A Framework for Analyzing the Robustness of Graph Models

Khaled Abdelaal
School of Computer Science
The University of Oklahoma
Norman, OK, USA
khaled.abdelaal@ou.edu

Richard Veras
School of Computer Science
University of Oklahoma
Norman, OK, USA
richard.m.veras@ou.edu

***Abstract*—Graphs – and sparse matrices – provide a powerful representation for expressing the complex structural relationship between elements in a set, which is why they are used extensively in graph machine learning, network analytics, and scientific computing. One of the challenges in this field is obtaining large scale graph data for performance evaluation. Here, parameterized graph models and their corresponding generators fill in the gap. While there is much work on how well these models represent real data, there are open questions as to how sensitive, or robust, these dials are to noise.**

In this paper we present a framework for evaluating parameterized graph models in order to study how perturbations to these parameters affect the global structure of the resulting graph. We discuss how this framework is extensible to any graph model and choice of graph features. Further, we provide a case study for Kronecker graphs and analyze the effects of varying the parameters of the Kronecker Graph's initiator matrix, along with injecting noise into the graph on global features. What we will see is that certain features have varying degrees of robustness relative to parameter being modified.

***Index Terms*—Sparse Matrix, Graph, Descriptor, Kronecker Graphs, Sensitivity Analysis**

I. Introduction

Graph models are incredibly important for generating large scale synthetic data when real data is not accessible, and when graph operations need to be evaluated for performance. Thus, generated data gives developers a proxy dataset to refine their code. The closer the model approximates the real data, the more likely developers are to produce code that is efficient for those real datasets. One question is how much of a predictive understanding of the global structures of the graph can these models provide, and are these structures robust enough, i.e. not highly sensitive to noise, that a developer can optimize for them?

To this end, we propose a novel framework to evaluate graph descriptors. Our framework takes an input vector of graph generation parameters P, and produces a set of graphs $G_s = \{G_1, G_2, G_3, ..., G_n\}$. It then evaluates the effect of varying P values on the structure of the output graph set G_s through observing the change in different properties distributions such as: degree, in-degree, out-degree, in-betweenness centrality, clustering coefficient, etc. To ensure the robustness of the parameter set P in describing the graph structure, our framework also evaluates the effect of adding gradual noise to the parameter set, and observes the noise threshold $alpha$ at which the structure of the graph is dominated by the induced noise.

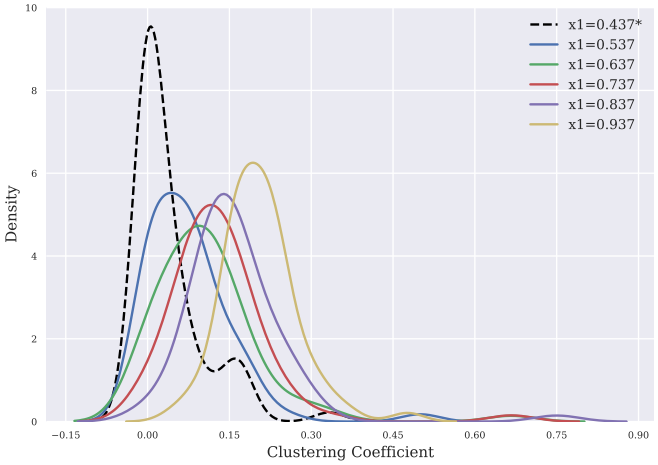


Fig. 1: An example of an output plot produced by our graph analysis framework. The plot shows the effect of changing a graph generation parameter x_1 on the node clustering coefficient distribution of the graph. The dashed line indicates the distribution for the initial graph G_0

Figure 1 shows an example output plot from our framework. This kernel denisty estimation (KDE) plot evaluates the effect of varying a graph generation parameter $x_1 \in P$ with an incremental step $\alpha = 0.1$ on the graph clustering coefficient distribution, as one of the graph structural properties. The original graph distribution is represented by the dashed black line. As discussed later, x_1 is an initiator matrix value of a Kronecker graph.

