

Deep latent position model for node clustering in graphs

Dingge Liang, Marco Corneli, Charles Bouveyron, Pierre Latouche

▶ **To cite this version:**

Dingge Liang, Marco Corneli, Charles Bouveyron, Pierre Latouche. Deep latent position model for node clustering in graphs. ESANN 2022 - 30th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, Oct 2022, Bruges, Belgium. 10.14428/esann/2022.ES2022-30 . hal-03874698

HAL Id: hal-03874698
https://hal.science/hal-03874698

Submitted on 28 Nov 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Deep latent position model for node clustering in graphs

Dingge Liang[1], Marco Corneli[1], Charles Bouveyron[1] and Pierre Latouche[2]

1- Université Côte d'Azur, INRIA, CNRS, Laboratoire J.A.Dieudonné - France
2- Université Paris Cité, CNRS, Laboratoire MAP5, UMR 8145 - France

Abstract. With the significant increase of interactions between individuals through numeric means, the clustering of vertex in graphs has become a fundamental approach for analysing large and complex networks. We propose here the deep latent position model (DeepLPM), an end-to-end clustering approach which combines the widely used latent position model (LPM) for network analysis with a graph convolutional network (GCN) encoding strategy. Thus, DeepLPM can automatically assign each node to its group without using any additional algorithms and better preserves the network topology. Numerical experiments on simulated data and an application on the Cora citation network are conducted to demonstrate its effectiveness and interest in performing unsupervised clustering tasks.

1 Introduction and related work

Networks are employed in a wide range of applications, from social media and email communications to protein-protein interactions, because they are simple structures yet are capable of modeling complex systems. In this context, vertex clustering is a key branch of clustering which attempts to partition the nodes of the graph into different groups to extract patterns summarizing the data.

On the one hand, a long series of statistical methods have been developed to discover the underlying features in networks. The stochastic block model [SBM, 1] is widely used to detect communities or more general clusters of nodes. Based on SBM, many extensions looking for overlapping clusters have been proposed, such as MMSBM [2] and OSBM [3]. On the other hand, a different approach to model network data relies on latent position models (LPMs) [4]. Afterwards, LPCM [5] was developed to incorporate a clustering structure into LPM. Nevertheless, these models have a challenging inference procedure that primarily relies on MCMC and do not scale easily to large and complex networks.

From another aspect, deep learning based techniques have been intensively investigated in clustering. In this line of methods, VGAE [6] adopts a graph convolutional network [GCN, 7] encoder to produce nodes embeddings in the latent space. By introducing adversarial learning into the generation process, ARVGA [8] enforced the latent representation to match a prior distribution. Lately, DGLFRM [9] combined OSBM with GCN by positing each node of the graph to have an embedding modeled by a Beta-Bernoulli process. These approaches adopt a two-step clustering procedure, simply relying on *external* clustering algorithms (e.g. k-means) to group the embedded nodes, independently from the generative model. More recently, a self-training clustering module was

developed as an alternative approach in DAEGC [10] that allows to jointly optimize graph embeddings and clusters. All of the aforementioned methods employ inner-product-based decoders, whereas we argue that a different solution, accounting for the Euclidean distance between nodes in the latent space might be more suited.

In order to overcome the limitations of the methods listed above, while exploring their benefits, we introduce the deep latent position model (DeepLPM), allowing to simultaneously learn vertex representations and obtain node partitions. By combining a GCN encoder with a LPM-based decoder, our model aims at capturing the best of both worlds described so far: it is a flexible representation learning tool based on the deep learning architecture, yet comprehensive and interpretable thanks to the statistical model considered.

2 Deep latent position model

Notations. In this work, networks are modeled as undirected, unweighted graphs $G = (V; E)$ with $N = |V|$ nodes. We introduce an $N \times N$ adjacency matrix A, where $A_{ij} = 1$ if there is a link between node i and node j, 0 otherwise. The set of the edges E can be associated with an additional covariate information, collected into the matrix $Y \in \mathbb{R}^{|E| \times D}$. The generic entry of Y, denoted y_{ij}, is the D-dimensional feature associated with the edge connecting i to j. For instance, y_{ij} could encode the text exchanged between users i and j in a communication network. We aim to learn well-represented, latent, node embeddings Z in a lower dimension P and to partition the nodes into K clusters.
Generative model. As in LPM, we assume that each node i, $i = \{1, \cdots, N\}$, has an unknown position $z_i \in \mathbb{R}^P$ in a latent space. The probability of a link between two nodes is modeled as a distance function between their latent positions. Our generative process is as follows. First, each node is assigned to a cluster via a random variable c_i encoding its cluster membership

$$c_i \overset{iid}{\sim} \mathcal{M}(1, \pi), \quad \text{with} \quad \pi \in [0,1]^K, \ \sum_{k=1}^{K} \pi_k = 1. \tag{1}$$

