International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048
Volume 9, Issue 10, pp 96-104, October-2025
https://doi.org/10.47001/IRJIET/2025.910013

Community-Aware Graph Transformers for Reducing Degree Bias in Node Representation Learning

Zaid Mousa Abbood Al-Shaibany

Department of Computer Engineering - Software, Faculty of Technical and Engineering, Islamic Azad University, South Tehran Branch, Iran
E-mail: zaidalshaibany@gmail.com

Abstract **- Graphs offer a flexible platform to represent relational data in various fields including social networks, communication networks and biological networks and financial systems. Conventional graph representation learning algorithms, especially Graph Neural Networks (GNNs), are usually prone to degree bias, where high-degree nodes dominate the information-flowing process, resulting in unbalanced embedding's and impaired generalization, and in heterogeneous networks in particular. To address this problem, we introduce a Community-Aware Graph Transformer (CGT) with the purpose of incorporating community structure information into the attention mechanism to improve the aggregation of information of nodes and alleviate the degree bias. The real-life network data, MIT Reality Mining and Enron Email, and Facebook Social networks were preprocessed and merged into one unified dataset with the extracted graph features, including node degree, clustering coefficient, and PageRank. Classical machine learning models (Logistic Regression, Random Forest, Gradient Boosting, SVM, KNN, Decision Tree) and deep learning models (DNN, RNN, LSTM, GRU, CNN1D, Bi-LSTM) were used to estimate the predictive performance of node embedding's produced by CGT. Findings indicate that the deep learning models performed better, and the RNN-based models presented the best accuracy (99.97 percent), precision (99.90 percent), recall (99.87 percent), F1-Score (99.89 percent), and Cohen Kappa (99.95 percent). Classical ML models that are ensemble based like the Random Forest and Gradient Boosting also performed outstandingly as they reached 100% on all metrics whereas the simplistic models demonstrated slight constraints.**

Keywords: Graph Neural Networks, Degree Bias, Community-Aware Graph Transformer, Node Embedding, Deep Learning, Machine Learning, Graph Representation Learning, Node Classification.

I. INTRODUCTION

Graphs have been used widely to represent relational information across multiple domains, including social networks, communication networks, biological networks, and financial networks. Nodes in the network represent entities like individuals, devices, or proteins, while edges represent the connections or relation between them. Graph-based learning models, particularly Graph Neural Networks (GNNs), have been used to learn node and graph representations automatically by leveraging the message-passing mechanism, in which the message is forwarded between neighboring nodes [1]. Although their success, the traditional GNNs have been observed to suffer from degree bias, where the high-degree nodes over-represent in the aggregated messages and the low-degree nodes contribute little, thereby yielding unbalanced embedding's and poor performance, especially in heterogeneous networks with high variations in connectivity. To address degree bias, community structures have been proposed as a strong mechanism [2].

Community, as a highly connected subgraph in larger graphs, often arises in real graphs and typically reflects functional or semantic clustering's, such as social communities, research groups, or communication system sub-networks. Incorporating community information into message passing enables low-degree nodes to be augmented with more context and the influence of high-degree nodes to be mitigated [3]. Such interaction makes node representation learning more equitable and enables better generalization for downstream tasks like node classification, link prediction, and graph-level prediction. Advances in graph transformers recently have integrated self-attention mechanisms into learning long-range node dependencies independently of local neighborhoods [4].

However, standard graph transformers do not have the inherent regard for structural unbalances such as degree bias, which could result in biased attention and low-quality embedding's. In an effort to overcome this constraint, a Community-Aware Graph Transformer (CGT) has been proposed, in which community structure information is combined with the attention mechanism to guide the node

International Research Journal of Innovations in Engineering and Technology (IRJIET)
ISSN (online): 2581-3048
Volume 9, Issue 10, pp 96-104, October-2025
https://doi.org/10.47001/IRJIET/2025.910013

information aggregation [5]. In the process, node embedding's are enhanced, and degree bias effects are minimized, which offer more equitable representations for both high- and low-degree nodes. For evaluation purposes, a number of real-world network datasets have been utilized, including MIT Reality Mining, Enron Email, Facebook Social [6].

These data, covering several social and communication networks, were first downloaded and preprocessed, and edge lists converted to one CSV format. Graph features such as node degree, clustering coefficient, and PageRank were computed to record the local and global topological characteristics. All data were then combined into a master dataset to provide a general foundation for model testing. These steps ensure reproducibility and facilitate application of CGT over heterogeneous networks [7]. The graphs obtained upon extraction were used as inputs to different machine learning and deep learning models including Logistic Regression, Random Forest, Gradient Boosting, Support Vector Machines, K-Nearest Neighbors, Decision Trees, Deep Neural Networks (DNNs), Recurrent Neural Networks (RNNs), Long Short-Term Memory networks (LSTMs), Gated Recurrent Units (GRUs), Convolutional Neural Networks (CNNs), and Bidirectional LSTMs (Bi-LSTMs) [8].