The main contributions of this work is as follows:

1) Propose a novel framework to evaluate graph descriptors and how sensitive graph structures are to them.
2) Evaluate the tolerance of the graph descriptor to random noise.
3) Demonstrate the usage of the proposed framework through a case study on the Kronecker Graph model.

II. Background and Related Work

A. Generative Graph Models

Various works presented different models to generate synthetic graphs [1]–[7], that are similar in terms of graph properties [8] to real world graph. The motivation behind the introduction of such models is, but not limited to: (1)understanding complex structures of large graph using smaller, well-formulated synthetic models, (2)tackling the privacy restrictions associated with accessing and studying real graphs, (3)predicting the evolution of large scale graphs, and (4) benchmarking graph neural networks (GNNs) [9]. In this paper, we focus on the Kronecker graph model as one example of such generative models.

B. Kronecker Graphs

Kronecker graphs [10] are a class of synthetic graphs that have been widely used to model real-world networks, and are generated by recursively applying the Kronecker product of a small base graph with itself. Let A and B be two matrices. Then, their Kronecker product $A \otimes B$ is given by

$$A \otimes B = \begin{pmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{pmatrix} \quad (1)$$

where a_{ij} are the entries of A. The resulting graph has a power-law degree distribution and exhibits a hierarchical structure that captures both the local and global connectivity patterns of the underlying real-world network.

To generate Kronecker graphs, an initiator matrix (typically of size 2×2 or 3×3) is chosen, and the Kronecker product is applied to this matrix by itself K times, where K is the Kronecker power. Then, a randomly generated probability matrix is used to mask out random values in the Kronecker matrix (remove edges from the Kronecker graph). Other compute-efficient methods can be used to generate Kronecker graphs such as ball dropping and grass hopping [11].

In [10], [12], [13], the authors present and analyze the Kronecker graph model as an approximation of scale-free graphs. Their analysis served as a template for ours, however in their work they compared their model against the real graphs that they were approximating, whereas as we analyze varying synthetic graphs against each other. In addition to Kronecker Graphs, many other graph models have been studied and surveyed [14]–[16].

In [17] the authors demonstrate that the Kronecker Graph model does not produce power law graphs, but that through the addition of noise the resulting model becomes a much stronger representation of power law graphs.

C. Graph Learning and Structure Prediction

In [18], the authors present an extensible framework for classifying sparse matrices structures using graph neural networks. The framework classifies an input matrix (graph), into one of a pre-defined classes of structure such as mesh network, random matrix, Kronecker graph, and combination of them. Additional classes can be integrated in the framework. Our proposed framework can be used in tandem with the classifier framework by evaluating more graph models (classes of structure), and then augmenting the classifier framework with additional structures that show a clear correlation between the input generation parameter set P and the output graph structure, and robustness to injected noise.

In addition, our framework can be used to evaluate graph generation parameters that can be then used in the feature set selection step of training graph neural networks for non-attributed graphs [19], given that these parameters prove to be robust representing the graph structure.

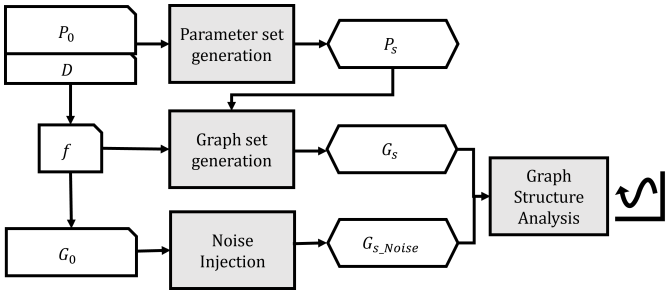


Fig. 2: High-level overview of the proposed framework. It takes initial generation parameters P_0 alongside with the domain of legal values D and the generation function f to generate a new parameters set P_s which is used to generate a new graph set G_s. Noise is also injected after generating the initial graph G_0 to synthesize the noisy graph set G_{s_Noise}. Finally, an analysis is performed on the output graph sets structure.

