



Article

Robust Graph Neural Networks via Ensemble Learning

Qi Lin ¹, Shuo Yu ¹, Ke Sun ¹, Wenhong Zhao ^{2,*}, Osama Alfarraj ³ , Amr Tolba ³  and Feng Xia ⁴

- ¹ School of Software, Dalian University of Technology, Dalian 116620, China; lq@mail.dlut.edu.cn (Q.L.); yushuo@dlut.edu.cn (S.Y.); sunke@mail.dlut.edu.cn (K.S.)
- ² Ultraprecision Machining Center, Zhejiang University of Technology, Hangzhou 310014, China
- ³ Computer Science Department, Community College, King Saud University, Riyadh 11437, Saudi Arabia; oalfarraj@ksu.edu.sa (O.A.); atolba@ksu.edu.sa (A.T.)
- ⁴ School of Engineering, IT and Physical Sciences, Federation University Australia, Ballarat, VIC 3353, Australia; f.xia@ieee.org
- * Correspondence: zhaowh@zjut.edu.cn or whzhao6666@outlook.com

Abstract: Graph neural networks (GNNs) have demonstrated a remarkable ability in the task of semi-supervised node classification. However, most existing GNNs suffer from the nonrobustness issues, which poses a great challenge for applying GNNs into sensitive scenarios. Some researchers concentrate on constructing an ensemble model to mitigate the nonrobustness issues. Nevertheless, these methods ignore the interaction among base models, leading to similar graph representations. Moreover, due to the deterministic propagation applied in most existing GNNs, each node highly relies on its neighbors, leaving the nodes to be sensitive to perturbations. Therefore, in this paper, we propose a novel framework of graph ensemble learning based on knowledge passing (called GEL) to address the above issues. In order to achieve interaction, we consider the predictions of prior models as knowledge to obtain more reliable predictions. Moreover, we design a multilayer DropNode propagation strategy to reduce each node's dependence on particular neighbors. This strategy also empowers each node to aggregate information from diverse neighbors, alleviating oversmoothing issues. We conduct experiments on three benchmark datasets, including Cora, Citeseer, and Pubmed. GEL outperforms GCN by more than 5% in terms of accuracy across all three datasets and also performs better than other state-of-the-art baselines. Extensive experimental results also show that the GEL alleviates the nonrobustness and oversmoothing issues.

Keywords: graph neural networks; graph learning; ensemble learning; multilayer DropNode propagation; knowledge passing

MSC: 68T07; 05C62



Citation: Lin, Q.; Yu, S.; Sun, K.; Zhao, W.; Alfarraj, O.; Tolba, A.; Xia, F. Robust Graph Neural Networks via Ensemble Learning. *Mathematics* **2022**, *10*, 1300. <https://doi.org/10.3390/math10081300>

Academic Editors: Zhao Kang and Xiao Wang

Received: 21 March 2022

Accepted: 11 April 2022

Published: 14 April 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Graphs serve as structural data to describe complex relationships between entities among numerous networks, such as social networks [1,2], academic networks [3], citation networks [4,5], and traffic networks [6]. There is evidence that mining graphs can be beneficial for solving many real-world applications, such as node classification and clustering. Many studies have shown that graph neural networks (GNNs) [7–10] are a powerful approach to exploring graph data and can achieve promising results on graph-based semi-supervised learning tasks.

Despite the remarkable ability of most existing GNNs, recent studies have demonstrated that GNNs suffer from nonrobustness issues [11,12], i.e., GNNs are vulnerable to a small perturbation in the graph structures. By adding fake edges randomly to change the graph structures, the performance of GNNs can degrade dramatically, which poses a great challenge for employing GNNs in sensitive scenarios, such as finance networks [13].

To mitigate the nonrobustness issues faced by most GNNs, one strategy is to incorporate an ensemble learning mechanism into GNNs to obtain a stronger robustness. Based

on ensemble learning, several strategies have focused on enhancing the ability of base models and then constructing an ensemble directly [14–18], e.g., average the predictions of enhanced base models. Their experimental results show that the nonrobustness issues can be improved by applying an ensemble mechanism. However, these approaches have some major shortcomings: (1) They ignore the interactions among multiple models when constructing an ensemble framework, which may lead to obtaining similar graph representations [19]. As a result, the whole ensemble models may be affected by the same perturbation due to the transferability of perturbations among models [14]. (2) In most GNNs models, they still apply deterministic propagation for each node to extract information. This propagation rule makes each node highly dependent on its neighbors, leaving each node to be easily sensitive to perturbations in the graph structures.

