On Positional and Structural Node Features for Graph Neural Networks on Non-attributed Graphs

Hejie Cui  
Department of Computer Science, Emory University  
Atlanta, GA, United States  
hejie.cui@emory.edu

Zijie Lu  
Department of Computer Science, University of Illinois at Urbana-Champaign  
Champaign, IL, United States  
zijielu2@illinois.edu

Pan Li  
Department of Computer Science, Purdue University  
West Lafayette, IN, United States  
panli@purdue.edu

Carl Yang*  
Department of Computer Science, Emory University  
Atlanta, GA, United States  
j.carlyang@emory.edu

ABSTRACT

Graph neural networks (GNNs) have been widely used in various graph-related problems such as node classification and graph classification, where the superior performance is mainly established when natural node features are available. However, it is not well understood how GNNs work without natural node features, especially regarding the various ways to construct artificial ones. In this paper, we point out the two types of artificial node features, i.e., positional and structural node features, and provide insights on why each of them is more appropriate for certain tasks, i.e., positional node classification, structural node classification, and graph classification. Extensive experimental results on 10 benchmark datasets validate our insights, thus leading to a practical guideline on the choices between different artificial node features for GNNs on non-attributed graphs. The code is available at https://github.com/zjzijielu/gnn-positional-structural-node-features.

CCS CONCEPTS

• Computing methodologies → Neural networks; Feature selection.

KEYWORDS

Graph neural networks, artificial node features, non-attributed graphs, positional and structural features

1 INTRODUCTION

Graphs provide a concise yet rich representation of data across different domains such as social networks, citation networks, gene-protein interactions, molecular structures and so on. How to effectively mine valuable information underneath graph data has become an appealing problem for data mining community. Recently, various kinds of powerful Graph Neural Networks (GNNs) demonstrate their privilege on common graph tasks such as node classification [23, 26], link prediction [25, 55, 57] and graph classification [2, 31, 47, 56]. GNNs combine both node features and graph structures by aggregating node features through links into low-dimensional vector representations. Recently, considerable efforts have been put on studying the complicated contents of networks, such as node types and informative [21, 49], pooling layers [28, 34], design spaces [10, 54], heterogeneous graph [32, 59], graph generation and transformation [15, 24, 46], graph learning schema [50], task-specific GNNs [11] and so on, where the superior performances are mainly established when natural node features (i.e., attributes) are available when applying GNNs.

However, a great number of graphs in the wild do not contain node attributes [8, 13], which deteriorates the performance of GNNs [5, 14]. For example, in the molecules dataset QM9 [39, 41], a graph represents a molecule, i.e., nodes are atoms and edges are chemical bonds. For typical tasks on this dataset such as predicting the properties of molecules, i.e., toxicity or biological activity, GNNs cannot be directly applied due to the lack of natural node features [8, 45]. Another example is the social network such as REDDIT. In this dataset, each graph represents a discussion thread, where each node corresponds to one user, and two nodes are connected by an edge if one user responded to a comment of the other [35]. The missing of node features for each user in these social networks will introduce extra difficulties in the task of sub-reddits prediction.

To apply GNNs on non-attributed graphs, several intuitive methods have been commonly practiced to initialize node features, such as degree-based [16], random [1, 42], one-hot [9], position-based [53], distance-based [29, 52] and so on. However, to the best of our knowledge, there exists no generic understanding or guideline towards the initialization of artificial node features based on the needs of downstream tasks. In this paper, we categorize common artificial node features and study their utility towards different tasks.
types of graph mining tasks. From a high level, these intuitive node feature initialization methods can be grouped into two categories, positional and structural ones [6] (Section 2). Take Figure 1 as an example. Positional features can help GNNs put node A and node B closer in the embedding space, whereas structural features facilitates putting node A and node C closer.

Extensive experiments are performed on 10 datasets with 8 common artificial features. Based on the information needs of different tasks, we further categorize them into multiple divisions, namely, positional node classification, structural node classification and graph classification (Section 3). Observations on the results validate our understanding that positional node features are more suitable for positional node classification, while structural node features benefit more for structural node classification and graph classification tasks. With appropriately designed artificial node features, the performance of GNNs can even surpass that with real features in some cases, as indicated in Table 3. Besides, our proposed novel degree-based node feature initialization method, i.e., degree bucket range, achieves state-of-the-art performance on structural node classification (Section 2.3). We believe this empirical study on the selection of artificial node features can facilitate the understanding of feature initialization on non-attributed graphs and inspire new designs of artificial node features, thus shedding light on various GNN applications on graphs in the wild.

