The Deep Latent Position Block Model for Clustering and Representation of Networks

Rémi Boutin[1], Pierre Latouche[2] and Charles Bouveyron[3]

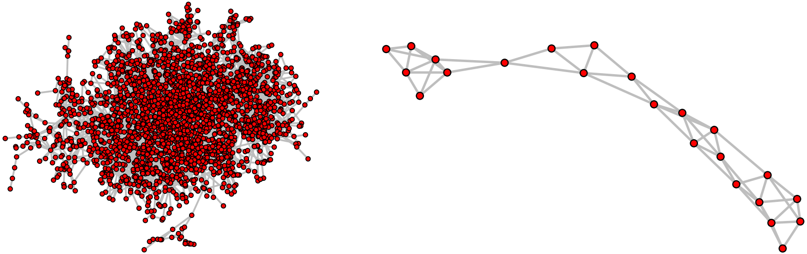
[1] LPSM - Sorbonne Université
[2] LMBP - Université Clermont Auvergne
[3] Maasai team - INRIA, Université Côte d'Azur

Journées de la Statistique, Bordeaux, May 2024

Introduction and motivation

The **networks** are a natural data structure to represent interactions between objects or individuals, such as:

- emails, co-authorship networks
- biological networks (protein-protein interactions networks)
- social websites (Facebook, Twitter)



(a) Cora network.

(b) Example of an enzyme network.

Figure: These networks representations were computed with the Fruchterman-Reingold algorithm[1].

[1] **fruchterman1991graph**.

Outline

Stochastic Block Model

The stochastic block model[2] assumes that each node is assigned to a single cluster:

$$\boldsymbol{\eta}_i \overset{i.i.d}{\sim} \mathrm{Multinomial}(1; \alpha = (\alpha_1, \ldots, \alpha_d)). \tag{1}$$

Hence,

$$\boldsymbol{\eta}_{ik} = \begin{cases} 1 & \text{if } i \text{ is in cluster } k, \\ 0 & \text{otherwise.} \end{cases}$$

Given the node cluster memberships, the probability of connection is given by:

$$A_{ij} \mid \{\boldsymbol{\eta}_{ik} = 1, \boldsymbol{\eta}_{jq} = 1\} \sim \mathcal{B}(\boldsymbol{\Pi}_{kq}).$$

[2] **holland1983stochastic**; **nowicki2001estimation**; **daudin2006**.
[3] **hoff2007modeling**; **daudin2010model**.

Stochastic Block Model

The stochastic block model[2] assumes that each node is assigned to a single cluster:

$$\boldsymbol{\eta}_i \overset{i.i.d}{\sim} \text{Multinomial}(1; \alpha = (\alpha_1, \ldots, \alpha_d)). \tag{1}$$

Hence,

$$\boldsymbol{\eta}_{ik} \;=\; \begin{cases} 1 & \text{if } i \text{ is in cluster } k, \\ 0 & \text{otherwise.} \end{cases}$$

Given the node cluster memberships, the probability of connection is given by:

$$A_{ij} \mid \{\boldsymbol{\eta}_{ik} = 1, \boldsymbol{\eta}_{jq} = 1\} \;\sim\; \mathcal{B}(\boldsymbol{\Pi}_{kq}) = \mathcal{B}(\eta_i^\top \boldsymbol{\Pi} \eta_j).$$

Can we relax the binary constraint from $\boldsymbol{\eta}_i \in \{0,1\}^Q$ to $\boldsymbol{\eta}_i \in \Delta_Q$ instead ?[3]

[2] **holland1983stochastic**; **nowicki2001estimation**; **daudin2006**.
[3] **hoff2007modeling**; **daudin2010model**.

