DAG-GNN: DAG Structure Learning with Graph Neural Networks

Yue Yu [*1] **Jie Chen** [*2 3] **Tian Gao** [3] **Mo Yu** [3]

Abstract

Learning a faithful directed acyclic graph (DAG) from samples of a joint distribution is a challenging combinatorial problem, owing to the intractable search space superexponential in the number of graph nodes. A recent breakthrough formulates the problem as a continuous optimization with a structural constraint that ensures acyclicity (Zheng et al., 2018). The authors apply the approach to the linear structural equation model (SEM) and the least-squares loss function that are statistically well justified but nevertheless limited. Motivated by the widespread success of deep learning that is capable of capturing complex nonlinear mappings, in this work we propose a deep generative model and apply a variant of the structural constraint to learn the DAG. At the heart of the generative model is a variational autoencoder parameterized by a novel graph neural network architecture, which we coin DAG-GNN. In addition to the richer capacity, an advantage of the proposed model is that it naturally handles discrete variables as well as vector-valued ones. We demonstrate that on synthetic data sets, the proposed method learns more accurate graphs for nonlinearly generated samples; and on benchmark data sets with discrete variables, the learned graphs are reasonably close to the global optima. The code is available at `https://github.com/fishmoon1234/DAG-GNN`.

1. Introduction

Bayesian Networks (BN) have been widely used in machine learning applications (Spirtes et al., 1999; Ott et al., 2004). The structure of a BN takes the form of a directed acyclic graph (DAG) and plays a vital part in causal inference (Pearl, 1988) with many applications in medicine, genetics, economics, and epidemics. Its structure learning problem is however NP-hard (Chickering et al., 2004) and stimulates a proliferation of literature.

Score-based methods generally formulate the structure learning problem as optimizing a certain score function with respect to the unknown (weighted) adjacency matrix A and the observed data samples, with a combinatorial constraint stating that the graph must be acyclic. The intractable search space (with a complexity superexponential in the number of graph nodes) poses substantial challenges for optimization. Hence, for practical problems in a scale beyond small, approximate search often needs to be employed with additional structure assumption (Nie et al., 2014; Chow & Liu, 1968; Scanagatta et al., 2015; Chen et al., 2016).

Recently, Zheng et al. (2018) formulate an equivalent acyclicity constraint by using a continuous function of the adjacency matrix (specifically, the matrix exponential of $A \circ A$). This approach drastically changes the combinatorial nature of the problem to a continuous optimization, which may be efficiently solved by using maturely developed blackbox solvers. The optimization problem is nevertheless nonlinear, thus these solvers generally return only a stationary-point solution rather than the global optimum. Nevertheless, the authors show that empirically such local solutions are highly comparable to the global ones obtained through expensive combinatorial search.

With the inspiring reformulation of the constraint, we revisit the objective function. The score-based objective functions generally make assumptions of the variables and the model class. For example, Zheng et al. (2018) demonstrate on the linear structural equation model (SEM) with a least-squares loss. While convenient, such assumptions are often restricted and they may not correctly reflect the actual distribution of real-life data.

Hence, motivated by the remarkable success of deep neural networks, which are arguably universal approximators, in this work we develop a graph-based deep generative model aiming at better capturing the sampling distribution faithful to the DAG. To this end, we employ the machinery of variational inference and parameterize a pair of encoder/decoder with specially designed graph neural networks (GNN). The objective function (the score), then, is the evidence lower

[*]Equal contribution [1]Lehigh University [2]MIT-IBM Watson AI Lab [3]IBM Research. Correspondence to: Yue Yu <yuy214@lehigh.edu>, Jie Chen <chenjie@us.ibm.com>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

bound. Different from the current flourishing designs of GNNs (Bruna et al., 2014; Defferrard et al., 2016; Li et al., 2016; Kipf & Welling, 2017; Hamilton et al., 2017; Gilmer et al., 2017; Chen et al., 2018; Veličković et al., 2018), the proposed ones are generalized from linear SEM, so that the new model performs at least as well as linear SEM when the data is linear.

Our proposal has the following distinct features and advantages. First, the work is built on the widespread use of deep generative models (specifically, variational autoencoders, VAE (Kingma & Welling, 2014)) that are able to capture complex distributions of data and to sample from them. Under the graph setting, the weighted adjacency matrix is an explicit parameter, rather than a latent structure, learnable together with other neural network parameters. The proposed network architecture has not been used before.

Second, the framework of VAE naturally handles various data types, notably not only continuous but also discrete ones. All one needs to do is to model the likelihood distribution (decoder output) consistent with the nature of the variables.

Third, owing to the use of graph neural networks for parameterization, each variable (node) can be not only scalar-valued but also vector-valued. These variables are considered node features input to/output of the GNNs.

Fourth, we propose a variant of the acyclicity constraint more suitable for implementation under current deep learning platforms. The matrix exponential suggested by Zheng et al. (2018), while mathematically elegant, may not be implemented or supported with automatic differentiation in all popular platforms. We propose a polynomial alternative more practically convenient and as numerically stable as the exponential.

We demonstrate the effectiveness of the proposed method on synthetic data generated from linear and nonlinear SEMs, benchmark data sets with discrete variables, and data sets from applications. For synthetic data, the proposed DAG-GNN outperforms DAG-NOTEARS, the algorithm proposed by Zheng et al. (2018) based on linear SEM. For benchmark data, our learned graphs compare favorably with those obtained through optimizing the Bayesian information criterion by using combinatorial search.

2. Background and Related Work

A DAG G and a joint distribution $\mathcal{P}$ are *faithful* to each other if all and only the conditional independencies true in $\mathcal{P}$ are entailed by G (Pearl, 1988). The faithfulness condition enables one to recover G from $\mathcal{P}$. Given independent and iid samples D from an unknown distribution corresponding to a faithful but unknown DAG, *structure learning* refers to recovering the DAG from D.