2 TWO TYPES OF ARTIFICIAL NODE FEATURES

Several node feature initialization methods have been proposed for non-attributed graphs and commonly applied in various GNN models. We group these artificial node features into two main families: positional node features and structural node features.

2.1 Positional Node Features

Positional node features help GNNs capture node distance information regarding their relative positions in the graph. For example, in Figure 1, nodes A and B are positionally close. A real case is the publication network, where two authors who cite each other and also cite / get cited by similar other authors should be close considering their graph positions, and recognized as sharing similar research interests. Some intuitive positional node features include:

- one-hot: A unique one-hot vector is initialized for each node [14, 53]. This feature is essentially equivalent to random, when the first linear layer of the GNN are randomly initialized.
- eigen: Eigen decomposition is performed on the normalized adjacency matrix and then the top k eigen vectors are used to generate a k-dimensional feature vector for each node [7, 22, 58], where the optimal value of k is decided by grid search [33].
- deepwalk: The initial feature of a node is generated based on the DeepWalk algorithm from [37] with the walk length set as 40 by default. Deep walk features with walk length longer than 2 can help to capture higher-order positional information in the graph. Correspondingly, positional node classifications target at grouping nodes with respect to their positions, which corresponds to coarse global information in the graph. For example, in Figure 1, nodes A and B should be classified into the same class in the task of positional node classification. Specifically, eigen and deepwalk methods which generate features by matrix decomposition [38], are essentially dimension reduction, where the complex graph structures (i.e., adjacency matrices) information are embedded into a low dimensional representation. Therefore, eigen and deepwalk methods also incorporate structural information. However, as the features based on eigen and deepwalk reflect the position of nodes, with some abuse of terminology, we keep calling them positional features.

2.2 Structural Node Features

On the other hand, structural node features help GNNs capture structural information of nodes, such as degree information and neighborhood connection patterns. For example, in Figure 1, node A and C are similar regarding their neighborhood structures in the graph, though they are far away from each other in position. A real case is the molecular network, where two nodes with similar degrees and connection patterns should be put close considering their structures, and recognized as atoms with similar properties or functions. Some intuitive node feature initialization methods focusing on the structural aspects include:

- shared: An initial feature vector is shared across all nodes [14]. The shared feature we used is simply a vector of all 1’s.
- degree: An one-hot degree vector is initialized for each node, whose dimension is decided by the max degree of all nodes [16, 47].
- pagerank: The original PageRank score [4] of a given node is calculated and then flattened into a vector in order to fully utilize the embedding dimensions of neural networks, where the dimension of the extended vector is selected by grid-search [33]. It can be viewed as generalized higher-order node degree information. Structural node classifications target at classifying nodes according to their structural patterns. For example, nodes A and C in Figure 1 should be put into the same class considering their similar “structural roles”. Different from positional features that characterize the position of nodes in a graph, structural node features target at representing structural roles. Recent distance-based features [29, 52] also help to learn node structural roles while GNNs that leverage distance-based features cannot make inference over multiple nodes in parallel, which increases the computational complexity. Interested readers may refer to the experiments in [51] to check how distance-based features help with learning node structural roles.
2.3 Byproduct: New SOTA for Structural Node Classification

Motivated by our empirical studies on structural node features, we propose a novel node feature initialization method based on bucketing node degrees, which we name as degree+. Specifically, we divide degree values into several buckets, then map the degree values distributed in each bucket range into one class, and finally construct a unique one-hot vector for each class. Our proposed degree+ feature can be regarded as an improved version of the original degree-based node feature, which better handles the sparse and skewed distribution of node degrees in the graph.

3 EXPERIMENTAL RESULTS

3.1 Basic Settings

To conduct a fair and unbiased evaluation on the effectiveness of node features, we adopt the popular GNN of GraphSAGE [18] with mean and sum aggregators for all the artificial node feature initialization methods across different types of graph mining tasks. Results with real features are also provided wherever natural node features are available. The train/test/validation split of each dataset follows the standard practice in the literature [14, 27, 40]. Graph level experiments are conducted with artificial features of sizes ranging from 100 to 500 with step 100. In addition, we perform comprehensive grid search for the best hyper-parameter settings including the learning rate, number of epochs and neighborhood sample size. The final performance of each feature initialization method is averaged over five runs under the optimal hyper-parameter settings.