Generative model

In this work, we assume that the node cluster membership assignment are not binary but continuous giving rise to the following assumptions:

$$\left.\begin{aligned} \mathbf{z}_i &\overset{i.i.d}{\sim} \mathcal{N}_d(0, \mathbf{I}_d) \\ \boldsymbol{\eta}_i &= \mathrm{softmax}(\mathbf{z}_i) \end{aligned}\right\} \text{ LogisticNormal distribution}$$

Generative model

In this work, we assume that the node cluster membership assignment are not binary but continuous giving rise to the following assumptions:

$$\left.\begin{aligned} \mathbf{z}_i &\overset{i.i.d}{\sim} \mathcal{N}_d(0, \mathbf{I}_d) \\ \boldsymbol{\eta}_i &= \operatorname{softmax}(\mathbf{z}_i) \end{aligned}\right\} \text{LogisticNormal distribution}$$

$$A_{ij} \mid \{\boldsymbol{\eta}_i, \boldsymbol{\eta}_j\} \sim \mathcal{B}(\boldsymbol{\eta}_i^\top \boldsymbol{\Pi} \boldsymbol{\eta}_j).$$

Inference

In this work, we aimed at maximising the **marginal log-likelihood** given by:

$$\log p(\mathbf{A} \mid \mathbf{\Pi}) = \log \int_{\mathbf{Z}} p(\mathbf{A}, \mathbf{Z} \mid \mathbf{\Pi}) d\mathbf{Z}. \tag{2}$$

The marginal likelihood being intractable, we rely on a variational inference to maximise it. In particular, for any distribution $R(\cdot)$ over the latent variable $\mathbf{Z}$, the following decomposition holds true:

$$\log p(\mathbf{A} \mid \mathbf{\Pi}) = \mathscr{L}(\mathbf{\Pi}; R) + \mathrm{KL}(R(\cdot) \mid\mid p(\mathbf{Z} \mid \mathbf{A})),$$

where

$$\mathscr{L}(\mathbf{\Pi}; R) = \mathbb{E}_{R(\mathbf{Z})}\left[\log \frac{p(\mathbf{A}, \mathbf{Z} \mid \mathbf{\Pi})}{R(\mathbf{Z})}\right]. \tag{3}$$

The quantity $\mathscr{L}(\mathbf{\Pi}; R)$ is called the **expected lower bound (ELBO)**.

Inference: variational assumption

Assuming that the variational distribution respect the mean-field hypothesis:

$$R_\phi(\mathbf{Z}) \;=\; \prod_{i=1}^{N} \mathcal{N}_d(\mu_\phi(\mathbf{A})_i, \sigma_\phi(\mathbf{A})_i^2 \mathbf{I}_d), \tag{4}$$

where the parameters are the ouput of a graph convolutional network[4]:

$$(\mu_\phi(\mathbf{A}), \log \sigma_\phi(\mathbf{A})^2) = \mathrm{GCN}_\phi(\mathbf{A}). \tag{5}$$

[4] **kipf2016variational**.

Details of the ELBO

Hence, the ELBO can be written as

$$
\begin{aligned}
\mathscr{L}(\boldsymbol{\Pi}; R_\phi) = & \sum_{j<i} \mathbb{E}_{R_\phi(\mathbf{Z})} \left[\log p(A_{ij} \mid \boldsymbol{\eta}_i, \boldsymbol{\eta}_j, \boldsymbol{\Pi})\right] - \sum_{i=1}^{N} \mathrm{KL}\left(R(\mathbf{z}_i) \mid p(\mathbf{z}_i)\right) \\
= & \sum_{j<i} A_{ij} \mathbb{E}_{R_\phi(\mathbf{Z})} \left[\log(\boldsymbol{\eta}_i^\top \boldsymbol{\Pi} \boldsymbol{\eta}_j)\right] + (1 - A_{ij}) \mathbb{E}_{R_\phi(\mathbf{Z})} \left[\log(1 - \boldsymbol{\eta}_i^\top \boldsymbol{\Pi} \boldsymbol{\eta}_j)\right] \\
& - \sum_{i=1}^{N} \frac{1}{2} \left(d\sigma_\phi(\mathbf{A})_i^2 + \|\mu_\phi(\mathbf{A})_i\|_2^2 - d \log \sigma_\phi(\mathbf{A})_i^2 - d\right).
\end{aligned} \tag{6}
$$

Next step: maximisation of $\mathscr{L}(\boldsymbol{\Pi}; R_\phi)$ with respect to $\boldsymbol{\Pi}$ and ϕ. We can directly optimise the previous quantity with respect to $\boldsymbol{\Pi}$ with gradient-based algorithm ... but not with respect to ϕ. Do you see the issue ?