To address the two challenges, in this paper, we propose a framework, i.e., graph ensemble learning based on knowledge passing (GEL), to alleviate the nonrobustness of GNNs and improve performance in the task of node classification. Specifically, we first present a knowledge-passing strategy to construct an ensemble model with interactions among base models. The motivation is that we hope the knowledge (predictions of prior models) can be passed to the next model, so that the next model can avoid degrading the performance caused by the same perturbations as in prior models, improving the robustness of the whole framework. Second, we design a multilayer DropNode propagation strategy, which is achieved by randomly dropping the entire feature matrix of each node during each propagation, with a different probability among base models. By doing this, each node aggregates information from diverse subsets of its neighbors rather than neighbors from a deterministic propagation, which reduces its dependence and sensitivity on particular neighbors and benefit from diverse neighborhoods, increasing the robustness of GEL. For instance, when some nodes are perturbed, in the deterministic propagation, the negative effect propagates to its neighbors. With our propagation rule, the effects are greatly reduced or even eliminated because the perturbed node may be excluded in the various subsets of neighbors depending on the different probability of DropNode. Furthermore, this propagation rule empowers each node to incorporate broader higher-hops information, mitigating the oversmoothing for GEL.

Finally, we conduct experiments on three public datasets with several popular GNNs models. Experimental results demonstrate that GEL outperforms GCN in terms of accuracy. We also show that the variants without the multilayer DropNode propagation or ensemble learning based on knowledge passing still gain improvements compared to GCN, which means that each strategy makes a difference to our framework. More importantly, we observe that GEL can mitigate nonrobustness and oversmoothing issues.

In summary, the contributions of this work are as follows:

- We propose a novel framework of graph ensemble learning based on knowledge passing (i.e., GEL) to address the robustness challenge of GNNs and improve the performance on semi-supervised learning tasks.
- We design a multilayer DropNode propagation strategy to reduce each node's dependence on particular neighbors, which can strengthen the robustness of GEL. Moreover, this propagation rule enables each node to extract knowledge from diverse subsets of neighbors, alleviating the oversmoothing issues.
- Experimental results on three public datasets show that our framework performs better than baseline methods in terms of classification accuracy and robustness.

The remainder of this paper is structured as follows. In Section 2, we outline the semi-supervised learning task on graphs and review related work. In Section 3, we elaborate on the proposed framework. Next, we conduct experiments to evaluate the performance of our framework in Section 4. Finally, we conclude the paper in Section 5.

2. Task Definition and Related Work

In this section, we describe the semi-supervised node classification tasks on graphs and introduce some notations in Table 1. Then, we review the related work.

Table 1. Description of key notations.

Symbols	Definitions
N	Number of base models
n	Number of nodes
m	Number of labeled nodes
p	Multilayer DropNode probability
K	Propagation Step
λ	KP loss coefficient
Θ	Set of model parameters
η	Learning rate
\mathbf{X}	Feature matrix
\mathbf{A}	Adjacency matrix
\mathbf{Y}	All possible labels
$\mathbf{H}^{(l)}$	Hidden node representations in the l th layer
$\mathbf{W}^{(l)}$	Weight matrix in the l th layer
$\tilde{\mathbf{X}}$	Perturbed matrix
\mathbf{X}'	Matrix after propagation
\mathbf{Z}	Predicted possibilities of matrix \mathbf{X}'

2.1. Semi-Supervised Node Classification

We describe a connected graph $G = (V, E)$, where $V = \{V_1, V_2, \dots, V_n\}$ is a node set including n nodes, and $E = \{e_{ij}\}_{1 \leq i, j \leq n}$ is an edge set indicating the connections between nodes. $\mathbf{A} \in \{0, 1\}^{n \times n}$ denotes the adjacency matrix of graph G . For an undirected graph, $\mathbf{A}_{ij} = 1$ indicates that there is an edge e_{ij} between node V_i and V_j , otherwise $\mathbf{A}_{ij} = 0$. We use $\mathbf{X} \in \mathbb{R}^{n \times d}$ to denote the feature matrix of graph G , wherein \mathbf{X}_i denotes the node V_i 's features and d is the dimension of the feature matrix.

