A Graph Data Augmentation Strategy with Entropy Preserving

Xue Liu\textsuperscript{a}, Dan Sun\textsuperscript{b}, Wei Wei\textsuperscript{b,c,d,e,*}

\textsuperscript{a}Beijing System Design Institute of Electro-Mechanic Engineering, Beijing, 100854, China
\textsuperscript{b}School of Mathematical Sciences, Beihang University, Beijing, 100191, China
\textsuperscript{c}Key Laboratory of Mathematics, Informatics and Behavioral Semantics, Ministry of Education, 100191, China
\textsuperscript{d}Beijing Advanced Innovation Center for Big Data and Brain Computing, Beihang University, Beijing, 100191, China
\textsuperscript{e}Peng Cheng Laboratory, Shenzhen, Guangdong, 518066, China

Abstract

The Graph Convolutional Networks (GCNs) proposed by Kipf and Welling are effective models for semi-supervised learning, but facing the obstacle of over-smoothing, which will weaken the representation ability of GCNs. Recently some works are proposed to tackle with above limitation by randomly perturbing graph topology or feature matrix to generate data augmentations as input for training. However, these operations have to pay the price of information structure integrity breaking, and inevitably sacrifice information stochastically from original graph. In this paper, we introduce a novel graph entropy definition as an quantitative index to evaluate feature information diffusion among a graph. Under considerations of preserving graph entropy, we propose an effective strategy to generate perturbed training data using a stochastic mechanism but guaranteeing graph topology integrity and with only a small amount of graph entropy decaying. Extensive experiments have been conducted on real-world datasets and the results verify the effectiveness of our proposed method in improving semi-supervised node classification accuracy compared with a surge of baselines. Beyond that, our proposed approach significantly enhances the robustness and generalization ability of GCNs during the training process.

Keywords: Graph representation, Information theory, Statistical physics

*Corresponding author

Email address: weiw@buaa.edu.cn (Wei Wei)
1. Introduction

Graph, as a ubiquitous data structure, is employed extensively in a wide range of applications, such as bioinformatics [1], citation networks [2] and social networks [3]. All of these domains and many more can be readily modeled as graphs, which contain information about connection between individual units. For instance, citation graph describes interactions among science research papers which are represented as nodes with labels to indicate category, and the citation links between papers are mapped into edges. Information from single node or local dense nodes propagates along edges, and this makes graphs be useful structured knowledge repositories for machine learning tasks like link prediction and node classification.

Graph Convolutional Networks (GCNs) [4, 5, 6, 7, 8] draw support from convolutional operation on graph to aggregate neighbor nodes information from low- to high-order hierarchical structures to get central node representation. In the course of time, GCNs and subsequent variants have emerged as powerful approaches for a variety of tasks like semi-supervised node classification [4, 9], which is also the main focus of this paper.

In order to enable GCNs with more expressivity to wider neighbors, one may stack more layers to the network. But unfortunately, deeper layer network model fails to achieve the expectation partly due to the phenomenon of over-smoothing [10], which is an inherent issue of graph convolutional calculation mechanism. It has been proven that graph convolution operation is a type of Laplacian smoothing, thus representations of nodes in same region converge to same values and tend to be indistinguishable across different classes in embedding space as model goes deeper [11]. An easy but effective way to tackle with over-smoothing is to generate perturbed graph data for training. DropNode [7], DropEdge [12], Dropout [13, 14] and GRAND [15] are four typical tricks but under different focuses. DropNode and DropEdge belong to topological structure based perturbation approaches, while Dropout and GRAND are in the category of graph feature matrix based perturbation methods. These four methods are shown in Figure 1 for brief introduction.

DropNode belongs to the kind of node sampling based methods. In particular, it samples subgraphs for mini-batch training by randomly removing a part of nodes as well as edges connected to the dropped nodes. As a consequence, this method will construct a subgraph $SG$ of original graph $G$, satisfying

$$V(SG) \subseteq V(G), E(SG) \subseteq E(G).$$  \hspace{1cm} (1)
DropEdge acts like a data augmenter by randomly dropping a certain rate of edges from the input graph. Formally, it randomly enforces $|E_p|$ non-zero elements of the adjacent matrix $A$ to be zeros, where $E_p$ is the dropped edges set with drop rate $p$. If we denote the resulting adjacency matrix as $A_{\text{DropEdge}}$, its relation with $A$ becomes

$$A_{\text{DropEdge}} = A - A',$$

(2)

where $A'$ is a sparse matrix expanded by a random subset of size $E_p$ from original edges set $E$. Like DropNode, DropEdge also constructs a subgraph of original graph $G$, but what sets it apart from DropNode is only edges being deleted instead of nodes being deleted, which would inevitably involve removing edges between nodes deleted from the graph. Since DropEdge is edge-oriented instead of node-oriented, it is possible to preserve all node features for the training, exhibiting more flexibility.

Dropout has been widely used for regularizing neural networks. This trick tries to perturb the feature matrix by randomly setting some elements in feature matrix $X$ to be zeros, i.e.,

$$\tilde{X}_{ij} = \epsilon_{ij}^1 - \delta X_{ij},$$

(3)

where $X_{ij}$ is the $j$-th element of the $i$-th row vector $X_i$ in feature matrix $X$, $\epsilon_{ij}$ draws from Bernoulli distribution $\text{Bernoulli}(1 - \delta)$. Dropout makes the input feature matrix $X$ noisy by randomly dropping some elements without affecting graph structures, since it only operates on nodes’ feature vector components.

