Imbalanced Node Classification Beyond Homophilic Assumption

Jie Liu¹, Mengting He¹, Guangtao Wang², Quoc Viet Hung Nguyen³, Xuequn Shang⁴† and Hongzhi Yin⁴†
¹Northwestern Polytechnical University
²Bytedance Inc
³Griffith University
⁴The University of Queensland
{jayliu,hmt468}@mail.nwpu.edu.cn, xjtuwgt@gmail.com, henry.nguyen@griffith.edu.au, shang@nwpu.edu.cn, h.yin1@uq.edu.au

Abstract

Imbalanced node classification widely exists in real-world networks where graph neural networks (GNNs) are usually highly inclined to majority classes and suffer from severe performance degradation on classifying minority class nodes. Various imbalanced node classification methods have been proposed recently which construct synthetic nodes and edges w.r.t. minority classes to balance the label and topology distribution. However, they are all based on the homophilic assumption that nodes of the same label tend to connect despite the wide existence of heterophilic edges in real-world graphs. Thus, they uniformly aggregate features from both homophilic and heterophilic neighbors and rely on feature similarity to generate synthetic edges, which cannot be applied to imbalanced graphs in high heterophily. To address this problem, we propose a novel GraphSANN for imbalanced node classification on both homophilic and heterophilic graphs. Firstly, we propose a unified feature mixer to generate synthetic nodes with both homophilic and heterophilic interpolation in a unified way. Next, by randomly sampling edges between synthetic nodes and existing nodes as candidate edges, we design an adaptive subgraph extractor to adaptively extract the contextual subgraphs of candidate edges with flexible ranges. Finally, we develop a multi-filter subgraph encoder which constructs different filter channels to discriminatively aggregate neighbors’ information along the homophilic and heterophilic edges. Extensive experiments on eight datasets demonstrate the superiority of our model for imbalanced node classification on both homophilic and heterophilic graphs.

1 Introduction

Graph Neural Networks (GNNs) successfully extend deep learning approaches to graph data and have exhibited powerful learning ability on node classification task [Kipf and Welling, 2017; Veličković et al., 2018; Liu et al., 2023]. Despite their effectiveness, most existing GNNs neglect the widely existing class-imbalance problem in real-world networks, where certain class(es) have significantly fewer node samples for training than other classes [Sun et al., 2021b]. For example, in online transaction networks, the majority of nodes are normal customers while only a small number are fraudsters; in molecular networks, there are much more low-mass atoms than high-mass atoms. Due to the dominating role of majority class nodes in the training set, classical GNNs are often highly inclined to majority classes, leading to severe performance degradation for minority node classification.

To address the class-imbalance problem in the node classification task, many methods have been proposed recently. They mainly relieve the imbalance problem by generating synthetic nodes for minority classes and further constructing synthetic edges between the generated nodes and the original nodes. For example, GraphSMOTE [Zhao et al., 2021] generates synthetic nodes by interpolating nodes of the same minority class through SMOTE [Chawla et al., 2002] and generates their linkages through a pre-trained edge generator. ImGAGN [Qu et al., 2021] also synthesizes minority class nodes and connects them to real minority class nodes through a generative adversarial network. GraphENS [Park
We design a novel imbalanced node classification model and code the acquired balanced graph for node classification. To construct a balanced graph, we apply a multi-filter GNN as node classifier to encode the subgraphs consisting of two-hop homogeneous and heterogeneous neighbors discriminatively. Finally, after generating imbalanced node classification methods suffer from three severe problems when applied to networks with a large portion of heterogeneous connections. P1: Most existing methods generate synthetic nodes based on homophilic interpolation, which restricts interpolated node pairs to be the same minority class. This causes synthetic nodes to lack diversity when real minority class nodes are very limited. P2: Existing models mainly resort to node feature similarity for synthetic edge construction. This strategy works well for homophilic edges which connect nodes with similar features but fail in constructing heterogeneous edges and would thus introduce structure bias (i.e., heterophilic/homophilic edge distribution drift). P3: Existing methods conduct uniform message passing for both homophilic and heterophilic edges when aggregating features, and consequently result in much noisy information from dissimilar neighbors derived from heterogeneous edges to be aggregated into the target nodes. This would seriously degrade the quality of node embeddings and hurt the following node classification task.

