DP-GCN: Node Classification Based on Both Connectivity and Topology Structure Convolutions for Risky Seller Detection

Chen Zhe  
Innovation Lab, PayPal  
Singapore  
zhen1@paypal.com

Aixin Sun  
Nanyang Technological University  
Singapore  
avsun@ntu.edu.sg

ABSTRACT
A payment network contains transactions between sellers and buyers. Detecting risky (or bad) sellers on such a payment network is crucial to payment service providers for risk management and legal compliance. In this research, we formulate this task as a node classification task. Specifically, we aim to predict a label for each seller in a payment network, by analysing its properties and/or interactions. Nodes residing in different parts of a payment network can have similar local topology structures. Such local topology structures reveal sellers’ business roles, e.g., supplier, drop-shopper, or retailer. We note that many existing solutions for graph-based node classification only consider node connectivity but not the similarity between node’s local topology structure. Motivated by business need, we present a dual-path graph convolution network, named DP-GCN, for node classification. DP-GCN considers both node connectivity and topology structure similarity. The proposed model consists of three main modules: (i) a C-GCN module to capture connectivity relationships between nodes, (ii) a T-GCN module to capture topology structure similarities between nodes, and (iii) a multi-head self-attention module to align both properties. We evaluate DP-GCN on seven benchmark datasets against diverse baselines. We also provide a case study of running DP-GCN on three large-scale payment networks from PayPal, one of the leading payment service providers. Experimental results demonstrate DP-GCN’s effectiveness and practicability.

1 INTRODUCTION
Node classification is to infer labels of unclassified nodes in a network by exploiting node properties [12]. Graph-based methods, especially graph convolution networks, have shown great success in various node classification applications. Example applications include payment network risk detection [19, 23], social network analysis, anomaly detection [44], and classification of molecules [42].

One limitation of existing graph convolution based solutions is that they only model connectivity between nodes, but not other types of node properties, e.g., nodes’ local topology structures. The main reason is that convolution process in graph convolution networks only propagates information along edges. Nodes residing at different parts of a graph may have similar local structural contexts, e.g., star topology, ring topology. We refer such local structural identity of nodes their local topology structures. Many real-world applications require to take nodes’ local topology structure into consideration. Take risky seller detection on payment network as an example. A large payment network typically contains millions of sellers and transactions. The risky sellers are those that result in revenue loss, due to different reasons e.g., fraud, bankrupt, bad suppliers. In order to detect such risky sellers in advance, we need to consider sellers’ connectivity relationships (e.g., interactions with other sellers, buyers, and suppliers), as well as their topology structural similarities (e.g., similarities on business models as supplier, drop-shopper, or retailer). We argue that existing graph network architectures are limited in modeling both properties.

Motivated by this real-world business need, in this paper, we propose a Dual-Path Graph Convolution Attention Network architecture, named DP-GCN, for node classification. DP-GCN combines both connectivity and topology structure properties, in an efficient and effective manner. Specifically, DP-GCN consists of a C-GCN module, a T-GCN module, and a multi-head self-attention module. The C-GCN module captures the connectivity relationships between nodes; the T-GCN module captures the topology structure similarities between nodes; and the multi-head self-attention module aligns both properties. To model topology structure efficiently, we propose role-based convolution, which extracts node topology structures into high-level latent roles for convolution. This strategy significantly reduces the computation cost for each node, but well retains all convolution paths between nodes that are similar in terms of topology structures.

In experiments, we evaluate DP-GCN on seven benchmark datasets. DP-GCN outperforms ten state-of-the-art baselines and achieves significant performance improvements on all datasets. We also conduct a careful ablation study and an embedding visualization to show that all components of DP-GCN are important for achieving the best performance. As a case study, we also evaluate DP-GCN on three large-scale payment networks from PayPal, to detect real risky sellers. This case study is challenging due to the size of networks, the diversity of risky seller types, and the high imbalance ratio. As expected, only a very small number of sellers are risky sellers. To summarize, we have made three contributions in this work.

- We propose DP-GCN, a dual-path graph convolution attention network, for effective node classification, based on both node connectivity and topology structure similarity.
- We propose role-based convolution, a novel and generalizable model, to make convolution process applicable to additional node properties aside connectivity.
- We evaluate DP-GCN on seven benchmark datasets and also provide a case study on large-scale real-world datasets. Together with ablation study and embedding analysis, we show DP-GCN’s effectiveness and practicability.

2 RELATED WORK
Our research is related to studies in four areas: node connectivity embedding, node topology structure embedding, graph neural network, and graph-based fraud detection.
Node Connectivity Embedding. Recent node embedding methods for graphs are largely based on the skip-gram model [4, 24]. In particular, Deepwalk [26] follows this approach to embed nodes to preserve nodes’ neighborhood relationships through random walk. Node2vec [8] optimizes this approach by adopting breadth-first and depth-first search in the random walk process. These methods become increasingly popular in both industry and research communities, as they outperform many existing methods with similar objectives. Due to the characteristics of random walk, those methods only focus on the connectivity relationship between nodes [8, 26, 33]; hence these models do not consider roles of nodes based on their local topology structures in networks.