GRAND is another graph feature matrix based perturbation method. Different from Dropout, GRAND randomly sets some nodes’ features to be zero vectors, i.e.,

$$\tilde{X}_i = \frac{\epsilon_i}{1 - \delta}X_i,$$

(4)

where $\tilde{X}_i$ denotes the $i$-th row vector of feature matrix $X$ and $\epsilon_i$ draws from Bernoulli distribution $\text{Bernoulli}(1 - \delta)$.

These four techniques utilize different perturbation strategies to generate graph data for training but lead to the damage of structure in respect to graph topology or feature information. As for DropNode and DropEdge, they perturb graph by stochastically breaking topological structure and notice that the propagating efficiency of information from one node to another relies on the integrity of graph topology. As for Dropout and GRAND, they operate on graph feature matrix
Figure 1: Illustration of the four graph data augmentation approaches that help alleviating over-smoothing. DropNode randomly drops nodes which are in grey color and generates new graphs for training. DropEdge generates new graphs by randomly deleting edges instead, which are marked in dotted lines. Dropout tries to perturb feature matrix by stochastically setting part of elements to be zeros. GRAND perturbs feature matrix by randomly setting some rows to be zero vectors, and the corresponding nodes are marked in green.
but keep topological structure unchanged. Motifs as higher-order connectivity patterns are crucial
to construct graph topology and control behaviors of a network [16]. In motifs, features from local
dense nodes are gathered into a whole as motif-based information structure to express information,
but this special aggregated information structure could be inevitably destroyed after various
perturbation operations.

As a fundamental concept of statistical physics and information theory, graph entropy serves as
a quantitative measure for graph dynamics [17] and could describe the change of graph topology
as well as graph features. The greater the entropy is, the more information the system reflects
due to higher uncertainty and more randomness. It is obvious that the original graph exhibits
highest entropy, since there exists no damage on both topological structure and feature information.
But the over-smoothing phenomenon because of information coupling weakens feature extraction
ability for machine learning model, and as a consequence, the information contained by the original
graph could not be fully learned, although it has highest entropy. Different graph augmentation
methods tackling with over-smoothing by deleting elements from graph topology or features, result
in the perturbations on features distribution. Usually graph entropy aims at a topological invariant
(e.g., vertex degrees, distances etc.), but the features attached on nodes from local areas still need
attention.

In this paper, we propose a new graph entropy as an index to describe the smoothness of
graph feature information diffusion and yield that the key point to control this kind of smoothness
lies in motif-based information structures. Afterwards, a novel graph data perturbation strategy
for over-smoothing problem on GCNs is provided, whereby graph entropy could be protected as
much as possible. The main steps in this augmentation strategy are as follows. Firstly, we tend
to keep original adjacent matrix \( A \) unchanged instead of dropping any nodes or edges from the
input graph for each training epoch. Afterwards nodes from specific shape motifs and nodes not
in motifs but selected with a certain probability are set as activated status. Only activated nodes’
features could be present in feature matrix while the remaining are represented in zero vectors.
In this study, we focus on the triangle motifs for its ubiquitousness in understanding interaction
of social networks [18]. Extensive experiments have been conducted on several real-world datasets
and the results demonstrate the effectiveness of the proposed method in reducing over-smoothing
and increasing robustness during the whole training process. Our results significantly improve
semi-supervised nodes classification performance compared to state-of-the-art methods.
We summarize the main contributions as follows:

(1) We provide a new graph entropy design to measure the smoothness of graph features distribution and conclude that motif-based information structures determine this graph entropy.

(2) We propose a novel graph data augmentation strategy that protects not only the integrity of topological structure but also the integrity of motifs-based information structures. Our strategy shows advantages in maintaining more graph entropy compared with other methods.

(3) Extensive experiments are conducted on several real-world datasets to show the effectiveness of our proposed method.

(4) Our approach significantly enhances the robustness and generalization ability for GCNs.

2. BACKGROUND AND PROBLEM STATEMENT

Let $G = (V, E)$ denote a graph with nodes set $V$ and edges set $E \subseteq V \times V$, $G$ has a feature matrix $X \in \mathbb{R}^{|V| \times \eta}$ with $i$-th row $X_i$ corresponding to the feature vector of node $v_i$ with length $\eta$, and training labels for all nodes are listed in $Y \in \{0, 1\}^{|V| \times c}$, where $c$ is the classes number. The adjacency matrix $A \in \mathbb{R}^{|V| \times |V|}$ encodes the node-wise connection of the network, whose entry $A_{ij} = 1$ if there exits an edge between node $v_i$ and $v_j$, otherwise $A_{ij} = 0$. We denote $D$ as the degree matrix, $D = \text{diag}(d_1, d_2, ..., d_{|V|})$, such that $d_j = \sum_{i=1}^{|V|} A_{ij}$.