Then, conditionally to its cluster, a latent embedding z_i is generated

$$z_i | (c_{ik} = 1) \sim \mathcal{N}(\mu_k, \sigma_k^2 I_P), \quad \text{with} \quad \sigma_k^2 \in \mathbb{R}^{+*}, \tag{2}$$

independently for each node. Finally, the probability of a connection between nodes i and j is modeled through a Bernoulli random variable related to the distance between latent positions

$$A_{ij} | z_i, z_j \sim \mathcal{B}(f_{\alpha,\beta}(z_i, z_j)), \tag{3}$$

with

$$f_{\alpha,\beta}(z_i, z_j) = \sigma(\alpha + \beta^T y_{ij} - ||z_i - z_j||^2), \tag{4}$$

where $f_{\alpha,\beta}$ can be seen as a *decoding*, one-layer, neural network parametrized by α and β and σ is the logistic sigmoid function and y_{ij} is the covariate of the

edge connecting i with j.

Model inference. By denoting $\Theta = \{\pi, \mu_k, \sigma_k^2, \alpha, \beta\}$ the set of the model parameters introduced so far, we rely on a variational approach to approximate the intractable (integrated) log-likelihood: $\log p(A|\Theta) = \mathcal{L}(q(Z,C);\Theta) + D_{KL}(q(Z,C)||p(Z,C|A,\Theta))$. In order to deal with a tractable family of distributions, $q(Z,C)$ is assumed to factorize into two terms

$$q(Z,C) = q(Z)q(C) = \prod_{i=1}^{N} q(z_i)q(c_i), \tag{5}$$

in which it is also supposed

$$q(z_i) = \mathcal{N}(z_i; \tilde{\mu}_\phi(\overline{A})_i, \tilde{\sigma}^2_\phi(\overline{A})_i I_P), \tag{6}$$

where $\tilde{\mu}_\phi(\cdot) : \mathbb{R}^{N\times N} \mapsto \mathbb{R}^{N\times P}$ (respectively $\tilde{\sigma}^2_\phi(\cdot) : \mathbb{R}^{N\times N} \mapsto \mathbb{R}^{+N}$) is the function mapping the normalized adjacency $\overline{A} = D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$ into the matrix of the variational means (vector of the standard deviations), parametrized by the two-layer GCN *encoder* g_ϕ.

Finally, a standard assumption is made for variational cluster probabilities

$$q(C) = \prod_{i=1}^{N} \mathcal{M}(c_i; 1, \gamma_i), \quad \text{with} \quad \sum_{k=1}^{K} \gamma_{ik} = 1, \tag{7}$$

where γ_{ik} represents the variational probability that node i is in cluster k.

Thanks to Equations (5)-(6)-(7), the evidence lower bound can be further developed as

$$\begin{aligned}
\mathcal{L} &= \int_Z \sum_C q(Z,C) \log \frac{p(A|Z,\alpha,\beta)p(Z|C,\mu_k,\sigma_k^2)p(C|\pi)dZ}{q(Z,C)} \\
&= \mathbb{E}\left[\sum_{i\neq j} A_{ij}\log\eta_{ij} + (1-A_{ij})\log(1-\eta_{ij})\right] - \sum_{i=1}^{N}\sum_{k=1}^{K}\gamma_{ik} D_{KL}(\mathcal{N}(\tilde{\mu}_\phi(\overline{A})_i, \tilde{\sigma}^2_\phi(\overline{A})_i I_P)||\mathcal{N}(\mu_k, \sigma_k^2 I_P)) \\
&\quad + \sum_{i=1}^{N}\sum_{k=1}^{K}\gamma_{ik}\log(\frac{\pi_k}{\gamma_{ik}}),
\end{aligned} \tag{8}$$

where $\eta_{ij} = \sigma(\alpha + \beta^T y_{ij} - ||z_i - z_j||^2)$, $D_{KL}(\cdot)$ denotes the KL divergence and the expectation is taken with respect to the variational probability $q(\cdot)$. The pseudo code of the optimization process is reported in Algorithm 1.

