Introduction to Probabilistic Graphical Models - MVA 2023/2024
A mixture model for random graphs, Daudin et al.

Jean Dimier de la Brunetiere jean.dimier-de-la-brunetiere@polytechnique.edu

December 22, 2023

Disclaimer: My partner of work stopped answering me a bit less than two weeks before the deadline. I chose to still produce something, even if the experiments are clearly less ambitious than I would have liked them to be.

1 Introduction

This article [1], aims at giving new interesting ways to study the properties of a given graph, by giving a new way to model them. As it has been done for point clouds in arbitrary dimensions for example with Gaussian Mixture Models, this article introduces a new model for random graphs, considering them too as Mixture Models.

The motivation to consider such a model comes, among other reasons, from the fact that the most used theoretical models to study real-world graphs do not actually fit so well to them. A well-known example is the Erdös-Rényi model, which has been used and studied a lot for its simplicity and the fact that it is easy to derive from it many theoretical properties, whether it be average or asymptotic ones, or for quantities of interest such as the connectivity or the clustering coefficient. Yet, a well-known drawback of the Erdös-Rényi model is that is does not fit well to real-world data, and the best way to be convinced of this is to look at the empirical degree distribution of these networks and the one offered by the model: from the model arises asymptotically a Poisson distribution, whereas the one from real-world data is often much more heavy-tailed. This model also tends to underestimate clustering coefficients encountered in the real-world.

To tackle these issues, the authors proposed to derive a Mixture Model framework, to account for the heterogeneity that is often naturally induced by real-world data, and to still produce interpretable results, which most other methods at the time were incapable of. The authors also provide a practical methods to estimate the parameters of this model from the data with a variational approach, and give a criterion to estimate the best number of classes the mixture should contain. The aim of this work is therefore to be able to estimate as well as possible the classes of the different vertices of a graph, which is tantamount to making a clustering of this graph.

2 Method

2.1 Presentation of the problem

We consider only indirected graphs with n vertices and we define the variable X_{ij} which equals 1 if i and j are connected, 0 otherwise. All the graphs we consider are without self-loops, hence we have for all i: $X_{ii} = 0$. We define the degree of vertex i as the number of edges connected to vertex i: $K_i = \sum_{j \neq i} X_{ij}$.

For example, the Erdös-Rényi model with parameter p is defined as follows:

$$\{X_{ij}\} \text{ i.i.d}, \qquad X_{ij} \sim \mathcal{B}(p)$$

Therefore, we have for all i: $K_i \sim \mathcal{B}(n-1, p) \approx \mathcal{P}(\lambda)$ with $\lambda = (n-1)p$ with the classical approximation of the binomial distribution by a Poisson distribution.

The distribution of the degrees is therefore not heavy-tailed, when it is often the case in practice, and therefore other models for the degree-distribution have been tested, such as the Zipf distribution, which is a power-law, but here it is useful mostly for the tail of the distribution rather than the whole dataset. The authors make the point that if the Erdös-Rényi is unable to capture the real underlying properties of the real graphs, it could be due to some heterogeneities between vertices, some being more connected than the others. That is to account for this heterogeneity that they suggested that the degrees of the vertices could be modeled by a mixture of Poisson distributions. But by considering vertices in such an independent way, the possible drawback is to miss the actual topology of the network under investigation: to fix that, the mixture model is actually extended to modeling the edges of the graph.

2.2 A Mixture model for the edges

We will assume that the n vertices of a graph G can be split into Q classes with prior probabilities $\{\alpha_1, ..., \alpha_Q\}$. To follow the article's notations, we use the indicator variables $\{Z_{iq}\}$ which equal 1 if vertex i belongs to class q, and 0 otherwise. We therefore have:

$$\alpha_q = \mathbb{P}(Z_{iq} = 1) = \mathbb{P}i \in q, \text{with} \sum_q \alpha_q = 1$$

Then, we introduce a mixing matrix $\pi \in \mathbb{R}^{Q \times Q}$ between classes, such that π_{ql} represents the probability for any vertex of class q to be connected to a given vertex of class l. The graph being undirected, this matrix is symmetric. Finally, we suppose that the edges $\{X_{ij}\}$ are conditionally independent given the classes of vertices i and j, by setting:

$$X_{ij} | \{i \in q, j \in l\} \sim \mathcal{B}(\pi_{ql})$$

It can be shown that this model is general enough to recover many specific well-studied models, such as Erdös-Rényi models of course, but also affiliation networks.

This model also presents nice theoretical properties, when it comes to estimating some core quantities describing its topology. For example, the degree distribution of a vertex given the class it belongs to follows a Poisson distribution:

$$K_i|\{i \in q\} \sim \mathcal{B}(n-1, \overline{\pi}_q) \approx \mathcal{P}(\lambda_q)$$

where $\overline{\pi}_q$ is a barycenter of the different connection coefficients of π with weights given by the $\{\alpha_q\}$:

$$\overline{\pi}_q = \sum_l \alpha_l \pi_{ql}$$

We therefore see that the distribution will be, for example in an independent model, be equal to a sum of Poisson distribution with different parameters. The ability of this model to recover more arbitrary distribution will therefore be by introducing a dependency between the classes, which will avoid this "sum of parabola" phenomenon.