2.1. Semi-Supervised Node Classification

In most cases, semi-supervised node classification task has common applications including classifying documents, videos or proteins and is one of the most basic tasks to verify the effectiveness of GCNs [4]. In this task, labels are only available for a small proportion of nodes, with the goal being to label the full graph relying on this small initial set. Formally, $m$ nodes ($0 < m \ll |V|$) from the graph have observed their labels $Y^L$ and the labels $Y^U$ of the remaining $|V| - m$ nodes are missing. The objective is to learn function $g : (G, X, Y^L) \rightarrow Y^U$ to predict the missing labels $Y^U$ for unlabeled nodes.
2.2. Graph Convolutional Networks

Graph Convolutional Networks (GCNs) generalize neural techniques into graph-structured data. The core operation in GCNs is graph propagation, in which information spreads from each node to its neighborhoods with some deterministic propagation rules. The feed forward propagation in GCNs model consists of \( k \) layers of graph convolution, which is similar to a perception but additionally has a neighborhood aggregation step motivated by spectral convolution. This framework is recursively conducted as

\[
H^{(l)} = \begin{cases} 
\sigma(\hat{A}H^{(l-1)}W^{(l-1)}), & \text{if } l \in [1, ..., k] \\
X, & \text{if } l = 0
\end{cases},
\]

(5)

where \( \hat{A} = \hat{D}^{-1/2}(A+I)\hat{D}^{-1/2} \) is a symmetrically normalized adjacency matrix with self-connections, \( \hat{D} \) is the degree matrix of \( A + I \), \( H^{(l+1)} = \{h_i^{l+1}, ..., h_{|V|}^{l+1}\} \) are the hidden vectors of the \((l+1)\)-th layer with \( h_i^{(l+1)} \) as the hidden features for node \( i \) for input activations \( H^{(l)} \) with \( H^{(0)} = X \), \( \sigma(\cdot) \) is a nonlinear function, and \( W^{(l)} \) is the corresponding weight matrix for layer \( l \).

3. Graph Entropy

The concept of entropy is a fundamental law of statistical physics, and the second law of thermodynamics shows that the entropy of macrosopic system is hard to decrease. Shannon introduced the concept of entropy into information theory as a characteristic measure to reveal information related to a system. As a representation of complex systems, real networks are usually very large, one can characterize graph information quantitatively in terms of macroscopic parameters using methods similar to entropy. Thus graph entropy is widely used to describe and understand the dynamics of graph quantitatively in terms of general topology or features. It was first introduced by Rashevsky [19], then Mowshowitz investigated graph entropy to measure structural information content of graphs [20, 21] and Körner applied a different definition of graph entropy into coding theory [22].

Most graph entropies are derived from basic Shannon’s entropy definition, whose details are as follows: for a discrete system \( X \), \( I(x_i) = -\log p(x_i) \) denotes the self-information of \( x_i \in X \) with occurring probability \( p(x_i) \). The entropy of system \( X \) is defined by \( H(X) \), as

\[
H(X) = -\sum_{i=1}^{|X|} p(x_i) \log p(x_i).
\]

(6)
Usually, information-theoretic measures for graphs are based on a graph invariant and then derive a partitioning \[21\]. Instead of determining partitions of elements based on a given invariant, Dehmer et al. developed an approach which is based on using so called information functional \(f\), mapping sets of vertices to the positive reals \[23\], via

\[
p_i = \frac{f_i}{|X|} \sum_{j=1}^{|X|} f_j.
\]

Then graph entropy measure is obtained by applying function \(6\) and \(7\).

Graph entropy measures the randomness or uncertainty in a statistical perspective. Maximum entropy description retains all of the uncertainty not removed from the original data, and it has been interpreted as the maximally noncommittal with respect to missing information \[24\]. Here we briefly present a novel graph entropy design in terms of features on nodes as well as neighborhoods relations to evaluate feature information diffusion.

### 3.1. Smoothness Index

Here we provide a new graph entropy design as an index to indicate the smoothness of global information distribution. Its idea comes from an application of entropy in image segmentation, in which each pixel of a digital image maps to vertices and one divides them into different communities based on image contrasts. Entropy plays a significant role in quantifying the smoothness of texture in various regions image analysis: high-entropy indicates more smoothness of the texture and less abrupt graphic blocks. As a consequence, more information will be contained in the target image due to the fact that it exhibits more uniform distribution \[25\] [26].

In our new graph entropy design, feature vector of each node is regarded as an individual, and then all of them constitute feature vector space. In particular, in accordance with previous definition, \(X \in \mathbb{R}^{(|V| \times \eta)}\) denotes the feature matrix for graph \(G = (V, E)\), where \(i\)-th row is the feature vector \(X_i\) with length \(\eta\) for node \(v_i, i = 1, \ldots, |V|\). We assign probability values to each individual node of a graph as

\[
p(v_i) = \frac{f(v_i)}{|V|} \sum_{j=1}^{|V|} f(v_j),
\]

where \(f(v_i)\) equals the average inner product between feature vector \(X_i\) and its first order neighbors’
features, i.e.,

\[ f(v_i) = \frac{1}{|N_{v_i}|} \sum_{v_k \in N_{v_i}} \langle X_i, X_k \rangle, \]  

(9)

where \( N_{v_i} \) denotes the first order neighbors set of target node \( v_i \). We apply average feature distance between a node and its neighbors as similarity measurement to express local features distribution. Neighboring nodes with larger inner product indicate more similarity in feature space and exhibit higher smoothness.

Relying on the definition of \( p(v_i) \) for each node, we yield the smoothness index of feature information diffusion on a graph as

\[ I(G) = -\sum_{i=1}^{|V|} p(v_i) \log p(v_i). \]  

(10)

It quantifies the randomness of features distribution by the ensemble average of \(-\log p(v_i)\) over each node \( v_i \), where \( p(v_i) \) represents the contribution of local features to the global scope in the form of probability. We could infer that features tend to scatter evenly around a graph \( G \) if \( I(G) \) reaches a relative high value.

