The Journal of Open Source Software

ProbINet: Bridging Usability Gaps in Probabilistic Network Analysis

Diego Baptista [1,2], **Martina Contisciani** [3], **Caterina De Bacco** [4], and **Jean-Claude Passy** [1]

1 Max Planck Institute for Intelligent Systems, Tübingen, Germany. **2** Graz University of Technology, Graz, Austria. **3** Central European University, Vienna, Austria. **4** Delft University of Technology, Delft, Netherlands.

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Anastassia Vybornova

Reviewers:

- @giorgospanay
- @sandorjuhasz

Submitted: 12 March 2025
Published: unpublished

Summary

Probabilistic **I**nference on **Net**works (ProbINet) is a Python package that provides a unified framework to perform probabilistic inference on networks, enabling researchers and practitioners to analyze and model complex network data. The package integrates code implementations from several scientific publications, supporting tasks such as community detection, anomaly detection, and synthetic data generation using latent variable models. It is designed to simplify the use of cutting-edge techniques in network analysis by providing a cohesive and user-friendly interface.

Statement of need

Network analysis is central to disciplines such as social sciences, biology, and fraud detection, where understanding relationships is essential. Probabilistic generative models (Contisciani et al., 2020, 2022; Safdari et al., 2021, 2022; Safdari & De Bacco, 2022) reveal hidden patterns, detect communities, identify anomalies, and generate synthetic data. Their broader use is limited by fragmented implementations that hinder comparisons and reproducibility. ProbINet addresses this gap by unifying recent approaches in a single framework, improving accessibility and usability across disciplines.

ProbINet stands out among network analysis tools. Graph-tool (Peixoto, 2014) provides community detection and general graph analysis tools, but it uses a different model family than our mixed-membership framework and does not account for reciprocity. CDlib (Rossetti et al., 2019) offers detection algorithms and evaluation routines, but ProbINet extends this with probabilistic MLE models, optional node attributes, and anomaly detection. pgmpy (Ankan & Textor, 2024) focuses on Bayesian network structure learning, while ProbINet uncovers latent patterns like communities and reciprocity.

Main features

ProbINet offers a feature-rich framework to perform inference on networks using probabilistic generative models. Key features include:

- **Diverse Network Models**: Integration of generative models for various network types and goals (see table below).
- **Synthetic Network Generation**: Ability to generate synthetic networks that closely resemble real ones for further analyses (e.g., testing hypotheses).

- **Simplified Parameter Selection**: A cross-validation module to optimize key parameters, providing performance results in a clear dataframe.
- **Rich Set of Metrics for Analysis**: Advanced metrics (e.g., F1 scores, Jaccard index) for link and covariate prediction performance.
- **Powerful Visualization Tools**: Functions for plotting community memberships and performance metrics.
- **User-Friendly Command-Line Interface**: An intuitive interface for easy access.
- **Extensible and Modular Codebase**: Future integration of additional models possible.

Algorithm's Name	Description	Network Properties
CRep	Models directed networks with communities and reciprocity (Safdari et al., 2021).	Directed, Weighted, Communities, Reciprocity
JointCRep	Captures community structure and reciprocity with a joint edge distribution (Contisciani et al., 2022).	Directed, Communities, Reciprocity
DynCRep	Extends CRep for dynamic networks (Safdari et al., 2022).	Directed, Weighted, Dynamic, Communities, Reciprocity
ACD	Identifies anomalous edges and node community memberships in weighted networks (Safdari & De Bacco, 2022).	Directed, Weighted, Communities, Anomalies
MTCOV	Extracts overlapping communities in multilayer networks using topology and node attributes (Contisciani et al., 2020).	Weighted, Multilayer, Attributes, Communities

The **Usage** section below illustrates these features with a real-world example.

Usage

Example: Analyzing a Social Network with ProbINet

This section shows how to use ProbINet to analyze a social network of 31 students and 100 directed edges representing friendships in a small Illinois high school (Coleman, 1964). We analyze the network using JointCRep in ProbINet to infer latent variables, assuming communities and reciprocity drive tie formation, a reasonable assumption for friendship relationships.

Steps to Analyze the Network with ProbINet

With ProbINet, you can load network data as an edge list and select an algorithm (e.g., JointCRep), fit the model to extract latent variables, and analyze results like soft community memberships, which show how nodes interact across communities. This is exemplified in Figure 1. On the left, a network representation of the input data is displayed alongside the lines of code required for its analysis using ProbINet. The result is shown on the right, where nodes are colored according to their inferred soft community memberships, while edge thickness and color intensity represent the inferred probability of edge existence.