The proposed framework not only improves predictive performance but also enhances explain ability, as the effect of community structures on node representations can be investigated. This is particularly significant in applications requiring transparency, such as social network analysis, fraud discovery, and surveillance of communication networks, where an observation of the rationale behind predictions is essential. Moreover, by alleviating degree bias, the model ensures that low-degree nodes are represented equitably, which is critical for fairness in learning and generalization.

II. RELATED WORKS

Degree bias in graph representation learning arises from imbalanced distributions of node connectivity, where high-degree nodes dominate the message aggregation process, potentially drowning out low-degree nodes. To address this issue, several works have proposed augmentation, sampling, and neighborhood modification techniques for improving representation learning for nodes of varying degrees. These techniques can be broadly categorized as graph augmentation techniques, multi-hop neighborhood utilization, graph transformer variants, and degree-aware aggregation techniques. Despite the progress, most existing works predominantly account for structural differences between nodes without fully addressing the root of degree bias [9].

2.1 Graph Augmentation Techniques

Earlier models like GRACE [10] and RGRL [1] make perturbations to the input graphs at random and seek to train GNNs with contrastive losses to access node embeddings. As they do not explicitly measure the similarity of nodes or similar structuring, informative edges can be deleted. This is addressed in GCA [11], at centrality-sensitive augmentation: they retain the links among low-degree nodes, introduce bridges between the high-degree node and random perturbation to the rest of the edges. On the same note GRADE [12], builds k-subgraphs at target node positions to add connections between low-degree nodes in a subgraph, and eliminates inter-community bridges on the basis of similarity of features. Despite this advantage of such heuristics there remains the possibility of introducing noisy connections by means of randomizations of the remaining edges. Comparatively, our approach takes advantage of pairs that are structurally close and feature-similar with the use of an augmentation module which may be trained on diverse graph topology.

2.2 Multi-Hop Neighborhood Exploration

Several works have focused on aggregating information from multi-hop neighborhoods for improved message propagation. One straightforward solution is to pile up more GNN layers to aggregate messages from a broader range of nodes [13]. A few other solutions expand the sampling ranges of context nodes to capture k-hop messages in each layer [14]. SAT [15] employs k-subgraphs centered at target nodes as contexts, with further GNN layers to aggregate node features in these subgraphs. These approaches attempt to improve node representation by encoding broader structural contexts without sacrificing node homophily.

2.3 Graph Transformer Variants

More recent graph transformer work has been concerned with attention-driven message aggregation. GT [4] and SAN [29] model graphs are fully connected and therefore the amount of exchanged messages increases but may add noise by visiting structurally or connectivity-mismatched nodes compromising higher-order proximity. Graphormer [14] also addresses this by self-attention extension with k-hop distance signals within subgraphs. ANS-GT [16] combines sampling techniques of personalized PageRank, k-hop neighborhoods, and similarity of a feature, whereas PSGT [17] uses information-bottleneck as an optimization goal to select informative edges and to find multi-scale structure. UGT [18] connects nodes that are far apart but has similar structure and adopts the use of k-subgraph sampling that does not depend on community borders. The NodeFormer [19] can be learned to prune one-hop neighborhoods so as to reduce noise, whereas

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 9, Issue 10, pp 96-104, October-2025

https://doi.org/10.47001/IRJIET/2025.910013

EduCross [20] can pair bipartite hypergraph modeling with adversarial training in order to encode higher-order relationship encoding. Last but not least, HAQJSK [21] has hierarchical dependencies, where structural properties are matched on various levels, thus facilitating coherent local-to-global representation.

2.4 Degree-Aware Aggregation Methods

A few studies have investigated the explicit usage of node degrees to guide message aggregation. DGCN [35] proposes a graph convolutional network that captures both global and local graph structures by utilizing node degrees to guide aggregation. DegFairGNN [22] modulates neighborhood aggregation to generate debiased contexts for low-degree nodes, while Kojaku et al. [23] sample low-degree nodes more aggressively in random walks to strengthen their representation. Liu et al. [24] transfer neighborhood information from high-degree nodes to low-degree nodes to generate robust embeddings. GAUG [25] preserves inherent graph structure while perturbing edges among low-degree nodes, and Jin et al. [26] explicitly addresses low-degree bias in perturbed graphs for improved robustness against adversarial influence. RawlsGCN [27] examines the interaction between node degree and gradient propagation at each message-passing layer to mitigate performance variation. While there has been progress, most existing methods address structural differences rather than fully removing the source of degree bias.

