NetSciX2026: International School and Conference on Network Science
February 17th-20th, 2026 - Auckland, New Zealand

Enhancing Graph Neural Networks with Random Graph Ensembles

random graphs, statistical ensembles, inference, machine learning, graph neural networks

Extended Abstract

In data on complex networks, the observed network is often assumed to accurately represent the underlying system. However, this overlooks the inherent uncertainty in network formation processes and measurements. Random graph ensembles allow to address this by providing a statistical baseline against which observed networks can be compared, enabling to distinguish significant structural patterns from random fluctuations. This approach has proven valuable in different network analysis tasks, such as node clustering [1], identifying significant temporal patterns in dynamic networks [2], and other applications [3]. Identifying significant patterns is crucial for Graph Neural Networks (GNNs) due to their message-passing mechanism. GNNs leverage the network structure of data to propagate information across nodes, capturing relational patterns for predictive modeling. This reliance on the network structure makes GNNs sensitive to noise in real-world datasets. However, to the best of our knowledge random graph ensembles have not been used to tackle this issue.

Addressing this gap, we show how random graph ensembles can improve the performance of GNNs in a node classification task. For this we define edge scores based on a soft configuration model [4] and use them to refine the network. The *soft* configuration model addresses limitations of the Molloy-Reed model [5], which is computationally expensive and not analytically tractable. The soft version provides a closed-form expression for the null model based on an urn problem. It relaxes the property of a fixed degree sequence to *expected* degrees. The model defines a matrix $\Xi \in \mathbb{N}^n \times \mathbb{N}^n$ that contains entries in the urn. Entries are possible stub combinations defined as product of in- d_j^{in} and out-degrees d_i^{out} in the observed graph $\mathscr{G}$: $\Xi_{ij} = d_i^{out} \cdot d_j^{in}$ The total number M of stub combinations or edge placements is $M = \sum_{ij} \Xi_{ij}$. A random graph is sampled by drawing $m = \sum_i d_i^{out} = \sum_j d_j^{in}$ stub combinations without replacement. This results in degrees that are equivalent to the observed graph only in the expected case. However, the model is analytically tractable, as the probability of observing edge A_{ij} between nodes i and j follows a hypergeometric distribution, i.e. $P(X_{ij} = A_{ij}) = \binom{M}{m}^{-1} \binom{\Xi_{ij}}{A_{ij}} \binom{M-\Xi_{ij}}{m-A_{ij}}$.

We use this to define *edge scores* by computing the cumulative probability of edge frequencies $P(X_{ij} = A_{ij})$ in the soft configuration model. Accumulating the observation likelihood to $P(X_{ij} \leq A_{ij})$ measures how likely an edge in a random graph instance occurs with a frequency lower than the observed one. $P(X_{ij} \leq A_{ij}) = 0.5$ means that half of the random realizations have a lower edge frequency than observed and the other half have a higher one. Hence, the observed frequency matches the expected frequency. A probability of $P(X_{ij} \leq A_{ij}) > 0.5$ means that for any random realization the chance is higher that the edge frequency is lower than observed. Hence, the observed edge is *over represented* compared to the null model and thus has a higher relevance. For $P(X_{ij} \leq A_{ij}) < 0.5$ the edge is *under represented.*

High edge frequencies can occur as a result of high node activities (i.e. degree) only, and may not carry information about how strongly two *specific* nodes are "related". In contrast, edge scores help to identify statistically relevant connections. We use edge scores to refine the graph topology in two ways: (I) We use edge scores as weights instead of edge frequencies.

Table 1: Comparison of GCN variants. Models 3-6 use at least one configuration-based augmentation (3 in Edge Scores or Filtered Edges). Best balanced accuracy per dataset is in bold.

	Edge	**Edge**	**Filtered**	**SocioPatterns**				
#	**Freq.**	**Scores**	**Edges**	Highschool11	Highschool12	Hospital	Workplace13	Workplace15
1	✗	✗	✗	50.93 ± 8.71	51.70 ± 5.79	61.58 ± 18.50	86.75 ± 13.39	63.06 ± 6.69
2	✓	✗	✗	**56.72 ± 11.37**	51.47 ± 3.40	59.83 ± 20.78	83.46 ± 14.80	83.88 ± 9.71
3	✗	✗	✓	49.52 ± 8.09	**58.55 ± 5.84**	67.75 ± 13.81	**87.17 ± 10.59**	67.74 ± 4.87
4	✓	✗	✓	54.62 ± 8.05	48.96 ± 8.96	62.00 ± 21.65	84.08 ± 15.40	**84.07 ± 8.19**
5	✗	✓	✗	54.15 ± 8.24	57.10 ± 7.06	**70.58 ± 22.16**	**87.17 ± 10.59**	64.68 ± 8.11
6	✗	✓	✓	52.37 ± 9.35	55.23 ± 5.63	66.50 ± 20.19	**87.17 ± 10.59**	68.25 ± 5.29

This way, over-represented edges have a strong impact while under-represented ones have a lower influence. (II) We filter the graph by removing under-represented edges, ensuring that the graph only contains connections that occur more frequently than expected.

We experimentally evaluate both graph refinement methods in a node classification task based on Sociopatterns data on face-to-face interactions. We predict student gender (two classes) for *highschool*, worker's department for *workplace* (five classes in 13, twelve in 15), and medical personnel roles or patient status (four classes) for *hospital*. We use the GCN architecture [6] with two layers with embedding size 32, ReLU activation, dropout and batch normalization. A final linear layer transforms the output for the downstream class. We train and compare six GCN variants, each differing in whether we remove under-represented edges (approach (II)), whether we use edge weights, and whether edge weights are frequencies or edge scores outlined in approach (I). Table 1 reports the balanced accuracy of our node classification experiments (with numbers in parentheses referring to table rows). As baselines we use a standard GCN without edge frequencies (1) and a GCN with edge frequencies (2). The remaining four variants incorporate network refinements based on the soft configuration model: GCN where we remove under-represented edges (3), GCN where we additionally use edge frequencies as weights (4), GCN with edge scores as weights (5), and GCN with edge scores and where we additionally remove under-represented edges (6). The results show that refining the network leads to improved performance in 4 out of 5 cases. We also compare the best performing refinement to the best performing baseline. For *workplace* (where baselines already achieve good performance) we get minor improvements of 0.48% and 0.23%. In contrast, we observe significant gains of 13.25% for *highschool12* and 14.62% for *hospital*.

Our results suggest a promising direction for further refinements. Our work further demonstrates how network science approaches can be used to improve deep graph learning methods.

References

[1] Mark EJ Newman. "Modularity and community structure in networks". In: *Proceedings of the national academy of sciences* 103.23 (2006), pp. 8577–8582.

[2] Timothy LaRock et al. "Hypa: Efficient detection of path anomalies in time series data on networks". In: *Proceedings of the 2020 SIAM international conference on data mining*. SIAM. 2020, pp. 460–468.

[3] Mikhail Drobyshevskiy and Denis Turdakov. "Random graph modeling: A survey of the concepts". In: *ACM computing surveys (CSUR)* 52.6 (2019), pp. 1–36.

[4] Giona Casiraghi and Vahan Nanumyan. "Configuration models as an urn problem". In: *Scientific Reports* 11.1 (2021).

[5] Michael Molloy and Bruce Reed. "A critical point for random graphs with a given degree sequence". In: *Random Struct. Alg.* 6.2-3 (1995), pp. 161–180.

[6] Thomas N. Kipf and Max Welling. "Semi-Supervised Classification with Graph Convolutional Networks". In: *ICLR*. 2017.