In light of this, we propose a novel Subgraph-aware Adaptive Graph Neural Network (GraphSANN) for imbalanced node classification on both homophilic and heterophilic graphs. GraphSANN consists of three major components, i.e., unified feature mixer, adaptive subgraph extractor, and multi-filter subgraph encoder. Specifically, to tackle P1, GraphSANN first applies a unified feature mixer to carry out both homophilic and heterophilic interpolation in a unified way. Next, to tackle P2, instead of generating edges based on feature similarity, we propose an adaptive subgraph extractor to extract the surrounding subgraphs of candidate synthetic edges with flexible ranges. In this way, distant but similar nodes can be absorbed into the subgraph whose general structural information will be encoded to predict the existence of the edge. To tackle P3 and encode the subgraphs consisting of both homophilic and heterophilic connections, we design a multi-filter subgraph encoder to aggregate messages only from similar nodes instead of dissimilar ones by fusing the output messages of three distinct filters. Finally, after generating synthetic nodes/edges and attaching them to the original graph, we apply a multi-filter GNN as node classifier to encode the acquired balanced graph for node classification.

The major contributions of this work are stated as follows:

- To the best of our knowledge, this paper is the first work to tackle the imbalanced node classification problem beyond the homophilic assumption.
- We design a novel imbalanced node classification model GraphSANN which is able to build balanced graph by generating synthetic nodes and constructing both homophilic and heterophilic synthetic edges between generated and original nodes, and aggregates the information from homophilic and heterophilic neighbors discriminatively.
- Extensive experiments on eight benchmark datasets show that GraphSANN acquires superior performance on both imbalanced homophilic and heterophilic graphs.

## 2 Related Work

### 2.1 Heterophilic Graph Neural Networks

Since most existing GNNs follow the homophily assumption and thus face significant performance degradation on heterophilic graphs, heterophily-based GNNs have been proposed, which can be roughly categorized into two groups [Zheng et al., 2022]: (1) Neighbor extension methods which aim to expand local neighborhood to absorb features from distant but informative nodes. For example, MixHop [Abu-El-Haija et al., 2019] aggregates messages from multi-hop neighbors respectively and mixes them together through concatenation. UGCN [Jin et al., 2021] further restricts nodes from two-hop neighbors to have at least two different paths to the ego node. (2) Adaptive message aggregation methods which design adaptive aggregation operations to learn discriminative information from homophilic and heterophilic linkages. For example, FAGCN [Bo et al., 2021] adopts a self-gating attention mechanism to uniformly learn low-frequency and high-frequency signals from neighbors. ACM [Luan et al., 2022] further designs a linear combination of low-pass and high-pass filters to adaptively learn information from different filter channels.

### 2.2 Imbalanced Node Classification

Generally, imbalanced node classification methods can be divided into two groups, generic and network-specific methods. Generic ones directly combine general class-imbalance approaches (e.g., oversampling, re-weight, SMOTE [Chawla et al., 2002], etc.) with GNNs to graph data. For example, Oversampling [Buda et al., 2018] replicates existing node embeddings learned from GNNs to produce more minority node representations: Re-weight [Yuan and Ma, 2012] assigns larger penalty weights to minority nodes when computing training loss. Network-specific methods usually take account of the sophisticated topology of graphs to generate synthetic nodes and further determine the connections between the generated nodes and original nodes [Chen et al., 2021; Xia et al., 2021]. Among them, DR-GCN [Shi et al., 2020] utilizes class-conditional adversarial training to enhance the separation of different classes. GraphSMOTE [Zhao et al., 2021] interpolates nodes from minority classes as synthetic nodes and generates linkages through a pre-trained edge generator. ImGAGN [Qu et al., 2021] synthesizes minority nodes and connects them to original nodes through a generative adversarial network. GraphENS [Park et al., 2022] demonstrates the overfitting problem of neighbor memorization and proposes to generate the whole ego-networks for synthetic nodes based on information from all classes.

