SEMI-SUPERVISED NODE CLASSIFICATION WITH IMBALANCED RECEPTIVE FIELD

Anonymous authors
Paper under double-blind review

ABSTRACT

The imbalanced data classification problem has aroused lots of concerns from both academia and industrial since data imbalance is a widespread phenomenon in many real-world scenarios. Although this problem has been well researched from the view of imbalanced class samples, we further argue that graph neural networks (GNNs) expose a unique source of imbalance from the influenced nodes of different classes of labeled nodes, i.e., labeled nodes are imbalanced in terms of the number of nodes they influenced during the influence propagation in GNNs. To tackle this previously unexplored influence-imbalance issue, we connect social influence maximization with the imbalanced node classification problem, and propose balanced influence maximization (BIM). Specifically, BIM greedily assigns the pseudo label to the node which can maximize the number of influenced nodes in GNN training while making the influence of each class more balance. Experiments on four public datasets demonstrate the effectiveness of our method in relieving influence-imbalance issue. For example, when training a GCN with the imbalance ratio of 0.1, BIM significantly outperforms the state-of-the-art baseline ReNode by 8.9%-13.5% in four public datasets in terms of the F1 score.

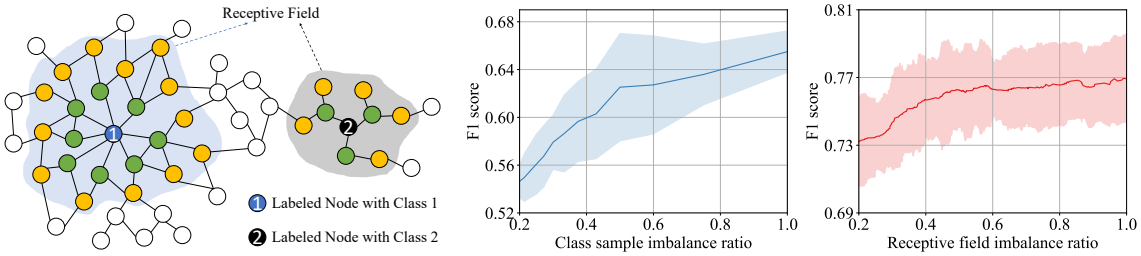
1 INTRODUCTION

Graph Neural Networks (GNNs) have achieved great success in many graph-based applications Zhang et al. (2020); Wu et al. (2020a;b); Guo et al. (2021); Wang et al. (2019). One common graph task is semi-supervised node classification, in which a small ratio of nodes are labeled. Despite the effectiveness and popularity, most GNNs assume a balanced label distribution Wu et al. (2019); Velickovic et al. (2018); Gasteiger et al. (2019), while this assumption is hard to be tenable due to the time-intensive and resource-expensive data annotation process. In many real-world scenarios, node classes are imbalanced in graphs, i.e., some classes have significantly fewer samples than other classes.

Take fake account detection as an example, the majority of users in a social network platform are real users while only a small portion of them are bots Salazar et al. (2018). As for the topic classification for website pages, the materials for some topics are scarce, compared to those on-trend topics. Due to the influence propagation in GNNs, the class for each node is no longer simply determined by its respective features but is also strongly impacted by its connected nodes Wang & Zhang (2007), and the majority class will influence more nodes. The topological interplay makes the imbalance node classification problem more serious. As a result, applying GNNs directly to imbalanced graph data tends to bias to majority classes in semi-supervised node classification.

While the imbalanced data classification problem has been well-studied previously Sun et al. (2009; 2007); Haixiang et al. (2017); Johnson & Khoshgoftaar (2019), most works ignore key characteristics of GNNs and make this problem still under-explored. A few methods are particularly proposed for GNNs to bridge this gap. For example, both DR-GCN Shi et al. (2020) and ReNode Chen et al. (2021) introduce more carefully designed loss functions during the training process. Motivated by the classical SMOTE algorithm Chawla et al. (2002), GraphSMOTE Zhao et al. (2021) re-balances the graph data by generating more pseudo nodes and edges. Although the above methods can alleviate the issue of imbalanced node classes, they ignore the influence propagation of GNNs, leading to sub-optimal performance.

Due to the recursive neighborhood expansion in the propagation process, the learned representation of each labeled node will be influenced by itself and its K-hop neighbors, i.e., nodes in its receptive field (RF) Ma et al. (2021); Zhang et al. (2021a). Consequently, a K-layer GNN can incorporate the unlabeled nodes within the RF of the labeled nodes into the model training, and thus benefits from such a semi-supervised training process Zhang et al. (2021c). Besides the well-studied problem of imbalanced class samples, GNNs especially suffer from the issue of imbalanced RF in



(a) Imbalanced RF (b) The impact of imbalanced class samples and RF

Figure 1: The imbalance RF and its impact on the node classification performance of GCN.

the semi-supervised node classification task, but this issue has not been investigated before. As shown in Figure 1(a), compared with the labeled node with class 2, the labeled node with class 1 lies in a denser region and can influence more unlabeled nodes in a 2-layer GNN, leading to the issue of imbalanced RF.

To measure the impact of the imbalanced class distribution and imbalanced RF on the semi-supervised node classification task of GNNs, we randomly select various labeled sets and train a 2-layer Graph Convolution Networks (GCN) for binary classification task on the Cora dataset Kipf & Welling (2017). Note that the class sample imbalance ratio is the proportion of minority class samples to majority class samples, and the RF imbalance ratio is calculated by the number of nodes influenced by different classes of samples. A smaller imbalance ratio represents a higher imbalance degree. Specifically, we fix the minority class size to 3 and RF to 165, and then vary the majority class size. As expected, the experimental results in the Figure 1(b)(left) show that a small imbalance ratio of the class sample will significantly decrease the F1 score on the test set. On the contrary, in the right figure, we control all the class sizes equally as 8 and train the GCN model on the node sets with various RF imbalance ratios. Surprisingly, we also find a similar phenomenon of performance degradation from the imbalanced RF. In fact, the problem of imbalanced RF will degenerate into the imbalanced class sample if we remove all edges from the graph data, thus it can be seen as a more general type of problem for imbalanced classification.

We name the nodes with large influence magnitude in RF as influenced nodes (See definition 2), and propose a fundamentally new imbalanced node classification method for GNN–*balanced influence maximization* (BIM)–by simultaneously maximizing and balancing the number of influenced nodes. By connecting *social influence maximization* Li et al. (2018); Golovin & Krause (2011); Chen & Krause (2013) with imbalanced node classification, BIM consists of both the *influence maximization* and *influence balance*. For *influence maximization*, BIM greedily assigns the pseudo label to the node which can incorporate more unlabeled nodes into the GNN training (i.e., maximize the influence). Considering the *influence balance*, BIM also requires this node to make the number of influenced nodes in each class more balanced.

In summary, the core contributions of this paper are 1) **New Problem.** To the best of our knowledge, we are the first to consider the influence imbalance issue in imbalanced node classification with GNNs; 2) **New Method.** Motivated by social influence maximization, we propose a new perspective to consider the influence maximization and influence balance in the imbalanced node classification of GNN, and the ablation study validates the effectiveness of these two components. Furthermore, we combine influence maximization and influence balance in a unified BIM framework; 3) **SOTA Performance.** The empirical study demonstrates that BIM significantly outperforms the compared baselines in different imbalanced ratios. For example, BIM outperforms the competitive baseline ReNode by 8.9%-13.5% in terms of F1 score when training a GCN with an imbalance ratio of 0.1.

