Learning to Discover Sparse Graphical Models

Solodkin Vladimir

MIPT, 2024

Learning to Discover Sparse Graphical Models

Problem formulation: structure discovery of undirected graphical models from observational data

Given the

1. $\mathbf{X} \in \mathbb{R}^{n \times p}$ - matrix with n rows of i.i.d. samples $x \sim P(x)$
2. $G = (V, E)$ - undirected graph associated with set of variables in x
3. $Y \in \mathcal{L}^{N_e}$, where $\mathcal{L} = \{0, 1\}$ and $N_e = \frac{1}{2}p(p-1)$ - indicator of presence or absence of edges in G

our goal is to find $g_w(\mathbf{X})$, which predicts the edge structure $\hat{Y} = g_w(\mathbf{X})$.

Learning to Discover Sparse Graphical Models

Gaussian case

The empirical covariance matrix $\hat{\Sigma}$ happens to be a sufficient statistic of the population covariance (therefore of the conditional dependency structure).

Problem reformulation

$$\text{Find } g_w(\mathbf{X}) := f_w(\hat{\mathbf{\Sigma}}) \text{ such that } f_w \text{ is parametrized by } w \text{ and } f_w \in \mathcal{F}.$$

Learning to Discover Sparse Graphical Models

What was before?

A standard approach to estimating structure with GGMs in high dimensions is based on the classic result that the zeros of a precision matrix correspond to zero partial correlation, a necessary and sufficient condition for conditional independence. Assuming only a few conditional dependencies corresponds to a sparsity constraint on the entries of the precision matrix, leading to a combinatorial problem.

Typical objective:

$$f_{gl}(\hat{\boldsymbol{\Sigma}}) = \arg\min_{\boldsymbol{\Theta} \succ 0} - \log |\boldsymbol{\Theta}| + Tr(\hat{\boldsymbol{\Sigma}}\boldsymbol{\Theta}) + \lambda \|\boldsymbol{\Theta}\|_1 - \text{ graphical lasso,}$$

which can be seen as a penalized maximum-likelihood estimator.

Learning to Discover Sparse Graphical Models

Current approach:

Defining a distribution $\mathbb{P}$ on $\mathbb{R}^{p \times p} \times \mathcal{L}^{N_e}$ such that $(\hat{\boldsymbol{\Sigma}}, Y) \sim \mathbb{P}$, we would like our estimator f_w to minimize the expected risk:

$$R(f) = \mathbb{E}_{(\hat{\boldsymbol{\Sigma}}, Y) \sim \mathbb{P}}[l(f(\hat{\boldsymbol{\Sigma}}), Y),$$

where $l : \mathcal{L}^{N_e} \times \mathcal{L}^{N_e} \rightarrow \mathbb{R}^+$ is the loss function.

It is sufficient to define a rich enough $\mathcal{F}$ over which we can minimize the empirical risk over the samples generated, giving us a learning objective over N samples $\{Y_k, \boldsymbol{\Sigma}_k\}_{k=1}^N$ drawn from $\mathbb{P}$:

$$\min_w \frac{1}{N} \sum\nolimits_{k=1}^{N} \hat{l}(f_w(\hat{\boldsymbol{\Sigma}}_k), Y_k),$$

where $\hat{l}$ is a convex surrogate of l, e.g., cross-entropy loss.

Learning to Discover Sparse Graphical Models

The process of learning f_w for sparse GMM is given below

Algorithm 1 Training a GMM edge estimator

```
for $i \in \{1, \ldots, N\}$ do
    Sample $G_i \sim \mathbb{P}(G)$
    Sample $\mathbf{\Theta}_i \sim \mathbb{P}(\mathbf{\Theta} | G = G_i)$
    $\mathbf{X} \leftarrow \{x_j \sim N(0, \mathbf{\Theta}_i^{-1})\}_{j=1}^{n}$
    Construct $(Y_i, \hat{\mathbf{\Sigma}}_i)$ pair from $(G_i, \mathbf{X}_i)$
end for
Select function class $\mathcal{F}$ (e.g. CNN)
Optimize: $\min_{f \in \mathcal{F}} \frac{1}{N} \sum_{k=1}^{N} \hat{l}(f(\hat{\mathbf{\Sigma}}_k), Y_k)$
```