However, these models are all based on the homophily assumption and thus suffer from performance degradation when applied to networks with strong heterophily.
3 Notations and Problem Definition

Definition 1. Graph Homophily and Heterophily. Given a graph $G = \{V, E\}$, where $V$ represents the node set, and for each edge $e = (v, \mu), v, \mu \in V$, if $v$ and $\mu$ have the same class label, the edge $e$ is homophilic. Otherwise, $e$ is heterophilic. Most graphs have both homophilic and heterophilic edges at the same time. We define the node homophily $H_{node}$ and edge homophily $H_{edge}$ to quantitatively measure the homophily degree of a graph as follows.

\[
H_{node} = \frac{1}{|V|} \sum_{v \in V} |\{\mu \in N(v) : y_v = y_\mu\}| / |N(v)|, \tag{1a}
\]

\[
H_{edge} = \frac{|\{(v, \mu) \in E : y_v = y_\mu\}|}{|E|}. \tag{1b}
\]

Based on the definitions of $H_{node}$, $H_{edge} \in [0, 1]$, the node heterophily and edge heterophily can be defined as $1 - H_{node}$ and $1 - H_{edge}$, respectively. Graphs with strong homophily have higher $H_{node}$ and $H_{edge}$, and vice versa.

Problem Definition. Let $G = \{V, E, X\}$ denote an attribute graph, where $X \in \mathbb{R}^{|V| \times d}$ is node feature matrix whose $i$-th row represents a $d$-dimensional feature vector of the $i$-th node. For node classification, each node is also associated with a one-hot node label $Y_v \in \mathbb{R}^C$ where $C$ is the number of node classes. If the class size distribution is imbalanced, we name it as an imbalanced node classification problem. The imbalance ratio is defined as $im\text{-ratio} = \min(|C_U|) / \max(|C_U|) < 1$, where $|C_U|$ denotes the node set with class label $c \in \{1, 2, \ldots, C\}$. Then, we give the formal definition of imbalanced node classification as follows.

Given an imbalanced graph $G = \{V, E, X\}$ composed of both homophilic and heterophilic edges, our goal is to learn a node classifier $f : f(V, E, X) \rightarrow Y$ that can well classify both majority and minority classes and can be well generalized on graphs with either low or high heterophily.

4 Methodology

In this section, we introduce our novel GraphSANN for imbalanced node classification. As illustrated in Figure 2, GraphSANN consists of three core components, (1) Unified Feature Mixer which carries out both homophilic and heterophilic interpolation to generate synthetic nodes (Subsection 4.1); (2) Adaptive Subgraph Extractor which adaptively extracts subgraphs around candidate synthetic edges (Subsection 4.2) and (3) Multi-filter based subgraph encoder which encodes the subgraphs extracted from component (2) into edge score using multiple passes of filters (Subsection 4.3). Please refer to Algorithm 1 in the Appendix for the forward propagation procedure of GraphSANN. We elaborate on each component in the following subsections.

4.1 Unified Feature Mixer

Most existing class-imbalance models apply homophilic interpolation on existing node pairs $<v_s, v_t>$ to generate synthetic nodes, which restricts $v_s$ and $v_t$ to be the nearest neighbors of the same minority class [Zhao et al., 2021; Qu et al., 2021]. The generated nodes under this strategy suffer from feature diversity problem especially when original minority nodes are very limited [Park et al., 2022]. To address this problem, we propose a unified feature mixer that conducts both homophilic and heterophilic interpolation in a unified way, which consists of two steps: (1) unified node pair sampling and (2) integrated gradient based feature mixup.

Unified Node Pair Sampling

Before conducting feature mixup to generate synthetic minority nodes, we first sample node pairs $<v_s, v_t>$ from the existing node set for interpolation. Here $v_s$ is sampled from minority classes while $v_t$ is from the entire classes, and thus it could have either the same or different class compared to $v_s$, formed as $S_{pair} = \{<v_s, v_t> \mid v_s \in V_{\text{minor}}, v_t \in V\}$. Let $C = \{1, 2, \ldots, C\}$ and $C_M \subset C$ be the entire class set and minority class set, the sampling distributions for $v_s$ and $v_t$ are denoted as $p_s(u \mid C_M)$ and $p_t(u \mid C)$, respectively. To obtain a uniformly sampling $v_t$ from all the classes, we define $p_s$ and $p_t$ as follows:

\[
p_s(u \mid C_M) = \frac{1}{|V_M|}, \quad m \in C_M, \tag{2a}
\]

\[
p_t(u \mid C) = \frac{\log(|V_U| + 1)}{(|V_U| + 1) \sum_{c \in C} \log(|V_U| + 1)}. \tag{2b}
\]

It is straightforward to get that $p_t$ reaches the peak when $|V_U| = 2$ and gets 0 when $|V_U| = 0$ or $|V_U| = \infty$. In this way, even if the sizes of majority classes are significantly larger than minority classes, nodes from the entire classes have roughly equal chances to be sampled as $v_t$. Here we use over-sampling scale $\zeta$ to control the amount of sampled node pairs.