Node Topology Structure Embedding. Recently, many methods are proposed to capture the topology structure of nodes in embedding. Struc2vec [28] constructs a multi-layer weighted graph that encodes the topology structure similarity between nodes. If two nodes have similar degree and their neighbors also have similar degrees, then these two nodes are similar in topology structure. Graphwave [6] captures the topology structure of nodes by leveraging spectral graph wavelet diffusion patterns. RoIX [10] extracts topology structure related features for nodes, then performs soft-clustering to group nodes based on those features. DeepGL [31] constructs role features by local graphlet decomposition. It learns graph functions where each represents a composition of relational operators that are applied to a graphlet/motif feature. Hone [30] describes a framework based on weighted k-step motif graphs to learn low-dimensional topology structure embeddings. Role2vec [1] derives topology structure features by computing motif patterns of nodes. It then computes random walk transition probabilities based on topology structure similarities, and embeds nodes using feature-based random walk. Recently, Riwalk [20] is proposed to model node topology structure by its degree and relative position. It can better integrate graph kernels with node embedding methods to leverage recent advancements in node embedding.

Graph Neural Networks. With advancement of deep learning, graph neural networks, especially graph convolution networks (GCN) have gained significant research interests [7, 9, 15, 18, 32, 35]. For the task of node classification, state-of-the-art solutions are dominated by GCN models. Different from skip-gram, GCN learns node representation by aggregating node’s local neighbourhood through functions. GraphSage [9] extends the vanilla GCN by using neighborhood sampling during the aggregation process. It introduces several aggregation functions such as “Mean Pooling”, “Max Pooling” and “LSTM function”. GCNII [3] is the deep graph neural network. Besides, GAT [35] introduces self-attention mechanism to graph neural network. The self-attention measures the importance of different nodes in neighborhood when performing aggregation. DEMO-Net [40] models node degree specifically. It introduces degree-specific mapping function for feature aggregation in the convolution.

There are also studies on dual-path convolution architecture. The work from [48] builds a PPMI matrix as the second path adjacency matrix to model graph global consistency. The authors in [46] use a dual-path architecture to embed image and text. It is a non-graph model but leverages dual-path to capture the relationship between image convolution and text convolution. DC-GCN [11] is a recent work for image question answering. It uses dual-path architecture to combine image and text for better answer prediction. AM-GCN [37] models node features with K-nearest neighbour (KNN) graph and propagates node features through both feature space and connectivity space simultaneously to better fusing node features and connectivity.

Different from all existing dual-path studies, our model considers both node connectivity and node topology structure similarities. In particular, the role-based topology convolution model in the T-GCN module of DP-GCN is a novel convolution model. This new model is generalizable and can be used to model other types of node properties. Moreover, the dual-path architecture designed in DP-GCN enables the two paths to be interactive, through a self-attention mechanism.

Graph-based Fraud Detection. Graph-based fraud detection usually focuses on analyzing graph connectivity and communities [27]. The system in [39] adapts HITS algorithm to detect fraudsters by propagating fraud score through graph. CatchSync [13] detects fraud by analyzing node properties (e.g., degree, HITS score, edge betweenness). The work [2, 22, 38] detect fraud by analyzing fraudulent communities. GCNEXT [16] is a detection model using expanded balance theory and GCN. In summary, those methods assume that fraudsters in network are usually connected in communities. Thus all methods are able to detect fraudsters by exploring nodes’ connectivity relationships. In our study, through extensive experiments, we show that combining both graph connectivity and topology structure is important and effective for graph fraud (risky seller) detection, and also node classification in general. Our work suggests a promising and practical direction for related applications.

3 DP-GCN ARCHITECTURE OVERVIEW

A graph is represented as $G = (V, E)$, where $V$ denotes the set of nodes and $E$ denotes the set of edges. The graph connectivity information is represented by a $|V|$ by $|V|$ adjacency matrix $A_c$, where $A_c[i][j] = 1$ if $v_i$ and $v_j$ are connected. Further, each node can be associated with node features. We use $F \in \mathbb{R}^{|V| \times d}$ to denote the feature matrix, where $d$ is the feature dimension. Usually, node features are specific to each individual node, e.g., demographic features of sellers in a payment network. Hence, node features are usually not related to the graph connectivity. In many related studies on graph convolution network, their solutions also support a non-feature option. In this case, an Identity matrix or Random matrix is used as feature matrix $F$ [25]. In our problem setting, we take both node features $F$ and graph $G$ as input, and predict the label for each node.

The overall architecture of DP-GCN is illustrated in Figure 1. We build our DP-GCN on top of the graph convolution framework, and its classification process is summarized as follows. Node representations are first initialized with the input features. We model node connectivity by the C-GCN module and model topology structure similarity by the T-GCN module, through a dual-path convolution process. The outputs from both modules i.e., C-GCN and T-GCN, are aligned by the multi-head self-attention module. This module automatically adjusts the importance of outputs from C-GCN and T-GCN, and learns the unified embedding representations of nodes.
The learnt unified embeddings are then fed to the next layer of dual-path convolution, and eventually fed into the classification module for class label prediction. In the following sections, we detail all the modules in DP-GCN.