Learning to Discover Sparse Graphical Models

Sampling procedure

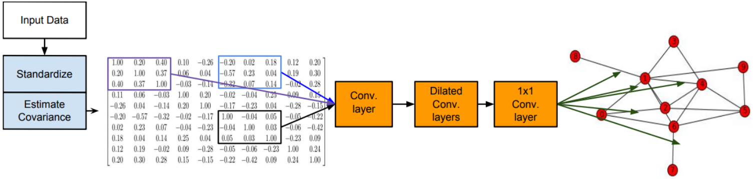
We first construct a lower diagonal matrix, L, where each entry has α probability of being zero. Non-zero entries are set uniformly between $-c$ and c. Multiplying LL^T gives a sparse positive definite precision matrix, $\mathbf{\Theta}$. This gives us our $P(\mathbf{\Theta}|G)$ with a sparse prior on $P(G)$. We sample from the Gaussian $N(0, \mathbf{\Theta}^{-1})$ to obtain samples of $\mathbf{X}$. Here α corresponds to a specific sparsity level in the final precision matrix.

Additional features

We use batch normalization at each layer. Additionally, we found that using the absolute value of the true partial correlations as labels, instead of hard binary labels, improves results.

Learning to Discover Sparse Graphical Models

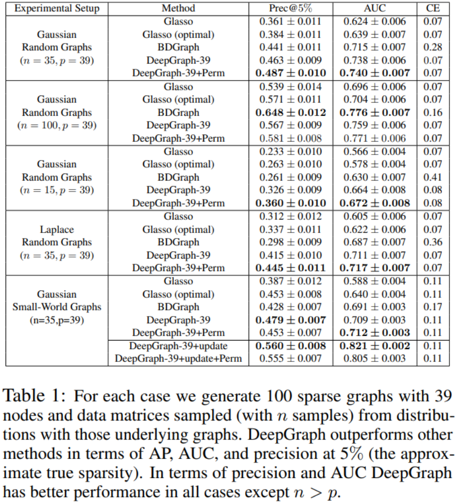
Proposed architecture



$$\begin{bmatrix}
1.00 & 0.20 & 0.40 & 0.10 & -0.26 & -0.20 & 0.02 & 0.18 & 0.12 & 0.20 \\
0.20 & 1.00 & 0.37 & 0.06 & 0.04 & -0.57 & 0.23 & 0.04 & 0.19 & 0.30 \\
0.40 & 0.37 & 1.00 & -0.03 & -0.14 & -0.32 & 0.07 & 0.14 & -0.02 & 0.28 \\
0.11 & 0.06 & -0.03 & 1.00 & 0.20 & -0.02 & -0.04 & 0.25 & 0.09 & 0.15 \\
-0.26 & 0.04 & -0.14 & 0.20 & 1.00 & -0.17 & -0.23 & 0.04 & -0.28 & -0.15 \\
-0.20 & -0.57 & -0.32 & -0.02 & -0.17 & 1.00 & -0.04 & 0.05 & -0.05 & -0.22 \\
0.02 & 0.23 & 0.07 & -0.04 & -0.23 & -0.04 & 1.00 & 0.03 & -0.06 & -0.42 \\
0.18 & 0.04 & 0.14 & 0.25 & 0.04 & 0.05 & 0.03 & 1.00 & -0.23 & 0.09 \\
0.12 & 0.19 & -0.02 & 0.09 & -0.28 & -0.05 & -0.06 & -0.23 & 1.00 & 0.24 \\
0.20 & 0.30 & 0.28 & 0.15 & -0.15 & -0.22 & -0.42 & 0.09 & 0.24 & 1.00
\end{bmatrix}$$

Figure 2: Diagram of the DeepGraph structure discovery architecture used in this work. The input is first standardized and then the sample covariance matrix is estimated. A neural network consisting of multiple dilated convolutions (Yu & Koltun, 2015) and a final 1×1 convolution layer is used to predict edges corresponding to non-zero entries in the precision matrix.

