Principle of Relevant Information for Graph Sparsification (Supp. Material)

Shujian Yu[1] **Francesco Alesiani**[2] **Wenzhe Yin**[3] **Robert Jenssen**[1,5,6] **Jose C. Principe**[4]

[1]UiT - The Arctic University of Norway, Norway
[2]NEC Laboratories Europe, Germany
[3]University of Amsterdam, Netherlands
[4]University of Florida, USA
[5]Norwegian Computing Center, Norway
[6]University of Copenhagen, Denmark

A PROOFS AND ADDITIONAL INFORMATION

A.1 ADDITIONAL INFORMATION ON THE RIGOR OF ASSUMPTION 1

Assumption 1. *Given an undirected graph $G = \{V, E\}$, let $G' = G + \{u, v\}$, where $V(G) = V(G')$ and $E(G) = E(G') \cup \{u, v\}$, we have $S_{vN}(L_{G'}) \geq S_{vN}(L_G)$, i.e., there exists a strictly monotonically increasing relationship between the number of edges $|G|$ and the von Neumann entropy $S_{vN}(L_G)$.*

Note that, one can find counterexamples about Assumption 1. The question of whether the factor $\frac{d_{G'}-2}{d_{G'}}$ can be removed from Eq. (12) in Theorem 1 was investigated in [Dairyko et al., 2017]. The answer is negative and adding an edge may decrease the von Neumann entropy slightly. For example, $K_{2,n-2}$ graph ($n > 5$) satisfies $S_{vN}(K_{2,n-2}) > S_{vN}(K_{2,n-2} + e)$ after adding edge e.

However, the Assumption 1 does hold for most of edges. To further corroborate our argument, we performed an additional experiment, where we generated a set of random graphs with 20 nodes by the Erdös-Rényi (ER) model, with the average node degree $\bar{d}$ of roughly 1.6, 2.5, 3, 4 and 5, respectively. For each graph, we add one edge to the original graph and re-evaluate the von Neumann entropy $S_{\text{vN}}(L)$. We traverse all possible edges and calculate the percentage that the difference of $S_{\text{vN}}(L)$ is non-negative before and after edge addition. As can be seen from the following table, we have more than 85% confidence that Assumption 1 holds. We made similar observations also for large graph with 200 nodes.

Table 1: The percentage that adding one edge may increase the von Neumann entropy for a random graph with 20 nodes generated by the Erdös-Rényi (ER) model.

Degree	1.6	2.5	3	4	5
Percentage (%)	95.65	88.57	88.82	88.13	87.25

A.2 PROOF TO COROLLARY 1

Corollary 1. *Under Assumption 1, suppose $G_s = \{V_s, E_s\}$ is a sparse graph obtained from $G = \{V, E\}$ (by removing edges), let $G'_s = G_s + \{u, v\}$, where $\{u, v\}$ is an edge from the original graph G, $V(G_s) = V(G'_s)$ and $E(G_s) = E(G'_s) \cup \{u, v\}$, we have $D_{QJS}(L_{G'_s} \| L_G) \leq D_{QJS}(L_{G_s} \| L_G)$, i.e., adding an edge is prone to decrease the QJS divergence.*

Before proving Corollary 1, we first present Lemma 1, Lemma 2 and Lemma 3. The proof of Corollary 1 is based on the results described in Lemma 3, which is based on the conclusion of Lemma 2. We present Lemma 1 as a more restrictive version of Lemma 2.

Lemma 1. *Let $\boldsymbol{\lambda} = \{\lambda_i\}$ be the eigenvalues of (trace normalized) Graph Laplacian $\tilde{L}$. Suppose the i-th eigenvalue λ_i has a minor and negligible increase and the remaining eigenvalues decrease proportionately to their existing values, such that*

Accepted for the 38th Conference on Uncertainty in Artificial Intelligence (UAI 2022).

$\sum_i \lambda_i = 1$. *Then, the total derivative of von Neumann entropy* $S(\boldsymbol{\lambda}) = -\sum_{i=1}^{N} \lambda_i \log_2 \lambda_i$ *with respect to* λ_i *is given by:*

$$\frac{dS}{d\lambda_i} = \frac{\partial S}{\partial \lambda_i} + \sum_{i \neq j} \frac{\partial S}{\partial \lambda_j} \frac{d\lambda_j}{d\lambda_i} = -\frac{S(\boldsymbol{\lambda}) + \log_2 \lambda_i}{1 - \lambda_i}. \tag{1}$$