## 4 DUAL-PATH CONVOLUTION

The dual-path convolution layer aims to learn node embeddings simultaneously for connectivity and for topology structure, through graph convolution in an end-to-end fashion.

Reviewed in Section 2, related studies [11, 46, 48] already show that dual-path convolution architecture is superior in learning unified node embedding. Our dual-path convolution consists of two interactive convolution modules: the C-GCN module and the T-GCN module. In our dual-path architecture, both modules take the unified embedding as input, thus the two modules will mutually reinforce each other, towards the common classification objective.

### 4.1 The C-GCN Module

The C-GCN module employs the standard graph convolution for modeling node connectivity relationship, which has been demonstrated to be effective in node representation learning. The input to the connectivity convolution layer is the node feature matrix \( F \), which represents the initial nodes’ embeddings. All nodes connectivity information are stored in the adjacency matrix \( A_c \), thus we directly apply convolution based on \( A_c \). Specifically, we use the symmetric normalizing trick described in [15] to normalize \( A_c \), to avoid changing the scale of feature vector:

\[
\hat{A}_c = \tilde{D}^{-\frac{1}{2}} (A_c + I) \tilde{D}^{-\frac{1}{2}}
\]

where \( \tilde{D} \) is the diagonal degree matrix and \( I \) is the Identity matrix. The Identity matrix is added to include each node’s self-feature into convolution. Each convolution layer takes the input from previous layers and generates node level embedding \( F^{(l+1)} \). The computation of the convolution is defined as:

\[
F^{(l+1)}_c = \sigma(\hat{A}_c H^l W)
\]

In this equation, \( W \) is the weight matrix, \( l \) denotes the \( l \)-th convolution layer. \( H \) denotes the unified embedding received from the multi-head self-attention module (see Figure 1), and the only exception is that \( H^1 = F \). The output of the C-GCN module \( F^{(l+1)}_c \) will be fed into the multi-head self-attention module together with the output from the T-GCN module, to produce the updated unified embedding \( H^{l+1} \).

Multi-layer convolutions enable the node to receive messages from a further neighbourhood [41]. As detailed above, our design is different from a typical convolution layer. The input to the successive convolution layers (i.e., when \( l > 1 \)) in the C-GCN module is the unified node embedding \( H^l \), rather than \( F^l \) as in typical convolution. This design allows the two modules C-GCN and T-GCN to mutually reinforce each other while modeling different types of node properties.

### 4.2 The T-GCN Module

The T-GCN module is designed to model the topology structure of nodes through graph convolution. The topology structure property is fine-tuned during the convolution, and finally integrated into the node embeddings.

Before we present our role-based topology convolution, we discuss threshold-based convolution model, a simple way to model topology structure by graph convolution. This simple model generates "augmented paths" between nodes if their topology structure similarity is above a threshold. Those augmented paths create a new
kind of neighborhoods for nodes that are similar in terms of local topology structures. In this way, graph convolution can aggregate nodes along the augmented paths, which naturally learn the node embedding based on topology structure similarity.

Specifically, a threshold-based convolution model is designed in four steps: (i) leverage local topology structure embedding algorithms to generate node features representing topology structure, (ii) calculate pair-wise similarity between nodes by topology structure features, (iii) create augmented paths between nodes if their topology structure similarity is above a predefined threshold, and (iv) perform graph convolution on augmented paths to fine-tune the node topology structure embedding. That is, we derive a topology structure adjacency matrix $A_t$ to capture the augmented paths between nodes. The convolution in Equation (1) can be directly adopted on the augmented paths, by replacing $A_e$ with $A_t$.

The time complexity for evaluating the pair-wise topology structure similarity between $n$ nodes is $O(n^2)$. When a graph is large, it is impractical to evaluate and generate the augmented paths between every pair of nodes, for enterprise-scale applications. To address this limitation, we propose a novel role-based convolution model to perform topology structure convolution.

### 4.3 Topology Role Construction

The main idea of role-based topology convolution is to summarize node topology structure with "latent topology roles". A topology role is a high-level representation of underlying topology structure identities of nodes, similar to the centroid of a cluster of nodes.

To initialize topology roles, we first obtain topology structure features for each node. These features can be generated by using node topology structure embedding models (see Section 2). Then we perform clustering or classification on topology structure features to discover the latent topology roles. Topology roles discovered by clustering preserve the original graph properties, while topology roles discovered by classification can be directly mapped into the task-specific space (i.e., node classification).

Once topology roles are discovered, we create dummy nodes on the graph to represent these topology roles. Then, belongingness edges between topology roles and their member nodes are added to retain their topology structure similarities. If two nodes are similar by their topology structure features, then the two nodes connect to the same topology role through belongingness edges.

In our implementation, we evaluate a few ways to generate topology structure features, e.g., Struc2vec [28], Graphwave [6], Role2vec [1] and Riwalk [20]. Based on the features, we discover the initial latent topology roles with $K$-means or regression classifier.