We formalize the semi-supervised node classification tasks on graphs. In a graph, each vertex V_i is associated with its label $\mathbf{Y}_i \in \mathbf{Y}$, where \mathbf{Y} denotes all possible labels. For a semi-supervised node classification, m nodes have known labels $\mathbf{Y}^L \subset \mathbf{Y}$ and the labels $\mathbf{Y}^U = \mathbf{Y} \setminus \mathbf{Y}^L$ of the remaining $n - m$ nodes are unknown. The target is to design a function F to predict the labels \mathbf{Y}^U of unlabeled nodes via its corresponding feature matrix \mathbf{X} . Therefore, the predictive function can be detailed as follows:

$$F : G, \mathbf{X}, \mathbf{Y}^L \rightarrow \mathbf{Y}^U. \tag{1}$$

Traditional methods to solve this problem are mostly based on graph Laplacian regularization [20,21]. Recently, GNNs have emerged as promising approaches for semi-supervised node classification [22–25], which are briefly introduced below.

2.2. Graph Neural Networks

In this part, we introduce some representative GNNs methods and GNN models' propagation rule. In GNNs, each node propagates information to its neighbors with some deterministic propagation rules. For instance, in the graph convolutional network (GCN) [7] for semi-supervised learning on graphs, the graph propagation rule is formulated as follows:

$$\mathbf{H}^{(l+1)} = ReLU(\widehat{\mathbf{A}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}), \tag{2}$$

where $\widehat{\mathbf{A}}$ denotes a symmetric normalized adjacency matrix, $\mathbf{W}^{(l)}$ denotes the weight matrix in the l th layer, $\mathbf{H}^{(l)}$ is the hidden node representation in the l th layer, i.e., $\mathbf{H}^{(0)} = \mathbf{X}$ and $ReLU$ is the activation function.

Some methods have been proposed to advance this architecture. For example, Hamilton et al. [26] defined the graph convolution as aggregating information from neighbors. Petar Veličković et al. [27] applied an attention mechanism to assign different weights in the aggregation of neighbors' features. Xu et al. [28] added residual and jumping connections to adapt neighbors' properties. Wu et al. [29] removed the nonlinear activation function

to simplify GCN. Abu-El-Haija et al. [30] studied a class of neighborhood-mixing relationships. Tu et al. [31] chose hyperparameters automatically to improve the effectiveness and efficiency. However, all the above works do not consider the robustness of GNNs models.

2.3. Ensemble Learning

Ensemble learning was proposed to combine the predictions of base learners into more accurate predictions [32]. Ensemble learning has shown its effectiveness in many real-world scenarios. It is also widely used in semi-supervised node classification tasks.

There are a few studies applying ensemble learning to graphs. For example, Hou et al. [33] leveraged a graph ensemble technique to help dependency-based approaches alleviate the influence of parsing errors in the sentiment analysis area. Zhang et al. [34] trained many GCN models and then created an ensemble of them in a way similar to BAN. Further works have adopted ensemble learning to improve the robustness issues [14,15,17,35,36]. Liu et al. [18] used a voting ensemble for generating a high accuracy output. Mun et al. [37] trained an ensemble of GNN classifiers with dependent codes to improve the robustness of the networks.

However, these works ignore the interaction among base models, which refers to the sharing of information, such as knowledge and experience, among base models. In other words, each model outputs its predictions independently, without taking the predictions of prior models as their own knowledge or experience. Compared with these works, the goal of our framework is to utilize the predictions from prior models as knowledge to train a new one, so that the new model can avoid degrading the performance caused by the same perturbations as in prior models and thus gain a more precise accuracy in node embeddings.