3.2 Positional Node Classification

Definition and Datasets. The tasks of positional node classification target at predicting the "positional role" of each node [20, 30, 44]. We consider three datasets for positional node classification, including Cora [43], Citeseer [43] and Pubmed [36]. These three citation networks consist of scientific publications as nodes, which can be classified into several content categories. Edges connecting those nodes denote the citation relationships between publications. Real features for each publication node are included in these three datasets, which are bag-of-word vectors indicating the word presence in the text content. In these three datasets, since the publications are connected by citation links, the research topic based node classification tasks should be mainly driven by the positions of nodes in the graph. Performance with the real node feature is also presented as a baseline for comparison.

Protocols and Performances. We train and test the GraphSAGE model using the same data splits as in [27], namely 20 randomly-selected samples for each class during training with a validation set of 500 samples. Experiment results of different node features on three positional node classification datasets are presented in Table 1, where $P$ and $S$ indicate the Type of artificial node features, corresponding to Positional or Structural respectively. Aggr denotes the aggregation method used in each GNN layer. Classification accuracy Acc(\%) is adopted here for evaluation.

Observations.

| Aggr. | Type | Feature | Cora Acc(%) | Pubmed Acc(%) | Citeseer Acc(%) |
|-------|------|---------|-------------|---------------|-----------------|
| $P$   | random | 56.1±1.6 | 42.3±1.4 | 36.0±1.0 |
|       | one-hot | 58.2±4.0 | 51.4±3.1 | 37.5±2.5 |
|       | eigen | 73.2±2.3 | 70.0±4.8 | 42.9±2.3 |
|       | deepwalk | 75.3±1.0 | 70.4±2.6 | 46.8±0.9 |
| Mean  | shared | 17.9±0.0 | 38.6±0.0 | 20.2±0.0 |
|       | degree | 37.4±2.1 | 41.1±2.9 | 36.0±1.3 |
|       | pagerank | 25.2±2.4 | 39.8±1.9 | 20.5±3.4 |
| $S$   | real feat. | 80.2±1.1 | 79.0±2.2 | 68.0±4.0 |

Table 1: Positional node classification results

- **Aggregation**: For positional node classification, mean aggregation shows better performance than sum aggregation, since mean aggregation can effectively filter out the influence of neighborhood size, which makes little contribution to and even impairs the performance on positional node classification. However, shared feature plus mean aggregation gives the same embedding for every node, so the results are constantly poor with no variance.
- **Cross Feature Type Comparison**: For positional node classification tasks, most positional node feature initialization methods achieve much better performance than structural node feature ones. The advantage of position node features over structural node features is especially remarkable with mean aggregation.
- **Within Feature Type Comparison**: Among all positional node features, random and one-hot achieve comparable results. This is because they are essentially the same: after passing through the first layer of neural network where the parameters are randomly initialized, one-hot initialization is equivalent to random initialization except for possible differences in dimensions (e.g., on Pubmed). Among all positional features, deepwalk and eigen demonstrate the best performance across all the datasets, which owes to the higher-order positional information they can capture.

3.3 Structural Node Classification

Definition and Datasets. The tasks of structural node classification target at predicting the "structural role" of each node [17, 19, 20]. Here we choose three datasets, namely American air-traffic network, Brazilian air-traffic network and European air-traffic network [40]. Given an airport node in the air-traffic network, the target is to predict passenger flow level of that node solely based on the structure of air-traffic network. These three datasets are chosen because the node labels of them indicate the structural roles (vary in four levels from hubs to switches), rather than the traditional community identifiers of nodes [16, 27, 43].

Protocols and Performances. Following struc2vec [40], we use 80% of nodes for training. To highlight the performance of our
novel degree+ method, we adopt logistic regression with L2 regularization to train the classifier using the representation learned by struc2vec [40], which demonstrates SOTA results on these datasets. Experiment results are presented in Table 2.

Observations.
• **Aggregation:** For structural node classification, sum aggregation outperforms mean aggregation because it can capture the number of neighbors, which is an important structural feature in graphs.
• **Cross Feature Type Comparison:** For structural node classification tasks, in most cases structural node features demonstrate superiority compared with positional ones, and our proposed structural node feature degree+ manifests the most distinct advantage over other positional features, reaching the new state-of-the-art.
• **Within Feature Type Comparison:** 1. among all four types of structural node features, degree+ improves on degree by using a degree bucket, where nodes with degree values in a range are projected together. This alleviates the node degree sparsity and skewness problem. 2. shared can only capture the sizes of multi-hop neighborhoods, but loses track of neighborhood structures, thus performing rather poorly. 3. In contrast, pagerank can be viewed as a generalized higher-order node degree, and we conjecture that its performance deterioration arises from over-smoothing which in the worst cases renders it as similar to shared.