2 Preliminary

2.1 Problem Formulation

Suppose we have a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}| = n$ nodes and $|\mathcal{E}| = m$ edges, the node adjacency matrix with self loops is denoted as $\tilde{\mathbf{A}} \in \mathbb{R}^{n \times n}$, the node feature matrix is $\mathbf{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2 ..., \boldsymbol{x}_n\}$ in which $\boldsymbol{x}_i \in \mathbb{R}^f$ represents the node attribute vector v_i, and $\mathbf{Y} = \{\boldsymbol{y_1}, \boldsymbol{y_2} ..., \boldsymbol{y_l}\}$ is the one-hot label matrix for c classes. The entire node set $\mathcal{V}$ is partitioned

into training set $\mathcal{V}_{train}$ (including both the labeled set $\mathcal{V}_l$ and unlabeled set $\mathcal{V}_u$), validation set $\mathcal{V}_{val}$ and test set $\mathcal{V}_{test}$. The goal is to predict the labels for nodes in the test set $\mathcal{V}_u$ with the supervision of labeled set $\mathcal{V}_l$.

2.2 Graph Neural Networks

Let $\mathbf{X}^{(k)}$ be the feature matrix of k-th layer, and $\mathbf{W}^{(k)}$ are the model weights of k-th GNN layer. Taking graph convolution network (GCN) (Kipf & Welling, 2017) as an example, each GCN layer can be formulated as:

$$\mathbf{X}^{(k+1)} = \delta\left(\widetilde{\mathbf{D}}^{-\frac{1}{2}}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}^{(k)}\mathbf{W}^{(k)}\right), \tag{1}$$

where $\mathbf{X}^{(0)}$ (equals $\mathbf{X}$) is the original node feature matrix, and $\widetilde{\mathbf{D}}$ is the diagonal node degree matrix used to normalize $\widetilde{\mathbf{A}}$. Due to the influence propagation along edges, each node in GNNs can enhance its learned representation by distant neighbours, and thus boosts the semi-supervised node classification performance. However, the influence propagation also makes the imbalance node classification problem more severe since RF (i.e., the influenced nodes) will be extremely imbalanced.

2.3 Imbalanced Node Classification

Recent solutions for imbalanced node classification in graph can be roughly categorized into re-sampling and re-weighting methods. Re-sampling methods (GraphSMOTE Zhao et al. (2021) and ImGAGN Qu et al. (2021)) re-balance the class size by generating minority class nodes in the training set. Apart from simulating the nodes' attributes, these methods have to learn complicated topological structure distribution at the same time and generate connections for pseudo nodes. In terms of re-weighting methods, they adjust the model training procedure to up-weight the minority class samples. DR-GCN Shi et al. (2020) and RA-GCN Ghorbani et al. (2022) leverage the adversarial training to learn higher weights for under-represented classes. ReNode Chen et al. (2021) studies topology-imbalance problem and adjusts the training weights for labeled nodes based on class boundaries. However, all these works ignore the issue of label imbalanced influence, which is a key property in semi-supervised node classification.

3 Proposed Method

This section presents BIM, the first method that considers both the influence maximization and influence balance of the imbalanced node classification with GNNs. To make more unlabeled nodes influenced and incorporated into the training process, we firstly introduce the influence maximization in Sec. 3.1, and then explain how to ensure the influence balance in Sec. 3.2. Last, we combine these two modules and introduce balanced influence maximization in Sec. 3.3.

3.1 Influence Maximization

Inspired by Zhang et al. (2021b); Wang & Leskovec (2020); Xu et al. (2018), we measure the influence of a node v_i on v_j by how much a change in the input label of v_j affects the *aggregated* label of v_i after k iterations label propagation. Specifically, we define the influence score of node v_i on node v_j as the gradient of $\boldsymbol{y}_j^{(k)}$ with respect to $\boldsymbol{y}_i$, and the final influence score is normalized as:

$$I(v_j, v_i, k) = \frac{\hat{I}(v_j, v_i, k)}{\sum_{v_w \in \mathcal{V}} \hat{I}(v_j, v_w, k)}, \quad \hat{I}(v_j, v_i, k) = \left\| \mathbb{E}[\partial \boldsymbol{y}_j^{(k)} / \partial \boldsymbol{y}_i] \right\|_1. \tag{2}$$

Larger $I(v_j, v_i, k)$ means that v_i has more probability to arrive at v_j after the k-steps random walk. Generally, the source node v_i should contribute more in measuring its influence on another node v_j if its label is reliable. Considering the influence of each class of labeled nodes, we define the node class reliability for different classes.

To enhance the label supervision and incorporate more unlabeled nodes into the GNN training, we consider both the labeled nodes and the reliable soft label predicted by the GNN model itself.

Definition 1 (**Node class reliability score**). *Suppose $\tilde{\boldsymbol{y}}_i$ is the predicted softmax outputs of GNN models on node v_i, the node class reliability score of node v_i on class n is defined as:*

$$r_{i,n} = \begin{cases} y_{i,n} & \text{if } v_i \in \mathcal{V}_l \\ \tilde{y}_{i,n} & \text{if } v_i \in \mathcal{V}_u \end{cases}, \tag{3}$$

where $\boldsymbol{y}_i = [y_{i,1}, y_{i,2}, ..., y_{i,c}]$ is the original one-hot label, and $\tilde{\boldsymbol{y}}_i = [\tilde{y}_{i,1}, \tilde{y}_{i,2}, ..., \tilde{y}_{i,c}]$ is the predicted soft label by the GNN model for c classes.

Larger $r_{i,n}$ means node v_i is more likely to belong to the n-th class. Considering the influence score, the n-th class reliable influence score of node v_i on node v_j after k-step influence propagation is

$$Q_n(v_j, v_i, k) = r_{i,n} I(v_j, v_i, k). \tag{4}$$

From the perspective of label propagation, node v_i will have larger influence on node v_j in the n-th class if (1) node v_i has large probability to have class n and (2) node v_i has more chance to propagate its label information to v_j after k-step influence propagation in a k-layer GNN.

Definition 2 (**Influenced nodes**). *Given a set of labeled seeds $\mathcal{V}'$ (including both the original labeled set $\mathcal{V}_l$ and pseudo labeled set $\mathcal{V}_p$), the influenced node set by class n is defined as:*

$$\sigma_n(\mathcal{V}') = \bigcup_{v_j \in \mathcal{V}, Q_n(v_j, \mathcal{V}', k) > T_i} \{v_j\}, \tag{5}$$

And the full influenced nodes from different classes are

$$\sigma(\mathcal{V}') = \bigcup_{n \in \{1,...,c\}} \{\sigma_n(\mathcal{V}')\}, \tag{6}$$

where $Q_n(v_j, \mathcal{V}', k) = \sum_{v_i \in \mathcal{V}'} Q_n(v_j, v_i, k) \quad s.t.\ \max(\mathbf{r}_i) > T_c$. Here we use the threshold T_c to filter out those unreliable soft labels. Besides, the threshold $T_i = 0$ means we consider an unlabeled node v is influenced as long as it is in RF of any labeled nodes, but the influence may be so weak for the GNN training. So, we assume an unlabeled node is influenced if the reliable influence score is larger than T_i. By setting an appropriate T_i and maximizing $\sigma(\mathcal{V}')$, more unbalanced nodes will get sufficient magnitude of influence from the labeled nodes, and thus improve the training process.

3.2 Influence Balance

To measure the ratio of imbalanced influence, we first get the distribution of the number of nodes influenced by each class. Specifically, we define the imbalance ratio of influence as:

$$B(\mathcal{V}') = \frac{\sum_{n=1}^{c} p_n(\mathcal{V}') log p_n(\mathcal{V}')}{log \frac{1}{c}}, \quad p_n(\mathcal{V}') = \frac{|\sigma_n(\mathcal{V}')|}{\sum_{n=1}^{c} |\sigma_n(\mathcal{V}')|}, \tag{7}$$

where $p_n(\mathcal{V}')$ is the distribution score of the n-th class. $B(\mathcal{V}') \in [0, 1]$, and larger $B(\mathcal{V}')$ means the number of influenced nodes by each class of nodes in $\mathcal{V}'$ is more balanced.