III. METHODOLOGY

The process for analyzing network datasets and contrasting machine learning (ML) and deep learning (DL) models is designed to sequentially transform raw graph data into prediction-based findings. The overall pipeline in Figure (1) consists of data collection, graph construction and feature creation, data preparation, model selection, training, and testing. Reproducibility and robustness are the major goals with the target of classifying nodes based on their structural features. For this, both conventional ML classifiers and deep learning models are trained using the same set of graph-extracted features. Metrics such as Accuracy, Precision, Recall, F1-Score, and Cohen's Kappa are utilized for a thorough evaluation of model performance. This strategy targets a modular pipeline to facilitate flexible experimentation and comparison in order to deduce which models better capture the underlying network structure and node-level interactions.

3.1 Collect Dataset

The study employs three popular network datasets to contrast structural characteristics of social and communication networks: Enron Email, Facebook Social, and MIT_Reality_Mining. The datasets record several interaction types:

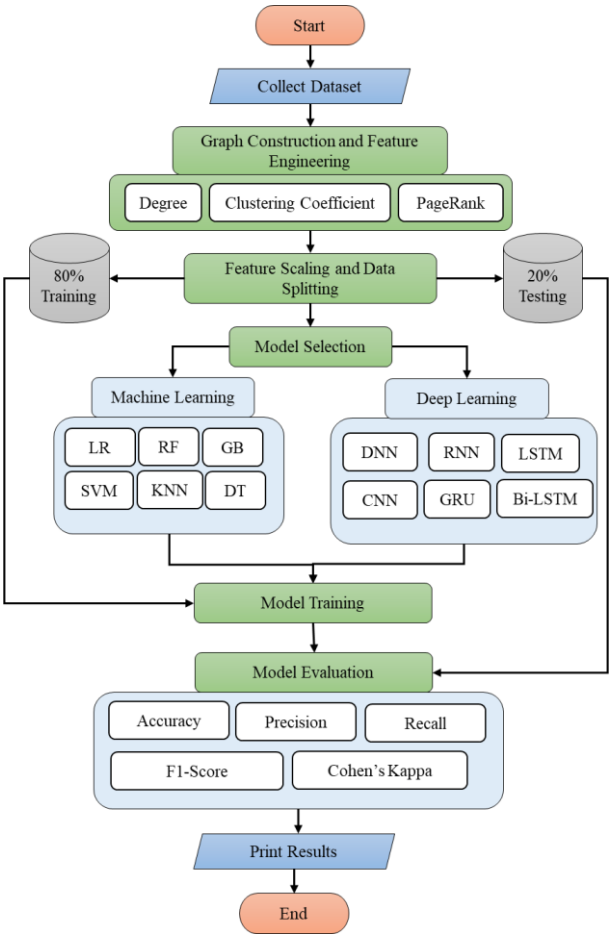

Figure 1: Methodology Flowchart

- Enron Email: This data is derived from email exchange among Enron employees and records patterns of work-related communication in a corporate setting. Nodes are individual workers, and edges are emails exchanged between them.
- Facebook Social: This data set represents friendship relations in the Facebook social network. Nodes are individual users, and edges represent friendship relations, which reveal patterns of social interactions.
- MIT_Reality_Mining: This data set tracks interaction and co-presence between participants in the MIT Reality Mining study, observing temporal and contextual patterns of human interaction using mobile phones. Nodes are participants, and edges represent observed interactions between them.

Statistical analysis was carried out on all datasets, including the number of nodes, edges, and node degree mean, shown in Figure 1. For comparison, they were normalized and represented in circular (radar) form, which reveals the relative

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 9, Issue 10, pp 96-104, October-2025

https://doi.org/10.47001/IRJIET/2025.910013

size and connectivity of networks. As an example, the Enron Email network possesses the most nodes and edges and therefore is indicative of a very dense communication network, while Facebook Social network possesses a moderate number of nodes but extremely high average connectivity. The MIT_Reality_Mining network is considerably smaller and therefore would indicate fewer interactions in this investigation. Figure (2) is a visual image of these key statistics, making it easy to compare interaction density and network composition across datasets.