III. Framework Overview

Our proposed framework can be visualized as shown in Figure 2. It takes as input a graph model (M) generator f and a set of initial parameters vector $P_0 = \{p_{00}, p_{01}, ..., p_{0m-1}\}$ such that one can generate a graph G_0 of model M as follows: $G_0 = f(P_0)$. In addition, it expects the domain of legal values for each of the parameters $D = \{d_0, d_1, ..., d_{m-1}\}$. Then, the framework generates a set of graphs G_s by varying each of the parameters in P_0 individually within D for n times, while fixing the rest of the parameters, generating a new set of parameters $P_s = \{P_1, P_2, P_3, ..., P_n\}$. With the new set of parameters, a new set of graphs G_s is generated using f such that $G_i = f(P_i)$ for $1 \leq i \leq n$.

For each of the generated graphs, the framework calculates and plots multiple graph structural properties such as degree centrality, in-degree, out-degree, betweenness centrality, closeness centrality, laplacian centrality, clustering coefficient, and singular values. The output vectors and figures enable the evaluation of the effect of changing the parameter set P_s on the structure of the original graph G_0.

In addition, the framework evaluates how graph structure reacts to injecting random noise, and the extent of noise to which it maintains its original underlying structure. This is done by adding random edges, represented as a random sparse

matrix (adjacency matrix) with density $alpha$. By varying $alpha$, different noise matrices are generated and are added to the original graph's (G_0) adjacency matrix. The same graph properties are calculated and plotted as output.

Such analysis enables the evaluation of the graph model M and the parameter set P. One can determine whether this model M is a representative model of a set of classes that share specific structural properties, that can be later exploited to design the algorithms and data structures for this class of graph, or this model does not introduce any exploitable characteristics (i.e. random).

Moreover, the analysis can predict whether the structure of the generated graphs is sensitive to the input parameter set. Hence, a decision can be made to use the input parameter set as a representative feature set for this class of graph. The choice of adequate representative set for graph is critical in many application such as compact graph representation for large graphs (compressing large graphs by only storing a small set of parameters that can be used to generate such graph on demand), and learning on graphs: choosing a feature set for (non-attributed) graphs in a graph neural network setting.

Noise analysis is crucial as well, since most of the real data (graphs) are expected to include a certain level of entropy. Gradual noise injection and observing the resulting graph structure permits a good estimate of a noise tolerance, where the original graph model maintains its structure before, and loses its special characteristics beyond. It is also useful in evaluating the graph generation parameter set: if adding noise with higher values does not affect the graph structure, this means that the initial graph structure was already random and the used parameter set and/or graph model cannot be effectively used to represent a meaningful unique class of graphs.

IV. Case Study: Analyzing Kronecker Graphs

In order to evaluate our framework, we analyze the stochastic Kronecker Graph Model. The generator f is the Kronecker graph generator. It takes the following parameters P : the Kronecker initiator matrix X and the Kronecker power K. For simplicity, we only consider X in this discussion.

X is assumed to be a 2×2 matrix and consists of the following values:

$$\begin{bmatrix} x_0 & x_1 \\ x_2 & x_3 \end{bmatrix}$$

Hence, each parameter vector can be represented as $P_i = \{x_{i0}, x_{i1}, x_{i2}, x_{i3}\}$. The domain D for the each $x_{ij} \in P_i$ is floating point values between 0 and 1.