The reparametrisation trick[5]

How to compute the gradient $\frac{\partial}{\partial \phi}\mathscr{L}(\mathbf{\Pi}; R_\phi)$? Based on the previous slide, we have:

$$\frac{\partial}{\partial \phi}\mathscr{L}(\mathbf{\Pi}; R_\phi) = \sum_{j<i} \frac{\partial}{\partial \phi}\mathbb{E}_{R_\phi(\mathbf{Z})}\left[\log p(A_{ij} \mid \boldsymbol{\eta}_i, \boldsymbol{\eta}_j, \mathbf{\Pi})\right] - \sum_{i=1}^{N} \frac{\partial}{\partial \phi} \overbrace{\mathrm{KL}\left(R_\phi(\mathbf{z}_i) \mid p(\mathbf{z}_i)\right)}^{\text{analytical form}}. \tag{7}$$

Issue: Since $R_\phi(\cdot)$ depends on ϕ, we cannot interchange the derivative and the integral in the term on the left-hand side.

[5] **kingma2014autoencoding**; **rezende2014stochastic**.

The reparametrisation trick[5]

How to compute the gradient $\frac{\partial}{\partial \phi}\mathscr{L}(\mathbf{\Pi}; R_\phi)$? Based on the previous slide, we have:

$$\frac{\partial}{\partial \phi}\mathscr{L}(\mathbf{\Pi}; R_\phi) = \sum_{j<i} \frac{\partial}{\partial \phi}\mathbb{E}_{R_\phi(\mathbf{Z})}\left[\log p(A_{ij} \mid \boldsymbol{\eta}_i, \boldsymbol{\eta}_j, \mathbf{\Pi})\right] - \sum_{i=1}^{N} \frac{\partial}{\partial \phi} \overbrace{\mathrm{KL}\left(R_\phi(\mathbf{z}_i) \mid p(\mathbf{z}_i)\right)}^{\text{analytical form}}. \quad (7)$$

Issue: Since $R_\phi(\cdot)$ depends on ϕ, we cannot interchange the derivative and the integral in the term on the left-hand side.
The **reparametrisation trick**[5] removes this dependency by the following sampling scheme:

$$\epsilon \sim \mathcal{N}_d(0, \mathbf{I}_d), \quad \text{and} \quad \mathbf{z}_i = \mu_\phi(\mathbf{A})_i + \sigma_\phi(\mathbf{A})_i \epsilon.$$

Hence, we can now interchange the integral and the derivative and use a Monte-Carlo estimate of the term on the right-hand side of the following equation:

$$\frac{\partial}{\partial \phi}\mathbb{E}_{R_\phi(\mathbf{Z})}\left[\log p(A_{ij} \mid \boldsymbol{\eta}_i, \boldsymbol{\eta}_j, \mathbf{\Pi})\right] = \frac{\partial}{\partial \phi}\mathbb{E}_{\epsilon}\left[\log p(A_{ij} \mid \boldsymbol{\eta}_i, \boldsymbol{\eta}_j, \mathbf{\Pi})\right] = \mathbb{E}_{\epsilon}\left[\frac{\partial}{\partial \phi}\log p(A_{ij} \mid \boldsymbol{\eta}_i, \boldsymbol{\eta}_j, \mathbf{\Pi})\right].$$

[5] **kingma2014autoencoding**; **rezende2014stochastic**.

Simulation setup

- Number of clusters $= 5$
- Number of nodes is set to 200
- $\beta \in \{0.1, 0.2, 0.3\}$ tunes for the high level of connectivity between clusters
- $\eta = 0.01$ in all our experiments