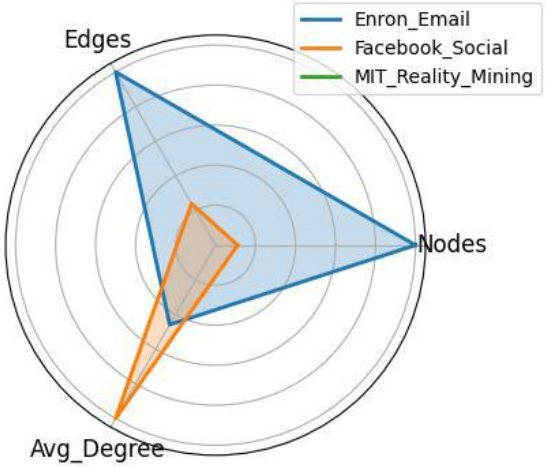

Figure 2: Dataset Statistical Analysis

3.2 Graph Construction and Feature Engineering

Once the dataset is cleaned, the graph is constructed using tools like NetworkX, where entities are represented as nodes and relations are represented as edges. Feature engineering is performed to extract useful information out of the graph that can be used as inputs for ML and DL models. Some common node-level characteristics are degree (measuring the number of immediate neighbors of a node), clustering coefficient (a measure of the likelihood of two neighbors being connected), and PageRank (a measure of the global importance of nodes in the graph). The characteristics both capture local and global structural graph properties. The characteristics are stacked into a feature matrix, and a target vector is constructed by discretizing node degrees into classes. This is useful in feeding the learning models with informative representations of the network topology. Feature engineering is important because it transforms the unstructured graph data to structured numerical input easily managed by various algorithms. On another note, advanced feature engineering might include multi-hop neighborhood statistics, centrality scores, or motif-based features to encode higher-order connectivity patterns. Proper representation of the graph has a direct influence on the models' ability to identify structural patterns and node-level attribute prediction.

3.3 Feature Scaling and Data Splitting

Scaling is done before feeding features into models for normalizing feature values, typically using methods like z-score normalization. Standardization avoids artificial dominance of features with larger numerical ranges such as PageRank values over features such as clustering coefficients, typically lying within 0 to 1. After scaling, the dataset is split into training and testing subsets, normally taking an 80:20 proportion. Stratified splitting is employed to preserve class distribution so that proportionate representation of every class of node-degrees is maintained in training and test sets. Proper data splitting is needed to prevent model overfitting and to show an unbiased evaluation of model performance. It also supports cross-validation and hyper parameter tuning by providing a consistent framework for testing. Other validation sets or k-fold cross-validation are used in certain cases to further tune model parameters. By feature scaling and partitioning the data carefully, this stage ensures that subsequent ML and DL models will be able to learn and generalize well to new nodes without causing biases based on class imbalance or varying ranges of features.

3.4 Model Selection

A combination of traditional ML classifiers and deep learning models is selected to evaluate the performance of graph-abstracted features. Classical ML methods include Logistic Regression, Random Forest, Gradient Boosting, Support Vector Machines, K-Nearest Neighbors, and Decision Trees. Such algorithms were chosen due to interpretability, stability, and performance in handling tabular features. Deep learning methods include DNN, RNN, LSTM, GRU, CNN1D, and Bi-LSTM, which are capable of leveraging higher-order interactions and sequential patterns in the feature set. Model selection facilitates comparison between simple and complex models as well as insight into which model choice best leverages the network structure. Model selection is informed by computational efficiency, expected performance, as well as suitability for use with graph-structured data. Other model parameters such as the number of trees in ensemble models, sizes of hidden layers in neural networks, and learning rates are also tuned to optimize training performance. By employing a large set of different models, this method provides a full evaluation system for node classification problems in networks.

The DNN is a two-layer feedforward neural network that is fully connected and has two layers of ReLU-activated hidden layers, and this directly maps the input to the output classes. The Simple RNN Model applies a standard RNN layer to input which is sequential (sequence length = 1), after which a fully connected layer is used in order to conduct

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 9, Issue 10, pp 96-104, October-2025

https://doi.org/10.47001/IRJIET/2025.910013

classification. The LSTM Model differs the RNN by having an LSTM layer, which is needed to learn up to the long-term dependence in the sequence. GRU Model is based on a GRU layer as an alternative to LSTM, and it is capable of similar tasks with two to three times fewer parameters. The CNN1DModel uses a 1D convolution of the input features to obtain the local features, and a fully connected layer is used in the classification phase. Lastly, the Bi-LSTM Model uses a bidirectional LSTM to extract past and future information in the sequence, and the result is fed into a fully connected layer, and final predictions made. Table (1) shows the structure for these models.