To evaluate the correlation between each of the initiator matrix values and the resulting Kronecker graph structure, we vary each of them within D, while fixing the other three values. We observe the effect of this variation on different graph and node-level attributes such as: degree, in-degree, out-degree, betweenness centrality, etc. The initial parameter vector P_0 is based on Kronfit's [12] estimated Kronecker initiator values [10] of the High Energy Physics - Phenomenology Collaboration (CA-HEP-PH) Graph [20]. Starting with P_0, we generate graphs of the 6th Kronecker Power ($K = 6$) for each of initiator matrices we evaluate $P_s = \{P_1, P_2, ..., P_n\}$.

For example, to evaluate the effect of changing x_0, we fix the values of x_1, x_2, and x_3. We then vary the value of x_0 for m times (5 in our experiments), by subtracting α (0.1) each time. So, for the first time, the new initiator matrix values will be as follows:

$$\begin{bmatrix} x_0 - \alpha & x_1 \\ x_2 & x_3 \end{bmatrix}$$

From this new initiator matrix, we generate a Kronecker Graph of Kronecker power 6. Then, we analyze the different graph and node-level attributes detailed in the following subsections.

For distributions, our framework generates Kernel Density Estimation plots, where the horizontal axis represents the property distribution (degree, betweenness centrality, etc.), and the vertical axis represents the density of the distribution at each point. A black dashed line the Figures 3, 4, 5 represents the corresponding distribution for the initial parameter set P_0, which is the starting point from which the framework begins to vary the parameter set values.

Figure 3 shows the effect of varying the different initiator matrix values on the degree distribution of the resulting Kronecker graph. To further investigate the structural effects of changing the different initiator matrix values, we additionally study the in-degree and out-degree behavior. Figure 4 shows a similar analysis for in-degree distribution. Figure 5 shows the analysis for out-degree distribution.

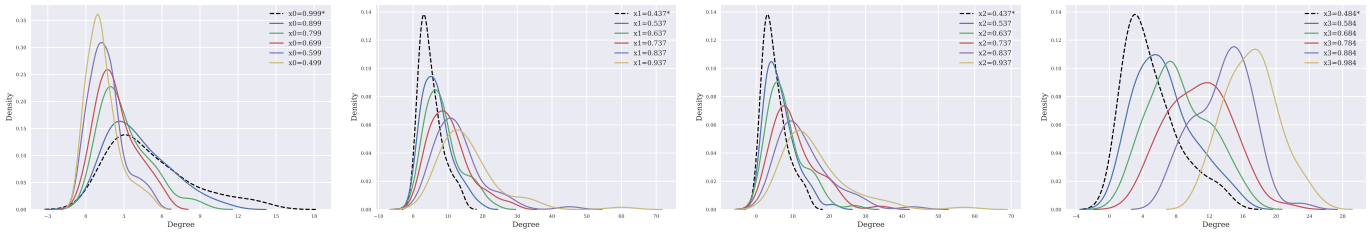
For noise analysis in Figure 6, we evaluate the effect of varying random noise (varying the random noise sparse matrix density: alpha) on: (a) Degree, (b) In-Degree, (c) Out Degree, (d) Betweenness Centrality, (e) Closeness Centrality, (f) Laplacian Centrality, (g) Clustering Coefficient, and (h) Scree Plot (singular values). The dashed black lines represent the distributions for the original initial graph G_0 with no noise injected.

A. Effect of Varying Kronecker Initiator Values

1) varying x_0: ***For the degree distribution***, Figure 3a shows that increasing x_0 value results in increasing the maximum degree across nodes, and decreasing it decreases the maximum degree. It is also seen that the minimum degree is not affected due to x_0 variation. This means that increasing the value of x_0 flattens the degree distribution of the graph, where the peak density (or frequency) of nodes with low degree is decreased (as compared to lower x_0 values) and the peak of the distribution gradually shifts towards the right bottom. This behavior is consistent across the different values we tested for x_0 within the legal range.