Proof. For simplicity, suppose we increase the element λ_i up to the value $\lambda_i + d\lambda_i$ for some infinitesimally small value $d\lambda_i$. As we increase this element, we decrease all the other elements proportionately to their values so that the constraint holds. Thus, for some infinitesimal value δ we update $\boldsymbol{\lambda}$ as:

$$\begin{aligned}
\lambda_1 &\mapsto \lambda_1(1-\delta) \\
&\vdots \\
\lambda_{i-1} &\mapsto \lambda_{i-1}(1-\delta) \\
\lambda_{i+1} &\mapsto \lambda_{i+1}(1-\delta) \\
&\vdots \\
\lambda_N &\mapsto \lambda_N(1-\delta)
\end{aligned}$$

or in general $\lambda_j \mapsto \lambda_j(1-\delta), j \neq i$. Due to the constraint $\sum_i \lambda_i = 1$, or

$$\begin{aligned}
\sum_{j \neq i} \lambda_j(1-\delta) + \lambda_i + d\lambda_i &= 1 \\
(1-\lambda_i)(1-\delta) + \lambda_i + d\lambda_i &= 1 \\
1 - \lambda_i - (1-\lambda_i)\delta + \lambda_i + d\lambda_i &= 1 \\
-(1-\lambda_i)\delta + d\lambda_i &= 0
\end{aligned}$$

thus we have:

$$\delta = d\lambda_i/(1-\lambda_i). \tag{2}$$

Then,

$$\frac{d\lambda_j}{d\lambda_i} = \frac{-\lambda_j \delta}{d\lambda_i} = -\frac{\lambda_j}{1-\lambda_i}, \quad j \neq i. \tag{3}$$

Therefore, the total derivative of von Neumann entropy $S(\boldsymbol{\lambda}) = -\sum_{i=1}^{N} \lambda_i \log_2 \lambda_i$ with respect to λ_i is given by:

$$\begin{aligned}
\frac{dS}{d\lambda_i} &= \frac{\partial S}{\partial \lambda_i} + \sum_{i \neq j} \frac{\partial S}{\partial \lambda_j} \frac{d\lambda_j}{d\lambda_i} \\
&= -(\frac{1}{\ln 2} + \log_2 \lambda_i) + \frac{1}{1-\lambda_i} \sum_{i \neq j} \lambda_j \left(\frac{1}{\ln 2} + \log_2 \lambda_j \right) \\
&= -\frac{(1-\lambda_i)\log_2 \lambda_i}{1-\lambda_i} + \frac{1}{1-\lambda_i} \sum_{i \neq j} \lambda_j \log_2 \lambda_j \\
&= -\frac{1}{1-\lambda_i} \{ \sum_{i=1}^{N} -\lambda_i \log_2 \lambda_i + \log_2 \lambda_i \} \\
&= -\frac{1}{1-\lambda_i} \{ S(\boldsymbol{\lambda}) + \log_2 \lambda_i \}
\end{aligned} \tag{4}$$

□

In addition of the previous lemma, we show a more general form (without assuming that all other eigenvalues decrease proportionately to their values).

Lemma 2. *Let $\boldsymbol{\lambda} = \{\lambda_i\}$ be the eigenvalues of (trace normalized) Graph Laplacian $\tilde{L}$. Then, the directional total derivative of von Neumann entropy $S(\boldsymbol{\lambda}) = -\sum_{i=1}^{N} \lambda_i \log_2 \lambda_i$ with respect to λ_i along a change in the eigenvalues defined by $\boldsymbol{v}$ such that $\boldsymbol{\lambda}' = \boldsymbol{\lambda} + \delta\boldsymbol{v}$, for $\delta \to 0$ is given by:*

$$\begin{aligned}\frac{dS}{d\lambda_i}|_{\boldsymbol{v}} &= \frac{\partial S}{\partial \lambda_i} + \sum_{j\neq i} \frac{\partial S}{\partial \lambda_j}\frac{d\lambda_j}{d\lambda_i}|_{\boldsymbol{v}} \\ &= -\frac{1}{v_i}\sum_j v_j \log_2 \lambda_j = -\frac{1}{v_i}\,\mathbb{E}_{\boldsymbol{v}} \log_2 \boldsymbol{\lambda}\end{aligned} \tag{5}$$

where $\sum_j v_j = 0$ and $0 \leq v_j + \lambda_j \leq 1$ is the directional variation of the eigenvalues $\boldsymbol{\lambda}' = \boldsymbol{\lambda} + \delta\boldsymbol{v}$. We denoted $\mathbb{E}_{\boldsymbol{v}}$ the sum over the element of $\boldsymbol{v}$ (i.e. $\mathbb{E}_{\boldsymbol{v}} \log_2 \boldsymbol{\lambda} = \sum_j v_j \log_2 \lambda_j$). In compact form (and with the abuse of the expectation operator)

$$\frac{dS}{d\boldsymbol{\lambda}}|_{\boldsymbol{v}} = -\boldsymbol{v}^{-1}\,\mathbb{E}_v \log_2 \boldsymbol{\lambda} \tag{6}$$

where the inverse of the vector $\boldsymbol{v}$ is element-wise.