3.3 Balanced Influence Maximization

To balance and maximize all the influenced nodes in the semi-supervised GNN training, we aim to select and annotate a subset of unlabeled nodes with pseudo soft labels and get the new labeled set $\mathcal{V}'$. Since different datasets may have different ratios of class sample imbalance and influence imbalance, we adopt a parameter α to adjust these two terms. Specifically, we propose to find a subset $\mathcal{V}'$ according to the following Eq. 8:

$$\max_{\mathcal{V}'} F(\mathcal{V}') = \frac{|\sigma(\mathcal{V}')|}{|\mathcal{V}|} + \alpha B(\mathcal{V}'). \tag{8}$$

Pseudo labeling. We first train the initial GNN model under the label supervision of the labeled set $\mathcal{V}_l$ (line 1), and then get the corresponding softmax output $\tilde{\mathbf{Y}}^{(0)}$ (line 2). With the average prediction ($\hat{\mathbf{Y}}^{(i)}$) of different iterations of

Algorithm 1: Working pipeline of BIM.

Input: The adjacent matrix $\widetilde{\mathbf{A}}$, degree matrix $\widetilde{\mathbf{D}}$, feature matrix $\mathbf{X}$, labeled set $\mathcal{V}_l$, unlabeled set $\mathcal{V}_u$, one-hot label matrix $\mathbf{Y}$, iteration number H and maximum class size M.

Output: The predicted soft label matrix $\tilde{\mathbf{Y}}$.

1 Train initial model $f^{(0)}$ with $\mathcal{V}_l$;
2 Use $f^{(0)}$ to predict the soft label matrix $\tilde{\mathbf{Y}}^{(0)}$, and set $\hat{\mathbf{Y}}^{(0)} = \tilde{\mathbf{Y}}^{(0)}$;
3 Initialize pseudo labeled set $\mathcal{V}_p = \varnothing$;
4 **for** $i = 0$ *to* $H - 1$ **do**
5 **while** *Minimum class size* < M **do**
6 Select the node $v_j = \arg\max_v F(\mathcal{V}_l \bigcup \mathcal{V}_p \bigcup \{v \mid v \in \mathcal{V}_u\})$ with $\hat{\mathbf{Y}}^{(i)}$ according to Eq. 8;
7 **if** $F(\mathcal{V}_l \bigcup \mathcal{V}_p \bigcup \{v_j\}) \leq F(\mathcal{V}_l \bigcup \mathcal{V}_p)$ **then**
8 **return** The predicted soft label matrix $\tilde{\mathbf{Y}}^{(i)}$.
9 Update $\mathcal{V}_p \leftarrow \mathcal{V}_p \bigcup \{v_j\}$;
10 Remove v_j from $\mathcal{V}_u$;
11 Set one-hot pseudo label $\boldsymbol{y}_j$ and the label reliability score r_j for v_j;
12 Train $f^{(i)}$ with the label supervision in $\mathcal{V}_l \bigcup \mathcal{V}_p$, and get model $f^{(i+1)}$;
13 Use $f^{(i+1)}$ to get the predicted soft label matrix $\tilde{\mathbf{Y}}^{(i+1)}$;
14 **if** $i > 0$ **then**
15 Update $\hat{\mathbf{Y}}^{(i+1)} \leftarrow average(\tilde{\mathbf{Y}}^{(i+1)}, ..., \tilde{\mathbf{Y}}^{(1)})$;
16 **return** The predicted soft label matrix $\tilde{\mathbf{Y}}^{(H)}$.

GNN models, if one class is imbalanced (i.e., the minimum class size is smaller than M), we select one unlabeled node v_j to maximize and balance the number of influenced nodes according to the objective function defined in Eq. 8 (line 6). If labeling v_j with the model prediction cannot increase the objective score, we directly break this algorithm and return the predicted soft label matrix $\tilde{\mathbf{Y}}^{(i)}$ (lines 7-8). Otherwise, we add the selected node to the pseudo labeled set $\mathcal{V}_p$ (line 9), and remove it from the unlabeled set $\mathcal{V}_u$ (line 10). According to $\tilde{\mathbf{Y}}^{(i)}$, the pseudo label $\boldsymbol{y}_j$ of node v_j is the class with the maximum probability score in its soft label, and we define this probability score as the label reliability r_j (line 11). Larger r_j means the GNN model is more confident in its prediction, and the predicted pseudo label is more reliable. We loop the above process until we cannot further maximize the objective score (lines 5-11).

Model training. After the pseudo labeling, we train the GNN model with both the original label and the pseudo label (line 12) and predicted the new softmax outputs (line 13). Considering that nodes with larger label reliability will contribute more to the training process, we adopt the weighted cross-entropy loss as: $\mathcal{L} = -\sum_{v_i \in \mathcal{V}_l \bigcup \mathcal{V}_p} r_i \boldsymbol{y_i} \log \tilde{\boldsymbol{y}}_i$ in the GNN training. Since these pseudo labels can be used to train a more accurate GNN model, and a better model can in turn generate better pseudo labels, we loop this process H times (line 4). Besides, we average the model predictions from all previous loops to further improve the accuracy of pseudo labels (line 15).

4 EXPERIMENTS

We test BIM on five real-world graphs to verify the effectiveness and aim to answer three questions. **Q1**: Compared with other state-of-the-art baselines, can BIM achieve better classification performance? **Q2**: How do each component (e.g., influence maximization, influence balance, iterative optimization and weighted loss) in BIM affect the model performance? **Q3**: Can BIM generalize well to different imbalanced ratios and different GNN models? **Q4**: How to explain the effectiveness of BIM?

4.1 EXPERIMENT SETTINGS

Datasets and Baselines. We evaluate BIM on five real-world networks: Cora, Citeseer, PubMed Kipf & Welling (2017), Ogbn-arxiv Hu et al. (2020) and Amazon-computers Shchur et al. (2018). Following the setting of previous works Zhao et al. (2021), all majority classes of training data have 20 nodes and minority classes include $20 \times imbalance\ ratio$

Table 1: Performance of different compared baselines with GCN as the base model. The best performance is bold

Method	Cora		Citeseer		PubMed		Ogbn-arxiv		Amazon-computers	
	F1 score	AUC-ROC	F1 score	AUC-ROC	F1 score	AUC-ROC	F1 score	AUC-ROC	F1 score	AUC-ROC
GCN	52.9±2.1	87.5±0.9	21.7±0.4	74.7±1.2	51.8±2.5	85.2±5.9	30.1±1.4	80.0±0.6	73.9±1.1	96.0±0.1
ROS	53.0±3.8	85.7±1.3	22.6±0.4	75.1±0.6	61.5±0.7	87.3±0.9	35.8±1.5	82.3±1.7	74.5±1.2	96.0±0.2
SMOTE	53.3±3.2	86.1±2.0	22.4±0.4	75.0±0.8	60.6±0.9	87.1±0.5	34.1±1.6	82.0±1.7	73.5±0.9	95.7±0.8
Reweight	55.9±1.4	88.1±0.5	22.1±0.4	75.3±1.2	59.4±1.2	88.4±0.6	33.7±1.2	80.7±0.7	76.0±0.1	95.9±0.3
DR-GCN	53.3±1.6	83.5±1.1	22.5±2.3	75.0±4.1	57.5±7.2	88.3±1.8	30.2±1.4	82.7±1.3	76.1±1.3	95.8±0.6
GraphSMOTE	58.9±2.0	87.4±0.4	22.2±0.3	75.3±0.1	68.2±0.9	85.4±0.3	39.7±1.6	79.3±1.2	76.7±0.9	96.9±0.4
ReNode	60.3±2.3	88.7±1.4	35.2±4.5	77.1±1.6	68.8±1.3	90.3±0.2	36.7±1.1	81.8±0.8	77.2±0.6	96.6±0.2
BIM w/o IO	63.7±3.1	87.5±1.7	39.8±5.6	78.4±2.3	77.6±1.7	90.8±0.9	45.1±1.9	83.8±1.2	78.1±0.8	96.1±0.2
BIM	**68.9±2.2**	**89.6±1.1**	**48.7±7.1**	**82.0±2.7**	**78.6±0.6**	**91.2±0.4**	**45.6±1.9**	**84.2±1.4**	**78.9±1.3**	**96.7±0.1**

nodes. If not specified otherwise, $imbalance\ ratio$ is set to 0.1 for all datasets and experiments. The other properties of these datasets are summarized in Appendix A.1.