In Figure 2, we take Cora, Citeseer and Pubmed datasets as examples to show how the graph entropy varies after feature information structures being damaged. All the curves achieve highest graph entropy equalling 7.6357, 7.9247 and 9.6724 on three datasets, respectively. After that, these curves appear to show different responses to the drop rate. GRAND leads to most severe decaying on graph entropy and DropEdge gives rise to the slightest loss on graph entropy as drop rate increasing. All curves decrease quickly after they meet 50% drop rate and stop at the lowest
values at 90% drop rate. From the results, it is clear that all these four methods are strongly sensitive to the drop rate, which reflects the damage extent of features on a graph. But these four methods result in such performances for different reasons: DropNode deletes parts of features by dropping nodes, DropEdge cuts off connections between features attached on nodes, while Dropout and GRAND remove some elements or rows of feature matrix directly.

3.2. Motif-Based Information Structure

Motif-based approaches are well used in graph learning tasks [27, 28, 29], and formally, a motif with $s$ nodes and $t$ edges can be denoted as:

$$M^t_s = \{V_M, E_q\}, \quad (11)$$

where $V_M \subseteq V$ represents the set of $s$ nodes and $E_M \subseteq E$ represents the set of $t$ edges. In particular, triangle motifs ($M^3_3$) which we focus on in this paper are significant in social graph learning [18].

![Figure 3: Examples of four triangle motifs: an undirected triangle (a), a triangle in any direction (b), a cycle (c), and a feed-forward loop (d).](image)

As higher-order connectivity structures, motifs characterize the building blocks of graph topological structure and are crucial to the organization of graph [30, 16, 31]. DropNode and DropEdge break the integrity of motifs, so that features on motifs are deleted at the same time. Dropout and GRAND break the integrity of features on motifs without perturbing graph topology. It is worth noting that features attached on nodes from motifs aggregate as an whole to express local dense characteristics among the whole graph due to the fact motifs exhibit higher connectivity. We define this special information structure as motif-based information structure, which plays a significant role in the graph entropy [32], and keeping this information structure integrity could be regarded as a criteria for designing new augmentation strategy that demands entropy preserving.
4. METHODOLOGY

Building on above, in this section, we describe a new data stochastically augmenting strategy for semi-supervised learning on graphs as illustrated in Figure 4 and Figure 5.

![Diagram](image)

**Figure 4:** Here we take triangle motifs $M_3$ to illustrate our proposed strategy. The original graph with feature matrix $X$ serves as an input and its perturbed data are generated through the following steps. In step 1, nodes will be activated depending on whether they are in triangle motifs $M_3$ or not, and the activated nodes are marked in red while others are in green. In step 2, nodes from the remaining part will be activated with a preset probability and turn into red from green. Only features on activated nodes could be revealed in $X$ and others are set as zero vectors.

4.1. Generate Graph Data Augmentations Using Entropy Preserving Strategy

For a graph $G = (V, E)$ with its adjacent matrix $A$ and feature matrix $X$, our method keeps the topological structure of $G$ unchanged and then takes two steps to generate multiple graph data augmentations: (1) activating nodes on motifs, (2) activating the remaining nodes with certain probability.

In the first step, in each training epoch, we set nodes on triangle motifs $M_3$ as activated status and the rest nodes are in dormant status. Afterwards it generates a feature matrix $X_{M_3}$, where only features on activated nodes could be revealed, while the others are set as zero vectors. For the second step, we randomly sample a binary mask $\alpha_i$ by $\text{Bernoulli}(1 - \delta)$ for each node $v_i$ in the remaining part to determine whether $v_i$ would be activated next or not.
To guarantee the perturbed feature vector is in expectation equal to original vector, we multiple coefficient $\frac{1}{1-\delta}$ and get the following as regularized perturbed feature vector

$$\tilde{X}_i = \frac{\alpha_i}{1-\delta} X_i.$$  \hfill (12)

In summary, our proposed method generates perturbed feature matrix $\tilde{X}$ such that

$$\tilde{X}_i = \begin{cases} X_i, & \text{if } v_i \in V_{M_i'} \\ \frac{\alpha_i}{1-\delta} X_i, & \text{otherwise} \end{cases},$$  \hfill (13)

where $X_i$ is the $i$-th row vector of original feature matrix $X$, and binary mask $\alpha_i$ draws from $Bernoulli(1 - \delta)$. The whole computation complexity is $O(n^3)$ and pseudo code is shown in Algorithm [1]

One benefit of this strategy is that neither nodes nor edges are deleted from the original graph in the whole procedure, therefore graph topology keeps unchanged. A further advantage is that this strategy could maintain graph entropy as much as possible since motif-based information structures are protected. In order to understand how our strategy preserves graph entropy, we calculate the
Algorithm 1 Graph Data Augmentation Strategy with Entropy Preserving

**Input:** graph $G$ with its node set $V$ and feature matrix $X$, target motif $M^t$, Bernoulli distribution $Bernoulli(1 - \delta)$.