Many exact and approximate algorithms for learning DAG from data have been developed, including score-based and constraint-based approaches (Spirtes et al., 2000a; Chickering, 2002; Koivisto & Sood, 2004; Silander & Myllymaki, 2006; Jaakkola et al., 2010; Cussens, 2011; Yuan & Malone, 2013; Gao & Wei, 2018). Score-based methods generally use a score to measure the goodness of fit of different graphs over data; and then use a search procedure—such as hill-climbing (Heckerman et al., 1995; Tsamardinos et al., 2006; Gmez et al., 2011), forward-backward search (Chickering, 2002), dynamic programming (Singh & Moore, 2005; Silander & Myllymaki, 2006), A* (Yuan & Malone, 2013), or integer programming (Jaakkola et al., 2010; Cussens, 2011; Cussens et al., 2016)—in order to find the best graph. Commonly used Bayesian score criteria, such as BDeu and Bayesian information criterion (BIC), are decomposable, consistent, locally consistent (Chickering, 2002), and score equivalent (Heckerman et al., 1995).

To make the DAG search space tractable, approximate methods make additional assumptions such as bounded tree-width (Nie et al., 2014), tree-like structures (Chow & Liu, 1968), approximation (Scanagatta et al., 2015), and other constraints about the DAG (Chen et al., 2016). Many bootstrap (Friedman et al., 1999) and sampling-based structure learning algorithms (Madigan et al., 1995; Friedman & Koller, 2003; Eaton & Murphy, 2012; Grzegorczyk & Husmeier, 2008; Niinimäki & Koivisto, 2013; Niinimaki et al., 2012; He et al., 2016) are also proposed to tackle the expensive search problem.

Constraint-based methods, in contrast, use (conditional) independence tests to test the existence of edges between each pair of variables. Popular algorithms include SGS (Spirtes et al., 2000b), PC (Spirtes et al., 2000b), IC (Pearl, 2003), and FCI (Spirtes et al., 1995; Zhang, 2008). Recently, there appears a suite of hybrid algorithms that combine score-based and constraint-based methods, such as MMHC (Tsamardinos et al., 2003), and apply constraint-based methods to multiple environments (Mooij et al., 2016).

Due to the NP-hardness, traditional DAG learning methods usually deal with discrete variables, as discussed above, or jointly Gaussian variables (Mohan et al., 2012; Mohammadi et al., 2015). Recently, a new continuous optimization approach is proposed (Zheng et al., 2018), which transforms the discrete search procedure into an equality constraint. This approach enables a suite of continuous optimization techniques such as gradient descent to be used. The approach achieves good structure recovery results, although it is applied to only linear SEM for ease of exposition.

Neural-network approaches started to surface only very re-

cently. Kalainathan et al. (2018) propose a GAN-style (generative adversarial network) method, whereby a separate generative model is applied to each variable and a discriminator is used to distinguish between the joint distributions of real and generated samples. The approach appears to scale well but acyclicity is not enforced.

3. Neural DAG Structure Learning

Our method learns the weighted adjacency matrix of a DAG by using a deep generative model that generalizes linear SEM, with which we start the journey.

3.1. Linear Structural Equation Model

Let $A \in \mathbb{R}^{m\times m}$ be the weighted adjacency matrix of the DAG with m nodes and $X \in \mathbb{R}^{m\times d}$ be a sample of a joint distribution of m variables, where each row corresponds to one variable. In the literature, a variable is typically a scalar, but it can be trivially generalized to a d-dimensional vector under the current setting. The linear SEM model reads

$$X = A^T X + Z, \tag{1}$$

where $Z \in \mathbb{R}^{m\times d}$ is the noise matrix. When the graph nodes are sorted in the topological order, the matrix A is strictly upper triangular. Hence, ancestral sampling from the DAG is equivalent to generating a random noise Z followed by a triangular solve

$$X = (I - A^T)^{-1} Z. \tag{2}$$

3.2. Proposed Graph Neural Network Model

Equation (2) may be written as $X = f_A(Z)$, a general form recognized by the deep learning community as an abstraction of parameterized graph neural networks that take node features Z as input and return X as high level representations. Nearly all graph neural networks (Bruna et al., 2014; Defferrard et al., 2016; Li et al., 2016; Kipf & Welling, 2017; Hamilton et al., 2017; Gilmer et al., 2017; Chen et al., 2018; Veličković et al., 2018) can be written in this form. For example, the popular GCN (Kipf & Welling, 2017) architecture reads

$$X = \widehat{A} \cdot \mathrm{ReLU}(\widehat{A} Z W^1) \cdot W^2,$$

where $\widehat{A}$ is a normalization of A and W^1 and W^2 are parameter matrices.

Owing to the special structure (2), we propose a new graph neural network architecture

$$X = f_2((I - A^T)^{-1} f_1(Z)). \tag{3}$$

The parameterized functions f_1 and f_2 effectively perform (possibly nonlinear) transforms on Z and X, respectively. If f_2 is invertible, then (3) is equivalent to $f_2^{-1}(X) = A^T f_2^{-1}(X) + f_1(Z)$, a generalized version of the linear SEM (1).

We will defer the instantiation of these functions in a later subsection. One of the reasons is that the activation in f_2 must match the type of the variable X, a subject to be discussed together with discrete variables.

3.3. Model Learning with Variational Autoencoder

Given a specification of the distribution of Z and samples $X^1, \ldots, X^n$, one may learn the generative model (3) through maximizing the log-evidence

$$\frac{1}{n}\sum_{k=1}^{n} \log p(X^k) = \frac{1}{n}\sum_{k=1}^{n} \log \int p(X^k|Z)p(Z)\, dZ,$$

which, unfortunately, is generally intractable. Hence, we appeal to variational Bayes.

To this end, we use a variational posterior $q(Z|X)$ to approximate the actual posterior $p(Z|X)$. The net result is the evidence lower bound (ELBO)

$$L_{\text{ELBO}} = \frac{1}{n}\sum_{k=1}^{n} L^k_{\text{ELBO}},$$

with

$$L^k_{\text{ELBO}} \equiv -D_{\text{KL}}\Big(q(Z|X^k)\,||\,p(Z)\Big) + \mathrm{E}_{q(Z|X^k)}\Big[\log p(X^k|Z)\Big]. \tag{4}$$

Each individual term L^k_{ELBO} departs from the log-evidence by $D_{\text{KL}}\Big(q(Z|X^k)\,||\,p(Z|X^k)\Big) \geq 0$, the KL-divergence between the variational posterior and the actual one.