Integrated Gradient based Feature Mixup

Afterward, we interpolate the raw features of each node pair $<v_s, v_t>$ to generate synthetic nodes. Since $v_t$ is sampled beyond the same minority class as $v_s$, we only preserve generic node attributes which are irrelevant to class prediction to avoid introducing distracting information. As introduced by [Sundararajan et al., 2017], integrating gradient can effectively evaluate the contributions of input features to the model prediction. Compared to directly using gradients to evaluate feature importance [Park et al., 2022], integrated gradient addresses the saturation and thresholding problems [Shrikumar et al., 2017] and can acquire more reliable feature importance. Specifically, the integrated gradient $IG_i(x)$ along the $i$-th dimension of input node feature $x \in \mathbb{R}^d$ is calculated as follows:

\[
IG_i(x) = x_i \int_{t=0}^{1} \frac{\partial \mathcal{L}_{cls}(t x, y)}{\partial x_i} dt, \tag{3}
\]

where $y$ represents the vector of true class labels and $\mathcal{L}_{cls}$ denotes the node classification loss. Then, we compute the distance $\psi_{st}$ between $v_s$ and $v_t$ by:

\[
\psi_{st} = \|W_p x_s - W_p x_t\|_2, \tag{4}
\]

where $W_p \in \mathbb{R}^{d \times d'}$ represents the projection matrix, and $x_s$ and $x_t$ are raw attributes of $v_s$ and $v_t$, respectively. We define $\hat{\psi}_{st} = \frac{1}{1 + \psi_{st}} \in [0, 1]$ as the similarity between $v_s$ and
Finally, we construct the masking vector \( M_t \in \mathbb{R}^d \) as follows:

\[
M_t = 1_{\mathbb{R}^d}(\kappa \psi_{st} \cdot I_t - D_t),
\]

where \( I_t \in \mathbb{R}^d \) is an all-ones vector, \( \kappa \) is a hyper-parameter, \( D_t = [I_G(x_t), \ldots, I_G(x_t)] \) is the feature importance vector for \( x_t \), and \( 1_{\mathbb{R}^d}(\cdot) \) is an indicator function which returns 1 when the input is positive otherwise 0. Thus, the mixed feature \( x_{\text{syn}} \) of synthetic nodes is formulated as follows:

\[
x_{\text{syn}} = (1 - M_t) \odot x_t + M_t \odot x_t.
\]

### 4.2 Adaptive Subgraph Extractor

After generating synthetic nodes \( v_{\text{syn}} \in \mathcal{V}_{\text{syn}} \) for the minority classes, we need to determine their connections to the original graph. The existence of an edge \( e = (v_{\text{syn}}, u) \) is highly related to the structural information embedded in its surrounding subgraph \( \mathcal{G}_e = \{V_e, E_e\} \) regardless of its homophily or heterophily [Zhang and Chen, 2018]. Here, we first randomly sample connections between synthetic and original nodes as candidate edges, then we adaptively extract the subgraphs of candidate edges and finally encode their structure information to predict edge existence. Specifically, given the node pair set \( S_{\text{pair}} \), and synthetic node set \( \mathcal{V}_{\text{syn}} \), the candidate synthetic edge set \( \mathcal{E}_{\text{syn}} \) is constructed as follows:

\[
\mathcal{E}_{\text{syn}} = \{(v, u) | v \in \mathcal{V}_{\text{syn}}, u \in \mathcal{V}_{\text{orig}}\},
\]

\[
\mathcal{V}_{\text{orig}} = \{N_1(v_{\text{syn}}) \cup N_1(v_t) \}_{\xi}, v_{\text{syn}}, v_t \in S_{\text{pair}},
\]