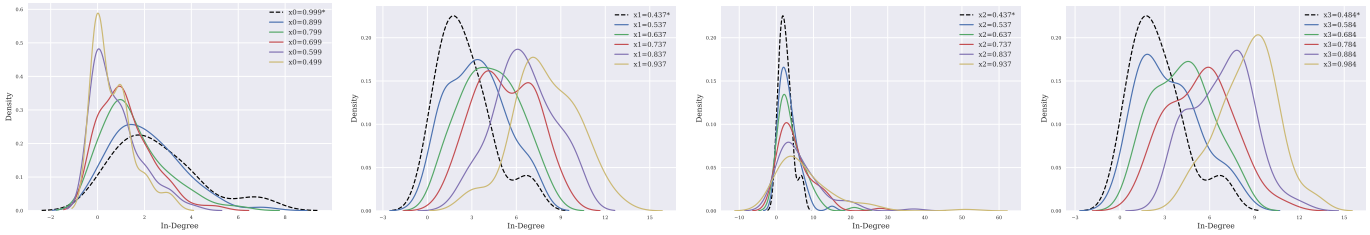
In-degree distribution exhibits a similar behavior to the overall degree distribution as shown in Figure 4a: maximum degree increases with increasing the value of x_0, and the minimum degree is still unaffected, which squishes the peak density of the lower degrees into a flatter curve with lower peaks as compared to smaller x_0 values.

As for out-degree, increasing x_0 still increases the maximum out-degree. However, x_0 contribution to increase of out-



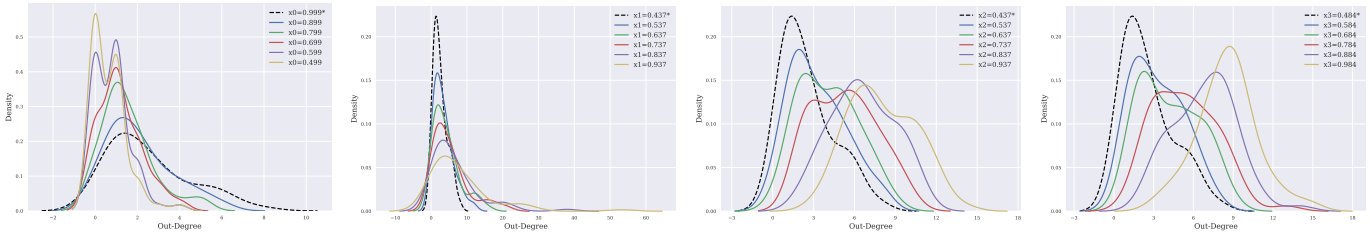
(a) (b) (c) (d)

Fig. 3: The effect of changing (a)x_0 , (b)x_1 , (c)x_2, and (d)x_3 on the degree distribution of the resulting $K6$ Kronecker Graph. Each of the sub-figure is a KDE plot where the degree distribution are on the horizontal axis, and the density is on the vertical axis.



(a) (b) (c) (d)

Fig. 4: The effect of changing (a)x_0 , (b)x_1 , (c)x_2, and (d)x_3 on the in-degree distribution of the resulting $K6$ Kronecker Graph. Each of the sub-figure is a KDE plot where the degree distribution are on the horizontal axis, and the density is on the vertical axis.



(a) (b) (c) (d)

Fig. 5: The effect of changing (a)x_0 , (b)x_1 , (c)x_2, and (d)x_3 on the out-degree distribution of the resulting $K6$ Kronecker Graph. Each of the sub-figure is a KDE plot where the degree distribution are on the horizontal axis, and the density is on the vertical axis.

degree is limited compared to in-degree as illustrated in Figure 5a.

2) Varying x_1: ***For the degree distribution***, Figure 3b shows the effect of varying x_1. A similar trend to x_0's is observed. However, increasing x_1 rapidly increases the maximum degree to a higher value.