**Output:** augmentations of graph feature matrix: $\tilde{X}$.

1: for $i = 1; i < |V|; i++$ do
2:     if node $v_i$ in $M^t$ then
3:         $\tilde{X}_i = X_i$
4:     else
5:         $\alpha_i \sim Bernoulli(1 - \delta)$
6:         $\tilde{X}_i = \frac{\alpha_i}{1 - \delta} X_i$
7:     end if
8: end for
9: return $\tilde{X}$

graph entropy of citation graphs in 7 contrasting scenarios as shown in Table 1: original graph without any operation on feature matrix, graph perturbed by DropNode, graph perturbed by DropEdge, graph perturbed by Dropout, graph perturbed by GRAND, graph with only motif-based information structures preserved, and graph perturbed by our entropy preserving strategy. Graph entropy derived from original graph without any operation exhibits highest entropy reaching 7.6358, 7.9247 and 9.6724 on three datasets respectively, since original graph contains all the features. DropNode and GRAND get the lowest entropy compared with others. Here especially note that the sixth scenario graph with only motif-based information structures preserved is corresponding to the first step in our strategy. It maintains almost graph entropy of original graph achieving 7.6028, 7.6996 and 9.6384, which are only a little bit worse than original graph but better than DropNode, DropEdge, Dropout and GRAND except for only one case. We can see our entropy preserving strategy achieves second highest graph entropy and the values are closest to the original graph scenario with only 0.0129, 0.1466 and 0.0151 less on each dataset.

We also explore the statistics of the number of triangle motifs and the number of nodes on triangle motifs, as shown in Table 2. The nodes number on motifs for Cora is 1470, accounting for more than 50% of the total nodes number. While, as for Citeseer and Pubmed, the numbers of nodes on motifs are 1183 and 4835 accounting for about 35.6% and 24.5%, respectively. These
Table 1: Graph entropy calculation results derived from 7 scenarios, which are shown in each column from left to right: original graph, graph perturbed by DropNode (with 50% drop rate), graph perturbed by DropEdge (with 50% drop rate), graph perturbed by Dropout (with 50% drop rate), graph perturbed by GRAND (with 50% drop rate), graph with only triangle motif-based information structures preserved and graph perturbed by our entropy preserving strategy (with activating probability drawing from \textit{Bernoulli}(0.5) distribution).

| Datasets   | Original | DropNode | DropEdge | Dropout | GRAND | Only Motif-based Information Structures Preserved | Our Strategy |
|------------|----------|----------|----------|---------|-------|-----------------------------------------------|--------------|
| Cora       | 7.6358   | 6.7793   | 7.3648   | 7.3063  | 6.6292| 7.6028                                        | 7.6229       |
| Citeseer   | 7.9247   | 6.8784   | 7.5570   | 7.7154  | 6.7884| 7.6996                                        | 7.7781       |
| Pubmed     | 9.6724   | 8.6353   | 9.3230   | 9.3119  | 8.5597| 9.6384                                        | 9.6573       |

Table 2: Number of nodes from original graph, number of triangle motifs and number of nodes on triangle motifs.

| Datasets | Nodes | Motifs | Nodes on Motifs |
|----------|-------|--------|-----------------|
| Cora     | 2708  | 4890   | 1470            |
| Citeseer | 3327  | 4137   | 1183            |
| Pubmed   | 19717 | 37649  | 4835            |

Observations confirm the facts: motif-based information structures determine the graph entropy of the whole graph although they are not in dominant in nodes number, and randomly activating feature operation plays more roles in generating perturbations.

4.2. Aggregate Mixed Order Information

Since for various datasets, the styles that how information from multiple order neighborhoods affects central node are different. Hence we adopt linear combination of different adjacent matrix powers with adaptive weights to target dataset, i.e. $\mathbf{X} = \tilde{\mathbf{A}} \mathbf{X}$, where

$$\tilde{\mathbf{A}} = \sum_{i=0}^{d} g_i(\theta_0, \theta_1, \ldots, \theta_d) \hat{\mathbf{A}}^i$$  \hspace{1cm} (14)
is the weighted average power of symmetrically normalized adjacency matrix \( \hat{A} \) from order 0 to \( d \). The weight \( g_i(\theta_0, \theta_1, ..., \theta_d) \) is defined by softmax function as

\[
g_i(\theta_0, \theta_1, ..., \theta_d) = \frac{\exp(\theta_i)}{\sum_{j=0}^{d} \exp(\theta_j)}.
\] (15)

Note that after enough iterative calculations, parameters \( \theta_0, \theta_1, ..., \theta_d \) will be updated and adjusted to best values until reaching final convergence state.

4.3. Make Prediction

Suppose that we generate \( K \) augmentations as input for each training epoch and derive perturbed feature matrix set \( \{ \tilde{X}^{(k)} \}_{k=1}^{K} \), each one from that set will be fed into networks to get prediction probabilities in the form of binary matrix \( \tilde{Z}^{(k)} \in [0, 1]^{n \times c} \):

\[
\tilde{Z}^{(k)} = \varphi(\tilde{X}^{(k)}, \Omega),
\] (16)

where \( \tilde{X}^{(k)} = A\tilde{X}^{(k)} \) and \( \Omega \) denotes the parameter.

4.4. Loss

Our work follows [33] and [15] to design the loss function, which is a combination of the supervised loss and the graph regularization loss, i.e.,

\[
\mathcal{L} = \mathcal{L}_s + \lambda \mathcal{L}_r.
\] (17)

The first term \( \mathcal{L}_s \) on the right-hand side is the supervised classification loss on labeled node set \( S_L = \{(v_i, Y_i^L)\}_{i=1}^{m} \), where \( v_i \) corresponds to the target node and \( Y_i^L \) is the ground-true label. GCNs model calculates each node \( v_i \) from \( S_L \) and outputs \( \tilde{Z}_i^{(k)} \) as corresponded prediction, then derives \( \mathcal{L}_s \) by the average cross-entropy loss over \( K \) data augmentations:

\[
\mathcal{L}_s = -\frac{1}{K} \sum_{k=1}^{K} \sum_{i=1}^{m} (Y_i^L)^T \log \tilde{Z}_i^{(k)}.\] (18)