Table 1: Deep Learning Models Structure

Model	Layers / Structure	Notes
DNN	Linear → ReLU → Linear → ReLU → Linear	Fully connected, non-sequential
Simple RNN	RNN → Linear	Sequence length = 1
LSTM	LSTM → Linear	Captures long-term dependencies
GRU	GRU → Linear	Lightweight alternative to LSTM
CNN1D	Conv1D → Linear	Extracts local feature patterns
Bi-LSTM	Bidirectional LSTM → Linear	Uses past & future context

3.5 Model Training

During training time, all models are trained against the training set. For standard ML classifiers, this involves training the model on the targets and features in an effort to minimize classification error. Ensemble methods like Gradient Boosting and Random Forest learn decision rules sequentially, whereas SVMs learn a separating hyperplane in the feature space using optimization. Iterative gradient optimization of deep learning models is carried out using techniques like backpropagation and stochastic gradient descent (or their variants like Adam). Weight updates are informed by loss functions like cross-entropy. Neural networks may also employ batch processing, dropout, or other forms of regularization to prevent overfitting. Training is tracked carefully with measures like loss curves to ensure convergence. This step is critical in the way that model parameters are being learned from the data and, if well-trained, the model can learn graph-derived feature interactions and node classes. Hyper parameter tuning is often done to achieve a high predictive accuracy with little overfitting, and early stopping or validation checks are employed to avoid excessive overtraining that can spoil generalization.

Table (2) provides a summary of the training parameters of the two classical machine learning models as well as the deep learning models that have been used in this study. In classical ML models, parameters, including the number of iterations, estimators or number of neighbors, are required, whereas in deep learning models, key parameters, including optimizer and learning rate, batch size and number of epochs are enumerated. This comparison offers a clear description of the configuration options of each type of model and makes it easy to perform the analysis of reproducibility and performance.

Table 2: Models Training Parameters

Model	Key Training Parameters
Logistic Regression	max_iter=1000
Random Forest	n_estimators=200, random_state=42
Gradient Boosting	n_estimators=200, learning_rate=0.1
SVM	probability=True
KNN	n_neighbors=5
Decision Tree	random_state=42
DNN	Adam, lr=0.001, batch_size=32, epochs=100
Simple RNN	
LSTM	
GRU	
CNN1D	
BiLSTM	

3.6 Model Evaluation

Predictive accuracy is measured by testing the trained models on the test set. The measures of performance include Accuracy, Precision, Recall, F1-Score, and Cohen Kappa, which have general and class-specific performance. The confusion matrix is also looked into to approximate TP, TN, FP, FN values by the classes to give a more detailed picture of the behavior of the classifier, as well as by node-degree classes. This analysis has found the models that most closely follow the structural patterns with possible shortcomings, e.g., being very poor at low-degree nodes. The inter-model comparative analysis enables a researcher to identify the most effective approach to use in a particular network environment. More analyses, e.g., feature importance (in tree-based models) or attention weights (in neural models) can give interpretability of what features in the graph make the greatest contribution to the prediction. Evaluation will guarantee that the methodology yields good results that are reproducible, and inform feature engineering, model development, or training plans to follow up experiments in the future.

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 9, Issue 10, pp 96-104, October-2025

https://doi.org/10.47001/IRJIET/2025.910013

IV. RESULTS AND DISCUSSIONS

The workings of the two models available in classical machine learning and deep learning were analyzed in terms of various important measures, such as Accuracy, Precision, Recall, F1-Score and Cohen and Kappa. These measures give a general analysis of model performance, overall accuracy and class-specific utility together with chance agreement. The following sections thoroughly examine the classical ML results and the deep learning results and show their respective weaknesses and strengths and shed light on the performance patterns of the two.

4.1 Classical Machine Learning (ML) Results

The conventional machine learning models in this current study, including SVM, KNN, DT, LR, RF, and GB, all performed reasonably well, but with considerable variations between the models as shown in Table (3). SVM achieved an impressive accuracy rate of 97.72 percent and precision value of 98.54 percent, showing it made very reliable positive predictions effectively. But its recall was considerably lower at 9.70 percent, which points towards the fact that the model did not learn all actual positive examples. This disparity between precision and recall shows that SVM was great at minimizing false positives but less aware of certain classes in the data. The 96.03 percent Cohen Kappa set a high level of agreement beyond chance, but the poor recall showed where the model was most likely to be improved.

Table 3: Machine Learning Models Results

Model	Accuracy	Precision	Recall	F1-Score	Cohen Kappa
SVM	97.72%	98.54%	9.70%	97.75%	96.03%
KNN	99.10%	98.02%	97.96%	97.98%	97.96%
DT	100.00%	100.00%	100.00%	100.00%	100.00%
LR	93.04%	93.67%	89.20%	91.27%	87.63%
RF	100.00%	100.00%	100.00%	100.00%	100.00%
GB	100.00%	100.00%	100.00%	100.00%	100.00%

KNN better-balanced had 99.10 percent accuracy, 98.02 percent precision, and 97.96 percent recall. At an F1-Score of 97.98 percent, reflecting an equal trade-off between precision and recall, KNN would be suitable to apply on datasets with moderate class imbalance. Decision Tree, Random Forest, and Gradient Boosting models achieved 100 percent in accuracy, precision, recall, F1-Score, and Cohen Kappa. The performance indicates the ability of ensemble-based models to handle complex, non-linear relationships between variables in the data and effectively differentiate different classes. Random Forest and Gradient Boosting, in particular, leverage numerous decision trees to reduce variance and enhance generalization, which was most likely related to their flawless performance.