Proof. We first observe that

$$\frac{d\lambda_j}{d\lambda_i}|_{\boldsymbol{v}} = \frac{\delta v_j}{\delta v_i} = \frac{v_j}{v_i}, \quad j \neq i.$$

for the definition of the variation. Therefore, the total derivative of von Neumann entropy $S(\boldsymbol{\lambda}) = -\sum_{i=1}^{N} \lambda_i \log_2 \lambda_i$ with respect to λ_i,along the direction $\boldsymbol{v}$, is given by:

$$\begin{aligned}\frac{dS}{d\lambda_i}|_{\boldsymbol{v}} &= \frac{\partial S}{\partial \lambda_i} + \sum_{j\neq i} \frac{\partial S}{\partial \lambda_j}\frac{d\lambda_j}{d\lambda_i}|_{\boldsymbol{v}} \\ &= -(\frac{1}{\ln 2} + \log_2 \lambda_i) - \sum_{j\neq i}\frac{v_j}{v_i}\left(\frac{1}{\ln 2} + \log_2 \lambda_j\right) \\ &= -\frac{1}{\ln 2} - \log_2 \lambda_i - \frac{1}{v_i}\sum_{j\neq i} v_j \left(\frac{1}{\ln 2} + \log_2 \lambda_j\right) \\ &= -\frac{1}{\ln 2} - \log_2 \lambda_i - \frac{1}{v_i}\sum_{j\neq i} v_j \frac{1}{\ln 2} - \frac{1}{v_i}\sum_{j\neq i} v_j \log_2 \lambda_j \\ &= -\frac{1}{\ln 2} - \log_2 \lambda_i - \frac{1}{v_i}\frac{-v_i}{\ln 2} - \frac{1}{v_i}\sum_{j\neq i} v_j \log_2 \lambda_j \\ &= -\frac{1}{\ln 2} - \log_2 \lambda_i + \frac{1}{\ln 2} - \frac{1}{v_i}\sum_{j\neq i} v_j \log_2 \lambda_j \\ &= -\log_2 \lambda_i - \frac{1}{v_i}\sum_{j\neq i} v_j \log_2 \lambda_j \\ &= -\frac{v_i}{v_i}\log_2 \lambda_i - \frac{1}{v_i}\sum_{j\neq i} v_j \log_2 \lambda_j \\ &= -\frac{1}{v_i}\sum_j v_j \log_2 \lambda_j\end{aligned}$$

□

Lemma 3. *Let $\boldsymbol{\lambda} = \{\lambda_i\}$ be the eigenvalues of (trace normalized) Graph Laplacian $\tilde{L}$ whose von Neumann entropy is $S(\boldsymbol{\lambda}) = -\sum_{i=1}^{N} \lambda_i \log_2 \lambda_i$. Let $\boldsymbol{\lambda}' = \boldsymbol{\lambda} + \boldsymbol{v}$, such that $\sum_i v_i = 0$ and $0 \leq v_j + \lambda_j \leq 1$ and $S(\boldsymbol{\lambda}') \geq S(\boldsymbol{\lambda})$, then*

$$\mathbb{E}_{\boldsymbol{v}} \log_2 \boldsymbol{\lambda}' \geq \mathbb{E}_{\boldsymbol{v}} \log_2 \boldsymbol{\lambda} \tag{7}$$

and

$$\boldsymbol{v}\frac{dS}{d\boldsymbol{\lambda}}(\boldsymbol{\lambda}')|_{\boldsymbol{v}} \leq \boldsymbol{v}\frac{dS}{d\boldsymbol{\lambda}}(\boldsymbol{\lambda})|_{\boldsymbol{v}} \tag{8}$$

Proof. From the definition we consider $\boldsymbol{v} = \boldsymbol{\lambda}' - \boldsymbol{\lambda}$, thus, omitting the vector indices and using vector notation, where operations are performed element-wise and sum is over the elements of the resulting vector:

$$\begin{aligned}\mathbb{E}_{\boldsymbol{v}} \log_2 \boldsymbol{\lambda} &= \sum_j (\boldsymbol{\lambda}'_j - \boldsymbol{\lambda}_j) \log_2 \boldsymbol{\lambda}_j \\ &= \sum_j \boldsymbol{\lambda}'_j \log_2 \boldsymbol{\lambda}_j - \sum \boldsymbol{\lambda}_j \log_2 \boldsymbol{\lambda}_j \\ &= \sum_j \boldsymbol{\lambda}'_j \log_2 \boldsymbol{\lambda}_j + S(\boldsymbol{\lambda})\end{aligned}$$

similarly

$$\begin{aligned}\mathbb{E}_{\boldsymbol{v}} \log_2 \boldsymbol{\lambda}' &= \sum_j (\boldsymbol{\lambda}'_j - \boldsymbol{\lambda}_j) \log_2 \boldsymbol{\lambda}'_j \\ &= \sum_j \boldsymbol{\lambda}'_j \log_2 \boldsymbol{\lambda}'_j - \sum_j \boldsymbol{\lambda}_j \log_2 \boldsymbol{\lambda}'_j \\ &= -S(\boldsymbol{\lambda}') - \sum_j \boldsymbol{\lambda}_j \log_2 \boldsymbol{\lambda}'_j\end{aligned}$$