The ELBO lends itself to a variational autoencoder (VAE) (Kingma & Welling, 2014), where given a sample X^k, the encoder (inference model) encodes it into a latent variable Z with density $q(Z|X^k)$; and the decoder (generative model) tries to reconstruct X^k from Z with density $p(X^k|Z)$. Both densities may be parameterized by using neural networks.

Modulo the probability specification to be completed later, the generative model (3) discussed in the preceding subsection plays the role of the decoder. Then, we propose the corresponding encoder

$$Z = f_4((I - A^T) f_3(X)), \tag{5}$$

where f_3 and f_4 are parameterized functions that conceptually play the inverse role of f_2 and f_1, respectively.

3.4. Architecture and Loss Function

To complete the VAE, one must specify the distributions in (4). Recall that for now both X^k and Z are $m\times d$ matrices.

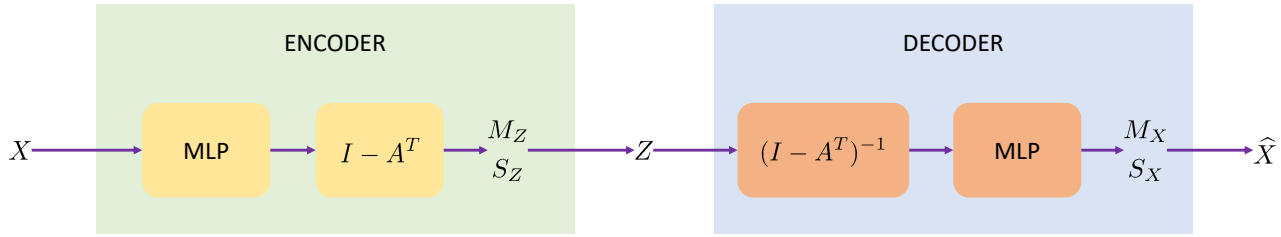

Figure 1. Architecture (for continuous variables). In the case of discrete variables, the decoder output is changed from M_X, S_X to P_X.

For simplicity, the prior is typically modeled as the standard matrix normal $p(Z) = \mathcal{MN}_{m\times d}(0, I, I)$.

For the inference model, we let f_3 be a multilayer perceptron (MLP) and f_4 be the identity mapping. Then, the variational posterior $q(Z|X)$ is a factored Gaussian with mean $M_Z \in \mathbb{R}^{m\times d}$ and standard deviation $S_Z \in \mathbb{R}^{m\times d}$, computed from the encoder

$$[M_Z|\log S_Z] = (I - A^T)\,\mathrm{MLP}(X, W^1, W^2), \quad (6)$$

where $\mathrm{MLP}(X, W^1, W^2) := \mathrm{ReLU}(XW^1)W^2$, and W^1 and W^2 are parameter matrices.

For the generative model, we let f_1 be the identity mapping and f_2 be an MLP. Then, the likelihood $p(X|Z)$ is a factored Gaussian with mean $M_X \in \mathbb{R}^{m\times d}$ and standard deviation $S_X \in \mathbb{R}^{m\times d}$, computed from the decoder

$$[M_X|\log S_X] = \mathrm{MLP}((I - A^T)^{-1}Z, W^3, W^4), \quad (7)$$

where W^3 and W^4 are parameter matrices.

One may switch the MLP and the identity mapping inside each of the encoder/decoder, but we find that the performance is less competitive. One possible reason is that the current design (7) places an emphasis on the nonlinear transform of a sample $(I - A^T)^{-1}Z$ from linear SEM, which better captures nonlinearity.

Based on (6) and (7), the KL-divergence term in the ELBO (4) admits a closed form

$$D_{\mathrm{KL}}\Big(q(Z|X)\,||\,p(Z)\Big) = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{d}(S_Z)_{ij}^2 + (M_Z)_{ij}^2 - 2\log(S_Z)_{ij} - 1, \quad (8)$$

and the reconstruction accuracy term may be computed with Monte Carlo approximation

$$\mathrm{E}_{q(Z|X)}\Big[\log p(X|Z)\Big] \approx \frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{m}\sum_{j=1}^{d} -\frac{(X_{ij} - (M_X^{(l)})_{ij})^2}{2(S_X^{(l)})_{ij}^2} - \log(S_X^{(l)})_{ij} - c, \quad (9)$$

where c is a constant and $M_X^{(l)}$ and $S_X^{(l)}$ are the outputs of the decoder (7) by taking as input Monte Carlo samples $Z^{(l)} \sim q(Z|X)$, for $l = 1, \ldots, L$.

Note that under the autoencoder framework, Z is considered latent (rather than the noise in linear SEM). Hence, the column dimension of Z may be different from d. From the neural network point of view, changing the column dimension of Z affects only the sizes of the parameter matrices W^2 and W^3. Sometimes, one may want to use a smaller number than d if he/she observes that the data has a smaller intrinsic dimension.

An illustration of the architecture is shown in Figure 1.

3.5. Discrete Variables

One advantage of the proposed method is that it naturally handles discrete variables. We assume that each variable has a finite support of cardinality d.

Hence, we let each row of X be a one-hot vector, where the "on" location indicates the value of the corresponding variable. We still use standard matrix normal to model the prior and factored Gaussian to model the variational posterior, with (6) being the encoder. On the other hand, we need to slightly modify the likelihood to cope with the discrete nature of the variables.

Specifically, we let $p(X|Z)$ be a factored categorical distribution with probability matrix P_X, where each row is a probability vector for the corresponding categorical variable. To achieve so, we change f_2 from the identity mapping to a row-wise softmax and modify the decoder (7) to

$$P_X = \mathrm{softmax}(\mathrm{MLP}((I - A^T)^{-1}Z, W^3, W^4)). \quad (10)$$

Correspondingly for the ELBO, the KL term (8) remains the same, but the reconstruction term (9) needs be modified to

$$\mathrm{E}_{q(Z|X)}\Big[\log p(X|Z)\Big] \approx \frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{m}\sum_{j=1}^{d} X_{ij}\log(P_X^{(l)})_{ij}, \quad (11)$$

where $P_X^{(l)}$ is the output of the decoder (10) by taking as input Monte Carlo samples $Z^{(l)} \sim q(Z|X)$, for $l = 1, \ldots, L$.