We compare representative and state-of-the-art methods for the class imbalance problem, which include: (1) **Random over-sampling(ROS)**: sample the nodes and their edges in minority classes to re-balance the classes; (2) **SMOTE** Chawla et al. (2002): interpolate a minority sample and its nearest neighbors in the same class. The edge of the synthetic minority sample is set to be the same as the target node; (3) **Reweight** Ren et al. (2018): a cost-sensitive method by increasing the category weight of classification loss function; (4) **DR-GCN** Shi et al. (2020): a representation learning method that enhances the separation of nodes from different classes by conditional adversarial training and distribution alignment; (5) **GraphSMOTE** Zhao et al. (2021): synthesize new minority nodes in graph embedding space and generate edges by training an edge generator; (6) **ReNode** Chen et al. (2021): address the topology-imbalance issue in the graph by re-weighting the influence of labeled nodes based on their relative positions to class boundaries.

Implementations. For a fair comparison, all these methods are tested on the same base GNN model. Besides, we tune or follow the original papers to find the optimal hyper-parameters for each baseline. To eliminate randomness, we repeat each experiment 10 times and report the average test accuracy and standard deviation. The implementation details are shown in Appendix A.2,

Evaluation metrics. The performance of each baseline is evaluated by three classification task criteria: classification accuracy(ACC), AUC-ROC score and F1 score. ACC is the ratio of corrected samples among test samples. AUC-ROC score shows the probability that the corrected class is ranked higher than other classes. F1 score is the harmonic mean of precision and recall for each class. Note that, because the AUC-ROC score and F-measure are the non-weighted averages over each class, they can avoid the majority classes dominating the final performance.

4.2 Experiment results

Imbalanced Classification Performance. To answer **Q1**, we use GCN as the base model and compare BIM with 7 baselines on five graph datasets. The average F1 and AUC results with standard deviation are shown in Table 1, and the ACC results can be found in Appendix A.3. From the tables, we observe that BIM consistently outperforms baselines in all datasets on different evaluation metrics, which validates the effectiveness of BIM. Compared with the naive GCN model, ROS, SMOTE and DR-GCN perform up and downs but ReNode, Reweight, and GraphSMOTE show performance improvement on different metrics. This indicates that cost-sensitive learning and re-sampling method designed for graph data could mitigate the class imbalance problem well. However, when ignoring graph structure information, simply duplicating the minority samples or distribution alignment approaches are counterproductive. Furthermore, compared with the generation model GraphSMOTE and DR-GCN, BIM considers the semi-supervised learning process and avoids introducing noises by referring to original unlabeled nodes and edges in the graph, contributing to better classification performance in imbalanced scenarios.

Ablation Study. BIM aims to maximize the influence and balance each class's influence at the same time. To answer **Q2** and verify the necessity of these two components, BIM is evaluated on the same base model while disabling one component at a time when the imbalance ratio is 0.1. Firstly, we compare BIM with pseudo label random selection strategy(RS) which selects pseudo labels until classes are balanced. Then, we evaluate BIM: (i) without influence maximization when selecting the pseudo labels (called "w/o IM"); (ii) without the influence balance when selecting

Table 2: Ablation study results on the Cora and Citeseer datasets. Note that the base model used here is GCN and the best performance is bold.

Method	Cora			Citeseer		
	F1 score	ACC	AUC-ROC	F1 score	ACC	AUC-ROC
RS	66.5±1.2	71.5±1.3	89.5±0.8	47.3±2.9	52.2±2.2	81.7±1.0
BIM w/o IB	66.4±4.0	72.4±2.8	88.5±1.6	47.8±8.2	52.4±6.9	82.0±3.0
BIM w/o IM	66.9±2.6	72.3±2.1	89.6±1.3	46.7±6.3	51.0±5.8	80.7±2.7
BIM w/o WL	68.0±1.3	73.8±1.2	90.2±0.9	47.0±5.2	51.2±4.7	82.0±1.6
BIM	**68.9±2.2**	**74.0±1.9**	**89.6±1.1**	**48.7±7.1**	**53.2±6.2**	**82.0±2.7**

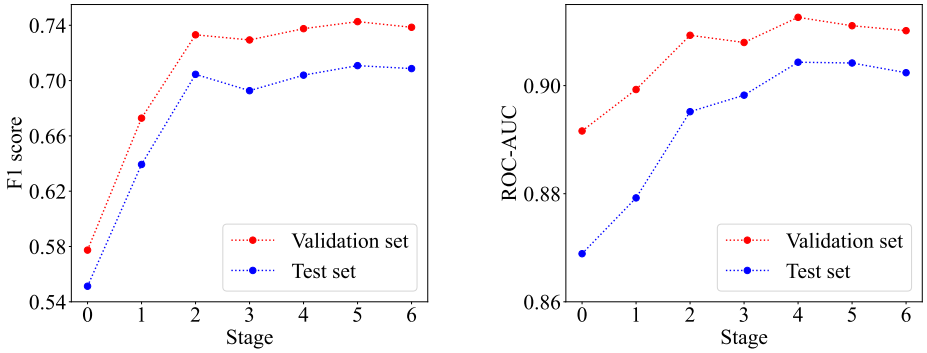


Figure 2: Iterative optimization of BIM on the Cora dataset.

the pseudo labels (called “w/o IB”); (iii) without the weighted loss in the training procedure (called “w/o WL”); (iv) without the iterative optimization for multiple times (called “w/o IO”). Table 2 displays the results of these settings.

First, the classification results will decrease on all three metrics if influence maximization is ignored. The F1 score gap in Citeseer dataset is as large as 2% if influence maximization is unused since adopting influence maximization can incorporate more unlabeled nodes to the GNN training under the semi-supervised training process. Besides, the influence balance component avoids the skewed label influence from different classes, contributing to a 2.5% improvement in the F1 score in Cora. Besides, RS achieves very competitive performance if we adopt BIM without influence maximization or influence balance. This means that both IM and IB are important objectives in our proposed algorithm, and we should not optimize one of them individually.

Due to the imbalanced training data distribution and weak presentation ability of the initial model, the confidence scores will be inaccurate and there will be lots of noise introduced in the pseudo label. To improve the reliability of pseudo labels, we use iterative optimization in the pseudo labeling procedure and weighted loss in the training procedure. The experimental results with the iterative optimization procedure are shown in Figure 2. With the increase of the iteration stage H (defined in Algorithm 1), both F1 score and AUC-ROC score increase gradually and then become stable, indicating the significant effect of iterative optimization on the noise handling ability of BIM. Furthermore, we compare BIM without iterative optimization(called "w/o IO") with the SOTA methods in Table1 and 7. Although iterative optimization helps improve classification performance, BIM w/o IO achieves competitive results on different metrics. As for training procedure, we evaluate BIM without weighted loss (called “w/o WL”) and performance decrease in Table 2 verifies the anti-noise ability of weighted loss strategy.