3. The Design of GEL

We designed a graph ensemble learning based on knowledge passing (GEL) framework for semi-supervised node classification tasks and mitigating nonrobustness issues. As illustrated in Figure 1, a connected graph G with its adjacent matrix A and feature matrix X are given. In order to empower each node to extract information from diverse subsets of its neighbors, we utilized a multilayer DropNode propagation strategy to achieve this for each model. Afterwards, the matrix after propagation was put into a classification model MLP. Finally, we leveraged ensemble learning for better performance under a semi-supervised setting. Each step of our framework is explained in detail below.

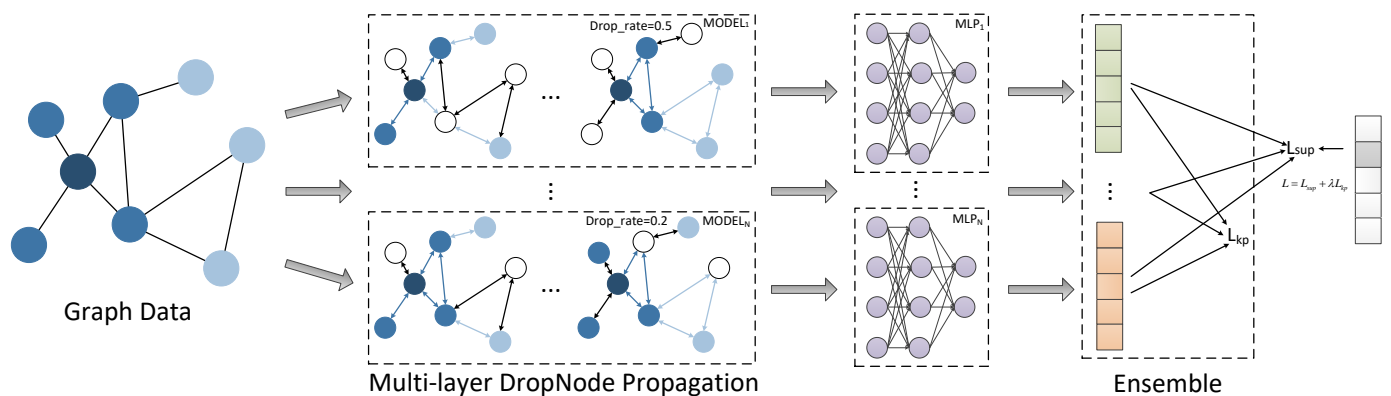


Figure 1. Illustration of GEL with multilayer DropNode propagation. GEL consists of multilayer DropNode propagation and graph ensemble learning based on knowledge passing.

3.1. MultiLayer DropNode Propagation

Multilayer DropNode Propagation. There were two steps in the multilayer DropNode propagation. First, before propagation, the feature matrix of each node was removed with probability p to gain a perturbed matrix \tilde{X} . Second, during propagation, we performed

label propagation to generate matrix \mathbf{X}' . Note that the above process was only implemented during training. During inference, we directly used the original feature matrix \mathbf{X} .

Formally, in the first step, we assigned a mask $p_i \sim \text{Bernoulli}(1 - \varepsilon)$ for each node v_i . Then, we gained the perturbed matrix $\tilde{\mathbf{X}}$ by multiplying each node's feature matrix with its mask, i.e., $\tilde{\mathbf{X}}_i = p \cdot \mathbf{X}_i$, where \mathbf{X}_i denotes the i th row of feature matrix \mathbf{X} . In doing so, we changed the graph structure before each propagation. Specifically, each node extracted information from its diverse subsets of neighbors rather than the same subsets of neighbor nodes, reducing its dependence on specific neighbors. Note that the probability p of removing node features was different for each model. Generally, the probability p is decreasing gradually from the first model to the last model. Thus, we obtained different node representations and ensured the diversity of the models.

In the second step, we adopted a multilayer label propagation, i.e., $\mathbf{X}' = \frac{1}{K+1} \sum_{k=0}^K \hat{\mathbf{A}}^k \tilde{\mathbf{X}}$, where $\hat{\mathbf{A}}^k = (\hat{\mathbf{A}}^{k-1} \cdot \tilde{\mathbf{X}}) \cdot p$, $\bar{\mathbf{A}} = \frac{1}{K+1} \sum_{k=0}^K \hat{\mathbf{A}}^k$ is the average of $\hat{\mathbf{A}}^0$ to $\hat{\mathbf{A}}^K$. Compared to directly using $\hat{\mathbf{A}}^K$, this propagation rule empowered each node to aggregate more local information, mitigating the risk of oversmoothing.