### 4.4 Topology Role Convolution

After topology roles are identified, the convolution process is designed to extract the topology roles’ representations, and to fine-tune their members’ embeddings, simultaneously. The convolution consists of information propagation in two stages, as illustrated in Figure 2.

In the first stage, information is propagated from member nodes to topology roles, through belongingness edges. This process learns or updates the embedding representations for topology roles. The convolution on each topology role $r_i \in R$ is expressed by:

$$r_i^{l+1} = \sigma \left( \frac{1}{m_{r_i}(i)}W \sum_{i \in m_{r_i}(i)} h_i^l \right)$$  (3)

where $l$ denotes the $l-th$ convolution layer, $W$ is the weight parameter, and $m_{r_i}(i)$ denotes the member nodes of role $r_i$. Similarly, the input to the successive topology role convolution layers ($l > 1$) are the unified node embedding $h \in H$.

In the second stage, the information is back-propagated from topology roles to their member nodes. This process fine-tunes the member nodes’ embeddings from their topology roles. Here, we assume that the underlying topology roles in the graph are non-overlapping. Thus we use a sharing scheme for this back-propagation. That is, nodes inherit the topology structure embeddings directly from their connected roles.

In implementation, the overall topology role convolution can be transformed into corresponding matrix multiplications as follows:

$$R^{l+1} = \sigma (\hat{A}_tH^TW)$$  (4)

where $R^{l+1}$ is the embedding matrix of topology roles and $\hat{A}_t$ is the $|R| \times |N|$ normalized membership matrix for topology roles. Then, we compute the topology embedding for each member node by

$$F_i^{l+1} = \hat{A}_t^TR_i^{l+1}$$  (5)

where $(\cdot)^T$ denotes transpose. Note that, we assume topology roles are non-overlapping; hence the topology role embedding is shared among all its members.

Role-based topology convolution is an effective way to model topology structure similarity between nodes. The strategy of using topology roles preserves the information propagation paths for all nodes that are similar in topology structures. As the convolution follows belongingness edges and each node has only one belongingness edge to its corresponding topology role, the complexity for the convolution is $O(m + n)$ for a graph with $n$ nodes and $m$ topology roles. Note that, the total number of topology roles is bounded by $O(n)$. In the worst case, each node is modeled by an individual topology role. But in this case, the topology structure information becomes trivial as there is no pair of nodes having similar topology structures.

With the T-GCN module, the proposed dual-path convolution framework is generalizable to model other types of node properties besides topology structure. It is also extendable to a multi-path convolution framework when there are more properties to be modeled.

### 5 MULTI-HEAD GRAPH SELF-ATTENTION

Based on previous studies [34, 35], we use a modified multi-head self-attention mechanism to enhance the correlation, and to adjust the importance, between connectivity relationship and topology structure similarities, in each layer of dual-path convolution. For simplicity, we denote node connectivity property by $c$ and topology structure similarity property by $t$. We obtain the attention coefficients for each property $j \in \{c, t\}$ of node $i$ as follows:

$$e_i^{l+1} = \text{LeakyReLU} \left( \alpha \left[ Wc_i \parallel W_t f_i^{l+1} \right] \right), j \in \{c, t\}$$  (6)

Again, $l$ denotes the $l-th$ convolution layer; $h_i$ is the unified embedding and $f_{ij}$ is the output from either C-GCN or T-GCN, i.e.,
Multi-Head Attention

weight can be naturally calculated by SoftMax function:

Thus the unified embedding is obtained by:

\[ L_{\text{LeakyReLU}} \]

\[
\text{to do prediction. The prediction layer is computed as follows:}
\]

\[
h_i^* = \text{elu}(h_i)
\]

\[
\hat{y}_i = \text{LogSoftMax}(h_i^*)
\]

According to [47] and also our empirical evaluations, it is beneficial to add an additional normalization layer on top of the unified embedding for classification. With normalization, the predictions are expressed as:

\[
h_i^* = h_i^*/\|h_i^*\|_2
\]

\[
\hat{y}_i = \log \left( \frac{\exp(h_i^*)}{\sum \exp(h_j^*)} \right)
\] (11)

The network is trained by using standard multi-class negative log-likelihood loss in tandem with log-SoftMax in Equation 11 [14]. The negative log-likelihood loss is expressed as

\[
L = - \sum_{i=1}^{n} [y_i \log \hat{y}_i + (1 - y_i) \log (1 - \hat{y}_i)]
\] (12)

where \( y_i \) is the ground-truth label, and \( \hat{y}_i \) is the output of the network.

7 EXPERIMENTS ON OPEN DATASETS

In this section, we evaluate DP-GCN classification model against ten state-of-the-art baselines on open networks. Through experiments, we also aim to find out (i) whether the T-GCN module can effectively utilize the topology structure information for classification, and (ii) whether the multi-head self-attention contributes to model performance. Lastly, we visualize the node representations learned from dual-path convolution. In Section 8, we conduct a case study on PayPal payment networks.

7.1 Experiment Setting and Results

We first evaluate our model on the task of node classification on five publicly available network datasets and two synthetic datasets, against baselines. All the five open network datasets come with ground-truth labels for validation.