the difference is

$$\begin{aligned}\mathbb{E}_v \log_2 \boldsymbol{\lambda}' - \mathbb{E}_v \log_2 \boldsymbol{\lambda} = &-S(\boldsymbol{\lambda}') - S(\boldsymbol{\lambda}) \\ &- \sum_j \boldsymbol{\lambda}_j \log_2 \boldsymbol{\lambda}'_j - \sum_j \boldsymbol{\lambda}'_j \log_2 \boldsymbol{\lambda}_j \geq 0\end{aligned}$$

which is the sum of two non-negative terms:

$$\begin{aligned}-S(\boldsymbol{\lambda}) - \sum_j \boldsymbol{\lambda}_j \log_2 \boldsymbol{\lambda}'_j \geq 0 \\ -S(\boldsymbol{\lambda}') - \sum_j \boldsymbol{\lambda}'_j \log_2 \boldsymbol{\lambda}_j \geq 0\end{aligned}$$

The last two inequalities follow from the property of the KL divergence, indeed we use the inequality $D(\boldsymbol{p}||\boldsymbol{q}) = \sum_i p_i \log_2 \frac{p_i}{q_i} = \mathbb{E}_{\boldsymbol{p}} \log_2 \boldsymbol{p} - \mathbb{E}_{\boldsymbol{p}} \log_2 \boldsymbol{q} = -H(\boldsymbol{p}) - \mathbb{E}_{\boldsymbol{p}} \log_2 \boldsymbol{q} \geq 0$.

Since for Lemma 2

$$\frac{dS}{d\boldsymbol{\lambda}}(\boldsymbol{\lambda})|_{\boldsymbol{v}} = -\boldsymbol{v}^{-1} \mathbb{E}_v \log_2 \boldsymbol{\lambda},$$

it follows from the first property ($\mathbb{E}_{\boldsymbol{v}} \log_2 \boldsymbol{\lambda}' \geq \mathbb{E}_{\boldsymbol{v}} \log_2 \boldsymbol{\lambda}$) that

$$\boldsymbol{v} \frac{dS}{d\boldsymbol{\lambda}}(\boldsymbol{\lambda}')|_{\boldsymbol{v}} \leq \boldsymbol{v} \frac{dS}{d\boldsymbol{\lambda}}(\boldsymbol{\lambda})|_{\boldsymbol{v}}$$

where the comparison is element wise, i.e.

$$\boldsymbol{v}_i \frac{dS}{d\boldsymbol{\lambda}_i}(\boldsymbol{\lambda}')|_{\boldsymbol{v}} \leq \boldsymbol{v}_i \frac{dS}{d\boldsymbol{\lambda}_i}(\boldsymbol{\lambda})|_{\boldsymbol{v}}$$

□

Now, we present proof to Corollary 1.

Proof. Suppose the addition of an edge makes the i-th eigenvalue λ_i has a minor change $\Delta\lambda_i$. By first-order approximation, we have:

$$S_{\text{vN}}\left(\frac{L_G + L_{G'_S}}{2}\right) - S_{\text{vN}}\left(\frac{L_G + L_{G_S}}{2}\right) \approx \frac{1}{2} \frac{dS_{\text{vN}}(L_{\bar{G}})}{d\lambda_i} \Delta\lambda_i, \tag{9}$$

and

$$S_{\text{vN}}(L_{G'_S}) - S_{vN}(L_{G_S}) \approx \frac{dS_{\text{vN}}(L_{G_s})}{d\lambda_i}\Delta\lambda_i, \tag{10}$$

in which $\frac{dS}{d\lambda_i}$ is the total derivative of von Neumann entropy $S(\boldsymbol{\lambda}) = -\sum_{i=1}^{N}\lambda_i \log_2 \lambda_i$ with respect to λ_i.

By Assumption 1, we have $S_{\text{vN}}(L_{\bar{G}}) \geq S_{\text{vN}}(L_{G_s})$. By applying Lemma 3, we obtain:

$$\frac{dS_{\text{vN}}(L_{\bar{G}})}{d\lambda_i}\Delta\lambda_i \leq \frac{dS_{\text{vN}}(L_{G_s})}{d\lambda_i}\Delta\lambda_i. \tag{11}$$

along the direction $\boldsymbol{v} = \boldsymbol{\lambda}(L_{\bar{G}}) - \boldsymbol{\lambda}(L_{G_s})$, where $\boldsymbol{\lambda}(L_{\bar{G}})$ and $\boldsymbol{\lambda}(L_{G_s})$ are the eigenvalues of the two normalized Graph Laplacian matrices. We used the variation $v_i = \Delta\lambda_i$ of Lemma 3.

