}$ from the true DAG.

Shows two DAG-DB methods STE_84 and IMLE_None, compared with continuous (GOLEM, NOTEARS) and combinatoric (FGES, PC) methods.

Experiments (2, real data)

Model		SHD_c	$prec_c$
IMLE_None	mean	12.7	0.869
	median	13	**1.000**
GOLEM		**11**	**1.000**
NOTEARS		**11**	0.467
FGES		**11**	0.750
PC		**11**	0.750

Tests on the Sachs cellular biochemistry dataset.[1] Best metric scores are in bold. SHD_c is un-normalised; $prec_c$ is a **precision** metric.

[1] Sachs, Perez, Pe'er, Lauffenburger & Nolan. *Causal protein-signaling networks derived from multiparameter single-cell data.* (2005)

Conclusion and future work

Take home

- Using discrete probabilistic backpropagation, DAG-DB performs competitively at structure identification, often better than combinatoric methods, though not as well as continuous approaches.
- May be easier than many other DAG identification methods to integrate with other neural nets.

- Extend to more types of Bayesian network: non-Gaussian, non-linear, discrete-valued, with interventions.
- What if no 'gold' DAG, e.g. we generate a latent DAG optimised to achieve some goal?

Thank you!

andrew.wren@ntlworld.com
arXiv:2210.15353