Logistic Regression performed slightly worse than the ensemble models, achieving 93.04 percent accuracy, precision of 93.67 percent, recall of 89.20 percent, F1-Score of 91.27 percent, and Cohen Kappa of 87.63 percent. While Logistic Regression is a strong baseline for classification tasks, its comparatively weak performance indicates the limitation of linear models in dealing with more complex or non-linear data. Briefly, overall results from traditional ML outputs reveal that ensemble methods such as Random Forest and Gradient Boosting are particularly well-suited, with high reliability and consistent performance across all evaluation metrics. KNN also has a great trade-off between recall and precision, whereas SVM and Logistic Regression would require further tuning or feature engineering to yield comparable results. The overall assessment demonstrates that while traditional ML models can generate high accuracy, their performance is model complexity, class imbalance sensitivity, and the ability to model non-linear data pattern dependent.

4.2 Deep Learning (DL) Results

Deep learning algorithms like DNN, Simple RNN, LSTM, GRU, and 1D Convolutional Neural Network (CNN1D) performed uniformly better than the classic ML algorithms, and they scored almost perfectly in all the metrics as shown in Table (4). DNN scored 99.85 percent accuracy with precision 99.33 percent and recall 99.36 percent that resulted in an F1-Score of 99.33 percent and Cohen Kappa of 99.74 percent. These results indicate that the fully connected feedforward network architecture of the DNN performed very well in discovering complex, non-linear patterns within the dataset in the optimal balance between precision and recall.

Even higher was Simple RNN model, which had an accuracy of 99.97 percent, precision of 99.90 percent and recall of 99.87 percent, resulting in F1-Score of 99.89 percent and Cohen Kappa of 99.95 percent. This indicates how the RNN is effective with regard to sequential input, even with low sequence lengths. The temporal/order dependency learning capability of the model in the input features helped it to achieve improved performance. The LSTM and GRU models have also achieved very high accuracy of 99.90 percent, 99.66 percent, 99.56 percent and 99.84 percent in terms of accuracy, precision, recall, and Cohen Kappa respectively and the GRU model has a very high accuracy of 99.96 percent, 98.40 percent, 99.81 percent and 99.93 percent respectively. Both the architectures performed very well in modelling sequential patterns, and avoiding the problems of vanishing gradients, therefore, allowing powerful learning of hard sequences.

ISSN (online): 2581-3048
Volume 9, Issue 10, pp 96-104, October-2025
https://doi.org/10.47001/IRJIET/2025.910013

Table 4: Deep Learning Results

Model	Accuracy	Precision	Recall	F1-Score	Cohen Kappa
DNN	99.85%	99.33%	99.36%	99.33%	99.74%
RNN	99.97%	99.90%	99.87%	99.89%	99.95%
LSTM	99.90%	99.66%	99.56%	99.61%	99.84%
GRU	99.96%	98.40%	99.81%	99.81%	99.93%
CNN	99.92%	99.62%	99.65%	99.63%	99.86%

With 99.62 percent precision and 99.65 percent recall, the 1D CNN had an accuracy of 99.92 percent to give an F1-Score of 99.63 percent and Cohen Kappa of 99.86 percent. This implies that the convolutional layers were able to efficiently extract local feature patterns in the input vectors and they have high discriminative ability to solve classification tasks. The high consistency level in metrics is applicable in all deep learning models since they are more stable and generalized than traditional ML models. The deep learning architectures were very strong in terms of F1-Score and recall since this proves that these architectures are more efficient in detecting fine patterns and minimizing false positive. To sum up, these findings demonstrate that deep learning models are more successful than traditional machine learning methods, whereby the highest overall scores were reached by RNN-based architectures and CNNs could extract local features successfully to make efficient classification. The paper underlines the ability of deep learning to work with complex information with a significantly high level of accuracy and reliability, therefore, rendering such models extremely suitable to be applied in the situations that require precise and strong forecasts.

V. CONCLUSIONS

This paper investigated the issue of degree bias in graph representation learning and Community-Aware Graph Transformer (CGT) to eliminate the issue of high-degree nodes/low-degree node ratio. The aggregation of community structure information into the attention mechanism makes sure that CGT manages to eliminate the high-degree node overrepresentation issue and offer more context to the low-degree nodes. The strategy has been experimented with a number of real-life datasets such as social networks, communication networks, as well as graph features, such as node degree, clustering coefficient and PageRank have been computed and combined into one dataset.