Combining Eqs. (9) to (11), we get:

$$\begin{aligned} &S_{\text{vN}}\left(\frac{L_G + L_{G'_S}}{2}\right) - S_{\text{vN}}\left(\frac{L_G + L_{G_S}}{2}\right) \\ &\qquad \leq \frac{1}{2}\left(S_{\text{vN}}(L_{G'_S}) - S_{\text{vN}}(L_{G_S})\right). \end{aligned} \tag{12}$$

Thus,

$$\begin{aligned} &S_{\text{vN}}\left(\frac{L_G + L_{G'_S}}{2}\right) - \frac{1}{2}\left(S_{\text{vN}}(L_{G'_S}) + S_{vN}(L_G)\right) \\ &\leq S_{\text{vN}}\left(\frac{L_G + L_{G_S}}{2}\right) - \frac{1}{2}\left(S_{\text{vN}}(L_{G_S}) + S_{vN}(L_G)\right), \end{aligned} \tag{13}$$

which completes the proof.

□

A.3 ADDITIONAL INFORMATION AND PROOF TO THEOREM 4

Theorem 4. *The gradient of Eq. (11) in the main manuscript with respect to edge selection vector* $\mathbf{w}$ *is:*

$$\nabla_{\mathbf{w}}\mathcal{J}_{\textit{Graph-PRI}} = Ug, \tag{14}$$

where $\tilde{\mathbf{w}}$ *is the normalised* $\mathbf{w}$ *(*$\tilde{\mathbf{w}} = \mathbf{w}/\sum_{i=1}^{M} w_i$*),* $\tilde{\mathbf{1}}_M = \frac{1}{M}\mathbf{1}_M$ *is the normalized version of the all-ones vector.* $\bar{\sigma}_{\mathbf{w}} = \frac{1}{2}\left(\tilde{\sigma}_{\mathbf{w}} + \tilde{\rho}\right) = \frac{1}{2}B\,\text{diag}\left(\tilde{\mathbf{w}} + \tilde{\mathbf{1}}_M\right)B^T$*.* $g = -\,\text{diag}\left(B^T\left[(1-\beta)\ln\tilde{\sigma}_{\mathbf{w}} + \beta\ln\bar{\sigma}_{\mathbf{w}}\right]B\right)$ *and* $U = \{u_{ij}\} \in \mathbb{R}^{M\times M}, u_{ij} = -\frac{\tilde{w}_j}{1-\tilde{w}_i}, \forall ij | i \neq j, u_{ii} = 1$.

Theorem 4 shows the closed-form gradient of the argument of Eq. (11) in the main manuscript. This gradient can be used to reduce the computational or memory requirement to compute the gradient, as compared to the use of automatic differentiation. It can also help in understanding the contribution of the gradient and design approximation of the gradient. In Theorem 4, $\mathbf{w}$ is edge selection vector, while $\tilde{\mathbf{w}}$ is its normalised version, i.e. $\tilde{\mathbf{w}} = \mathbf{w}/\sum_{i=1}^{M} w_i$. Similarly, $\tilde{\mathbf{1}}_M = \frac{1}{M}\mathbf{1}_M$ is the normalized version of the all-ones vector. In Theorem 4, g is the gradient of the Von Neumann entropy with respect to the normalized Laplacian matrix, while U is a matrix that normalizes the gradient with respect to the edge selection vector values.

The gumbel-softmax distribution can be used on both $H(G)$ and $S_{\text{vN}}(L_G)$ and does not require the results of Theorem 4.

Proof of Theorem 4. Theorem 4 follows by definition of Eq. (10) in the main manuscript and substituting the definition of Eq. (10) and the use of result from Theorem 5. The total derivative of the cost function w.r.t. to the normalized selection vector $\tilde{\mathbf{w}}$, is given by $\frac{dJ}{dw_i} = \frac{\partial J}{\partial w_i} + \sum_{i\neq j}\frac{\partial J}{\partial w_j}\frac{dw_j}{dw_i}$. With the normalized selector vector, we have that $\sum_k w_k = 1$ before and after the change. If we consider, as in Lemma 1, $w_i \to w_i + \delta$ and $w_j \to w_j(1-\gamma), j \neq i$, then $\gamma = \frac{\delta}{1-w_i}$ and $\frac{dw_j}{dw_i} = -\frac{\gamma w_j}{\delta} = -\frac{w_j}{1-w_i}$ □

Theorem 5. *The gradient of the von Neumann entropy w.r.t. the edge selection vector* $\mathbf{w}$ *is*

$$\nabla_{\mathbf{w}} S(\sigma_{\mathbf{w}}) = -\operatorname{diag}\left(B^T \log(B \operatorname{diag}(\mathbf{w}) B^T) B\right), \tag{15}$$

where $S(\sigma) = -\operatorname{tr}(\sigma \log \sigma - \sigma) = -\sum_i (\lambda_i \log \lambda_i - \lambda_i)$ *and* $\sigma_{\mathbf{w}} = B \operatorname{diag}(\mathbf{w}) B^T$.