Compared with GRAND [38], we performed a DropNode strategy in each layer propagation, rather than only once before propagation. By doing so, we obtained stochastic subsets of neighbors for each node. Consequently, each node extracted information randomly from its diverse neighbors and thus its features could be affected by more remote neighbors. Moreover, we guaranteed the reduction of the distraction from noise nodes' information, enhancing the robustness of the model and meanwhile alleviating the oversmoothing problem.

Prediction. For each model, the matrix \mathbf{X}' were generated after performing the multilayer random DropNode propagation. Then, each matrix was fed into an MLP model to obtain the corresponding classification results:

$$\mathbf{Z} = f_{mlp}(\mathbf{X}', \Theta), \tag{3}$$

where $\mathbf{Z} \in (0,1)$ denotes the predicted possibilities of the matrix \mathbf{X}' and Θ is a set of model parameters.

3.2. Graph Ensemble Learning Based on Knowledge Passing

We designed a graph ensemble learning method based on knowledge passing. The main idea was to consider not just the performance of a single model, but the interactions among base models. Our goal was for the model to learn the knowledge of prior models, which could make the model perform well.

Ensemble learning. We adopted an ensemble learning strategy, which was to use multiple MLPs to complete the classification task. For each model, we obtained the matrix \mathbf{X}' from the multilayer DropNode propagation. Afterwards, it was fed into an MLP model to output its corresponding predictions. Except for the first model, each base model used the classification results of the prior models as knowledge.

Specifically, we designed the serialization ensemble learning method as shown in Figure 2. First, the classifier MLP_1 outputted the classification result $\mathbf{Z}^{(1)}$, and $\mathbf{Z}^{(1)}$ was delivered to the next model MLP_2 . When the classifier MLP_2 performed classification, it received the result delivered by the first model and gave its own predictions. It is not difficult to conclude that each i th model received knowledge from the $(i - 1)$ th model. The optimization objective which aligned the predictions between the i th model and $(i - 1)$ th model can be formulated as

$$\min_{\Theta} \sum_{v \in V} \text{distance}(f_{mlp_i}(v), f_{mlp_{(i-1)}}(v)), \tag{4}$$

where $distance(\cdot, \cdot)$ denotes the distance between the predictions of two models. In doing so, the i th model received the prior knowledge transferred from the $(i - 1)$ th model and thus the i th model performed better.

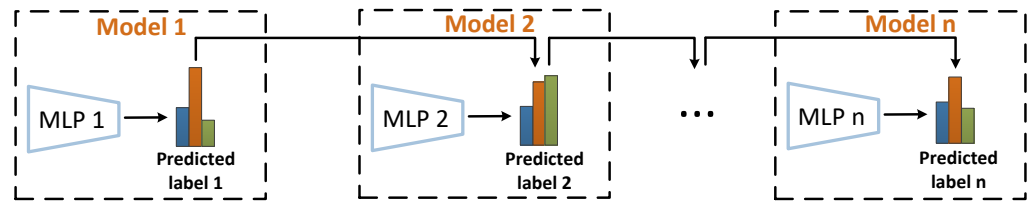


Figure 2. Serialization ensemble learning. Each i th base model receives knowledge from $(i - 1)$ th base model.

Meanwhile, we also designed a parallel ensemble learning method entitled P-GEL as one of the comparison algorithms shown in Figure 3. Different from the serialized ensemble learning method, the first $N - 1$ models were trained independently and outputted classification results $\mathbf{Z}^{(1)}$ to $\mathbf{Z}^{(N-1)}$, respectively. When the i th model was trained, it received classification results from the first model up to the $(i - 1)$ th model and outputted its own classification result.