Varying Imbalance Ratio. To answer **Q3**, we test the performance of different methods with respect to the imbalance ratio on the Cora dataset. All methods are based on GCN and the imbalance ratio varies as 0.1, 0.3, 0.5, 0.7. The experimental results are shown in Table 3. We observe that, among the traditional methods (i.e., ROS, SMOTE, and Reweight), Reweight has the best performance with different imbalance ratios. Our proposed method BIM could successfully generalize to different imbalance ratios and consistently outperforms other compared methods on all three evaluation metrics. Especially for the extreme imbalance scenario, the performance gain of BIM is more significant. For

Table 3: Experiment results of different compared baselines on Cora under various imbalance ratios. The base model used here is GCN.

	Imbalance ratios											
	0.1			0.3			0.5			0.7		
Method	F1 score	ACC	AUC-ROC	F1 score	ACC	AUC-ROC	F1 score	ACC	AUC-ROC	F1 score	ACC	AUC-ROC
GCN	52.9±2.1	62.7±1.4	87.5±0.9	69.8±0.7	73.3±0.6	95.0±0.2	76.3±0.8	77.6±0.7	96.1±0.1	80.1±0.6	81.1±0.5	96.4±0.1
ROS	53.0±3.8	63.4±2.3	85.7±1.3	73.2±1.1	75.5±0.7	94.9±0.2	77.3±1.0	78.4±0.8	95.9±0.1	80.5±0.6	81.4±0.5	96.5±0.1
SMOTE	53.3±3.2	63.5±2.1	86.1±2.0	70.7±0.9	73.7±0.8	94.9±0.2	77.4±0.9	78.6±0.8	95.9±0.2	80.3±0.5	81.3±0.5	96.5±0.1
Reweight	55.9±1.4	65.3±1.0	88.1±0.5	75.4±0.9	77.0±0.9	95.4±0.2	78.2±0.7	79.4±0.6	96.2±0.1	80.8±0.4	81.5±0.6	96.7±0.1
DR-GCN	53.3±1.6	63.4±1.5	83.5±1.1	72.9±1.9	75.4±1.8	95.4±0.4	78.1±1.3	79.3±1.5	95.9±0.1	80.9±0.6	81.5±0.6	96.2±0.2
GraphSMOTE	58.9±2.0	67.8±1.5	87.4±0.4	72.8±1.4	73.9±1.0	91.4±0.8	79.1±1.4	80.0±1.3	96.3±0.3	80.4±0.6	81.2±0.5	96.3±0.2
ReNode	60.3±2.3	67.2±2.5	88.7±1.4	76.3±1.5	78.0±1.3	95.4±0.5	79.1±1.1	80.0±1.2	96.4±0.4	81.2±0.8	81.8±0.8	96.8±0.1
BIM	**68.9±2.2**	**74.0±1.9**	**89.6±1.1**	**79.2±1.4**	**80.2±1.5**	**95.6±0.3**	**81.1±0.9**	**82.0±0.8**	**96.4±0.3**	**82.1±0.7**	**82.8±0.7**	**96.8±0.1**

Table 4: Experiment results on Cora with GraphSAGE as base model.

Methods	F1 score	ACC	AUC-ROC
GraphSAGE	52.5±3.5	61.0±2.6	83.9±1.4
ROS	53.0±2.7	61.4±1.4	84.1±1.2
SMOTE	53.4±2.8	62.6±1.7	81.8±2.2
Reweight	55.1±2.2	62.8±1.2	85.3±1.2
GraphSMOTE	65.8±0.5	70.5±0.4	88.4±0.2
ReNode	58.9±2.4	66.2±1.5	88.9±1.2
BIM	**66.1±1.4**	**70.9±1.6**	**89.0±0.9**

example, BIM exceeds the the best baseline, i.e., ReNode, by a margin of 8.6%, 2.8%, and 0.9% in terms of the F1 score, ACC and AUC-ROC, respectively. Under a high imbalance ratio scenario, generation methods (i.e., GraphSMOTE and DR-GCN) have to generate a large number of pseudo nodes and edges, but the noise will also be introduced to decrease the learning performance. On the contrary, BIM still has a stable performance gain on all three metrics, which indicates the superiority of introducing original unlabeled nodes and edges into the training procedure.

Influence of Base Model. To answer **Q3** and validate the generalization ability of BIM, we set the imbalance ratio is set as 0.1 and test BIM with another widely used GNN model GraphSAGE. For a fair comparison, other model-free methods, including ROS, SMOTE, Reweight, ReNode, and GraphSMOTE, all adopt GraphSAGE as the base model. The experimental results on the Cora dataset are shown in Table 4. Similar to the results in GCN, all these compared baselines can alleviate the imbalance issue and improve the classification performance of GraphSAGE. Moreover, the graph-specific over-sampling method (i.e., GraphSMOTE) is more effective and gets a higher performance gain when using GraphSAGE as base model. Compared with all these methods, BIM consistently reaches the highest classification results on all three evaluation metrics. For example, it outperforms the competitive baseline ReNode by a margin of 7.2% and 4.7% in F1 score and ACC, respectively.

Parameter Sensitivity Analysis. In this part, we analyze the influence of hyper-parameters on BIM. The experiment is conducted on Cora and the base model is selected as GCN. As shown in Figure 3, T_c is quite important which could significantly affect the classification result. For example, T_c should be controlled between 0.7 and 1 to get better performance. Small T_c will introduce lots of noise to the pseudo labels and lead to performance degradation. Following the increase of α, the model performance first increases and then decreases. This verified the effectiveness of both influence maximization and influence balance, and the best results can be obtained with the appropriate value. To sum up, BIM gets stable and high classification performance when α ranges from 1 to 5 and T_c ranges from 0.7 to 1, indicating that the performance of BIM is robust to different parameters. More analyzes of other hyper-parameters are summarized in Appendix A.4

Interpretability. To answer **Q4**, we intuitively explain the effectiveness of the proposed BIM from the perspective of influence maximization and influence balance. Specifically, we set the imbalance ratio to 0.1, and evaluate the distribution of all the influenced nodes and influenced nodes by minority classes for GCN and BIM on Cora. As shown in Figure 4, the number of all influenced nodes can be increased by a large extent if we adopt BIM in GCN, which means more unlabeled nodes can get sufficient influence and contribute to the training process. Besides, we also find that the influenced nodes by minority classes in GCN can all be effectively increased with BIM. This means the influence of

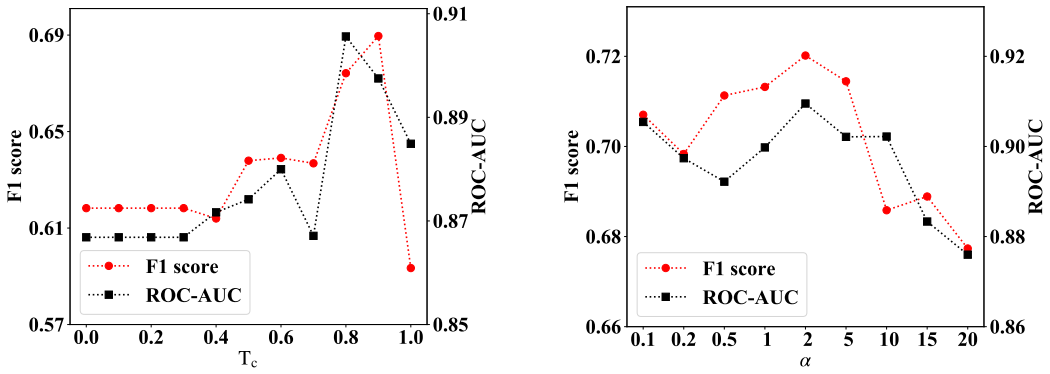


Figure 3: Parameter sensitivity results on the Cora dataset.