Proof of Theorem 5. Theorem 5 follows from $\nabla_\sigma S(\sigma) = -\log \sigma$ and the use of gradient of the trace of a function of a matrix. Here we use the un-normalized Laplacian matrix for simplicity. □

B PRINCIPLE OF RELEVANT INFORMATION (PRI) FOR SCALAR RANDOM VARIABLES

In information theory, a natural extension of the well-known Shannon's entropy is the Rényi's α-entropy [Rényi, 1961]. For a random variable $\mathbf{X}$ with PDF $f(x)$ in a finite set $\mathcal{X}$, the α-entropy of $H(\mathbf{X})$ is defined as:

$$H_\alpha(f) = \frac{1}{1-\alpha} \log \int_{\mathcal{X}} f^\alpha(x) dx. \tag{16}$$

On the other hand, motivated by the famed Cauchy-Schwarz (CS) inequality:

$$\left| \int f(x) g(x) dx \right|^2 \leq \int |\, f(x) \,|^2 \, dx \int |\, g(x) \,|^2 \, dx, \tag{17}$$

with equality if and only if $f(x)$ and $g(x)$ are linearly dependent (e.g., $f(x)$ is just a scaled version of $g(x)$), a measure of the "distance" between the PDFs can be defined, which was named the CS divergence [Jenssen et al., 2006], with:

$$\begin{aligned} D_{cs}(f \| g) &= -\log(\int fg)^2 + \log(\int f^2) + \log(\int g^2) \\ &= 2H_2(f;g) - H_2(f) - H_2(g), \end{aligned} \tag{18}$$

the term $H_2(f;g) = -\log \int f(x) g(x) dx$ is also called the quadratic cross entropy [Principe, 2010].

Combining Eqs. (16) and (18), the PRI under the 2-order Rényi entropy can be formulated as:

$$\begin{aligned} f_{\text{opt}} &= \arg\min_f H_2(f) + \beta(2H_2(f;g) - H_2(f) - H_2(g)) \\ &\equiv \arg\min_f (1-\beta) H_2(f) + 2\beta H_2(f;g), \end{aligned} \tag{19}$$

the second equation holds because the extra term $\beta H_2(g)$ is a constant with respect to f.

As can be seen, the objective of naïve PRI for $i.i.d.$ random variables (i.e., Eq. (19)) resembles its new counterpart on graph data (i.e., Eq. (11) in the main manuscript). The big difference is that we replace $H_2(f)$ with $S_{\text{vN}}(\tilde{\sigma})$ and $H_2(f;g)$ with $S_{\text{vN}}\left(\frac{\tilde{\sigma}+\tilde{\rho}}{2}\right)$ to capture structure information.

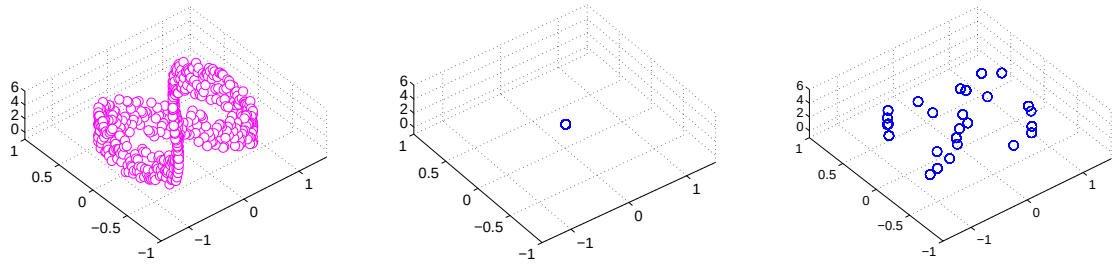
If we estimate $H_2(f)$ and $H_2(f;g)$ with the Parzen-window density estimator and optimize Eq. (19) by gradient descent. Fig. 1 demonstrates the structure learned from an original intersect data by different values of β.

Interestingly, when $\beta = 0$, we obtained a single point, very similar to what happens for Graph-PRI that learns a nearly star graph such that edges concentrates on one node. Similarly, when $\beta \to \infty$, both naïve PRI and Graph-PRI get back to the original input as the solution.

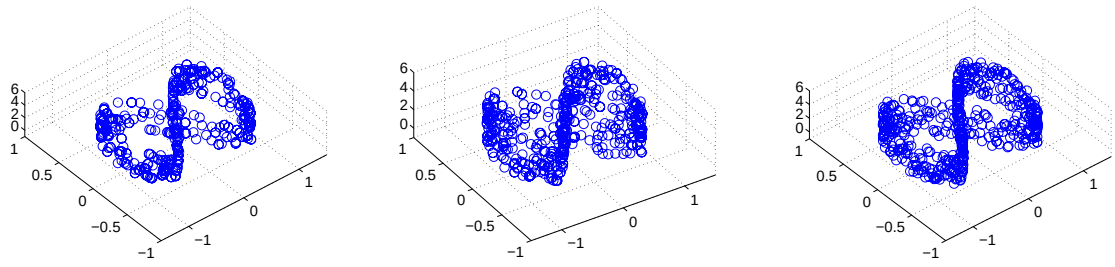
C DETAILS OF USED DATASETS IN SECTION 4.2 AND SECTION 4.3