Another interesting quantity to look at is the clustering coefficient. It is usually defined empirically when no probabilistic model is available, which equals 1 for a clique (every vertex is connected to all the others) and 0 for a graph without connections.

A definition proposed in [2] can be to use the clustering coefficient vertex-wise:

$$C_i = \frac{\Delta_i}{\frac{K_i(K_i)-1)}{2}}$$

where Δ_i is the number of edges between the neighbors of vertex i: $\Delta_i = \frac{1}{2}\sum_{i,j,k} X_{ij}X_{jk}X_{ik}$ which equals 0 as minimum value (for example without connections between vertices) and $\frac{K_i(K_i)-1)}{2}$ for a clique.

And then to take the empirical mean of those coefficients:

$$\hat{c} = \frac{1}{n}\sum_i C_i$$

But with a probabilistic model, a probabilistic definition of this coefficient can be given, and it is shown that in the case of the mixture model, it is equal to:

$$c = \mathbb{P}\left\{X_{ij}X_{jk}X_{ki} = 1 \mid X_{ij}X_{ik} = 1\right\} = \frac{\sum_{q,\ell,m} \alpha_q\alpha_\ell\alpha_m\pi_{q\ell}\pi_{qm}\pi_{\ell m}}{\sum_{q,\ell,m} \alpha_q\alpha_\ell\alpha_m\pi_{q\ell}\pi_{qm}}$$

Thus, for synthetic data for which we know the different values of the $\{\alpha_q\}$ and of the coefficients of π, it is easily computable.

Now that we have built a model, we have to make it fit to our data, and to be able to evaluate how well it is able to recover this data. Thus, the approach is to evaluate the likelihood of our model and its parameters given our data.

We have a formula for the complete data log-likelihood:

$$\begin{aligned}\log \mathcal{L}(\mathcal{X}, \mathcal{Z}) = &\sum_i \sum_q Z_{iq} \log \alpha_q \\ &+ \frac{1}{2}\sum\sum Z_{iq}Z_{j\ell} \log b\left(X_{ij}; \pi_{q\ell}\right)\end{aligned} \tag{1}$$

Of course, we do not know the values of the variables $\{Z_{iq}\}$ and therefore we cannot evaluate this sum. That is why, as a common approach in variational inference [3], we will try to optimize a lower-bound of this quantity, given by:

$$\mathcal{J}\left(R_{\mathcal{X}}\right) = \log \mathcal{L}(\mathcal{X}) - \mathrm{KL}\left[R_{\mathcal{X}}(\cdot), \mathbb{P}(\cdot \mid \mathcal{X})\right] \tag{2}$$

with $R_{\mathcal{X}}(\cdot)$ being a "convenient" class of distributions of data, that will allow us to make estimations in a computable way, to find the best distribution among this class to approximate our data $\mathcal{X}$.

The approximation chosen by the authors is therefore the mean-field approximation, given by:

$$R_{\mathcal{X}}(\mathcal{Z}) = \prod_i h(\mathcal{Z}_i; \boldsymbol{\tau}_i)$$

where $\boldsymbol{\tau}_i = (\tau_{i1}, ..., \tau_{iQ})$ and $h(\cdot; \boldsymbol{\tau})$ stands for the multinomial distribution with parameter $\boldsymbol{\tau}$. Therefore, we decouple the different nodes in the way we model their class belongings.

Practically, we alternatively update the parameters ($\boldsymbol{\alpha}, \boldsymbol{\pi}$ and $\boldsymbol{\tau}$: for the update of ***tau***, it is a maximization problem:

$$\begin{aligned} \text{maximize} \quad & \mathcal{J}\left(R_{\mathcal{X}}\right) \\ \text{subject to} \quad & \sum_q \tau_{iq} = 1, \forall i \in \{1, ..., n\} \\ & \boldsymbol{\tau} \geq 0 \end{aligned}$$

The article uses a fixed-point algorithm, but I have not managed to figure out exactly how to implement it: I only used a minimization routine.

Then, the other parameters are updated with a straight-forward formulas:

$$\widehat{\alpha}_q = \frac{1}{n} \sum_i \widehat{\tau}_{iq}, \quad \widehat{\pi}_{q\ell} = \sum_{i \neq j} \widehat{\tau}_{iq} \widehat{\tau}_{j\ell} X_{ij} / \sum_{i \neq j} \widehat{\tau}_{iq} \widehat{\tau}_{j\ell}. \tag{3}$$

The formula for $\widehat{\alpha}_q$ is extremely intuitive: it just corresponds to a kind a empirical mean on the nodes of the current probabilities for node i to belong to class q. This procedure reminds a bit of the EM algorithm, even if the two are fundamentally different, the EM algorithm requiring to compute the conditional distribution $\mathbb{P}(\mathcal{Z}|\mathcal{X})$, which is in our case completely intractable.

The paper also gives a theoretical guarantee on the pertinence of the optimization procedure described above: each update of parameters necessarily increases our lower-bound, even if we have no guarantee over its convergence towards a global maximum.