3.6. Connection to Linear SEM

One has seen from the forgoing discussions how the proposed model is developed from linear SEM: We apply nonlinearality to the sampling procedure (2) of SEM, treat the resulting generative model as a decoder, and pair with it a variational encoder for tractable learning. Compared with a plain autoencoder, the variational version allows a modeling of the latent space, from which samples are generated.

We now proceed, in a reverse thought flow, to establish the connection between the loss function of the linear SEM considered in Zheng et al. (2018) and that of ours. We first strip off the variational component of the autoencoder. This plain version uses (5) as the encoder and (3) as the decoder. For notational clarity, let us write $\widehat{X}$ as the output of the decoder, to distinguish it from the encoder input X. A typical sample loss to minimize is

$$\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{d}(X_{ij}-\widehat{X}_{ij})^2+\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{d}Z_{ij}^2,$$

where the first term is the reconstruction error and the second term is a regularization of the latent space. One recognizes that the reconstruction error is the same as the negative reconstruction accuracy (9) in the ELBO, up to a constant, if the standard deviation S_X is 1, the mean M_X is taken as $\widehat{X}$, and only one Monte Carlo sample is drawn from the variational posterior. Moreover, the regularization term is the same as the KL-divergence (8) in the ELBO if the standard deviation S_Z is 1 and the mean M_Z is taken as Z.

If we further strip off the (possibly nonlinear) mappings f_1 to f_4, then the encoder (5) and decoder (3) read, respectively, $Z=(I-A^T)X$ and $\widehat{X}=(I-A^T)^{-1}Z$. This pair results in perfect reconstruction, and hence the sample loss reduces to

$$\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{d}Z_{ij}^2=\frac{1}{2}\|(I-A^T)X\|_F^2, \tag{12}$$

which is the least-squares loss used and justified by Zheng et al. (2018).

3.7. Acyclicity Constraint

Neither maximizing the ELBO (4) nor minimizing the least-squares loss (12) guarantees that the corresponding graph of the resulting A is acyclic. Zheng et al. (2018) pair the loss function with an equality constraint, whose satisfaction ensures acyclicity.

The idea is based on the fact that the positivity of the (i,j) element of the k-th power of a nonnegative adjacency matrix B indicates the existence of a length-k path between nodes i and j. Hence, the positivity of the diagonal of B^k reveals cycles. The authors leverage the trick that the matrix exponential admits a Taylor series (because it is analytic on the complex plane), which is nothing but a weighted sum of all nonnegative integer powers of the matrix. The coefficient of the zeorth power (the identity matrix $I_{m\times m}$) is 1, and hence the trace of the exponential of B must be exactly m for a DAG. To satisfy nonnegativity, one may let B be the elementwise square of A; that is, $B=A\circ A$.

Whereas the formulation of this acyclicity constraint is mathematically elegant, support of the matrix exponential may not be available in all deep learning platforms. To ease the coding effort, we propose an alternative constraint that is practically convenient.

Theorem 1. *Let $A\in\mathbb{R}^{m\times m}$ be the (possibly negatively) weighted adjacency matrix of a directed graph. For any $\alpha>0$, the graph is acyclic if and only if*

$$\operatorname{tr}[(I+\alpha A\circ A)^m]-m=0. \tag{13}$$

We use (13) as the equality constraint when maximizing the ELBO. The computations of both $(I+\alpha B)^m$ and $\exp(B)$ may meet numerical difficulty when the eigenvalues of B have a large magnitude. However, the former is less severe than the latter with a judicious choice of α.

Theorem 2. *Let $\alpha=c/m>0$ for some c. Then for any complex λ, we have $(1+\alpha|\lambda|)^m\le e^{c|\lambda|}$.*

In practice, α may be treated as a hyperparameter and its setting depends on an estimation of the largest eigenvalue of B in magnitude. This value is the spectral radius of B, and because of nonnegativity, it is bounded by the maximum row sum according to the Perron–Frobenius theorem.

3.8. Training

Based on the foregoing, the learning problem is

$$\begin{aligned}\min_{A,\theta}\quad & f(A,\theta)\equiv -L_{\text{ELBO}}\\ \text{s.t.}\quad & h(A)\equiv\operatorname{tr}[(I+\alpha A\circ A)^m]-m=0,\end{aligned}$$

where the unknowns include the matrix A and all the parameters θ of the VAE (currently we have $\theta=\{W^1,W^2,W^3,W^4\}$). Nonlinear equality-constrained problems are well studied and we use the augmented Lagrangian approach to solve it. For completeness, we summarize the algorithm here; the reader is referred to standard textbooks such as Section 4.2 of Bertsekas (1999) for details and convergence analysis.

Define the augmented Lagrangian

$$L_c(A,\theta,\lambda)=f(A,\theta)+\lambda h(A)+\frac{c}{2}|h(A)|^2,$$

where λ is the Lagrange multiplier and c is the penalty parameter. When $c=+\infty$, the minimizer of $L_c(A,\theta,\lambda)$

must satisfy $h(A) = 0$, in which case $L_c(A, \theta, \lambda)$ is equal to the objective function $f(A, \theta)$. Hence, the strategy is to progressively increase c, for each of which minimize the unconstrained augmented Lagrangian. The Lagrange multiplier λ is correspondingly updated so that it converges to the one under the optimality condition.

There exist a few variants for updating λ and increasing c, but a typical effective rule reads:

$$(A^k, \theta^k) = \underset{A,\theta}{\operatorname{argmin}} L_{c^k}(A, \theta, \lambda^k), \tag{14}$$

$$\lambda^{k+1} = \lambda^k + c^k h(A^k), \tag{15}$$

$$c^{k+1} = \begin{cases} \eta c^k, & \text{if } |h(A^k)| > \gamma |h(A^{k-1})|, \\ c^k, & \text{otherwise,} \end{cases} \tag{16}$$

where $\eta > 1$ and $\gamma < 1$ are tuning parameters. We find that often $\eta = 10$ and $\gamma = 1/4$ work well.