Algorithm 1 Estimation of DeepLPM

Input: adjacency matrix A, edge features Y
pretrain_model = *pretrain*(A, 50 epochs) ▷ pre-training to save initial weights
while $\mathcal{L}$ increases **do**
  $\tilde{\mu}_\phi, \tilde{\sigma}^2_\phi$ = GCN(A)
  update γ_{ik}, π_k, μ_k and σ_k^2 by calculating derivations ▷ explicit optimization
  calculate the training loss (negative ELBO) $-\mathcal{L}$
  update neural net parameters ϕ, α and β via SGD ▷ implicit optimization
Output: reconstructed graph $\hat{A}$, cluster probability matrix $\hat{\gamma}$

3 Numerical experiments

To benchmark DeepLPM with other competitors, we designed three types of synthetic networks based on LPCM, SBM and from circle data. We emphasize that all networks are generated by models different from ours. The goal here is to assess the stability and the quality of the inference procedure we proposed. By varying the value of two parameters δ and $\delta^{'}$ in scenarios A (assortative) and B (dissortative), we can model the proximity between each cluster and thus test the robustness of our model in both simple and difficult cases. Contrary to standard communities with strong transitivity (your-friend-is-my-friend effect), scenario C describes the construction of three groups of nodes with little transitivity in each. As demonstrated in Figure 1, DeepLPM has the highest ARI with a small variance in all situations. Figure 2 shows the learned embeddings by three deep models in scenario C, only DeepLPM preserves the original network topology.

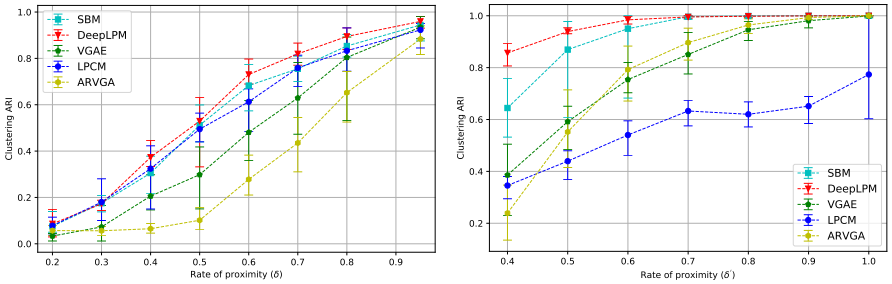


Fig. 1: Clustering ARI with different proximity rates in Sc.A and Sc.B.

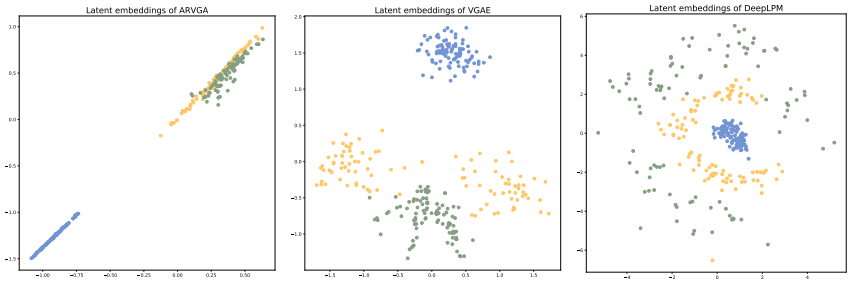


Fig. 2: Embeddings learned by ARVGA, VGAE and DeepLPM in Sc.C.

4 Analysis of Cora network

The Cora dataset contains 2,708 scientific publications classified in seven classes and consists of 5,429 citation links. Most related works assume that the number of clusters is equal to the number of classes used in supervised classification tasks, whereas we argue that the class labels might not be in a one-to-one relation with the detected communities in unsupervised clustering. Instead, an appropriate cluster number should be obtained through model selection. Thus, we decided

to use the class membership of each paper to build a tensor Y of dimension $D = 7 \times 7$ encoding the similarities between articles. For each pair of papers i and j with category labels s_i and s_j, $Y_{s_i s_j} = 1$ indicates that paper i belongs to the class s_i and j belongs to the class s_j, 0 otherwise.

Results. The model selection was conducted by varying the number of clusters from 5 to 11, with the dimensionality of the latent space equal to 16. Based on the evolution of training loss, the number of groups was estimated to be $K = 6$ with a clear minimum. Figure 4 shows the paper distributions when considering the class labels for six groups. In contrast to the fact that each group contains only one defined class, it is clear that new similarities between papers in different categories emerge as a result of the addition of paper labels as covariates.