where \( N_1(v) \) returns 1-hop neighbors of \( v \) and \( v_t \) itself, \( \lfloor \cdot \rceil_\xi \) is random sampling with sampling ratio \( \xi \). After obtaining \( \mathcal{E}_{\text{syn}} \), we extract the enclosing subgraph \( \mathcal{G}_e \) for each \( e \in \mathcal{E}_{\text{syn}} \). Considering that nodes with high structural and semantic similarities might be distant from each other in heterogeneous graphs, instead of fixing the subgraph to \( h \)-hop neighbors, we adaptively adjust the range of a subgraph based on a relevance score function \( f_{rel}(\cdot) \). For any \( h \)-hop neighbor \( k \in N_h(v) \cup N_h(u) \), \( f_{rel}(k) \) is calculated as follows:

\[
f_{rel}(k) = \delta(k, v) + \delta(k, u) + \min[\delta(k, v), \delta(k, u)] + \alpha_{k,i,T} W_k \alpha_{l,j} \cdot \left( \frac{\alpha_{k,i} T \alpha_{l,j} W_k}{\alpha_{k,i} T \alpha_{l,j} W_k + \alpha_{k,i} T \alpha_{l,j} W_k} \right),
\]

\[
\alpha_{k,i} = \frac{1}{|N_1(k)|} \sum_i \alpha_{k,i}. \tag{10}
\]

Where \( \delta(k, v) \) is the length of shortest path from \( k \) to \( v \), \( \alpha_{k,i} \in \mathbb{R}^{d_1 \times 1} \) represents the multi-pass weight vector between node \( k \) and \( i \) (further illustrated in Subsection 4.3), \( \alpha_{k,i} \) is the mean value of all the coefficient vectors between \( k \) and its 1-hop neighbors, which reflects the homophily status of node \( k \). \( W_k \in \mathbb{R}^{d_1 \times d_1} \) is a weight matrix. Then we select top \( M \) nodes from \( f_{rel}(k) \) along with central nodes \( u \) and \( v \) to construct the enclosing subgraph. The \( M \) is calculated based on subgraph density:

\[
M = \lceil \sqrt{\frac{1}{1 + \frac{2 |E_e|}{|V_e| (|V_e| - 1)}}} \rceil.
\]

### 4.3 Multi-filter Subgraph Encoder

In this section, we propose a novel subgraph encoder to embed the subgraph surrounding a candidate edge \( e \in \mathcal{E}_{\text{syn}} \) into a vector, and then predict whether the edge \( e \) should be generated or not based on that. Let \( \mathcal{G}_e \) be the extracted enclosing subgraph of the candidate edge \( e \). Considering the widely existing heterophilic connections in \( \mathcal{G}_e \), we design a multi-filter subgraph encoder that can discriminatively aggregate information from homophilic and heterophilic neighbors. Specifically, let \( h_{l-1}^{(l-1)} \in \mathbb{R}^{d_{l-2} \times 1} \) denote the \( (l-1) \)-th layer feature of node \( u \in \mathcal{G}_e \), \( h_u^{(0)} = [x_u \mid z_u] \) where \( z_u \) is one-hot labeling feature acquired by DRNL [Zhang and Chen, 2018]. We now compute the weight coefficients \( \alpha_{l-1, u(k)}^{(l)}, \alpha_{l, u(k)}^{(l)}, \alpha_{l, u(k)}^{(l)} \) which reflect the importance of
different frequencies of signals as follows:

\[
\begin{align*}
\alpha_{L,(u,k)}^{(l)} &= \sigma \left( g_L^T \left[ W_L^{(l)} h_u^{(l-1)} \parallel W_L^{(l)} h_k^{(l-1)} \right] \right), \\
\beta_{H,(u,k)}^{(l)} &= \sigma \left( g_H^T \left[ -W_H^{(l)} h_u^{(l-1)} \right] \right), \\
\gamma_{I,(u,k)}^{(l)} &= \sigma \left( g_I^T \left[ W_I^{(l)} h_u^{(l-1)} \right] \right).
\end{align*}
\]