The subproblem (14) may be solved by using blackbox stochastic optimization solvers, by noting that the ELBO is defined on a set of samples.

4. Experiments

In this section, we present a comprehensive set of experiments to demonstrate the effectiveness of the proposed method DAG-GNN. In Section 4.1, we compare with DAG-NOTEARS, the method proposed by Zheng et al. (2018) based on linear SEM, on synthetic data sets generated by sampling generalized linear models, with an emphasis on nonlinear data and vector-valued data ($d > 1$). In Section 4.2, we showcase the capability of our model with discrete data, often seen in benchmark data sets with ground truths for assessing quality. To further illustrate the usefulness of the proposed method, in Section 4.3 we apply DAG-GNN on a protein data set for the discovery of consensus protein signaling network, as well as a knowledge base data set for learning causal relations.

Our implementation is based on PyTorch (Paszke et al., 2017). We use Adam (Kingma & Ba, 2015) to solve the subproblems (14). To avoid overparameterization, we parameterize the variational posterior $q(Z|X)$ as a factored Gaussian with constant unit variance, and similarly for the likelihood $p(X|Z)$. When extracting the DAG, we use a thresholding value 0.3, following the recommendation of Zheng et al. (2018). For benchmark and application data sets, we include a Huber-norm regularization of A in the objective function to encourage more rapid convergence.

4.1. Synthetic Data Sets

The synthetic data sets are generated in the following manner. We first generate a random DAG by using the Erdős–Rényi model with expected node degree 3, then assign uniformly random weights for the edges to obtain the weighted adjacency matrix A. A sample X is generated by sampling the (generalized) linear model $X = g(A^T X) + Z$ with some function g elaborated soon. The noise Z follows standard matrix normal. When the dimension $d = 1$, we use lowercase letters to denote vectors; that is, $x = g(A^T x) + z$. We compare DAG-GNN with DAG-NOTEARS and report the structural Hamming distance (SHD) and false discovery rate (FDR), each averaged over five random repetitions. With sample size $n = 5000$, we run experiments on four graph sizes $m \in \{10, 20, 50, 100\}$. In Sections 4.1.1 and 4.1.2 we consider scalar-valued variables ($d = 1$) and in Section 4.1.3 vector-valued variables ($d > 1$).

4.1.1. Linear Case

This case is the linear SEM model, with g being the identity mapping. The SHD and FDR are plotted in Figure 2. One sees that the graphs learned by the proposed method are substantially more accurate than those by DAG-NOTEARS when the graphs are large.

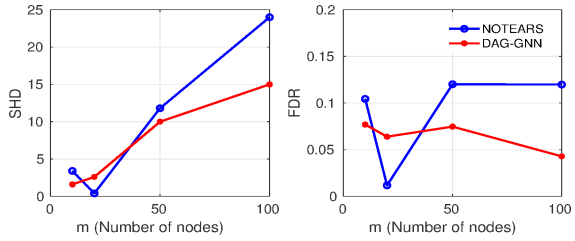

Figure 2. Structure discovery in terms of SHD and FDR to the true graph, on synthetic data set generated by $x = A^T x + z$.

4.1.2. Nonlinear Case

We now consider data generated by the following model

$$x = A^T h(x) + z,$$

for some nonlinear function h. Taking first-order approximation $h(x) \approx h(0)\mathbf{1} + h'(0)x$ (ignoring higher-order terms of x), one obtains an amendatory approximation of the graph adjacency matrix, $h'(0)A$. This approximate ground truth maintains the DAG structure, with only a scaling on the edge weights.

We take $h(x) = \cos(x + \mathbf{1})$ and plot the SHD and FDR in Figure 3. one observes that DAG-GNN slightly improves over DAG-NOTEARS in terms of SHD. Further, FDR is substantially improved, by approximately a factor of three, which indicates that DAG-GNN tends to be more accurate on selecting correct edges. This observation is consistent with the parameter estimates shown in Figure 4, where the ground truth is set as $-\sin(1)A$. The heat map confirms that DAG-GNN results in fewer "false alarms" and recovers a relatively sparser matrix.

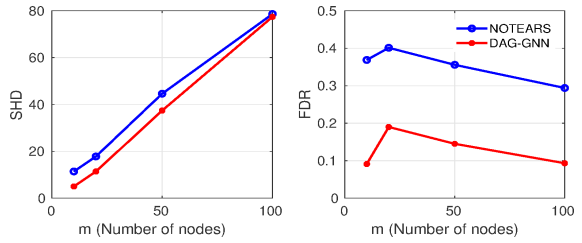

Figure 3. Structure discovery in terms of SHD and FDR to the true graph, on synthetic data set generated by $x = A^T \cos(x + \mathbf{1}) + z$.

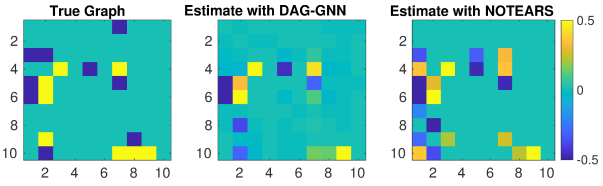

Figure 4. Parameter estimates (before thresholding) of the graph on synthetic data set generated by $x = A^T \cos(x + \mathbf{1}) + z$.

We further experiment with a more complex nonlinear generation model, where the nonlinearity occurs after the linear combination of the variables, as opposed to the preceding case where nonlinearity is applied to the variables before linear combination. Specifically, we consider

$$x = 2\sin(A^T(x + 0.5 \cdot \mathbf{1})) + A^T(x + 0.5 \cdot \mathbf{1}) + z,$$

and plot the results in Figure 5. One sees that with higher nonlinearity, the proposed method results in significantly better SHD and FDR than does DAG-NOTEARS.