The second term denotes the graph regularization loss as \( \mathcal{L}_r \) to guide the prediction of unlabeled node \( v_i \) close to its expected label over \( K \) augmentations by minimizing the distance between \( \tilde{Z}_i^{(k)} \) and \( \bar{Z}_i \):

\[
\mathcal{L}_r = \frac{1}{K} \sum_{k=1}^{K} \sum_{i=1}^{|V|} \| \tilde{Z}_i^{(k)} - \bar{Z}_i \|.
\] (19)
Table 3: Statistics of the benchmark graph datasets. The columns are: name of dataset, number of classes, number of nodes, number of edges, number of features, number of training nodes, number of validation nodes and number of test nodes.

| Datasets | Classes | Nodes | Edges | Features | Training Nodes | Validation Nodes | Test Nodes |
|----------|---------|-------|-------|----------|----------------|------------------|-----------|
| Cora     | 7       | 2708  | 5429  | 1433     | 1208           | 500              | 1000      |
| Citeseer | 6       | 3327  | 4732  | 3703     | 1827           | 500              | 1000      |
| Pubmed   | 3       | 19717 | 44338 | 500      | 18217          | 500              | 1000      |

In equation (19), $\bar{Z}_i'$ represents the possible distribution on the basis of expected label for node $v_i$, which is defined in the form of

$$\bar{Z}_i = \frac{1}{K} \sum_{k=1}^{K} \tilde{Z}_{i}^{(k)}.$$  \hspace{1cm} (20)

Each $j$-th element of $\bar{Z}_i'$ refers to the probability of node $v_i$ on the $j$-th class, and denotes as

$$\bar{Z}_{ij}' = \frac{\bar{Z}_{ij}^{\frac{1}{c}}}{\sum_{t=1}^{c} \bar{Z}_{it}^{\frac{1}{c}}}, 1 \leq j \leq c,$$  \hspace{1cm} (21)

where the categorical distribution is controlled by hyperparameter $\kappa \in [0, 1]$, and $\bar{Z}_i'$ will converge to a one-hot distribution as $\kappa$ getting close to 0 [15].

5. Experiments

With the proposed model above, in this section we follow the experiment setup in [4] to evaluate the effectiveness of our model on semi-supervised node classification task.

5.1. Datasets

We evaluate our model on real-world citation datasets Cora, Citeseer and Pubmed [34] and the data partition [35] details are covered in Table 3. Each citation network provides all the relevant information for each single paper represented as node and citing link between two documents by an edge. Nodes are assigned with labels under their categories. In node semi-supervised classification task, a number of nodes with labels selected by a preset rate from the whole as training set for learning to predict the labels of nodes in test set and validation set.
5.2. BASELINES

To validate the performance of our approach, we compare it with graph embedding methods, graph neural networks and graph augmentation methods on datasets in Table 3. Here are the details of the learning methods that are compared:

- **Graph Embedding Methods:** SemiEbd [36] provides a semi-supervised embedding algorithms which can be easily applied to deep multi-layer learning model. DeepWalk [37], as a representation of factorization-based approach, takes advantage of random walks to get embeddings from graph.

- **Graph Convolution Methods:** GCN [4] proposes convolutional architecture via a local first-order approximation containing both local graph structure and features of nodes. GAT [6] leverages self-attention layers to specify different weights to different nodes in the neighbors. MixHop [5] learns mixing feature representation of neighbors at different orders. SGC [38] improves GCN by reducing excess complexity via removing nonlinearities and collapsing weight matrices between consecutive layers. Graph Markov Neural Network (GMNN) [39] models the joint distribution of labels with a conditional random field, and uses graph neural networks for classification learning. GraphSAGE [40] proposes a general inductive framework to embed the target node by sampling and aggregating features from its local neighborhood. FastGCN [7] interprets graph convolutions as integral transforms of embedding function under probability measures, which are evaluated through Monte Carlo approximation.

- **Graph Augmentation Methods:** DropNode [41] randomly removes a certain number of nodes from the input graph at each training epoch, while DropEdge [12] randomly removes edges instead. Dropout [13, 14] perturbs the feature matrix by stochastically setting elements to be zeros. GRAND [15] utilizes a random propagation strategy to generate new graph feature matrix for training by randomly setting a rate of nodes’ feature vectors as zero vectors.

5.3. Parameter Settings

We follow exactly the same experimental procedure (such as features and data splits) as the standard GCNs settings on semi-supervised graph learning [4, 42, 38]. We randomly select nodes out of triangles motifs under Bernoulli distribution $\text{Bernoulli}(1 - \delta)$, where $\delta = 0.5$. The mixture
Table 4: Semi-supervised node classification accuracy (standard deviation) (%) of our method and baselines on benchmark datasets Cora, Citeseer and Pubmed. Bold font marks the best performance in a column.