Finally, the article provides a way to estimate the best number of classes we have to fuel our algorithm with, but I did not explore this parameter in my experiments.

3 Implementation and Results

As for the implementation, I implemented from scratch the generation of the graphs and the fitting procedure (except for the optimization routine using *scipy* as already said). I made sure that every graph generated can be accessed jointly with its underlying parameters $\boldsymbol{\alpha}$ and π.

An example of a graph generated can be seen on Figure 1

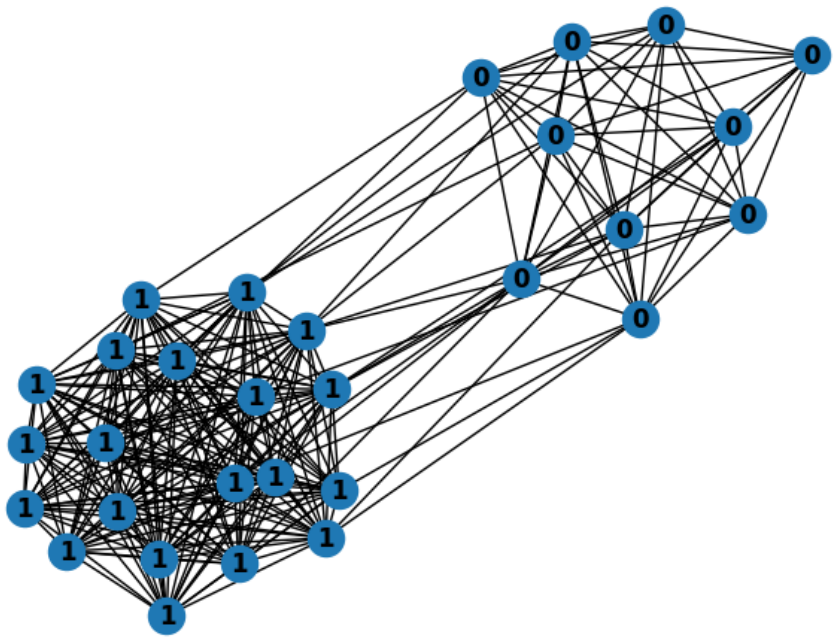

Figure 1: Example of an affiliation graph recovered via a mixture model, with $\alpha_2 = 2\alpha_1$ and $\pi = \begin{pmatrix} 1 & 0.1 \\ 0.1 & 1 \end{pmatrix}$

whose degree distribution is (2)

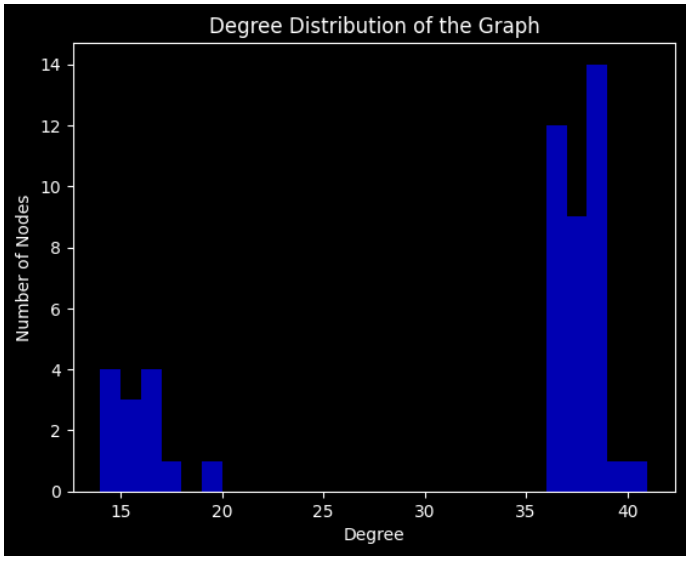

Figure 2: Degree distribution of the graph given in figure 1

I eventually lost too much time trying to run the optimization procedure, to get in the end an update of my parameter τ which always return the initialization I set to it. It is clearly due to the fact that the most adapted method to update this parameter would be to use a fixed-point algorithm, but I did not figure out how to implement it (given the fact that this is not even clearly a *fixed-point* equality). Therefore the fitting procedure gets obviously immediately stuck at its first point and is not able to recover the hidden parameters. I decided to keep this version of the code as it was the clearest one and had the benefit of making clear the algorithm used. I also have to acknowledge that with the tests I ran, in any case my implementation would have not worked for graphs with more than a few hundreds of vertices, due maybe to some easy vectorization procedures I let aside, but I think mostly because of the unadapted optimization routine.

As a result, my data simulation worked out great, but I was not able to use it as a ground truth to test the ability of my algorithm to recover the parameters with which I generated my graphs. If it worked, I would have taken as class q for any vertex i the value for which the obtained τ_{iq} is maximal, defined my labels according to this class, reordered my graph thanks to a small function I defined, and vizualized the resulting adjacency matrix.

References

[1] J.-J. Daudin, F. Picard, and S. Robin. A mixture model for random graphs. *Statistics and Computing*, 18(2):173–183, December 2007.

[2] Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1):47–97, January 2002.

[3] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, April 2017.