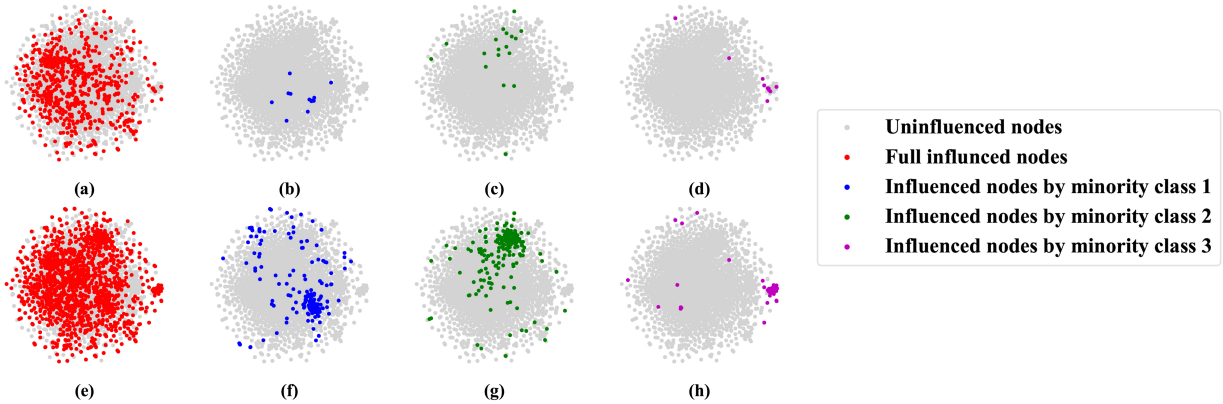


Figure 4: Influenced nodes of the Cora dataset for GCN and BIM. (a) Full Influenced nodes for GCN. (b) Influenced nodes by minority class 1 for GCN. (c) Influenced nodes by minority class 2 for GCN. (d) Influenced nodes by minority class 3 for GCN. (e) Full influenced nodes for BIM. (f) Influenced nodes by minority class 1 for BIM. (g) Influenced nodes by minority class 2 for BIM. (h) Influenced nodes by minority class 3 for BIM.

minority classes is enhanced, and the influence imbalance issue can be effectively alleviated. The balanced and large number of influenced nodes explains why BIM is effective to improve the performance of GCN in imbalanced node classification. Besides the visualization of influenced nodes, we also visualize the node embeddings in Appendix A.6.

5 Conclusion

In this work, we investigate the imbalanced node classification problem with GNNs. Besides the widely known class sample imbalance issue, we further find that the influence-imbalance issue widely exists in GNNs and hinders the learning of semi-supervised node classification, but this issue has not been well studied before. By considering both the influence maximization and influence balance, we propose BIM, a unified framework to maximize and balance the influenced nodes of GNNs. Specifically, BIM greedily assigns the pseudo label to the node which can maximize the number of influenced nodes in GNN training while making the influence of each class more balance. Extensive empirical results have verified the effectiveness of BIM in four public graph datasets. For future work, we are extending BIM to heterogeneous graphs so that it can adapt to more scenarios.

REFERENCES

Sinan Aral and Paramveer S Dhillon. Social influence maximization under empirical influence models. *Nature human behaviour*, 2(6):375–382, 2018.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

Deli Chen, Yankai Lin, Guangxiang Zhao, Xuancheng Ren, Peng Li, Jie Zhou, and Xu Sun. Topology-imbalance learning for semi-supervised node classification. *Advances in Neural Information Processing Systems*, 34, 2021.

Yuxin Chen and Andreas Krause. Near-optimal batch mode active learning and adaptive submodular optimization. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, pp. 160–168, 2013.

Golnoosh Farnad, Behrouz Babaki, and Michel Gendreau. A unifying framework for fairness-aware influence maximization. In *Companion Proceedings of the Web Conference 2020*, pp. 714–722, 2020.

Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019.

Johannes Gasteiger, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *International Conference on Learning Representations (ICLR)*, 2019.

Shay Gershtein, Tova Milo, Brit Youngmann, and Gal Zeevi. Im balanced: influence maximization under balance constraints. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pp. 1919–1922, 2018.

Mahsa Ghorbani, Anees Kazi, Mahdieh Soleymani Baghshah, Hamid R. Rabiee, and Nassir Navab. RA-GCN: Graph convolutional network for disease prediction problems with imbalanced data. *Medical Image Analysis*, 75:102272, 2022. ISSN 1361-8415.

Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *J. Artif. Intell. Res.*, 42:427–486, 2011.

Qipeng Guo, Xipeng Qiu, Xiangyang Xue, and Zheng Zhang. Syntax-guided text generation via graph neural network. *Sci. China Inf. Sci.*, 64(5), 2021.

Guo Haixiang, Li Yijing, Jennifer Shang, Gu Mingyun, Huang Yuanyue, and Gong Bing. Learning from class-imbalanced data: Review of methods and applications. *Expert systems with applications*, 73:220–239, 2017.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

Justin M Johnson and Taghi M Khoshgoftaar. Survey on deep learning with class imbalance. *Journal of Big Data*, 6(1): 1–54, 2019.

David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 137–146, 2003.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

Yuchen Li, Ju Fan, Yanhao Wang, and Kian-Lee Tan. Influence maximization on social graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 30(10):1852–1872, 2018.

Xiaojun Ma, Junshan Wang, Hanyue Chen, and Guojie Song. Improving graph neural networks with structural adaptive receptive fields. In *Proceedings of the Web Conference 2021*, pp. 2438–2447, 2021.

George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions - I. *Math. Program.*, 14(1):265–294, 1978.

Liang Qu, Huaisheng Zhu, Ruiqi Zheng, Yuhui Shi, and Hongzhi Yin. ImGAGN: Imbalanced network embedding via generative adversarial graph networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 1390–1398, 2021.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *International conference on machine learning*, pp. 4334–4343. PMLR, 2018.

Addisson Salazar, Gonzalo Safont, and Luis Vergara. Semi-supervised learning for imbalanced classification of credit card transaction. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7. IEEE, 2018.

Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*, 2018.

Min Shi, Yufei Tang, Xingquan Zhu, David Wilson, and Jianxun Liu. Multi-class imbalanced graph convolutional network learning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20)*, 2020.

Yanmin Sun, Mohamed S Kamel, Andrew KC Wong, and Yang Wang. Cost-sensitive boosting for classification of imbalanced data. *Pattern recognition*, 40(12):3358–3378, 2007.

Yanmin Sun, Andrew KC Wong, and Mohamed S Kamel. Classification of imbalanced data: A review. *International journal of pattern recognition and artificial intelligence*, 23(04):687–719, 2009.

Alan Tsang, Bryan Wilder, Eric Rice, Milind Tambe, and Yair Zick. Group-fairness in influence maximization. In *IJCAI*, 2019.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada*, 2018.

Fei Wang and Changshui Zhang. Label propagation through linear neighborhoods. *IEEE Transactions on Knowledge and Data Engineering*, 20(1):55–67, 2007.

Hongwei Wang and Jure Leskovec. Unifying graph convolutional neural networks and label propagation. *arXiv preprint arXiv:2002.06755*, 2020.

Yishu Wang, Ye Yuan, Yuliang Ma, and Guoren Wang. Time-dependent graphs: Definitions, applications, and algorithms. *Data Sci. Eng.*, 4(4):352–366, 2019.

Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 6861–6871. PMLR, 2019.

Shiwen Wu, Fei Sun, Wentao Zhang, and Bin Cui. Graph neural networks in recommender systems: a survey. *arXiv preprint arXiv:2011.02260*, 2020a.

Shiwen Wu, Yuanxing Zhang, Chengliang Gao, Kaigui Bian, and Bin Cui. Garg: Anonymous recommendation of point-of-interest in mobile networks by graph convolution network. *Data Science and Engineering*, 5(4):433–447, 2020b.

Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 5449–5458, 2018.

Wentao Zhang, Yuezihan Jiang, Yang Li, Zeang Sheng, Yu Shen, Xupeng Miao, Liang Wang, Zhi Yang, and Bin Cui. Rod: reception-aware online distillation for sparse graphs. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 2232–2242, 2021a.