| Algorithm   | Cora   | Citeseer | Pubmed  |
|-------------|--------|----------|---------|
| SemiEbd     | 68.0   | 45.3     | 63.0    |
| DeepWalk    | 67.2   | 43.2     | 65.3    |
| GCN         | 81.5   | 70.3     | 79.0    |
| GAT         | 83.0 (0.7) | 72.5 (0.7) | 79.0 (0.3) |
| MixHop      | 81.9 (0.4) | 71.4 (0.8) | 80.8 (0.6) |
| SGC         | 81.0 (0.0) | 71.9 (0.1) | 78.9 (0.0) |
| GMNN        | 83.7   | 72.9     | 81.8    |
| GraphSAGE   | 78.9 (0.8) | 67.4 (0.7) | 77.8 (0.6) |
| FastGCN     | 81.4 (0.5) | 68.8 (0.9) | 77.6 (0.5) |
| DropNode    | 80.7   | 70.8     | 77.5    |
| DropEdge    | 82.8   | 72.3     | 79.6    |
| Dropout     | 81.9   | 70.0     | 78.9    |
| GRAND       | 85.4 (0.4) | 75.4 (0.4) | 82.7 (0.6) |
| **Our**     | **85.6 (0.4)** | **77.5 (0.3)** | **86.6 (0.3)** |
order of aggregated adjacent matrix \( \bar{A} \) is set as \( d = 8 \). For each training epoch, 4 perturbed feature matrixes will be generated as input and the model utilizes stochastic gradient descent algorithm to calculate the loss function for 1000 epoches. The regularization hyper-parameter \( \lambda \) equals 1 and hyper-parameter \( \kappa \) is set as 0.5.

### 5.4. Comparison Results and Discussion

Comparison results of semi-supervised node classification tasks on Cora, Citeseer and Pubmed datasets are reported in Table 4 where the scores of our method are averaged over 10 times and the standard deviations are also reported. Figure 6 shows how the loss curves vary on training and validation sets with the increase of training epoch compared with DropNode, DropEdge, Dropout and GRAND.

As can be seen in Table 4, the results of our proposed model go beyond other baselines on all three datasets. Compared with graph embedding methods, our method improves the classification accuracy upon SemiEbd by 17.6\%, 32.2\%, and 23.6\% on Cora, Citeseer and Pubmed, respectively, and improves 18.4\%, 34.3\%, 21.3\% upon DeepWalk respectively. As for graph convolution methods, our method gains in at least 1.9\%, 4.6\% and 4.8\% on Cora, Citeseer, and Pubmed. As a new data augmentation approach, our method also performs best in this category, reaching 0.2\%, 2.1\% and 3.9\% higher in accuracy on Cora, Citeseer, and Pubmed. This strongly demonstrates the effectiveness of our proposed model on semi-supervised learning task.

In Figure 6, we utilize training and validation losses decreasing performance to show the training characteristic of our model compared to DropNode, DropEdge, Dropout and GRAND on Cora, Citeseer and Pubmed. It appears that both training and validation curves of our proposed model apparently decrease smoothly and then level off at successively inferior values along with the training on all three datasets, while other methods tend to fluctuate on different levels. This obvious superiority of our model over others suggests that our model achieves more stability and robustness during training process. Another novel finding we need to note is that for all three datasets, the validation loss curves of our model are all under the training curves, which suggests our model could achieve higher generalization performance.

### 5.5. Conclusion

In summary, we firstly introduce a new graph entropy design which acts as an index to indicate the smoothness of graph feature information diffusion. Keeping motif-based information structure
integrity is very much key point that determines such graph entropy. We adopt preserving entropy as a criteria and propose a novel graph data augmentation strategy for over-smoothing problem in GCNs. Compared with other graph data augmentation methods, our strategy maintains randomness with only a small amount of graph entropy loss and without breaking graph topological structure. Experimental results show that our model exhibits higher generalization ability and achieves good accuracy on semi-supervised node classification tasks among a series of baselines. Another advantage is that it performs more stable during the whole training process, which enhances robustness.

There are a lot of interesting directions for future work. Research on graph entropy defined by different pairwise distance is still warranted for further study. In addition, using entropy tool to investigate control problems in graph dynamics (e.g., pinning control problem [43]) is another interesting topic we aim to focus on in the future.
Acknowledgements

This work is supported by the Research and Development Program of China (No.2018AAA0101100), the Fundamental Research Funds for the Central Universities, the Beijing Natural Science Foundation (1192012, Z180005) and National Natural Science Foundation of China (No.62050132).

References

[1] J. B. Lee, R. Rossi, X. Kong, Graph classification using structural attention, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery &amp; Data Mining, 2018, pp. 1666–1674.

[2] A. K. McCallum, K. Nigam, J. Rennie, K. Seymore, Automating the construction of internet portals with machine learning, Information Retrieval 3 (2) (2000) 127–163.

[3] L. Backstrom, J. Leskovec, Supervised random walks: predicting and recommending links in social networks, in: the Fourth International Conference on Web Search and Web Data Mining, 2011, p. 635–644.

[4] T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: Proceedings International Conference on Learning Representations, 2017.

[5] S. Abu-El-Haija, B. Perozzi, A. Kapoor, N. Alipourfard, K. Lerman, H. Harutyunyan, G. Ver Steeg, A. Galstyan, Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing, in: international conference on machine learning, PMLR, 2019, pp. 21–29.

[6] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, in: Proceedings International Conference on Learning Representations, 2018.

[7] J. Chen, T. Ma, C. Xiao, Fastgcn: fast learning with graph convolutional networks via importance sampling, in: Proceedings International Conference on Learning Representations, 2018.

[8] R. Li, S. Wang, F. Zhu, J. Huang, Adaptive graph convolutional neural networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 32, 2018.
[9] M. Zhang, Z. Cui, M. Neumann, Y. Chen, An end-to-end deep learning architecture for graph classification, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 32, 2018.

[10] W. Huang, Y. Rong, T. Xu, F. Sun, J. Huang, Tackling over-smoothing for general graph convolutional networks, arXiv e-prints (2020) arXiv–2008.

[11] Q. Li, Z. Han, X.-M. Wu, Deeper insights into graph convolutional networks for semi-supervised learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 32, 2018.