Where \( W_L^{(l)}, W_H^{(l)}, W_I^{(l)} \in \mathbb{R}^{d_i \times d_{i-1}} \) are the weight matrices that project \( h_u^{(l-1)} \) into low-frequency, high-frequency and identity messages, respectively. \( g_L^{(l)}, g_H^{(l)}, g_I^{(l)} \in \mathbb{R}^{d_i \times 1} \) are the convolutional vectors, \( \sigma(\cdot) \) is the Sigmoid function. Then we compute weight vector \( \alpha_{(u,k)}^{(l)} \) by normalizing the importance weights of different frequencies:

\[
\begin{align*}
\alpha_{(u,k)}^{(l)} &= \frac{\alpha_{L,(u,k)}^{(l)}}{\sum_{i \in \{L,H,I\}} \alpha_{(u,k)}^{(l)}}, \\
\tilde{\alpha}_{i,(u,k)}^{(l)} &= \frac{\exp(\alpha_{i,(u,k)}^{(l)})}{\sum_{i \in \{L,H,I\}} \exp(\alpha_{i,(u,k)}^{(l)})}.
\end{align*}
\]

Next, we aggregate multi-frequency messages of neighbor nodes \( k \) with weight vector \( \alpha_{(u,k)}^{(l)} \) to compute the central node embedding \( h_u^{(l)} \):

\[
\begin{align*}
h_u^{(l)} &= \omega h_u^{(l-1)} + \sum_{k \in \mathcal{N}_i(u)} \alpha_{(u,k)}^{(l)} h_k^{(l-1)}, \\
H_k^{(l-1)} &= \text{ReLU} \left( [W_L^{(l)} h_k^{(l-1)}, W_H^{(l)} h_k^{(l-1)}, W_I^{(l)} h_k^{(l-1)}]^T \right).
\end{align*}
\]

Where \( \omega \) is a hyper-parameter, \( h_u^{(l)} \) denotes the aggregated node embedding of node \( u \in \mathcal{G} \), after stacking \( L \)-layer encoders. To acquire structural information from different orders of neighbors, we concatenate node embeddings from different layers and utilize a mean readout to compute the existence probability \( p_c \) of subgraph \( \mathcal{G}_c \):

\[
\begin{align*}
h_u &= h_1^{(l)} \parallel h_2^{(l)} \parallel \cdots \parallel h_u^{(l)}, \\
p_c &= \frac{1}{|\mathcal{V}_c|} \sum_{u \in \mathcal{V}_c} W_\text{pool} h_u.
\end{align*}
\]

Where \( W_\text{pool} \in \mathbb{R}^{1 \times (d_1 + \cdots + d_L)} \) projects the latent embeddings into a scalar \( p_c \) which reflects the existence probability of synthetic edge \( e \). We remove the edges with low \( p_c \) from \( \mathcal{E}_\text{syn} \) based on threshold \( \eta \) and attach the rest of synthetic edges to the original graph to construct the adjacency matrix \( \hat{A} \) after over-sampling:

\[
\hat{A}(v_{\text{syn}}, u) = \begin{cases} 
1, & \text{if } p_c > \eta \\
0, & \text{otherwise}.
\end{cases}
\]

### 4.4 Optimization Objective

In this section, we introduce the optimization objective of our proposed GraphSANN for imbalanced node classification, which consists of two optimization tasks: 1) adjacency matrix reconstruction and 2) node classification.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Nodes</th>
<th>Edges</th>
<th>Features</th>
<th>Classes</th>
<th>( \mathcal{H}_{\text{edge}} )</th>
<th>( \mathcal{H}_{\text{node}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cora</td>
<td>2,708</td>
<td>5,429</td>
<td>1,433</td>
<td>7</td>
<td>0.8100</td>
<td>0.8252</td>
</tr>
<tr>
<td>Pubmed</td>
<td>19,717</td>
<td>44,338</td>
<td>500</td>
<td>3</td>
<td>0.9024</td>
<td>0.7924</td>
</tr>
<tr>
<td>Citeseer</td>
<td>2,277</td>
<td>4,732</td>
<td>3,703</td>
<td>6</td>
<td>0.7362</td>
<td>0.7175</td>
</tr>
<tr>
<td>Chameleon</td>
<td>5,201</td>
<td>36,101</td>
<td>2,325</td>
<td>5</td>
<td>0.2795</td>
<td>0.2470</td>
</tr>
<tr>
<td>Squirrel</td>
<td>5,201</td>
<td>217,073</td>
<td>2,089</td>
<td>5</td>
<td>0.2416</td>
<td>0.2156</td>
</tr>
<tr>
<td>Film</td>
<td>7,600</td>
<td>33,544</td>
<td>931</td>
<td>5</td>
<td>0.2200</td>
<td>0.2400</td>
</tr>
<tr>
<td>Amazon-CP</td>
<td>13,381</td>
<td>245,778</td>
<td>767</td>
<td>10</td>
<td>0.7721</td>
<td>0.7853</td>
</tr>
<tr>
<td>Amazon-PH</td>
<td>7,487</td>
<td>110,043</td>
<td>745</td>
<td>8</td>
<td>0.8727</td>
<td>0.8365</td>
</tr>
</tbody>
</table>