### Table 2: Structural node classification results.

| Aggr. | Type | Initial. | USA-air Acc(%) | Brazil-air Acc(%) | Europe-air Acc(%) |
|-------|------|----------|----------------|------------------|------------------|
| Mean  | random | 59.3±1.8 | 45.7±5.9 | 44.9±5.8 |
|      | one-hot | 59.2±2.6 | 48.6±7.4 | 44.0±0.7 |
|      | eigen | 55.3±1.5 | 40.9±6.9 | 31.6±2.1 |
|      | deepwalk | 58.1±2.8 | 42.1±9.6 | 41.5±3.3 |
|      | shared | 25.0±0.0 | 25.0±0.0 | 25.0±0.0 |
|      | degree | 53.8±1.9 | 48.6±4.1 | 42.7±2.7 |
|      | degree+ | 59.2±2.7 | 60.0±3.0 | 50.6±3.9 |
|      | pagerank | 39.7±2.9 | 47.9±7.4 | 25.9±0.0 |
| Sum  | random | 60.7±3.2 | 47.9±7.4 | 48.9±5.1 |
|      | one-hot | 59.2±3.3 | 50.7±8.5 | 48.9±5.4 |
|      | eigen | 67.8±2.5 | 57.8±5.3 | 49.4±4.5 |
|      | deepwalk | 68.8±3.0 | 65.0±6.4 | 54.1±2.8 |
|      | shared | 55.7±2.0 | 61.4±4.7 | 45.4±1.0 |
|      | degree | 63.6±3.0 | 70.0±4.1 | 58.0±3.6 |
|      | degree+ | 69.1±2.6 | 76.4±4.1 | 61.2±3.8 |
|      | pagerank | 58.8±2.0 | 73.6±5.4 | 45.9±1.0 |
| SOTA | struc2vec | 63.8±1.6 | 73.6±9.6 | 58.8±3.0 |

### Table 3: Graph classification results.

| Aggr. | Type | Initial. | MUTAG Acc(%) | PROTEINS Acc(%) | IMDB-B Acc(%) | IMDB-M Acc(%) |
|-------|------|----------|--------------|----------------|--------------|--------------|
| Mean  | random | 64.9±4.1 | 67.2±4.2 | 58.0±2.9 | 36.1±1.9 |
|      | one-hot | 65.8±4.0 | 67.8±2.6 | 56.9±3.4 | 36.8±3.2 |
|      | eigen | 63.8±2.1 | 60.4±1.0 | 50.2±3.3 | 33.4±0.7 |
|      | deepwalk | 65.1±3.8 | 68.1±4.0 | 52.0±3.4 | 35.7±1.9 |
|      | shared | 66.7±0.0 | 59.6±0.0 | 50.0±0.0 | 33.3±0.0 |
| Sum  | degree | 84.4±7.7 | 69.5±3.2 | 69.7±5.1 | 41.5±2.6 |
|      | pagerank | 65.8±1.9 | 68.0±5.5 | 54.4±4.0 | 35.4±1.7 |
| SOTA | real feat. | 71.4±4.4 | 74.0±4.2 | - | - |

### Observations.
- **Aggregation:** Similar to structural classification tasks, sum aggregation outperforms mean aggregation on graph classification tasks, since the number of neighbors contributes as an important type of structural information for graph classification tasks.
- **Cross Feature Type Comparison:** For graph classification, though the best performance is not consistently achieved on a particular feature across four datasets, it always falls in the category of structural ones. This is because we do not care about positional information such as the specific position of each node in graph classification. Instead, similar to structural node classification, the overall structural information of the graph matters.
- **Within Feature Type Comparison:** 1. Among the structural node features, pagerank demonstrates better performance in most of the cases. 2. Impressively, the performances of GNN on degree on MUTAG and pagerank on PROTEIN with the sum aggregator even surpass those with real features. This further demonstrates the importance of choosing the appropriate artificial node features, sometimes even when natural node features are available.

### 4 CONCLUSION

Graphs in the real world do not always have natural node features available, due to the lack of task-specific node attributes, privacy concerns and/or difficulties in data collection. In this paper, we study the usage of artificial node features when applying GNNs on non-attributed graphs. We categorize commonly used artificial node features into two groups, positional node features and structural node features, based on what kind of information they can help GNNs capture. Extensive empirical experiments are conducted across three graph mining tasks, positional node classification, structural node classification and graph classification. The results validate our insights that positional node features are more suitable for positional node classification, while structural node features benefit more for structural node classification and graph classification tasks. We hope our empirical study can provide a generic and practical guideline for choosing the appropriate artificial node features and exploring more useful ones based on the needs of downstream tasks.