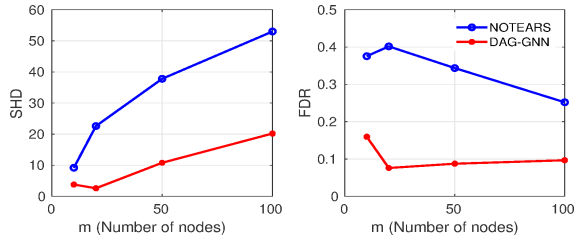

Figure 5. Structure discovery in terms of SHD and FDR to the true graph, on synthetic data set generated by $x = 2\sin(A^T(x + 0.5 \cdot \mathbf{1})) + A^T(x + 0.5 \cdot \mathbf{1}) + z$.

4.1.3. Vector-Valued Case

The proposed method offers a modeling benefit that the variables can be vector-valued with $d > 1$. Moreover, since Z resides in the latent space of the autoencoder and is not interpreted as noise as in linear SEM, one may take a smaller column dimension $d_Z < d$ if he/she believes that the variables have a lower intrinsic dimension. To demonstrate this capability, we construct a data set where the different dimensions come from a randomly scaled and perturbed sample from linear SEM. Specifically, given a graph adjacency matrix A, we first construct a sample $\tilde{x} \in \mathbb{R}^{m\times 1}$ from the linear SEM $\tilde{x} = A^T\tilde{x} + \tilde{z}$, and then generate for the k-th dimension $x^k = u^k\tilde{x} + v^k + z^k$, where u^k and v^k are random scalars from standard normal and z^k is a standard normal vector. The eventual sample is $X = [x^1|x^2|\cdots|x^d]$.

We let $d = 5$ and $d_Z = 1$ and compare DAG-GNN with DAG-NOTEARS. The SHD and FDR are plotted in Figure 6. The figure clearly shows the significantly better performance of the proposed method. Moreover, the parameter estimates are shown in Figure 7, compared against the ground-truth A. One sees that the estimated graph from DAG-GNN successfully captures all the ground truth edges and that the estimated weights are also similar. On the other hand, DAG-NOTEARS barely learns the graph.

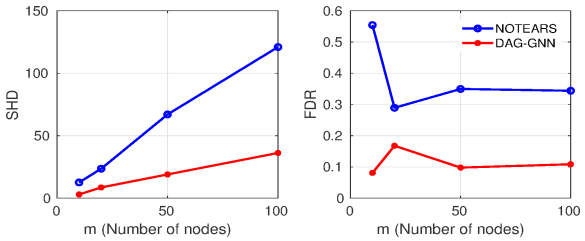

Figure 6. Structure discovery in terms of SHD and FDR to the true graph, on synthetic vector-valued data set.

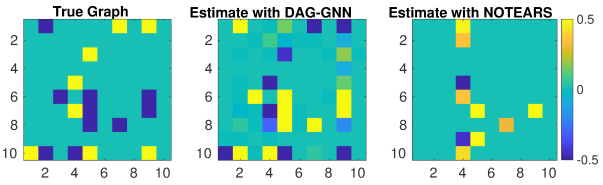

Figure 7. Parameter estimates (before thresholding) of the graph on synthetic vector-valued data set.

4.2. Benchmark Data Sets

A benefit of the proposed method is that it naturally handles discrete variables, a case precluded by linear SEM. We demonstrate the use of DAG-GNN on three discrete benchmark data sets: Child, Alarm, and Pigs (Tsamardinos et al., 2006). For comparison is the state-of-the-art exact DAG solver GOPNILP (Cussens et al., 2016), which is based on a constrained integer programming formulation. We use 1000 samples for learning.

One sees from Table 1 that our results are reasonably close to the ground truth, whereas not surprisingly the results of GOPNILP are nearly optimal. The BIC score gap exhibits by DAG-GNN may be caused by the relatively simple autoencoder architecture, which is less successful in approximating multinomial distributions. Nevertheless, it is encouraging that the proposed method as a unified frame-

work can handle discrete variables with only slight changes in the network architecture.

Table 1. BIC scores on benchmark datasets of discrete variables.

Dataset	m	Groundtruth	GOPNILP	DAG-GNN
Child	20	-1.27e+4	-1.27e+4	-1.38e+4
Alarm	37	-1.07e+4	-1.12e+4	-1.28e+4
Pigs	441	-3.48e+5	-3.50e+5	-3.69e+5

4.3. Applications

We consider a bioinformatics data set (Sachs et al., 2005) for the discovery of a protein signaling network based on expression levels of proteins and phospholipids. This is a widely used data set for research on graphical models, with experimental annotations accepted by the biological research community. The data set offers continuous measurements of expression levels of multiple phosphorylated proteins and phospholipid components in human immune system cells, and the modeled network provides the ordering of the connections between pathway components. Based on $n = 7466$ samples of $m = 11$ cell types, Sachs et al. (2005) estimate 20 edges in the graph.

In Table 2, we compare DAG-GNN with DAG-NOTEARS as well as FSG, the fast greedy search method proposed by Ramsey et al. (2017), against the ground truth offered by Sachs et al. (2005). The proposed method achieves the lowest SHD. We further show in Figure 8 our estimated graph. One observes that it is acyclic. Our method successfully learns 8 out of 20 ground-truth edges (as marked by red arrows), and predicts 5 indirectly connected edges (blue dashed arrows) as well as 3 reverse edges (yellow arrows).

Table 2. Results on protein signaling network: comparison of the predicted graphs with respect to the ground truth.

Method	SHD	# Predicted edges
FGS	22	17
NOTEARS	22	16
DAG-GNN	19	18

For another application, we develop a new causal inference task over relations defined in a knowledge base (KB) schema. The task aims at learning a BN, the nodes of which are relations and the edges indicate whether one relation suggests another. For example, the relation person/Nationality may imply person/Language, because the spoken language of a person naturally associates with his/her nationality. This task has a practical value, because most existing KBs are constructed by hand. The success of this task helps suggest meaningful relations for new entities and reduce human efforts. We construct a data set from FB15K-237 (Toutanova et al., 2015) and list in Table 3 a few extracted causal relations. Because of space limitation, we defer the details and more results in the supplementary material. One sees that these results are quite intuitive. We plan a comprehensive study with field experts to systematically evaluate the extraction results.