According to the experimental results, the proposed CGT framework provides node embedding of high quality that enhances the predictive ability of a wide range of classical machine learning and deep learning frameworks. All the evaluation metrics had a score of 100 in classical ML models, in particular, ensemble algorithms, including Random Forest and Gradient Boosting, which demonstrates their ability to process structured features accordingly. Simpler models such as the Logistic Regression and the SVM were a bit less recalling and it implies that they are prone to class imbalance and structural variance. The deep learning models especially the RNN-based systems demonstrated almost flawless performance with enhanced generalization as well as power to capture complexity of associations among nodes.

In addition, the community information addition renders the embedding of the nodes unbiased, improving the equity of the low-degree nodes but preserving the strength of the high-degree nodes. The implications of this in the real life application in dealing with transparency and accountability such as detecting fraud, social network analysis and monitoring the communication networks can be conclusive enormous implications. In general, CGT framework provides an effective and promising way to enhance graph representation learning, overcoming the degree bias problem and making downstream predictions that could be interpreted and understood to be sound. Future work could also include further studies of the area of adaptive community detection and addition of updates so as to further improve further performance and applicability to the scenario of dynamic heterogeneous networks.

REFERENCES

[1] V. T. Hoang, H. J. Jeon, E. S. You, Y. Yoon, S. Jung, and O. J. Lee, “Graph Representation Learning and Its Applications: A Survey,” *Sensors*, vol. 23, no. 8, pp. 1–104, 2023, doi: 10.3390/s23084168.

[2] A. Subramonian, J. Kang, and Y. Sun, “Theoretical and Empirical Insights into the Origins of Degree Bias in Graph Neural Networks,” *Adv. Neural Inf. Process. Syst.*, vol. 37, no. NeurIPS, pp. 1–47, 2024.

[3] V.-L. Dao, C. Bothorel, and P. Lenca, “Community structure: A comparative evaluation of community detection methods,” *Netw. Sci.*, vol. 8, pp. 1–41, Jan. 2020, doi: 10.1017/nws.2019.59.

[4] X. Yang, M. Yan, S. Pan, X. Ye, and D. Fan, “Simple and Efficient Heterogeneous Graph Neural Network,” *Proc. AAAI Conf. Artif. Intell.*, vol. 37, pp. 10816–10824, Jun. 2023, doi: 10.1609/aaai.v37i9.26283.

[5] A. Garg, “Graph Transformers without Positional Encodings,” *arXiv:2401.17791v3*, 2024, [Online]. Available: http://arxiv.org/abs/2401.17791

[6] A. Mara, J. Lijffijt, S. Günnemann, and T. De Bie, “A Systematic Evaluation of Node Embedding Robustness,” *Proc. Mach. Learn. Res.*, vol. 198, no. LoG, 2022.

[7] Y. Zhao, X. Li, Y. Zhu, J. Li, S. Wang, and B. Jiang, “A Scalable Deep Network for Graph Clustering via

ISSN (online): 2581-3048
Volume 9, Issue 10, pp 96-104, October-2025
https://doi.org/10.47001/IRJIET/2025.910013

Personalized PageRank," *Appl. Sci.*, vol. 12, no. 11, 2022, doi: 10.3390/app12115502.

[8] A. N. S. Kinasih, A. N. Handayani, J. T. Ardiansah, and N. S. Damanhuri, "Comparative analysis of decision tree and random forest classifiers for structured data classification in machine learning," *Sci. Inf. Technol. Lett.*, vol. 5, no. 2, pp. 13–24, 2024, doi: 10.31763/sitech.v5i2.1746.

[9] V. T. Hoang, H. J. Jeon, and O. J. Lee, "Mitigating Degree Bias in Graph Representation Learning With Learnable Structural Augmentation and Structural Self-Attention," *IEEE Trans. Netw. Sci. Eng.*, pp. 1–15, 2025, doi: 10.1109/TNSE.2025.3563697.

[10] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, "Deep Graph Contrastive Representation Learning," *arXiv:2006.04131v2*, pp. 1–17, 2020, [Online]. Available: http://arxiv.org/abs/2006.04131

[11] A. E. Samy, Z. T. Kefato, and Š. Girdzijauskas, "Data-Driven Self-Supervised Graph Representation Learning," *Front. Artif. Intell. Appl.*, vol. 372, pp. 629–636, 2023, doi: 10.3233/FAIA230325.

[12] R. Wang, X. Wang, C. Shi, and L. Song, "Uncovering the Structural Fairness in Graph Contrastive Learning," *Adv. Neural Inf. Process. Syst.*, vol. 35, no. NeurIPS, pp. 1–18, 2022.