C.1 MULTI-TASK LEARNING

Synthetic data. This dataset consists of 20 regression tasks with 100 samples each. Each task is a 30-dimensional linear regression problem in which the last 10 variables are independent of the output variable y. The 20 tasks are related in a



(a) original data (b) $\beta = 0$ (c) $\beta = 1$



(d) $\beta = 3$ (e) $\beta = 6$ (f) $\beta = 100$

Figure 1: Illustration of the structures revealed by the naïve PRI for (a) Intersect data set. As the values of β increase the solution passes through (b) a single point, (c) modes, (d) and (e) principal curves at different dimensions, and in the extreme case of (f) $\beta \to \infty$ we get back the data themselves as the solution.

group-wise manner: the first 10 tasks form a group and the remaining 10 tasks belong to another group. Tasks' coefficients in the same group are completely related to each other, while totally unrelated to tasks in another group.

Tasks' data are generated as follows: weight vectors corresponding to tasks 1 to 10 are $\mathbf{w}_k = \mathbf{w}_a \odot \mathbf{b}_k + \xi$, where $\odot$ is the element-wise Hadamard product; and tasks 11 to 20 are $\mathbf{w}_k = \mathbf{w}_b \odot \mathbf{b}_k + \xi$, where $\xi \sim \mathcal{N}(\mathbf{0}, 0.2\mathbf{I}_{20})$. Vectors $\mathbf{w}_a$ and $\mathbf{w}_b$ are generated from $\mathcal{N}(\mathbf{0}, \mathbf{I}_{20})$, while $\mathbf{b}_k \sim \mathcal{U}(0, 1)$ are uniformly distributed 20-dimensional random vectors.

Input and output variables for the t-th ($t = 1, \cdots, 20$) task, X_t and y_t, are generated as $X'_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{20})$ and $y_t = X'_t\mathbf{w}_t + \mathcal{N}(0, 1)$. 10-dimensional unrelated variables $X''_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{10})$ are then concatenated to X'_t to form the final input data $X_t = [X'_t \quad X''_t]$.

Parkinsons's disease dataset. This is a benchmark multi-task regression data set, comprising a range of biomedical voice measurements taken from 42 patients with earlystage Parkinson's disease. For each patient, the goal is to predict the motor Unified Parkinson's Disease Rating Scale (UPDRS) score based 18-dimensional record: age, gender, and 16 jitter and shimmer voice measurements. For the categorical variable "gender", we applied label encoding that converts genders into a numeric representation. We treat UPDRS prediction for each patient as a task, resulting in 42 tasks and $5,875$ observations in total.

C.2 BRAIN NETWORK CLASSIFICATION

For both datasets, the Automated Anatomical Labeling (AAL) template was used to extract ROI-averaged time series from the 116 ROIs. Meanwhile, to construct the initial brain network topology (i.e., the adjacency matrix A), we only keep edge if its weight (i.e., the absolute correlation coefficient) is among the top 20% of all absolute correlation coefficients in the network.

As for the node features, we only use the correlation coefficients for simplicity. That is, the node feature for node i can be represented as $\mathbf{x}_i = [\rho_{i1}, \rho_{i2}, \cdots, \rho_{in}]^T$, in which ρ_{ij} is the Pearson's correlation coefficient for node i and node j. One can expect performance gain by incorporating more discriminative network property features such as the local clustering coefficient [Rubinov and Sporns, 2010], although this is not the main scope of our work.

The first one is the eyes open and eyes closed (EOEC) dataset [Zhou et al., 2020], which contains the rs-fMRI data of 48 (22

females) college students (aged 19-31 years) in both eyes open and eyes closed states. The task is to predict two states based on brain network FC.

The second one is from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database[1]. We use the rs-fMRI data collected and preprocessed in [Kuang et al., 2019] which includes 31 AD patients aged 60–90 years. They were matched by age, gender, and education to mild cognitive impairment (MCI)[2] and 37 normal control (NC) subjects, together comprising 106 participants been selected. In this work, we only focus on distinguishing MCI group from NC group.

EOEC is publicly available from https://github.com/zzstefan/BrainNetClass/.

ADNI preprocessed by [Kuang et al., 2019] is publicly available from http://gsl.lab.asu.edu/software/ipf/.

D NETWORK ARCHITECTURE AND HYPERPARAMETER TUNING

D.1 FMRI-BASED BRAIN NETWORK CLASSIFICATION

The classification problem is solved using graph neural networks composed of two graph convolutional networks of size 32 and with relu activation function. We also use node feature drop with probability 10^{-1}. The node pooling is the sum of the node features, while the node classification minimizes the cross entropy loss. Hyper parameter search is applied to all method with time budget of $3'000$ seconds, over 3 runs. The learning rates, λ, β and the softmax temperature are optimized using early pruning. Each graph neural network is fed with graphs generated from the full correlation matrix by selecting edges among the strongest 20% absolute correlation values. For the Graph-PRI method, we used the GCN [Kipf and Welling, 2017] as graph classification network.