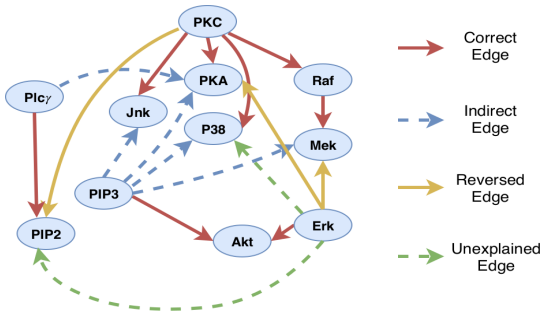

Figure 8. Estimate protein signaling network.

Table 3. Examples of extracted edges with high confidence.

film/ProducedBy	⇒	film/Country
film/ProductionCompanies	⇒	film/Country
person/Nationality	⇒	person/Languages
person/PlaceOfBirth	⇒	person/Languages
person/PlaceOfBirth	⇒	person/Nationality
person/PlaceLivedLocation	⇒	person/Nationality

5. Conclusion

DAG structure learning is a challenging problem that has long been pursued in the literature of graphical models. The difficulty, in a large part, is owing to the NP-hardness incurred in the combinatorial formulation. Zheng et al. (2018) propose an equivalent continuous constraint that opens the opportunity of using well developed continuous optimization techniques for solving the problem. In this context, we explore the power of neural networks as functional approximators and develop a deep generative model to capture the complex data distribution, aiming at better recovering the underlying DAG with a different design of the objective function. In particular, we employ the machinery of variational autoencoders and parameterize them with new graph neural network architectures. The proposed method handles not only data generated by parametric models beyond linear, but also variables in general forms, including scalar/vector values and continuous/discrete types. We have performed extensive experiments on synthetic, benchmark, and application data and demonstrated the practical competitiveness of the proposal.

Acknowledgements

Y. Yu would like to acknowledge support from the National Science Foundation under award DMS 1753031.

References

Bertsekas, D. P. *Nonlinear Programming*. Athena Scientific, 2nd edition, 1999.

Bruna, J., Zaremba, W., Szlam, A., and LeCun, Y. Spectral networks and locally connected networks on graphs. In *ICLR*, 2014.

Chen, E. Y.-J., Shen, Y., Choi, A., and Darwiche, A. Learning Bayesian networks with ancestral constraints. In *NIPS*, 2016.

Chen, J., Ma, T., and Xiao, C. FastGCN: Fast learning with graph convolutional networks via importance sampling. In *ICLR*, 2018.

Chickering, D. M. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 2002.

Chickering, D. M., Heckerman, D., and Meek, C. Large-sample learning of Bayesian networks is NP-hard. *Journal of Machine Learning Research*, 5:1287–1330, 2004.

Chow, C. and Liu, C. Approximating discrete probability distributions with dependence trees. *IEEE transactions on Information Theory*, 14(3):462–467, 1968.

Cussens, J. Bayesian network learning with cutting planes. In *UAI*, 2011.

Cussens, J., Haws, D., and Studenỳ, M. Polyhedral aspects of score equivalence in Bayesian network structure learning. *Mathematical Programming*, pp. 1–40, 2016.

Defferrard, M., Bresson, X., and Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*, 2016.

Eaton, D. and Murphy, K. Bayesian structure learning using dynamic programming and MCMC. *arXiv preprint arXiv:1206.5247*, 2012.

Friedman, N. and Koller, D. Being Bayesian about network structure. a Bayesian approach to structure discovery in Bayesian networks. *Machine learning*, 50(1-2):95–125, 2003.

Friedman, N., Goldszmidt, M., and Wyner, A. Data analysis with Bayesian networks: A bootstrap approach. In *UAI*, 1999.

Gao, T. and Wei, D. Parallel bayesian network structure learning. In *International Conference on Machine Learning*, pp. 1671–1680, 2018.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *ICML*, 2017.

Grzegorczyk, M. and Husmeier, D. Improving the structure MCMC sampler for Bayesian networks by introducing a new edge reversal move. *Machine Learning*, 71(2-3):265, 2008.

Gmez, J., Mateo, J., and Puerta, J. Learning Bayesian networks by hill climbing: efficient methods based on progressive restriction of the neighborhood. *Data Mining and Knowledge Discovery*, 22(1-2):106–148, 2011.

Hamilton, W. L., Ying, R., and Leskovec, J. Inductive representation learning on large graphs. In *NIPS*, 2017.

He, R., Tian, J., and Wu, H. Structure learning in Bayesian networks of a moderate size by efficient sampling. *Journal of Machine Learning Research*, 17(1):3483–3536, 2016.

Heckerman, D., Geiger, D., and Chickering, D. M. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine learning*, 20(3):197–243, 1995.

Jaakkola, T., Sontag, D., Globerson, A., and Meila, M. Learning Bayesian network structure using LP relaxations. 2010.

Kalainathan, D., Goudet, O., Guyon, I., Lopez-Paz, D., and Sebag, M. SAM: Structural agnostic model, causal discovery and penalized adversarial learning. *arXiv preprint arXiv:1803.04929*, 2018.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Kingma, D. P. and Welling, M. Auto-encoding variational Bayes. In *ICLR*, 2014.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

Koivisto, M. and Sood, K. Exact Bayesian structure discovery in Bayesian networks. *Journal of Machine Learning Research*, 5:549–573, 2004.

Li, Y., Tarlow, D., Brockschmidt, M., and Zemel, R. Gated graph sequence neural networks. In *ICLR*, 2016.

Madigan, D., York, J., and Allard, D. Bayesian graphical models for discrete data. *International Statistical Review/Revue Internationale de Statistique*, pp. 215–232, 1995.

Mohammadi, A., Wit, E. C., et al. Bayesian structure learning in sparse Gaussian graphical models. *Bayesian Analysis*, 10(1):109–138, 2015.

Mohan, K., Chung, M., Han, S., Witten, D., Lee, S.-I., and Fazel, M. Structured learning of Gaussian graphical models. In *NIPS*, 2012.

Mooij, J. M., Magliacane, S., and Claassen, T. Joint causal inference from multiple contexts. *arXiv preprint arXiv:1611.10351*, 2016.