### 5 Experiment

In this section, we conduct extensive experiments on eight public datasets to evaluate the effectiveness of GraphSANN, which aim to answer five research questions: **RQ1**: How does GraphSANN perform compared to other baselines in imbalanced node classification on both homophilic and heterophilic graphs? **RQ2**: How effective is GraphSANN under different imbalance ratios? **RQ3**: How does each core component of GraphSANN contribute to the performance gain? **RQ4**: How do different hyper-parameter values affect the performance of GraphSANN? **RQ5**: Can GraphSANN learn effective node representation to separate different classes of nodes in the embedding space?
5.1 Experimental Setup

**Datasets.** To thoroughly evaluate the performance of GraphSANN, we conduct experiments on eight benchmark datasets including six artificial imbalanced datasets and two genuine ones. Among the artificial datasets, Cora, Citeseer and Pubmed are three citation networks with high homophily, while Chameleon, Squirrel and Film are three Wikipedia networks with high heterophily. 3, 3, 2, 2, and 2 classes are randomly selected as minority classes for these six datasets by down-sampling. Following [Zhao et al., 2021], all majority classes have 20 nodes while minority classes only have \(20 \times \text{im}_\text{ratio} \) nodes in the training set. For two Amazon product networks whose class distributions are genuinely imbalanced, we use their original class ratios. The detailed statistical information of the six datasets is summarized in Table 1.

**Baselines.** We compare GraphSANN with eight state-of-the-art baselines for imbalanced node classification problem, including two vanilla models: GCN [Kipf and Welling, 2017] and ACM [Luan et al., 2022]; two generic class-imbalance methods: Oversampling and Reweight; and four network-specific methods: DR-GCN [Shi et al., 2020], ImGAGN [Qu et al., 2021], GraphSMOTE [Zhao et al., 2021] and GraphENS [Park et al., 2022]. Please refer to Appendix 7.5 for detailed descriptions of each baseline.

**Evaluation Metrics.** Following existing works [Zhao et al., 2021] in evaluating imbalanced classification, three evaluation metrics are adopted in this paper: Accuracy, AUC-ROC, and Macro-F1, where both AUC-ROC and Macro-F1 are reported by averaging the metrics over each class.

**Parameter settings.** The following hyper-parameters are set for hidden dimensions 64 and 32 for both edge generation and node classification. Adam optimizer with learning rate \(lr = 0.001\) for homophilic graphs and \(0.01\) for heterophilic graphs. Dropout rate \(\gamma = 0.7\). Epochs = 2000 with early stop strategy. Weight decay = 5e−4. Hyper-parameters \(\kappa = 1.05, \omega = 0.3\). Initial hop \(h = 2\), threshold \(\eta = 0.5\) and loss weight \(\lambda = 1e−6\). Sampling ratio of candidate edges \(\xi = 0.3\). Over-sampling scale \(\zeta = 1.0\).

5.2 Imbalanced Node Classification (RQ1)

To answer RQ1, we compare the node classification performance of GraphSANN with other baselines across all eight datasets and report the average performances along with standard deviations of each metric. Table 2 shows the node classification results for six imitative datasets and two genuine datasets. From the table, we can observe that: (1) GraphSANN outperforms all the other baselines by all metrics on all eight datasets. This indicates our proposed model consistently acquires better performance on either homophilic or heterophilic networks. (2) On three heterophilic datasets, most class-imbalance baselines only acquire slightly better, or even worse performance (e.g. DR-GCN on Film and GraphSMOTE on Squirrel) than vanilla models i.e., GCN or ACM. This is because these baselines rely on homophilic assumption and generate synthetic edges based on feature similarity. Thus they perform poorly on heterophilic graphs whose edges link nodes with dissimilar features. GraphSANN, however, thanks to the adaptive subgraph extractor and multi-filter encoder blocks, can discriminatively aggregate similar node features to generate heterophilic edges, and thus achieve significant performance gains over the baselines. (3) On genuine datasets, compared to the most competitive baseline GraphENS, GraphSANN still acquires 4.5%
5.3 Influence of Imbalance Ratio (RQ2)