JOSS
The Journal of Open Source Software

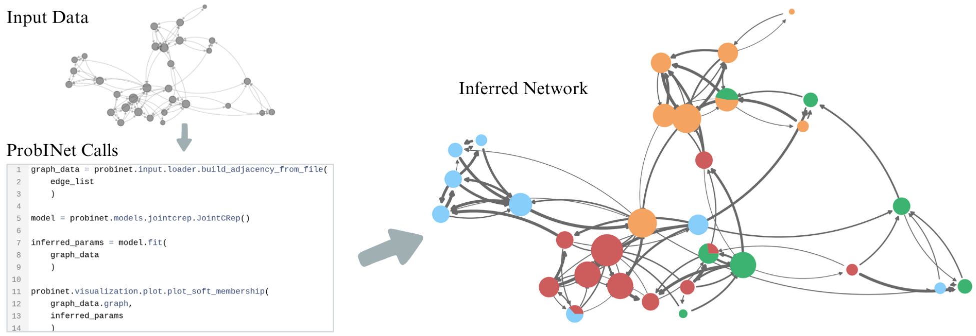

Figure 1: Usage of ProbINet on a social network. (Top-left) A network representation of the input data. (Bottom-left) A snapshot of the code used. (Right) The resulting output.

For more tutorials and use cases, see the package documentation.

Running Times of Algorithms

The table below summarizes algorithm runtimes on the tutorial data. **N** and **E** represent the number of nodes and edges, respectively. Edge ranges indicate variation across layers or time steps. **L/T** indicates the number of layers or time steps, and **K** represents the number of communities.

Algorithm	N	E	L/T	K	Time (mean ± std, in seconds)
CRep	600	5512	1	3	3.00 ± 0.35
JointCRep	250	2512	1	2	3.81 ± 0.69
DynCRep	100	234-274	5	2	1.48 ± 0.06
ACD	500	5459	1	3	27.8 ± 3.2
MTCOV	300	724-1340	4	2	1.51 ± 0.14

These benchmarks were performed on a 12th Gen Intel Core i9-12900 CPU, using `hyperfine` (Peter, 2023) and 10 runs. Runs required small amounts of RAM (less than 1 GB).

Acknowledgements

We thank the contributors of the integrated publications and Kibidi Neocosmos, Valkyrie Felso, and Kathy Su for their feedback.

References

Ankan, A., & Textor, J. (2024). Pgmpy: A python toolkit for bayesian networks. *Journal of Machine Learning Research*, *25*(265), 1–8. http://jmlr.org/papers/v25/23-0487.html

Coleman, J. S. (1964). Introduction to mathematical sociology. *London Free Press Glencoe*.

Contisciani, M., Power, E. A., & De Bacco, C. (2020). Community detection with node attributes in multilayer networks. *Scientific Reports*, *10*(1), 15736. https://doi.org/10.1038/s41598-020-72626-y

Contisciani, M., Safdari, H., & De Bacco, C. (2022). Community detection and reciprocity in networks by jointly modelling pairs of edges. *Journal of Complex Networks*, *10*(4), cnac034. https://doi.org/10.1093/comnet/cnac034

Peixoto, T. P. (2014). The graph-tool python library. *Figshare*. https://doi.org/10.6084/m9.figshare.1164194

Peter, D. (2023). *hyperfine* (Version 1.16.1). https://github.com/sharkdp/hyperfine

Rossetti, G., Milli, L., & Cazabet, R. (2019). CDlib: A python library to extract, compare and evaluate communities from complex networks. *Applied Network Science*, *4*(1), 52. https://doi.org/10.1007/s41109-019-0165-9

Safdari, H., Contisciani, M., & De Bacco, C. (2021). Generative model for reciprocity and community detection in networks. *Physical Review Research*, *3*(2), 023209. https://doi.org/10.1103/PhysRevResearch.3.023209

Safdari, H., Contisciani, M., & De Bacco, C. (2022). Reciprocity, community detection, and link prediction in dynamic networks. *Journal of Physics: Complexity*, *3*(1), 015010. https://doi.org/10.1088/2632-072X/ac52e6

Safdari, H., & De Bacco, C. (2022). Anomaly detection and community detection in networks. *Journal of Big Data*, *9*(1), 122. https://doi.org/10.1186/s40537-022-00669-1