Datasets. We utilize five network datasets that take both connectivity and topology structure into consideration, for node classification. Among the five datasets, \textit{Euro}, \textit{Brazil} and \textit{USA} are well known airline networks [6, 28]. Those networks encode direct flights between different airports. The label to each airport is its level of activity, which expects the topology structure equivalence due to hub-and-spoke structure existing in the datasets. The \textit{BA} dataset is a labeled network for node classification research from [29] and \textit{Cora} is a labeled citation network. The relationship in Cora network focuses more on connectivity instead of topology structure. The two synthetic networks \textit{Acm1} and \textit{Acm2} are generated by [21], and are included in our experiments for comprehensiveness. Data statistics are reported in Table 1.

Baseline Methods. We consider both traditional and state-of-the-art graph-based methods as baselines. As different baselines use different strategies to study the graph, we ensure the baselines in our experiments demonstrate sufficient diversity. We include the following 10 baselines in our evaluation. \textit{Struct2vec} [28] is a widely used node topology structure embedding method. It encodes topology structure explored by random walk. \textit{Graphwave} [6] learns node topology structure representation by diffusion patterns.
| Dataset | #Nodes | #Edges | #Classes | Type of data |
|---------|--------|--------|----------|--------------|
| Euro    | 399    | 5,995  | 3        | Real data    |
| Brazil  | 131    | 1,074  | 3        | Real data    |
| USA     | 1,190  | 13,599 | 3        | Real data    |
| Cora    | 2,708  | 5,429  | 5        | Real data    |
| Acm1    | 1,024  | 22,008 | 6        | Synthetic data|
| Acm2    | 4,096  | 116,012| 12       | Synthetic data|

Table 1: Statistics of four open and two synthetic networks

Role2vec [1] captures node topology structure similarity by analyzing node motif patterns. Riwalk [20] models node topology structure by degree and relative position. Node2vec [26] is an extension of the skip-gram embedding model by capturing breadth-first and depth-first search based neighborhood relationship. GCN [15] is a classic graph convolution neural network. GCNII [3] is the deep graph neural network. GAT [35] adapts self-attention to measure the neighborhood importance when performing node convolution. DEMO-Net models node degree specifically and AM-GCN models node feature with KNN graph to better fusing its features and connectivity. To summarize, Struc2vec, Graphwave, Role2vec and Riwalk are node topology structure based methods, Node2vec is connectivity based method. GCN, GCNII, GAT, DEMO-Net and AM-GCN are graph neural networks.

In our experiments, we split data for evaluation. Specifically, 67% of nodes are used for training and the remaining 33% of nodes for testing (fully supervised). For GCN, GAT, DEMO-Net and DP-GCN, we use 2-layers of convolution and 120 hidden dimensions. For GCNII, we use 64-layers and 120 hidden dimensions. For AM-GCN, we use 1-layer of convolution and perform parameter search for k (from 2 to 7) to build KNN graph. We use the Identity matrix as the initial feature input. For graph neural networks, we follow the standard strategy for training [15, 35]. In specific, we build the entire graph for convolution, but only back-propagate the loss from the training nodes. For traditional methods, we implement the same classifier from DP-GCN and add additional dense layers for classification. We perform parameter search for hyper parameters (e.g., number of hidden dimensions, number of layers) and adapt the optimal parameters for different datasets correspondingly.

Recall that the T-GCN module requires initialization of topology roles. We evaluate four different ways of topology roles initialization. Specifically, we initialize the topology roles in DP-GCN from Struc2vec (SV), Graphwave (GW), Role2vec (RV) and Riwalk (RiW) by k-means clustering (k = 100) respectively. Accordingly, we have four variants of DP-GCN and we denote them by DP-GCN-SV, DP-GCN-GW, DP-GCN-RV and DP-GCN-RiW respectively.

Experimental Results. We report both classification accuracy and Fscore for quantitative evaluation in Table 2.1

All four DP-GCN variants clearly outperform existing state-of-the-art models over a large margin. Specifically, the improvements tend to have better performance. For BA, Cora and Acm1 networks, we observe the opposite. However, DP-GCN achieves consistently best results as our model utilizes both sides of information and uses self-attention to align them properly. Overall, results of all variants DP-GCN-SV, DP-GCN-GW, DP-GCN-RV and DP-GCN-RiW are comparable.

7.2 Ablation Study

We now conduct ablation studies to evaluate the effectiveness of each module in the proposed DP-GCN model. Table 3 reports the results with different settings. As there are different ways to initialize the topology roles, we use the average Fscore of DP-GCN-SV/GW/RV/RiW when applicable. For instance, the first row in Table 3 is the average results of the four variants of DP-GCN.

To evaluate the effectiveness of the component modules, we start with removing each individual module at a time, i.e., C-GCN, T-GCN, and the multi-head attention module. When the multi-head attention module is removed, we combine the outputs from C-GCN and T-GCN unweighted. Alternatively, we also evaluate the setting of replacing the multi-head attention module by a single-head setting, or by using a single convolution layer.