To answer RQ2, in this subsection, we further compare the performance of GraphSANN with other baselines under different imbalance ratios. The imbalance ratio varies from 0.1 to 0.6. Each experiment is repeated 5 times and the average results are reported in Figure 3. From Figure 3, we can observe that (1) GraphSANN consistently outperforms other baselines across all the imbalance ratios on all the datasets, this demonstrates the generalization and robustness of our model under different imbalanced scenarios. (2) Generally, GraphSANN has more significant performance improvement over other baselines under more extreme imbalance ratios. As imbalance ratio increases, the datasets become more balanced, which offsets the effects brought by node/edge augmentation.

5.4 Ablation Study (RQ3)

To further investigate the contribution of each component of GraphSANN, we perform an ablation study and report the results in Table 3. w/o UFM replaces the unified feature mixer component with simple SMOTE strategy [Chawla et al., 2002]; w/o ASE replaces the adaptive subgraph extractor with fixed 2-hop neighbors of target edge to form subgraphs; and w/o MSE replaces multi-filter subgraph encoder component with a raw GCN. GraphSANN represents the full model with all the components available. From this table, we can observe that: (1) All three components contribute to the performance improvement of GraphSANN; (2) Adaptive subgraph extractor and multi-filter encoder exhibit crucial effects on heterophilic networks in view of sharp performance drops between w/o ASE and GraphSANN and between w/o MSE and GraphSANN on Chameleon.

5.5 Parameter Sensitivity Study (RQ4)

In this subsection, we investigate the impact of two crucial hyper-parameters, i.e., dropout rate $\gamma$ of adaptive classifier and sampling ratio $\xi$ of candidate synthetic edges on model performance. We vary $\gamma$ from 0.1 to 0.9 with step size 0.1 and vary $\xi$ from 0.01 to 0.9. The experiments are conducted on both Cora and Chameleon and test accuracy curves are shown in Figure 4. From Figure 4, we can observe that: (1) As $\gamma$ increases, model performance gradually rises and reaches peak values when $\gamma$ reaches 0.7 on both datasets. Then, as dropout rate keeps increasing, the performance gradually drops. (2) The performance decreases as the sampling ratio is under 0.3 or over 0.5. Our explanation is that the original edge distribution is not sufficiently simulated at low $\xi$, while a high $\xi$ impedes feature aggregations by introducing too many noisy edges, resulting in performance degradation.
5.6 Visualization (RQ5)
In this subsection, we project the latent node embeddings of GraphSANN and the most competitive baseline GraphENS on Cora into two-dimensional space using t-SNE [Van der Maaten and Hinton, 2008] and color the nodes based on their class labels. As shown in Figure 5, we can observe that the minority class representations of GraphENS (e.g. blue and dark cyan dots) are hard to be distinguished and have large mixed areas with majority clusters, while node representations of GraphSANN are clustered tightly together with clear boundaries for both majority and minority classes. This proves the superiority of GraphSANN in terms of embedding quality for separating different classes despite the class-imbalance problem.

6 Conclusion
In this paper, we design a novel GraphSANN for imbalanced node classification on both homophilic and heterophilic graphs. The elaborately designed three components within it can unifiedly interpolate synthetic nodes, adaptively extracts surrounding subgraphs of candidate synthetic edges, and discriminatively encode them to predict the existence of synthetic edges. Extensive experiments on eight benchmark datasets have demonstrated the superiority of GraphSANN.

Acknowledgments
This work is partially supported by The National Key Research and Development Program of China (Grant No. 2020AAA0108504), Australian Research Council Future Fellowship (Grant No. FT210100624), Discovery Project (Grant No. DP190101985), and Discovery Early Career Research Award (Grant No.DE200101465).

References