$$
\mathbf{\Pi}^\star = \overset{\text{Communities}}{\begin{pmatrix} \beta & \epsilon & \dots & \dots & \epsilon \\ \epsilon & \beta & \epsilon & \dots & \epsilon \\ \vdots & \epsilon & \beta & \dots & \epsilon \\ & \epsilon & \dots & \beta & \epsilon \\ \epsilon & \epsilon & \dots & \dots & \beta \end{pmatrix}} \quad \overset{\text{Disassortative}}{\begin{pmatrix} \epsilon & \beta & \dots & \dots & \beta \\ \beta & \epsilon & \beta & \dots & \beta \\ \beta & \beta & \epsilon & \dots & \beta \\ \beta & \beta & \dots & \epsilon & \beta \\ \beta & \beta & \dots & \dots & \epsilon \end{pmatrix}} \quad \overset{\text{Hub}}{\begin{pmatrix} \beta & \beta & \dots & \dots & \beta \\ \beta & \beta & \epsilon & \dots & \epsilon \\ \beta & \epsilon & \beta & \dots & \epsilon \\ \beta & \epsilon & \dots & \beta & \epsilon \\ \beta & \epsilon & \dots & \dots & \beta \end{pmatrix}}
$$

Benchmark

Table: ARI between the true node groups and the estimated ones. The closer to 1 the better.

		Communities	Disassortative	Hub
$\beta = 0.1$	SBM	0.13 ± 0.12	0.03 ± 0.03	0.41 ± 0.14
	ARGVA	0.42 ± 0.08	0.00 ± 0.01	0.07 ± 0.03
	VGAE	$\mathbf{0.55 \pm 0.11}$	0.01 ± 0.00	$\mathbf{0.20 \pm 0.08}$
	Deep LPBM	0.46 ± 0.20	0.02 ± 0.02	0.18 ± 0.04
$\beta = 0.2$	SBM	0.71 ± 0.15	0.47 ± 0.29	0.83 ± 0.15
	ARGVA	0.85 ± 0.03	0.01 ± 0.01	0.28 ± 0.06
	VGAE	$\mathbf{0.99 \pm 0.01}$	0.00 ± 0.01	0.78 ± 0.15
	Deep LPBM	$\mathbf{0.99 \pm 0.01}$	$\mathbf{0.39 \pm 0.14}$	$\mathbf{0.90 \pm 0.05}$
$\beta = 0.3$	SBM	0.77 ± 0.12	0.81 ± 0.21	0.98 ± 0.07
	ARGVA	0.88 ± 0.03	0.06 ± 0.04	0.56 ± 0.22
	VGAE	$\mathbf{1.00 \pm 0.00}$	0.00 ± 0.00	0.93 ± 0.07
	Deep LPBM	$\mathbf{1.00 \pm 0.00}$	$\mathbf{1.00 \pm 0.00}$	$\mathbf{1.00 \pm 0.00}$

Real dataset: the French political blogosphere[6]

- This dataset is composed of 194 nodes
- Each node corresponds to a political blog
- An edge exists between two blogs if a link exists between the two

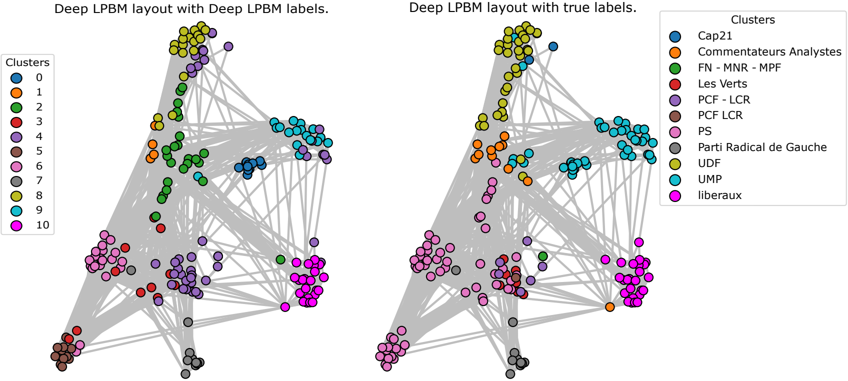


Figure: Representation obtained with Deep LPBM.

[6] **zanghi2008Fastonline**.

Real dataset: the French political blogosphere[7]

- the blogs related to the Socialist party are split in two clusters (5 and 6) revealing two different connectivity patterns

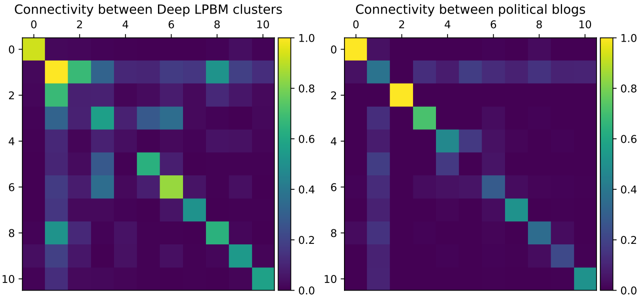


Figure: Connectivity between the estimated clusters and between the blog associated to the political parties.