Next, to better understand the clustering results, we plotted the latent positions learned by DeepLPM using PCA in Figure 5, highlighting nodes with degrees higher than 10. Those papers are more often cited by other papers and can be more representative. Interestingly, when looking at this figure from left to right, the content is changing from applied research to more theoretical learning, and then from bottom to top, the topic of the articles is changing from case-based methods and reinforcement learning to genetic algorithms, and finally to neural networks and statistical models.

Furthermore, based on the publications ID, we selected several articles with relatively large degree from each group and analyzed the information. For instance, according to paper titles, we find that group #1 (red) focuses on dynamic or temporal learning algorithms using probabilistic methods or reinforcement learning; in group #3 (blue), papers are largely based on the analysis and development of case studies; then, group #4 (cyan) contains articles on applications of genetic algorithms and neural networks; group #6 (yellow) typically involves statistical and machine learning models, etc.

Finally, we emphasize that, unlike most related works that consider supervised class labels as clusters in unsupervised learning, we encode this information into edge features and estimate the number of clusters via model selection, which

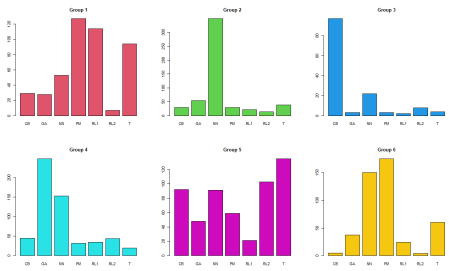


Fig. 4: Partitions with covariates taking into account class categories in each group on Cora.

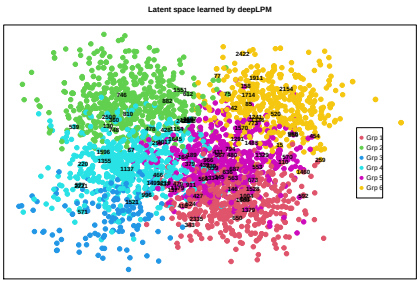


Fig. 5: Learned hidden space highlighting nodes with degrees higher than 10.

aids in the discovery of new node similarities hidden behind the supervised information, as demonstrated by the results.

5 Conclusion

We introduced DeepLPM to perform node clustering in an end-to-end manner by integrating the GCN encoder with the LPM-based decoder. Numerical experiments show that DeepLPM outperforms state-of-the-art methods. Moreover, real-world application on a scientific citation network was also proposed to illustrate the interest of the methodology for unsupervised analysis.

Acknowledgements

This work has been supported by the French government, through the 3IA Côte d'Azur, Investment in the Future, project managed by the National Research Agency (ANR) with the reference number ANR-19-P3IA-0002.

References

[1] Krzysztof Nowicki and Tom A B Snijders. Estimation and prediction for stochastic blockstructures. *Journal of the American statistical association*, 96(455):1077–1087, 2001.

[2] Edoardo Maria Airoldi, David M Blei, Stephen E Fienberg, and Eric P Xing. Mixed membership stochastic blockmodels. *Journal of machine learning research*, 2008.

[3] Pierre Latouche, Etienne Birmelé, and Christophe Ambroise. Overlapping stochastic block models with application to the french political blogosphere. *The Annals of Applied Statistics*, pages 309–336, 2011.

[4] Peter D Hoff, Adrian E Raftery, and Mark S Handcock. Latent space approaches to social network analysis. *Journal of the american Statistical association*, 97(460):1090–1098, 2002.

[5] Mark S Handcock, Adrian E Raftery, and Jeremy M Tantrum. Model-based clustering for social networks. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 170(2):301–354, 2007.

[6] Thomas N Kipf and Max Welling. Variational graph auto-encoders. In *NeurIPS Workshop on Bayesian Deep Learning (NeurIPS-16 BDL)*, 2016.

[7] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations (ICLR-17)*, 2016.

[8] Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, Lina Yao, and Chengqi Zhang. Adversarially regularized graph autoencoder for graph embedding. In *International Joint Conference on Artificial Intelligence (IJCAI-18)*, pages 2609–2615, 2018.

[9] Nikhil Mehta, Lawrence Carin Duke, and Piyush Rai. Stochastic blockmodels meet graph neural networks. In *International Conference on Machine Learning*, pages 4466–4474. PMLR, 2019.

[10] Chun Wang, Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, and Chengqi Zhang. Attributed graph clustering: A deep attentional embedding approach. In *International Joint Conference on Artificial Intelligence (IJCAI-19)*, 2019.