Wentao Zhang, Yexin Wang, Zhenbang You, Meng Cao, Ping Huang, Jiulong Shan, Zhi Yang, and Bin Cui. Rim: Reliable influence-based active learning on graphs. *Advances in Neural Information Processing Systems*, 34, 2021b.

Wentao Zhang, Mingyu Yang, Zeang Sheng, Yang Li, Wen Ouyang, Yangyu Tao, Zhi Yang, and Bin Cui. Node dependent local smoothing for scalable graph learning. *Advances in Neural Information Processing Systems*, 34, 2021c.

Ziwei Zhang, Peng Cui, and Wenwu Zhu. Deep learning on graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

Tianxiang Zhao, Xiang Zhang, and Suhang Wang. Graphsmote: Imbalanced node classification on graphs with graph neural networks. In *Proceedings of the 14th ACM international conference on web search and data mining*, pp. 833–841, 2021.

A APPENDIX

The appendix is organized as follows:

A.1 More details about the datasets.

A.2 More details about the experimental implementation.

A.3 Accuracy comparison of different methods in different datasets.

A.4 Hyperparameter sensitivity analysis.

A.5 The analysis of time complexity.

A.6 The visualization of node embeddings.

A.7 More related works about social influence maximization.

A.1 DATASETS DETAILS

We select five well-known benchmark graph datasets to verify our proposed method, including Cora, Citeseer, PubMed, Ogbn-arxiv and Amazon-computers. In these graphs, papers from different topics are considered as nodes, and edges are citations among the papers. Each paper's topic is regarded as a node class. For the first three datasets, the node attributes are binary word vectors. For ogbn-arxiv, the papers' feature vectors are obtained by averaging the embeddings of words in their title and abstract, and 10 classes with the maximum class size are selected for more flexible control of experimental settings. In Amazon-computers network, nodes represent goods and edges represent that two goods are frequently bought together. Given product reviews as bag-of-words node features, the task is to map goods to their respective product category. The properties of these datasets are summarized in Table 5. We follow the public validation/test split in GCN Kipf & Welling (2017), where 500 nodes for validation and 1000 nodes for the test. Following the setting of previous works Zhao et al. (2021), all majority classes of training data have 20 nodes and minority classes include $20\times imbalance\ ratio$ nodes. If not specified otherwise, $imbalance\ ratio$ is set to 0.1 for all datasets and experiments.

Table 5: Properties of five datasets.

Dataset	# Nodes	# Edges	# Features	# Classes	# Minority Classes
Cora	2708	5429	1433	7	3
Citeseer	3327	4732	3703	6	3
PubMed	19717	44338	500	3	1
Ogbn-arxiv	13515	23801	128	10	5
Amazon-computers	13752	491722	767	10	5

A.2 IMPLEMENTATION DETAILS

The experiments are conducted on an Ubuntu 16.04 system with Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz, 4 NVIDIA GeForce GTX 1080 Ti GPUs and 256 GB DRAM. All the experiments are implemented in Python 3.9 with Pytorch 1.12.1 on CUDA 10.1, and we use the ADAM optimization algorithm to train all the models. The model that performs best on the validation set will be evaluated by test set. The hyper-parameters used in experiments are searched by the grid search method or follow the original papers. The values for T_c, T_i, α and M are searched from {0.6, 0.65, 0.7, 0.75, 0.80, 0.85, 0.90, 0.95}, {1e-5, 5e-5, 1e-4, 5e-4, 1e-3, 5e-3}, {0.1, 0.2, 0.5, 1, 2, 5, 10} and {20, 22, 24, 26, 28, 30} respectively. For the GCN-based model, we set the hidden embedding size as 64, the L2 norm regularization weight decay as 1e-5. The values for learning rate, dropout rate, epoch, T_c, T_i, α and M are listed in Table 6. For the GraphSAGE-based model, we set the hidden embedding size as 64, the dropout rate as 0.3, the L2 norm regularization weight decay as 1e-5, and the learning rate as 0.05. The values for epoch, T_c, T_i, α and M are set as 500, 0.95, 1e-4, 0.2 and 20. Moreover, to eliminate randomness, each method is repeated ten times and the mean test accuracy and standard deviation are reported.

Table 6: Hyper-parameters for tested datasets.

	Cora	Citeseer	PubMed	Ogbn-arxiv	Amazon-computers
Learning rate	0.08	0.08	0.09	0.03	0.08
Dropout	0.2	0.3	0.5	0.5	0.5
#Epochs	300	500	300	1000	300
T_c	0.85	0.65	0.65	0.80	0.60
T_i	5e-5	1e-5	5e-5	1e-4	5e-5
α	10	2	5	2	1
M	30	20	20	20	20

A.3 ACCURACY COMPARISON

Due to page limitations, we list the ACC comparison results in Table 7. We use GCN as the base model and imbalance ratio=0.1. BIM is compared with 7 baselines on five graph datasets and achieves the best performance. Notice that even without iterative optimization(BIM w/o IO), our proposed method shows competitive classification results.

Table 7: Accuracy performance of different compared baselines with GCN as the base model.

	Cora	Citeseer	PubMed	Ogbn-arxiv	Amazon-computers
GCN	62.7±1.4	30.5±0.5	56.6±0.9	37.4±0.8	74.7±1.3
ROS	63.4±2.3	31.0±0.3	63.5±0.5	40.4±0.9	75.2±1.2
SMOTE	63.5±2.1	30.9±0.2	62.7±0.6	38.7±1.6	76.1±1.2
Reweight	65.3±1.0	30.9±0.4	61.9±1.0	39.4±0.6	76.7±0.3
DR-GCN	63.4±1.5	31.2±1.8	60.8±5.0	38.0±1.2	76.3±1.6
GraphSMOTE	67.8±1.5	30.7±0.4	68.8±0.6	41.9±1.1	76.6±1.2
ReNode	67.2±2.5	38.4±3.7	69.6±1.2	41.8±1.1	77.7±0.5
BIM w/o IO	70.2±2.5	45.3±4.8	78.1±1.8	47.0±1.3	79.2±0.6
BIM	**74.0±1.9**	**53.2±6.2**	**79.2±0.6**	**47.1±1.7**	**79.7±0.1**

A.4 PARAMETER SENSITIVITY ANALYSIS

Apart from T_c and α, we also analyze the effect of hyper-parameter T_i and M on BIM. The experiment is conducted on Cora and the base model is GCN. As shown in Figure 5, limiting T_i to a smaller range works best. A larger T_i means a more strict constraint for influenced nodes, leading to an inaccurate measurement of the information diffusion and only a small number of nodes can be influenced by each class. Thus, T_i should be selected smaller than 0.2. The results in

terms of M show that increasing class size could help enhance the classification performance, but also introduce more unreliable labels in model training. M should be controlled in the range of [20, 30] for better performance.

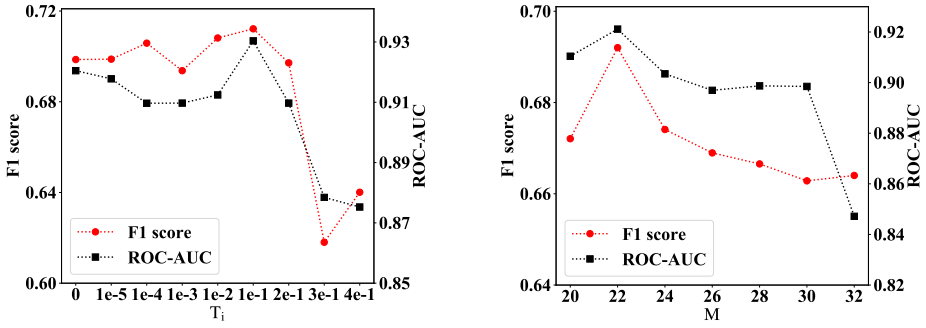


Figure 5: Parameter sensitivity results on Cora.