Nie, S., Mauá, D. D., De Campos, C. P., and Ji, Q. Advances in learning Bayesian networks of bounded treewidth. In *NIPS*, 2014.

Niinimaki, T., Parviainen, P., and Koivisto, M. Partial order MCMC for structure discovery in Bayesian networks. *arXiv preprint arXiv:1202.3753*, 2012.

Niinimäki, T. M. and Koivisto, M. Annealed importance sampling for structure learning in Bayesian networks. In *IJCAI*, 2013.

Ott, S., Imoto, S., and Miyano, S. Finding optimal models for small gene networks. In *Pacific symposium on biocomputing*, 2004.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in PyTorch. 2017.

Pearl, J. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers, Inc., 2 edition, 1988.

Pearl, J. Causality: models, reasoning, and inference. *Econometric Theory*, 19(46):675–685, 2003.

Ramsey, J., Glymour, M., Sanchez-Romero, R., and Glymour, C. A million variables and more: the fast greedy equivalence search algorithm for learning high-dimensional graphical causal models, with an application to functional magnetic resonance images. *International Journal of Data Science and Analytics*, 3(2):121–129, 2017.

Sachs, K., Perez, O., Peer, D., Lauffenburger, D. A., and Nolan, G. P. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308(5721): 523–529, 2005.

Scanagatta, M., de Campos, C. P., Corani, G., and Zaffalon, M. Learning Bayesian networks with thousands of variables. In *NIPS*, 2015.

Silander, T. and Myllymaki, P. A simple approach for finding the globally optimal Bayesian network structure. In *UAI*, 2006.

Singh, A. P. and Moore, A. W. Finding optimal Bayesian networks by dynamic programming. Technical report, Carnegie Mellon University, 2005.

Spirtes, P., Meek, C., and Richardson, T. Causal inference in the presence of latent variables and selection bias. In *UAI*, 1995.

Spirtes, P., Glymour, C. N., and Scheines, R. *Computation, Causation, and Discovery*. AAAI Press, 1999.

Spirtes, P., Glymour, C., Scheines, R., Kauffman, S., Aimale, V., and Wimberly, F. Constructing Bayesian network models of gene expression networks from microarray data, 2000a.

Spirtes, P., Glymour, C. N., Scheines, R., Heckerman, D., Meek, C., Cooper, G., and Richardson, T. *Causation, prediction, and search*. MIT press, 2000b.

Toutanova, K., Chen, D., Pantel, P., Poon, H., Choudhury, P., and Gamon, M. Representing text for joint embedding of text and knowledge bases. In *EMNLP*, 2015.

Tsamardinos, I., Aliferis, C. F., and Statnikov, A. Time and sample efficient discovery of markov blankets and direct causal relations. In *SIGKDD*, 2003.

Tsamardinos, I., Brown, L., and Aliferis, C. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65(1):31–78, 2006.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph attention networks. In *ICLR*, 2018.

Yuan, C. and Malone, B. Learning optimal Bayesian networks: A shortest path perspective. 48:23–65, 2013.

Zhang, J. On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias. *Artificial Intelligence*, 172(16-17): 1873–1896, 2008.

Zheng, X., Aragam, B., Ravikumar, P., and Xing, E. P. DAGs with NO TEARS: Continuous optimization for structure learning. In *NIPS*, 2018.

A. Proofs

Proof of Theorem 1. Let $B = A \circ A$. Clearly, B is nonnegative. The binomial expansion reads

$$(I + \alpha B)^m = I + \sum_{k=1}^{m} \binom{m}{k} \alpha^k B^k.$$

It is known that there is a cycle of length k if and only if $\mathrm{tr}(B^k) > 0$ when $B \geq 0$. Because if there is a cycle then there is a cycle of length at most m, we conclude that there is no cycle if and only if $\mathrm{tr}[(I + \alpha B)^m] = \mathrm{tr}(I) = m$. □

Proof of Theorem 2. Write

$$(1 + \alpha|\lambda|)^m = \left(1 + \frac{c|\lambda|}{m}\right)^m.$$

For given c and $|\lambda|$, the right-hand side of the equality is a function of m. This function monotonically increases for positive m and has a limit $e^{c|\lambda|}$. Hence, for any finite $m > 0$, $(1 + \alpha|\lambda|)^m \leq e^{c|\lambda|}$. □

B. Structure Learning over KB Relations

We construct the data set from triples in FB15K-237 (Toutanova et al., 2015), which is a subset of FreeBase with approximately 15k entities and 237 relations. Each sample corresponds to an entity and each variable corresponds to a relation in this knowledge base. Each sample has on average 7.36 relations (i.e. 7.36 non-zero entries in each row).

Table 4 gives additional examples learned by our model with highest confidence scores. For each target relation on the right-hand side, we show the highest ranked relations within the same domain (i.e. the contents in the field before "/" such as "film" and "tvProgram"). On the left-hand side, we omit the relations that are common to the associated entity types, e.g. "profession" and "gender" to persons and "genre" to films, because almost all entities with these types will contain such a relation.

Table 4. (Continued from Table 3) Examples of extracted edges with high confidence. The dot $\cdot$ appearing in $R_1.R_2$ means that the sample entity is connected to a virtual node (i.e. compound value types in FreeBase) via relation R_1, followed by a relation R_2 to a real entity.

film/ProducedBy	$\Rightarrow$	film/Country
film/ProductionCompanies	$\Rightarrow$	film/Country
tvProgram/CountryOfOriginal	$\Rightarrow$	tvProgram/Language
tvProgram/RegularCast.regularTv/AppearanceActor	$\Rightarrow$	tvProgram/Language
person/Nationality	$\Rightarrow$	person/Languages
person/PlaceOfBirth	$\Rightarrow$	person/Languages
person/PlaceOfBirth	$\Rightarrow$	person/Nationality
person/PlaceLivedLocation	$\Rightarrow$	person/Nationality
organization/Headquarters.mailingAddress/Citytown	$\Rightarrow$	organization/PlaceFounded
organization/Headquarters.mailingAddress/StateProvince	$\Rightarrow$	organization/PlaceFounded