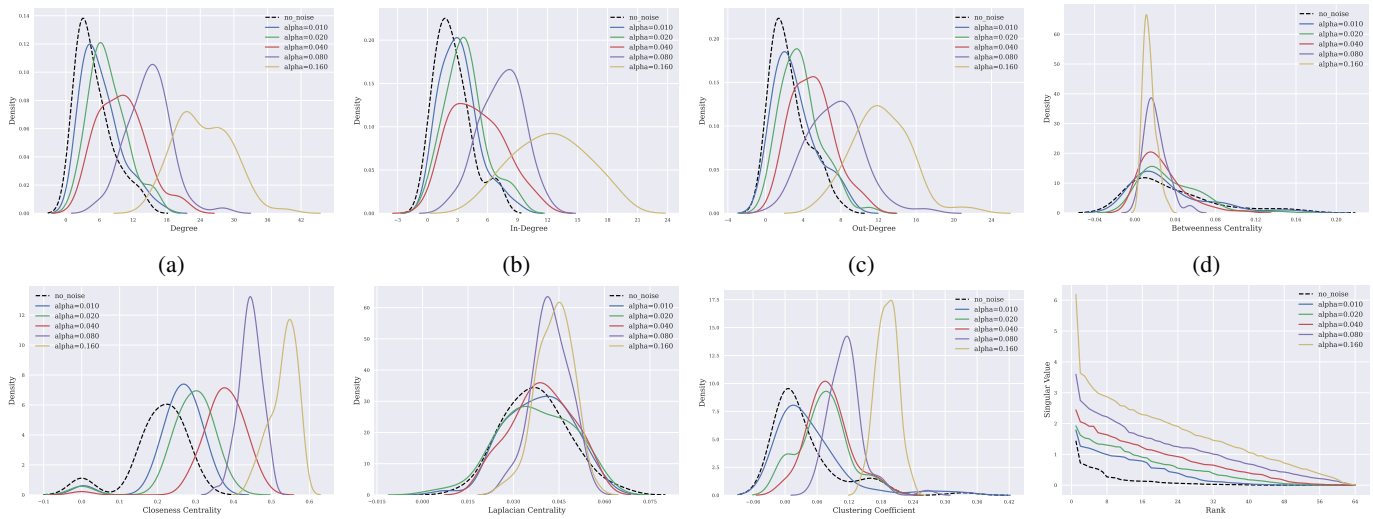
On the other hand, changing x_1 shows a different trend of increasing both maximum and minimum ***In-degree*** as shown in Figure 4b, even though the increase in the minimum degree is slower. Also, the maximum in-degree for the maximum x_1 value is significantly lower than that of the overall degree. So, most of the degree increase accounts to the out-degree increase and not the in-degree. An extended range of maximum degree increase proves this behavior in Figure 5b.

3) Varying x_2: The effect of varying x_2 on the resulting graph degree is similar to that of x_1 as shown in Figure 3c, and with similar values for maximum degree.

However, the opposite is true for ***In-Degree and Out-Degree***. Figure 4c illustrates that in-degree increase dominates the majority of the degree increase. The peak density still decreases with increasing x_2, but the range of in-degrees that fall within the peak density is narrower than what is observed for x_1.

x_2 changes the out-degree distribution in smaller steps than it does for the in-degree distribution as illustrated by 5c.

4) Varying x_3: Figure 3d demonstrates the effect of varying x_3 on the ***overall degree*** distribution, where increasing x_3 value affects both the minimum and maximum degree across nodes. Smaller x_3 values have both smaller minimum and



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 6: The effect of changing the injected sparse random noise matrix density on (a) degree, (b) in-degree, (c) out-degree, (d) betweenness centrality, (e) closeness centrality, (f) Laplacian centrality, and (d) scree plot of the resulting Graph. Each of the sub-figure is a KDE plot where the degree distribution are on the horizontal axis, and the density is on the vertical axis.

maximum degree values as compared to higher x_3 values. Changing x_3 values shifts the degree distribution to the right or the left (change min and max degree), but has a smaller effect on the degree density across nodes (peak density/frequency is slightly affected), which is different from the effect of changing x_0, x_1 , and x_2 on the degree density as described before.