[12] Y. Rong, W. Huang, T. Xu, J. Huang, Dropedge: Towards deep graph convolutional networks on node classification, in: Proceedings International Conference on Learning Representations, 2020.

[13] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, The journal of machine learning research 15 (1) (2014) 1929–1958.

[14] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, R. R. Salakhutdinov, Improving neural networks by preventing co-adaptation of feature detectors, Computer Science 3 (4) (2012) págs. 212–223.

[15] W. Feng, J. Zhang, Y. Dong, Y. Han, H. Luan, Q. Xu, Q. Yang, E. Kharlamov, J. Tang, Graph random neural networks for semi-supervised learning on graphs, Advances in Neural Information Processing Systems 33 (2020).

[16] A. R. Benson, D. F. Gleich, J. Leskovec, Higher-order organization of complex networks, Science 353 (6295) (2016) 163–166.

[17] G. S. Ya, On the concept of entropy for dynamical systems (in russian) dokl, Akad. Nauk SSSR 124 (1959) 768–771.

[18] M. E. Newman, J. Park, Why social networks are different from other types of networks, Physical review E 68 (3) (2003) 036122.

[19] N. Rashevsky, Life, information theory, and topology, The bulletin of mathematical biophysics 17 (3) (1955) 229–235.
A. Mowshowitz, M. Dehmer, Entropy and the complexity of graphs revisited, Bulletin of Mathematical Biophysics 30 (12) (1968) 387–414.

A. Mowshowitz, Entropy and the complexity of graphs: I. an index of the relative complexity of a graph, The bulletin of mathematical biophysics 30 (1) (1968) 175–204.

J. Körner, Coding of an information source having ambiguous alphabet and the entropy of graphs, in: 6th Prague conference on information theory, 1973, pp. 411–425.

M. Dehmer, Information processing in complex networks: Graph entropy and information functionals, Applied Mathematics and Computation 201 (1-2) (2008) 82–94.

J. Edward, M. Fitelson, Notes on maximum-entropy processing (corresp.), IEEE Transactions on information Theory 19 (2) (1973) 232–234.

S. J. Wernecke, L. R. D’Addario, Maximum entropy image reconstruction, IEEE Transactions on Computers 26 (4) (1977) 351–364.

M. T. Rahman, N. Kehtarnavaz, Q. R. Razlighi, Using image entropy maximum for auto exposure, Journal of Electronic Imaging 20 (1) (2011) 1917–1929.

A. Prat-Pérez, D. Dominguez-Sal, J.-M. Brunat, J.-L. Larriba-Pey, Put three and three together: Triangle-driven community detection, ACM Transactions on Knowledge Discovery from Data (TKDD) 10 (3) (2016) 1–42.

H. Yin, A. R. Benson, J. Leskovec, D. F. Gleich, Local higher-order graph clustering, in: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, 2017, pp. 555–564.

P.-Z. Li, L. Huang, C.-D. Wang, J.-H. Lai, Edmot: An edge enhancement approach for motif-aware community detection, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 479–487.

R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, U. Alon, Network motifs: simple building blocks of complex networks, Science 298 (5594) (2002) 824–827.

A. Sarajič, N. Malod-Dognin, Ö. N. Yaveroglu, N. Pržulj, Graphlet-based characterization of directed networks, Scientific reports 6 (1) (2016) 1–14.
[32] M. Dehmer, F. Emmert-Streib, Z. Chen, X. Li, Y. Shi, Mathematical Foundations and Applications of Graph Entropy, 2016.

[33] J. Weston, F. Ratle, H. Mobahi, R. Collobert, Deep learning via semi-supervised embedding, in: Neural networks: Tricks of the trade, Springer, 2012, pp. 639–655.

[34] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, T. Eliassi-Rad, Collective classification in network data, AI magazine 29 (3) (2008) 93–93.

[35] Z. Yang, W. W. Cohen, R. Salakhutdinov, Revisiting semi-supervised learning with graph embeddings, in: International Conference on International Conference on Machine Learning, 2016.

[36] J. Weston, F. Ratle, H. Mobahi, R. Collobert, Deep learning via semi-supervised embedding, in: Neural networks: Tricks of the trade, Springer, 2012, pp. 639–655.

[37] B. Perozzi, R. Al-Rfou, S. Skiena, Deepwalk: Online learning of social representations, in: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, 2014, pp. 701–710.

[38] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, K. Weinberger, Simplifying graph convolutional networks, in: International conference on machine learning, PMLR, 2019, pp. 6861–6871.

[39] M. Qu, Y. Bengio, J. Tang, Gmnn: Graph markov neural networks, in: International conference on machine learning, PMLR, 2019, pp. 5241–5250.

[40] W. L. Hamilton, R. Ying, J. Leskovec, Simplifying graph convolutional networks, in: Advances in Neural Information Processing Systems, 2017, pp. 1025–1035.

[41] W. Huang, T. Zhang, Y. Rong, J. Huang, Adaptive sampling towards fast graph representation learning, in: Advances in Neural Information Processing Systems, 2017, pp. 1025–1035.

[42] D. Zhu, Z. Zhang, P. Cui, W. Zhu, Robust graph convolutional networks against adversarial attacks, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 1399–1407.
[43] H. Liu, X. Xu, J.-A. Lu, G. Chen, Z. Zeng, Optimizing pinning control of complex dynamical networks based on spectral properties of grounded laplacian matrices, IEEE Transactions on Systems, Man, and Cybernetics: Systems (2018).