Results



Experimental Setup	Method	Prec@5%	AUC	CE
Gaussian Random Graphs ($n = 35, p = 39$)	Glasso	0.361 ± 0.011	0.624 ± 0.006	0.07
	Glasso (optimal)	0.384 ± 0.011	0.639 ± 0.007	0.07
	BDGraph	0.441 ± 0.011	0.715 ± 0.007	0.28
	DeepGraph-39	0.463 ± 0.009	0.738 ± 0.006	0.07
	DeepGraph-39+Perm	$\mathbf{0.487 \pm 0.010}$	$\mathbf{0.740 \pm 0.007}$	0.07
Gaussian Random Graphs ($n = 100, p = 39$)	Glasso	0.539 ± 0.014	0.696 ± 0.006	0.07
	Glasso (optimal)	0.571 ± 0.011	0.704 ± 0.006	0.07
	BDGraph	$\mathbf{0.648 \pm 0.012}$	$\mathbf{0.776 \pm 0.007}$	0.16
	DeepGraph-39	0.567 ± 0.009	0.759 ± 0.006	0.07
	DeepGraph-39+Perm	0.581 ± 0.008	0.771 ± 0.006	0.07
Gaussian Random Graphs ($n = 15, p = 39$)	Glasso	0.233 ± 0.010	0.566 ± 0.004	0.07
	Glasso (optimal)	0.263 ± 0.010	0.578 ± 0.004	0.07
	BDGraph	0.261 ± 0.009	0.630 ± 0.007	0.41
	DeepGraph-39	0.326 ± 0.009	0.664 ± 0.008	0.08
	DeepGraph-39+Perm	$\mathbf{0.360 \pm 0.010}$	$\mathbf{0.672 \pm 0.008}$	0.08
Laplace Random Graphs ($n = 35, p = 39$)	Glasso	0.312 ± 0.012	0.605 ± 0.006	0.07
	Glasso (optimal)	0.337 ± 0.011	0.622 ± 0.006	0.07
	BDGraph	0.298 ± 0.009	0.687 ± 0.007	0.36
	DeepGraph-39	0.415 ± 0.010	0.711 ± 0.007	0.07
	DeepGraph-39+Perm	$\mathbf{0.445 \pm 0.011}$	$\mathbf{0.717 \pm 0.007}$	0.07
Gaussian Small-World Graphs (n=35,p=39)	Glasso	0.387 ± 0.012	0.588 ± 0.004	0.11
	Glasso (optimal)	0.453 ± 0.008	0.640 ± 0.004	0.11
	BDGraph	0.428 ± 0.007	0.691 ± 0.003	0.17
	DeepGraph-39	$\mathbf{0.479 \pm 0.007}$	0.709 ± 0.003	0.11
	DeepGraph-39+Perm	0.453 ± 0.007	$\mathbf{0.712 \pm 0.003}$	0.11
	DeepGraph-39+update	$\mathbf{0.560 \pm 0.008}$	$\mathbf{0.821 \pm 0.002}$	0.11
	DeepGraph-39+update+Perm	0.555 ± 0.007	0.805 ± 0.003	0.11

Table 1: For each case we generate 100 sparse graphs with 39 nodes and data matrices sampled (with n samples) from distributions with those underlying graphs. DeepGraph outperforms other methods in terms of AP, AUC, and precision at 5% (the approximate true sparsity). In terms of precision and AUC DeepGraph has better performance in all cases except $n > p$.

Summary

1. An estimator for determining the structure of an undirected graphical model was introduced.
2. A network architecture and sampling procedure for learning such an estimator for the case of sparse GGMs was proposed.
3. Empirical results show that the proposed method works particularly well compared to other approaches.