We make four observations from Table 3. First, both T-GCN and C-GCN modules play important roles in DP-GCN as the performance decreases significantly after removing either of two modules. Second, the attention mechanism is also an important component in DP-GCN. It is intuitive that different types of node information take different levels of importance in the prediction task. Combining multiple types of node information without carefully analyzing their importance may not be effective in the classification task, and sometimes may even introduce noise to the model. Third, tuning the number of heads for attention is not trivial. It helps the model to better learn the information alignment towards the task-specific objective from multiple representation subspaces. Finally, we observe that the C-GCN module does not contribute much to the Euro, Brazil, and USA datasets. This is due to the characteristics of the datasets. In those three datasets, the topology structure information is more important for the corresponding classification objective. However, on BA, Cora, Acm1, and Acm2 datasets, the C-GCN module contributes more than the T-GCN module. Overall, this study suggests that all modules in the architecture are crucial for achieving the best classification results. If we combine the information modeled by both C-GCN and T-GCN modules with a proper attention mechanism, we are able to achieve the best classification results over all datasets.

7.3 Node Representation Visualization

We now visualize the node representations learned by the dual-path convolution process in DP-GCN. This visualization may provide insights into the inner workings of our model. In specific, we visualize the node representations generated by DP-GCN on the well known Zachary’s Karate Club network [43].
Table 2: Evaluation on four open networks and two synthetic datasets; best results are in boldface and second best highlighted.

| Dataset     | Accuracy | Fscore |
|-------------|----------|--------|
|             | Euro     | Brazil | USA   | BA    | Cora  | Acm1   | Acm2   | Euro   | Brazil  | USA   | BA    | Cora  | Acm1   | Acm2   |
| Struc2vec   | 0.588    | 0.630  | 0.534 | 0.230 | 0.602 | 0.263  | 0.112  | 0.550  | 0.407   | 0.508 | 0.224 | 0.568 | 0.263  | 0.093  |
| Graphwave   | 0.438    | 0.444  | 0.576 | 0.168 | 0.210 | 0.136  |       | 0.152  | 0.312   | 0.573 | 0.161 | 0.151 | 0.137  |
| Role2vec    | 0.365    | 0.519  | 0.534 | 0.217 | 0.624 | 0.470  | 0.530  | 0.206  | 0.344   | 0.492 | 0.218 | 0.603 | 0.454  | 0.490  |
| RiWalk      | 0.575    |       | 0.815 | 0.601 | 0.199 | 0.421  | 0.115  | 0.482  | 0.538   | 0.365 | 0.185 | 0.363 | 0.187  | 0.100  |
| Node2vec    | 0.263    | 0.333  | 0.479 | 0.236 | 0.671 | 0.612  | 0.517  | 0.221  | 0.175   | 0.477 | 0.232 | 0.646 | 0.567  | 0.506  |
| GCN         | 0.400    | 0.481  | 0.462 | 0.224 | 0.891 | 0.675  | 0.500  | 0.368  | 0.300   | 0.454 | 0.073 | 0.878 | 0.667  | 0.466  |
| GCNI1       | 0.522    | 0.500  | 0.577 | 0.237 | 0.899 | 0.746  | 0.414  | 0.460  | 0.494   | 0.523 | 0.230 | 0.881 | 0.732  | 0.330  |
| GAT         | 0.424    | 0.568  | 0.504 | 0.224 | 0.886 | 0.509  | 0.393  | 0.371  | 0.440   | 0.477 | 0.142 | 0.870 | 0.474  | 0.344  |
| DEMO-Net    | 0.525    | 0.593  | 0.588 | 0.273 | 0.710 | 0.502  | 0.545  | 0.545  | 0.529   | 0.557 | 0.250 | 0.664 | 0.492  | 0.549  |
| AM-GCN      | 0.375    | 0.482  | 0.332 | 0.217 | 0.884 | 0.761  | 0.418  | 0.187  | 0.163   | 0.228 | 0.114 | 0.864 | 0.732  | 0.345  |
| DP-GCN-SV   | 0.613    | 0.741  | 0.576 | 0.286 | 0.884 | 0.766  | 0.571  | 0.582  | 0.662   | 0.562 | 0.253 | 0.871 | 0.749  | 0.527  |
| DP-GCN-GW   | 0.625    | 0.704  | 0.605 | 0.267 | 0.893 | 0.746  |       | 0.586  | 0.488   | 0.589 | 0.203 | 0.875 | 0.718  |
| DP-GCN-RV   | 0.513    | 0.667  | 0.555 | 0.280 | 0.887 | 0.737  | 0.573  | 0.478  | 0.632   | 0.540 | 0.235 | 0.873 | 0.716  | 0.552  |
| DP-GCN-RW   | 0.650    | 0.704  | 0.613 | 0.255 | 0.886 | 0.756  | 0.560  | 0.598  | 0.594   | 0.603 | 0.173 | 0.873 | 0.739  | 0.499  |