A.5 TIME COMPLEXITY

We analyze the time complexity of BIM from 2 aspects: (i) overall running time comparison and (ii) pseudo labeling time complexity.

Overall running time comparison. BIM has two procedures: model training and pseudo labeling. As a model-free method, model training time is equal to base-model. We conduct the running time experiments on the Cora dataset and compare BIM with baselines. We evaluate two versions of our proposed method: BIM without iterative optimization(called "BIM w/o IO") and BIM optimized for 6 iterations (called "BIM"). The imbalance ratio is set as 0.1. For GCN, GraphSMOTE, DR-GCN, and ReNode, we use the codes released by their authors. ROS and SMOTE are implemented based on GCN code. For a fair comparison, the training epoch for a single GCN is set as 300. The running time and the corresponding F1 scores are shown in Table 8. Experiments are repeated 3 times and the average time is reported. For traditional ML methods, ROS and SMOTE show similar running times with GCN but only have a slight classification performance increase. GraphSMOTE and DR-GCN are two generative models and require more training epochs(1000 epochs) to reach convergence and higher F1 scores. Even for 300 epochs, these methods are quite time-consuming. Based on PyG Fey & Lenssen (2019), ReNode shows competitive results for both running time and F1 score. BIM w/o IO achieves a great balance between running time and classification performance. As for BIM, iterative optimization not only could help achieve a better F1 score, but also controls the running time lower than generative models.

Table 8: The running time comparison on Cora. The F1 score in parentheses is under epoch=1000.

	Running time(s)	F1 score
GCN	9.03±1.72	52.9±2.1
ROS	9.97±2.08	53.0±3.8
SMOTE	10.73±1.41	53.3±3.2
GraphSMOTE	91.55±3.55	56.7±1.3 (58.9±2.0)
DR-GCN	79.69±5.77	52.1±0.2 (53.3±1.6)
ReNode	6.73±0.30	60.3±2.3
BIM w/o IO	21.73±2.96	63.7±3.1
BIM	70.69±7.95	68.9±2.2

Pseudo labeling time complexity. During the pseudo labeling procedure and labeling for one pseudo node, all unlabelled nodes should be visited, and the time complexity is $\mathcal{O}(|\mathcal{V}_u|)$, which is linear with the unlabeled data size. Assume that if we label m nodes to achieve class balance, the time complexity for pseudo labeling is $\mathcal{O}(m\,|\mathcal{V}_u|)$. However, not all unlabelled nodes should be measured according to the objective function in Eq.8. There are two constraints in the node selection procedure. (i) Unlabelled nodes will be filtered by their class reliability score.

According to Eq.6, only nodes with class reliability score $\max(\mathbf{r}_i) > T_c$ is selected as trustworthy candidates. (ii) Trustworthy candidates will be further filtered by their pseudo labels. According to line 5 in Algorithm 1, the nodes with the majority class pseudo-labels are not included in the selection procedure.

As a result, only a small number of left nodes will be treated as candidates and measured by Eq. 8. In the Table 9, we show the number of nodes at above stages in the node selection procedure on PubMed (The largest one among our tested datasets). Compared with 42 training nodes, there are more than 18000 unlabelled nodes in the dataset. It could be observed that nodes are successfully filtered, and the number of nodes decreases rapidly. During the 7th pseudo node selection, there are only 30 nodes that should be ranked according to the objective function Eq.8.

Table 9: The node selection procedure on PubMed.

Selected node index	#Unlabeled nodes	#Trustworthy nodes	#Nodes with minority class pseudo-label	#Ranked nodes
1	18157	11333	1168	1162
4	18154	11330	1165	298
7	18151	11327	1162	30
10	18148	11324	1159	27
13	18145	11321	1156	23
16	18141	11318	1153	19

Benefiting from the above constraints, the node selection time will not present a significant overhead compared with the time-consuming GNN training. Specifically, the concrete time of line 6 in Algorithm 1 and the full process in different datasets are as Tabel 10. For PubMed, the node selection time is only 9.3% of the total time. As for the smaller dataset Cora, there are 1122 unlabeled nodes, and the pseudo labeling time is negligibly small, which is only 1.5%.

Table 10: The running time of pseudo labeling procedure.

	Total running time(s)	Pseudo labeling time(s)	Pseudo labeling time ratio	Pseudo labeling time per node(s)
PubMed	160.59±0.79	14.97±0.35	9.32%	0.83±0.12
Cora	70.69±7.95	1.10±0.01	1.54%	0.02±0.01

A.6 Interpretability of Embeddings

Besides the visualization of influenced nodes, we set the imbalance ratio to 0.1, and use t-SNE to map the embeddings of Cora generated by GCN and BIM into 2-dimensional space for visualization. As shown in Figure 6, we observe that GCN shows poor representation ability for minority classes (class 5, 6, and 7), and the embedding of different classes of nodes is hard to distinguish. On the contrary, BIM learns more discriminative embeddings, contributing to a better classification result.

A.7 Social Influence Maximization

The goal of social influence maximization is to select a subset of $\mathcal{B}$ influential seed nodes that can maximize the influence propagation in a social network (Aral & Dhillon, 2018). Specifically, given the full node set $\mathcal{V}$ and the seeding budget $\mathcal{B}$, the node selection process can be formulated as:

$$\max_{S} \ |\sigma(S)|, \ \mathbf{s.t.}\ S \subseteq \mathcal{V}, \ |S| = \mathcal{B}, \tag{9}$$

where $\sigma(S)$ is a set of nodes influenced by the seed set S under the influence propagation models, such as linear threshold (LT) and independent cascade (IC) models (Kempe et al., 2003). This node selection process in Eq. 9 is known to be NP-hard, and a greedy algorithm can provide an approximation guarantee of $(1 - \frac{1}{e})$ if $\sigma(S)$ is nondecreasing and submodular with respect to S (Nemhauser et al., 1978).

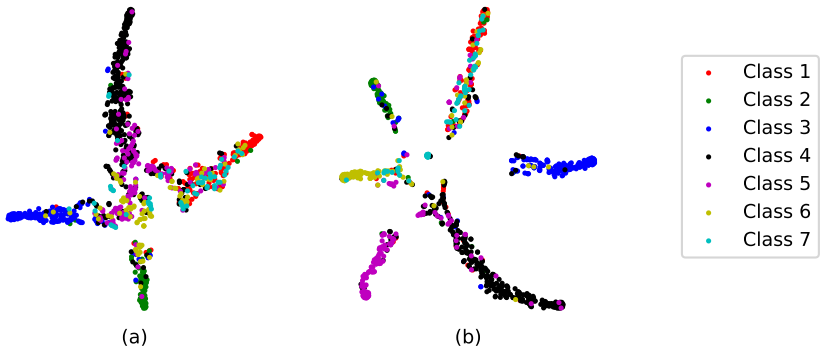


Figure 6: Embedding visualization on Cora. (a) GCN embeddings. (b)BIM embeddings.

Recently, a line of work proposes to maximize the influence while balancing the influence for the different users from subpopulations. For fairness in influence maximization, (Tsang et al., 2019) modifies the classic influence maximization problem with additional fairness provisions based on legal and game theoretic concepts. Besides, (Farnad et al., 2020) studies the trade-offs between enforcing fairness and the loss of total influence, and (Gershtein et al., 2018) introduces the balance constraint as a two-stage problem. Although we also propose to maximize the influence, our work differs from these previous studies in two perspectives. 1) Different motivations of influence balance. Previous works are proposed to address fairness-aware influence maximization problems, while our method is specifically to balance the influence from a different class of labeled nodes. 2) Different utilizations of influence balance. As explained in Section 3.2, we especially defined a new influence balance concept for GNN in the semi-supervised node classification setting.

Following these previous work, we propose to maximize the number of influenced nodes and make these influenced nodes balanced in our BIM framework.