Similarly, for ***In-Degree*** both tails of the density distribution are extended without a significant change in the peak density as exhibited in Figure 4d. x_3 variation has an almost identical effect on both in-degree and ***out-degree***.

5) General Observations: From the above analysis, it is obvious that certain features are more sensitive to certain parameters than others. For example: Figure 3 shows that degree is more sensitive to varying x_3 than x_0. In-degree (Figure 4) is highly sensitive to changes in x_1, while out-degree is more reactive to changes of x_2. Also, as x_3 values increases (gets closer to x_0), degree distributions start to diverge quickly (the mean degree shifts significantly). Additionally, Figure 4b shows that x_1 change has the highest effect of in-degree, and Figure 5c demonstrates that x_2 has a significant impact on out-degree distribution.

B. Varying noise

To evaluate the robustness of the correlation between the graph descriptor (Kronecker initiator matrix values) and the graph structure, we inject random noise to observe how the structure reacts to noise. The noise injected is a random sparse matrix, that is added to the adjacency matrix of the generated Kronecker graph. Then, we vary the density (sparsity) of the random matrix, and evaluate the effect of noise with different values for density (alpha).

Figures 6a, 6b, and 6c show that for lower values of alpha (up to around 2%), the resulting graph maintains a very similar degree distribution, compared to the original graph with no noise. Beyond that point, the random noise starts to take over until the Kronecker characteristics of the original graph are no longer recognizable.

Figure 6d shows a similar behavior for the betweenness centrality of the graphs. Random noise with alpha above 4% distorts the original graph betweenness centrality distribution.

Closeness centrality distribution is more sensitive to random noise as shown in Figure 6e, where the entire distribution shifts to the right, with the minimum closeness tail almost fixed. However, the pattern still persists where for injected noise with alpha beyond 4%, the distribution completely changes.

Laplacian centrality in Figure 6f is less sensitive to the noise as compared to closeness centrality, for lower values of alpha (up to 4%). With higher alpha, the distribution skews and is mainly random noise.

The Clustering coefficient by nature is highly sensitive to graph structure changes as shown in Figure 6g. Any noise with alpha greater than 1% immediately disrupts the distribution. Such behavior is expected since the clustering coefficient captures the local connections in a graph, and how it propagates globally. When adding a few new edges (by injecting random noise with low alpha value), the local connectivity patterns are immediately affected: local clusters may be disrupted or new clusters may form, eventually leading to changes in the clustering coefficient values for individual nodes, and then the overall distribution. However, one can still notice the significant difference between inducing noise with lower alpha values (up to 4%) and inducing noise with higher alpha values.

The Scree plot of a graph captures the relative importance

of components (nodes) in the graph. The importance of a component in a graph is mainly represented by the number of connections (degree) of the component. Inducing the random noise in our experiments simply adds new edges to the graph, so its relative effect on scree plot is not expected to be significant as shown in Figure 6h. However, adding noise with higher alpha values increases the (absolute) spectral gap between adjacent singular values.

V. Conclusion

In this paper, we proposed a novel framework to evaluate graph descriptors. The framework takes as input the graph descriptor (generation parameters) and generation function, and evaluates the co-relation between the descriptor values and the underlying graph structure. It captures the sensitivity of the graph structure to the descriptor values, as well as to injected random noise. Changes in graph structure are detected by observing the change in different graph structural properties such as degree, in-degree, out-degree, betweenness centrality, closeness centrality, laplacian centrality, clustering co-efficient, and singular values. We provided a case study of using the framework by analysing the sensitivity of stochastic Kronecker graph structure to the initiator matrix values used to generate them, and to induced random noise (sparse random matrix) with varying density. The design of the framework is modular so that it can evaluate different existing and future graph models, and additional graph properties can be easily calculated for evaluated graphs.