We follow [28] to construct a mirrored network of the same club network. Then we obtain a unified network which consists of the original and the mirrored networks (i.e., two copies of the same Karate Club network). In this unified network, each node is assigned a unique node id; so corresponding nodes in the original and mirrored networks will be assigned different node ids, as shown in Figure 4. We build DP-GCN with randomly initialized weights on this unified network. To enable role convolution, we make a simple assumption that the same corresponding nodes in the original and the mirrored networks have the same topology role (i.e., modelled by the same topology role). Without any training, we input the adjacency matrix of the unified Zachary’s karate club and the Identity matrix $X = I$ (i.e., without any additional features) to the model. Then DP-GCN performs both connectivity and topology role convolution on all nodes in the unified network. The final node representations are obtained through a random weighted attention which combines both sides of information (as we do not have a task-specific loss function here to guide the attention). As a comparison, we generate node representations from the standard GCN through the same process. In this analysis, we set the dimension of node representations to be 10 and feed the learned node representations to a log-SoftMax function. The corresponding dimension obtained from the log-SoftMax is the node cluster label. Figure 4 shows nodes clusters in different colors.

We observe that GCN tends to group nodes based on connectivity relationships, thus nodes within the same local community have very close representations. The convolution from GCN does not...

Table 3: Ablation Analysis

| Module change | Accuracy | Fscore |
|---------------|----------|--------|
| Dataset       | Euro     | Brazil | USA   | BA    | Cora  | Acm1   | Acm2   | Euro   | Brazil  | USA   | BA    | Cora  | Acm1   | Acm2   |
| DP-GCN (Avg)  | 0.561    | 0.594  | 0.573 | 0.216 | 0.873 | 0.730  | 0.526  |        |        |
| Remove C-GCN  | 0.515    | 0.499  | 0.512 | 0.118 | 0.349 | 0.151  | 0.222  |        |        |
| Remove T-GCN  | 0.397    | 0.466  | 0.485 | 0.138 | 0.869 | 0.668  | 0.435  |        |        |
| Remove Att    | 0.490    | 0.458  | 0.554 | 0.160 | 0.852 | 0.698  | 0.495  |        |        |
| l-Head Att    | 0.540    | 0.603  | 0.568 | 0.133 | 0.868 | 0.697  | 0.523  |        |        |
| l-Layer GCN   | 0.507    | 0.575  | 0.555 | 0.155 | 0.854 | 0.730  | 0.457  |        |        |

Figure 4: Clusters of node representations from GCN, DP-GCN without training (Best viewed in color)
8 CASE STUDY

As a case study, we evaluate the proposed DP-GCN model on payment networks from PayPal. The task is to detect risky sellers, i.e., the sellers who result in revenue loss due to various reasons like fraud, bankrupt, and bad suppliers.

8.1 PayPal Datasets

We conduct experiments on three network datasets with ground-truth labels from PayPal: SG-Jan, SG-Apr, and SG-Jul. Each network contains a subset of one-week payment activities of anonymous sellers (seller ids are encrypted) in January, April, and July 2020 respectively. The network includes all sellers located in Singapore region while the buyers are from all over the world. The sizes of the three datasets are comparable and each contains about 0.5 million nodes and 0.5 million edges. The risky seller labels are provided by an internal department for various risky types. The data is imbalanced, due to the fact that only a very small number of sellers are risky sellers.

Due to memory constraints, we manage to compare our model DP-GCN with Role2vec, Riwalk, GCN, and GCNII. Again, Role2vec and Riwalk are node topology structure based methods. GCN and GCNII are the graph convolution networks. We use 2-layers and 120 hidden dimensions for DP-GCN and GCN. We use 64-layers and 120 hidden dimensions for GCNII. We perform parameter search on the over-and under-sampling ratio for the imbalanced labels and use the same ratio for all methods. For DP-GCN, we use topology roles generated from Role2vec and Riwalk embeddings, and use 1000-dimensional binary vectors as node initial features. For GCN and GCNII, we evaluate the performance by using either 1000-dimensional binary vectors or topology structure embeddings as node initial features, and then select the best one to represent its final performance. As the labels are much imbalanced, we use macro-averaged precision, recall, and Fscore for performance evaluation, instead of accuracy. Macro-averaged metrics are known to be more appropriate for imbalanced data as it avoids dominance of majority classes [5, 45].

8.2 Result Analysis

Results of risky seller detection are reported in Table 4. DP-GCN is consistently at the top across all the three datasets by Fscore. Specifically, the improvements are about 4% to 6%. Role2vec and Riwalk are the best performing models among baselines, which suggests that modelling topology structure is important for identifying real risky sellers. We also observe that simply feeding topology structure embeddings as the initial features to graph convolution network leads to very marginal improvement. GCNII tends to provide conservative prediction but catches less risky seller. This result further emphasizes the importance of using dual-path convolution to model the different types of information in the graph neural network. Our results on enterprise-scaled data suggests that combining both connectivity and topology structure information for risky seller detection is a promising direction on payment network. Our model is well aligned with the real-world enterprise needs.