[7] **zanghi2008Fastonline**.

Conclusion

- The combination of graph neural networks with block modelling provides insightful results
- The model selection working without GNN still works in the variational autoencoder setting
- Need to test it on other datasets (in the presence of connectivity patterns different from communities)

Conclusion

- The combination of graph neural networks with block modelling provides insightful results
- The model selection working without GNN still works in the variational autoencoder setting
- Need to test it on other datasets (in the presence of connectivity patterns different from communities)

Thank you for your attention !

Details about VGAE[8]

Denoting $\tilde{\mathbf{A}} = \mathbf{f}^{-1/2}(\mathbf{A} + \mathbf{I}_N)\mathbf{f}^{-1/2}$, the graph convolutional network can be summarised as

[8] **kipf2016variational**.

Details about VGAE[8]

Denoting $\tilde{\mathbf{A}} = \mathbf{f}^{-1/2}(\mathbf{A} + \mathbf{I}_N)\mathbf{f}^{-1/2}$, the graph convolutional network can be summarised as

$$\mu_\phi(\mathbf{A}) = \tilde{\mathbf{A}}\,\mathrm{ReLU}(\tilde{\mathbf{A}}\boldsymbol{\Omega}_0)\boldsymbol{\Omega}_\mu,$$
$$\log\sigma^2_\phi(\mathbf{A}) = \tilde{\mathbf{A}}\,\mathrm{ReLU}(\tilde{\mathbf{A}}\boldsymbol{\Omega}_0)\boldsymbol{\Omega}_\sigma,$$

where

- $\mathrm{ReLU}(x) = (\max(0, x_1), \ldots, \max(0, x_F))$ if $x \in \mathbb{R}^F$,
- $\boldsymbol{\Omega}_0 \in \mathcal{M}_{N\times D}(\mathbb{R})$ with $D = 64$ in all the experiments we carried out,
- $\boldsymbol{\Omega}_\mu, \boldsymbol{\Omega}_\sigma \in \mathcal{M}_{D\times(Q-1)}(\mathbb{R})$.

[8] **kipf2016variational**.

Model Selection

In this methodology, we consider the following three model selection criterion:

$$\mathrm{AIC}(Q,\mathcal{M}) = \log p(\mathbf{A} \mid \mathbf{Z}) - \frac{Q*(Q+1)}{2} - N(Q-1),$$

$$\mathrm{BIC}(Q,\mathcal{M}) = \log p(\mathbf{A} \mid \mathbf{Z}) - \frac{Q*(Q+1)/2}{2} \log\left(\frac{N*(N-1)}{2}\right) - \frac{Q-1}{2}\log(N),$$

$$\mathrm{ICL}(Q,\mathcal{M}) = \log p(\mathbf{A} \mid \mathbf{Z}) - \frac{Q*(Q+1)/2}{2} \log\left(\frac{N*(N-1)}{2}\right) + \log p(\mathbf{Z}) - \frac{Q-1}{2}\log(N).$$

Model Selection

Table: Comparison of AIC (2a) and BIC (2b) and ICL (2c) to select the best number of clusters for Deep LPBM with $\beta = 0.3$, the true number of cluster corresponds to the shadowed row.

(a) AIC

Q	Commu	Disass	Hub
2	0	0	0
3	0	0	0
4	0	0	1
$5^\star$	**10**	**10**	**9**
6	0	0	0
8	0	0	0
9	0	0	0
10	0	0	0
16	0	0	0

(b) BIC

Commu	Disass	Hub
0	0	0
0	0	0
0	0	0
0	**7**	**6**
10	3	4
0	0	0
0	0	0
0	0	0
0	0	0

(c) ICL

Commu	Disass	Hub
0	0	0
0	0	0
0	0	0
0	**8**	**7**
10	2	3
0	0	0
0	0	0
0	0	0
0	0	0