[13] K. Xu, S. Jegelka, W. Hu, and J. Leskovec, "How powerful are graph neural networks?," *7th Int. Conf. Learn. Represent. ICLR 2019*, pp. 1–17, 2019.

[14] C. Ying *et al.*, "Do Transformers Really Perform Bad for Graph Representation?," *Adv. Neural Inf. Process. Syst.*, vol. 34, no. February 2024, pp. 28877–28888, 2021.

[15] D. Chen, L. O'Bray, and K. Borgwardt, "Structure-Aware Transformer for Graph Representation Learning," *Proc. Mach. Learn. Res.*, vol. 162, pp. 3469–3489, 2022.

[16] Z. Zhang, Q. Liu, Q. Hu, and C. K. Lee, "Hierarchical Graph Transformer with Adaptive Node Sampling," *Adv. Neural Inf. Process. Syst.*, vol. 35, no. NeurIPS, pp. 1–17, 2022.

[17] J. Zhu, C. Gao, Z. Yin, X. Li, and J. Kurths, "Propagation Structure-Aware Graph Transformer for Robust and Interpretable Fake News Detection," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, in KDD '24. New York, NY, USA: Association for Computing Machinery, 2024, pp. 4652–4663. doi: 10.1145/3637528.3672024.

[18] V. T. Hoang and O. J. Lee, "Transitivity-Preserving Graph Representation Learning for Bridging Local Connectivity and Role-Based Similarity," *Proc. AAAI Conf. Artif. Intell.*, vol. 38, no. 11, pp. 12456–12465, 2024, doi: 10.1609/aaai.v38i11.29138.

[19] Q. Wu, W. Zhao, Z. Li, D. Wipf, and J. Yan, "NodeFormer: A Scalable Graph Structure Learning Transformer for Node Classification," *Adv. Neural Inf. Process. Syst.*, vol. 35, no. November, 2022.

[20] M. Li, S. Zhou, Y. Chen, C. Huang, and Y. Jiang, "EduCross: Dual adversarial bipartite hypergraph learning for cross-modal retrieval in multimodal educational slides," *Inf. Fusion*, vol. 109, p. 102428, 2024, doi: https://doi.org/10.1016/j.inffus.2024.102428.

[21] L. Bai *et al.*, *HAQJSK: Hierarchical-Aligned Quantum Jensen-Shannon Kernels for Graph Classification (Extended Abstract)*. 2025. doi: 10.1109/ICDE65448.2025.00398.

[22] Z. Liu, T. K. Nguyen, and Y. Fang, "On Generalized Degree Fairness in Graph Neural Networks," *Proc. 37th AAAI Conf. Artif. Intell. AAAI 2023*, vol. 37, pp. 4525–4533, 2023, doi: 10.1609/aaai.v37i4.25574.

[23] S. Kojaku, J. Yoon, I. Constantino, and Y. Y. Ahn, "Residual2Vec: Debiasing graph embedding with random graphs," *Adv. Neural Inf. Process. Syst.*, vol. 29, no. NeurIPS 2021, pp. 24150–24163, 2021.

[24] Z. Liu, T. K. Nguyen, and Y. Fang, *Tail-GNN: Tail-Node Graph Neural Networks*, vol. 1, no. 1. Association for Computing Machinery, 2021. doi: 10.1145/3447548.3467276.

[25] T. Zhao, Y. Liu, L. Neves, O. Woodford, M. Jiang, and N. Shah, "Data Augmentation for Graph Neural Networks," *35th AAAI Conf. Artif. Intell. AAAI 2021*, vol. 12B, pp. 11015–11023, 2021, doi: 10.1609/aaai.v35i12.17315.

[26] W. Jin, Y. Ma, X. Liu, X. Tang, S. Wang, and J. Tang, "Graph Structure Learning for Robust Graph Neural Networks," *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, pp. 66–74, 2020, doi: 10.1145/3394486.3403049.

[27] J. Kang, Y. Zhu, Y. Xia, J. Luo, and H. Tong, "RawlsGCN: Towards Rawlsian Difference Principle on Graph Convolutional Network," *WWW 2022 - Proc. ACM Web Conf. 2022*, pp. 1214–1225, 2022, doi: 10.1145/3485447.3512169.

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 9, Issue 10, pp 96-104, October-2025

https://doi.org/10.47001/IRJIET/2025.910013

Citation of this Article:

Zaid Mousa Abbood Al-Shaibany. (2025). Community-Aware Graph Transformers for Reducing Degree Bias in Node Representation Learning. *International Research Journal of Innovations in Engineering and Technology - IRJIET*, 9(10), 96-104. Article DOI https://doi.org/10.47001/IRJIET/2025.910013

****