References

[1] P. Erdős, A. Rényi *et al.*, "On the evolution of random graphs," *Publ. math. inst. hung. acad. sci*, vol. 5, no. 1, pp. 17–60, 1960.

[2] R. Albert and A.-L. Barabási, "Statistical mechanics of complex networks," *Rev. Mod. Phys.*, vol. 74, pp. 47–97, Jan 2002. [Online]. Available: https://link.aps.org/doi/10.1103/RevModPhys.74.47

[3] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[4] X. Guo and L. Zhao, "A systematic survey on deep generative models for graph generation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 5, pp. 5370–5390, 2022.

[5] M. Suhail, A. Mittal, B. Siddiquie, C. Broaddus, J. Eledath, G. Medioni, and L. Sigal, "Energy-based learning for scene graph generation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 13 936–13 945.

[6] R. Liao, Y. Li, Y. Song, S. Wang, W. Hamilton, D. K. Duvenaud, R. Urtasun, and R. Zemel, "Efficient graph generation with graph recurrent attention networks," *Advances in neural information processing systems*, vol. 32, 2019.

[7] M. Yoon, Y. Wu, J. Palowitch, B. Perozzi, and R. Salakhutdinov, "Graph generative model for benchmarking graph neural networks," 2023.

[8] D. Chakrabarti and C. Faloutsos, "Graph mining: Laws, generators, and algorithms," *ACM Comput. Surv.*, vol. 38, no. 1, p. 2–es, jun 2006. [Online]. Available: https://doi.org/10.1145/1132952.1132954

[9] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2009.

[10] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani, "Kronecker graphs: An approach to modeling networks," *J. Mach. Learn. Res.*, vol. 11, p. 985–1042, mar 2010.

[11] A. S. Ramani, N. Eikmeier, and D. F. Gleich, "Coin-flipping, ball-dropping, and grass-hopping for generating random graphs from matrices of edge probabilities," *SIAM Review*, vol. 61, no. 3, pp. 549–595, 2019.

[12] J. Leskovec and C. Faloutsos, "Scalable modeling of real graphs using kronecker multiplication," in *Proceedings of the 24th International Conference on Machine Learning*, ser. ICML '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 497–504. [Online]. Available: https://doi.org/10.1145/1273496.1273559

[13] J. Kepner, "Analytic theory of power law graphs," in *SIAM Conference on Parallel Processing for Scientific Computing*, 2008.

[14] A. Zaki, M. A. Attia, D. Hegazy, and S. E.-S. Amin, "Comprehensive survey on dynamic graph models," *International Journal of Advanced Computer Science and Applications*, vol. 7, 2016.

[15] M. Drobyshevskiy and D. Turdakov, "Random graph modeling: A survey of the concepts," *ACM Comput. Surv.*, vol. 52, no. 6, dec 2019. [Online]. Available: https://doi.org/10.1145/3369782

[16] S. Ghafouri and S. H. Khasteh, "A survey on exponential random graph models: an application perspective," *PeerJ Computer Science*, vol. 6, 2020.

[17] C. Seshadhri, A. Pinar, and T. G. Kolda, "An in-depth analysis of stochastic kronecker graphs," *J. ACM*, vol. 60, no. 2, may 2013. [Online]. Available: https://doi.org/10.1145/2450142.2450149

[18] K. Abdelaal and R. Veras, "Observe locally, classify globally: Using gnns to identify sparse matrix structure," 2023. [Online]. Available: https://arxiv.org/abs/2309.02442

[19] C. Cai and Y. Wang, "A simple yet effective baseline for non-attribute graph classification," *arXiv preprint arXiv:1811.03508*, 2018.

[20] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graph evolution: Densification and shrinking diameters," *ACM Trans. Knowl. Discov. Data*, vol. 1, no. 1, p. 2–es, mar 2007. [Online]. Available: https://doi.org/10.1145/1217299.1217301