9 CONCLUSION

In this paper, we propose an interactive dual-path graph convolution network to explore both node connectivity and topology structure information, for risky seller detection and also node classification in general. We propose a generalizable role-based convolution to explicitly model the topology structure similarity of nodes with latent topology roles. With the notion of topology role, the computational cost of convolution over node’s local topology structure similarity becomes much lower. Furthermore, we align node connectivity and topology structure information with multi-head graph self-attentions to reduce the gaps between the two types of information. Via visualization and qualitative analysis, we show that our proposed model achieves highly competitive results on both open datasets and PayPal’s risky seller detection task. Our case study also shows that the proposed model can be applied to large networks with million nodes.

REFERENCES

[1] Nesreen K Ahmed, Ryan A Rossi, John Boaz Lee, Theodore L Willke, Rong Zhou, Xiangnan Kong, and Hoda Eldardiry. 2019. role2vec: Role-based network embeddings. In Proc. DLG KDD.
[2] PV Bindu, Rahul Mishra, and P Santhi Thilagam. 2018. Discovering spammer communities in twitter. Journal of Intelligent Information Systems 51, 3 (2018), 503–527.
[3] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. 2020. Simple and deep graph convolutional networks. In International Conference on Machine Learning. PMLR, 1725–1733.
Winnie Cheng, Chris Greaves, and Martin Warren. 2006. From n-gram to skipgram to conecgram. International journal of corpus linguistics 11, 4 (2006), 411–433.

Teerawit Chokkrirong and Porravat Vateekul. 2016. Improve accuracy of defect severity categorization using semi-supervised approach on imbalanced data sets. In Proceedings of the International MultiConference of Engineers and Computer Scientists, Vol. 1.

Claire Donnat, Marinka Zitnik, David Hallac, and Jure Leskovec. 2018. Learning structural node embeddings via diffusion wavelets. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 1320–1329.

Hongyang Gao, Zhengyang Wang, and Shuaiwang Ji. 2018. Large-scale learnable graph convolutional networks. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 1416–1424.

Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, 855–864.

Will Hamilton, Zhifan Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In Advances in neural information processing systems. 1024–1034.

Keith Henderson, Brian Gallagher, Tina Eliassi-Rad, Hanghang Tong, Sugato Basu, Leman Akoglu, Danai Koutra, Christos Faloutsos, and Lei Li. 2012. Rok: structural role extraction & mining in large graphs. In Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, 1231–1239.

Qinghao Huang, Jielong Wei, Yi Cai, Changmeng Zheng, Junying Chen, Ho-fung Leung, and Qiang Li. 2020. Aligned Dual Channel Graph Convolutional Network for Visual Question Answering. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 7166–7176.

Ling Jian, Jundong Li, and Huan Liu. 2018. Toward online node classification on streaming networks. Data Mining and Knowledge Discovery 32, 1 (2018), 231–257.

Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. 2014. Catch2Sync: catching synchronized behavior in large directed graphs. In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, 941–950.

Sekitoshi Kanai, Yasuhito Fujiwara, Yuki Yamanaka, and Shuichi Adachi. 2018. Sigforsmax: Reanalysis of the softmax bottleneck. In Advances in Neural Information Processing Systems. 286–296.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016).

Wataru Kudo, Mao Nishiguchi, and Fujio Toriumi. 2020. GCNEXT: graph convolutional networks for information retrieval with expanded balance theory for fraudulent user detection. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 1341–1350.

LeLv, Jianbo Cheng, Nanbo Peng, Min Fan, Dongbin Zhao, and Jianhong Zhang. 2019. DP-GCN: Node Classification Based on Both Connectivity and Topology Structure Convolutions for Risky Seller Detection. In 2019 IEEE Conference on Computational Intelligence for Financial Computation and Applications (TOMM), 1024–1034.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Maosong Sun. 2020. Open Graph Benchmark. http://networkrepository.com.

Sekitoshi Kanai, Yasuhito Fujiwara, Yuki Yamanaka, and Shuichi Adachi. 2018. Sigforsmax: Reanalysis of the softmax bottleneck. In Advances in neural information processing systems. 941–950.

Will Hamilton, Zhifan Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In Advances in neural information processing systems. 1024–1034.

Mary McGlohon, Stephen Bay, Markus G Anderle, David M Steier, and Christos Faloutsos. 2016. node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, 855–864.

Jian Tang, Meng Qu, Mingwang Wang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In Proceedings of the 24th international conference on world wide web. 1067–1077.

Ashish Goel, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Advances in neural information processing systems. 5998–6008.

Qiuyu Zhu, Zikuang He, Tao Zhang, and Wennan Cui. 2020. Improving Classification Performance of Softmax Loss Function Based on Scalable Batch Normalization. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 406–415.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Maosong Sun. 2020. Open Graph Benchmark. http://networkrepository.com.

Yiyuan Tian, Yanwei Fu, Yizhou Sun, and Hang Li. 2017. Graph convolutional networks with fast localized message passing. In Advances in neural information processing systems. 4616–4627.

Sarah M Amini, Hiroshi Kageyama, and Ryusuke Hasegawa. 2019. Large-scale Adversarial Graph Convolutional Networks for Network Anomaly Detection. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management, 1265–1274.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013).

Huy Hoang Park and Jennifer Neville. 2019. Exploiting Interaction Links for Node Clustering with Deep Graph Neural Networks. In IJCAI. 3224–3230.