For SVM, we use the Gaussian kernel and set kernel size equals to 1. For LASSO, we set the hyperparameter as 0.1. For t-test, we set the significance level as 0.05.

E MINIMAL IMPLEMENTATION OF GRAPH-PRI IN PYTORCH

Algorithm 1 PRI for Graph Sparsification

Input: $\rho = BB^T$, β, learning rate η, number of samples S
Output: $\sigma_{\mathbf{w}}$

$B \leftarrow$ incident matrix of ρ;
Initialize $\theta = \{\theta_1, \theta_2, \cdots, \theta_M\}$;
while not converged **do**
  $L = 0$;
  for $i = 1, 2, \cdots, S$ **do**
    $\mathbf{w}^i \leftarrow \text{GumbelSoftmax}(\theta)$;
    $\sigma_{\mathbf{w}^i} = B\,\text{diag}\,(\mathbf{w}^i)B^T$;
    $L = L + \mathcal{J}_\beta(\rho, \sigma_{\mathbf{w}^i})$;
  end for
  $L = \frac{1}{S}L$
  $\theta \leftarrow \theta - \eta\nabla_\theta L$;
end while
$\mathbf{w} \leftarrow \text{GumbelSoftmax}(\theta)$;
return $\sigma_{\mathbf{w}} = B\,\text{diag}\,(\mathbf{w})B^T$;

We additional provide PyTorch implementation of Graph-PRI.

```
import torch
import networkx as nx
import numpy as np
```

[1] http://adni.loni.usc.edu/
[2] MCI is a transitional stage between AD and NC.

```

def vn_entropy(k, eps=1e-20):

    k = k / torch.trace(k)
    eigv = torch.abs(torch.symeig(k, eigenvectors=True)[0])
    entropy = -torch.sum(eigv[eigv>0] * torch.log(eigv[eigv>0] + eps))
    return entropy

def entropy_loss(sigma, rho, beta):

    assert(beta>=0), "beta shall be >=0"
    if beta > 0:
        return 0.5 * (1 - beta) / beta * vn_entropy(sigma) + vn_entropy(0.5 * (sigma + rho))
    else:
        return vn_entropy(sigma)

def sparse(G, tau, n_samples, max_iteration, lr, beta):
  '''
  Args:
    G: networkx Graph
    n_samples: number of samples for gumbel softmax
  '''

  E = nx.incidence_matrix(g1, oriented=True)
    E = E.todense().astype(np.double)
    E = torch.from_numpy(E)

    rho = E @ E.T

    m, n = G.number_of_edges(), G.number_of_nodes()
    theta = torch.randn(m, 2, requires_grad=True)
    optimizer = torch.optim.Adam([theta], lr=lr)

    for itr in range(max_iteration):
        cost = 0
        for sample in range(n_samples):
            # Sampling
            z = F.gumbel_softmax(theta, tau, hard = True)
            w = z[:, 1].squeeze()
            sigma = E @ torch.diag(w) @ E.T
            _loss = entropy_loss(sigma, rho, beta)
            cost = cost + _loss

        cost = cost / n_samples
        cost.backward()
        optimizer.step()
        optimizer.zero_grad()

    z = F.gumbel_softmax(theta, tau, hard=True)
    w = z[:,1].squeeze()

    sigma = E @ torch.diag(w) @ E.T # sparse laplacian

    return sigma, w
```

Listing 1: Graph-PRI PyTorch

References

Michael Dairyko, Leslie Hogben, Jephian C-H Lin, Joshua Lockhart, David Roberson, Simone Severini, and Michael Young. Note on von neumann and rényi entropies of a graph. *Linear Algebra and its Applications*, 521:240–253, 2017.

Robert Jenssen, Jose C Principe, Deniz Erdogmus, and Torbjørn Eltoft. The cauchy–schwarz divergence and parzen windowing: Connections to graph theory and mercer kernels. *Journal of the Franklin Institute*, 343(6):614–629, 2006.

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

Liqun Kuang et al. A concise and persistent feature to study brain resting-state network dynamics: Findings from the alzheimer's disease neuroimaging initiative. *Human brain mapping*, 40(4):1062–1081, 2019.

Jose C Principe. *Information theoretic learning: Renyi's entropy and kernel perspectives*. Springer Science & Business Media, 2010.

Alfréd Rényi. On measures of entropy and information. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 547–561, 1961.

Mikail Rubinov and Olaf Sporns. Complex network measures of brain connectivity: uses and interpretations. *Neuroimage*, 52(3):1059–1069, 2010.

Zhen Zhou et al. A toolbox for brain network construction and classification (brainnetclass). *Human Brain Mapping*, 41 (10):2808–2826, 2020.