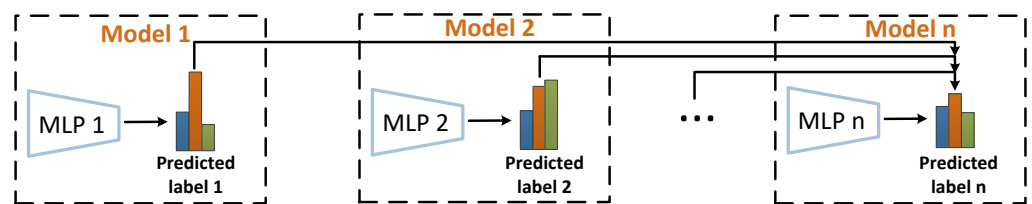


Figure 3. Parallel ensemble learning. The first $N - 1$ models are trained independently. The N th model receives classification results from the first model up to the $(N - 1)$ th model.

It can be seen from Section 4.2 that the classification results of the serialized ensemble learning algorithm are better than that of the parallel ensemble learning algorithm, which indicates that knowledge passing strategy among base models is significant. When a single model encounters perturbation and outputs poor classification results, the knowledge from prior models can improve the poor performance. Consequently, the robustness of the models is greatly strengthened.

Training and Inference. In the whole algorithm, the loss function was mainly divided into two parts: the loss in a model and the loss between models. The loss in a model referred to the supervised loss. With m labeled nodes among n nodes, the supervised loss in every epoch was formulated as the cross-entropy loss:

$$\mathcal{L}_{sup} = \sum_{i=0}^{m-1} \mathbf{Y}_i^T \log \mathbf{Z}_i. \tag{5}$$

The loss between modes was knowledge passing loss. Concretely, when the $(i - 1)$ th model passed knowledge to the i th model, we minimized the distance between $\mathbf{Z}^{(i)}$ and $\mathbf{Z}^{(i-1)}$:

$$\mathcal{L}_{kp} = \frac{1}{N} \sum_{i=0}^{N-1} \sum_{j=0}^{n-1} \left\| \mathbf{Z}_j^{(i)} - \mathbf{Z}_j^{(i-1)} \right\|_2^2. \tag{6}$$

The final loss of our algorithm was:

$$\mathcal{L} = \mathcal{L}_{sup} + \lambda \mathcal{L}_{kp}, \tag{7}$$

where λ is a hyperparameter to control the balance between the two losses. Algorithm 1 summarizes the training process of GEL.

Algorithm 1 GEL.

Require: Graph G , adjacent matrix \mathbf{A} , feature matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$, the number of models N , DropNode probability p , learning rate η , an MLP model: $f_{mlp}(\mathbf{X}', \Theta)$.

Ensure: Prediction results \mathbf{Z} .

```

1: while not convergence do
2:   for  $n = 1 : N$  do
3:     Perturb the feature matrix:  $\tilde{\mathbf{X}} \sim \text{DropNode}(\mathbf{X}, p)$ 
4:     Perform multilayer DropNode propagation:  $\mathbf{X}' = \frac{1}{K+1} \sum_{k=0}^K \hat{\mathbf{A}}^k \tilde{\mathbf{X}}$ 
5:     Predict the distribution of class using an MLP:  $\mathbf{Z} = f_{mlp}(\mathbf{X}', \Theta)$ 
6:   end for
7:   Calculate supervised classification loss  $\mathcal{L}_{sup}$  via Equation (5) and knowledge passing loss via Equation (6).
8:   Update the parameters  $\Theta$  by gradient descending:  $\Theta = \Theta - \eta \nabla_{\Theta} (\mathcal{L}_{sup} + \lambda \mathcal{L}_{kp})$ .
9: end while
10: Output prediction  $\mathbf{Z}$  via Equation (3).

```

3.3. Computational Complexity

For a model, the time complexity of the multilayer DropNode propagation is $\mathcal{O}(Kd(n + |E|))$, where K represents propagation step, d is the dimension of node feature, n is the number of nodes and $|E|$ is the count of edges. The next step is the prediction module and it can be accomplished in $\mathcal{O}(nd_h(d + C))$, where d_h represents its hidden size and C is the number of classes. The total computational complexity of GEL is $\mathcal{O}(N((Kd(n + |E|)) + (nd_h(d + C))))$, where N denotes the number of models.

4. Experiments

In this section, we begin with introducing the details of the datasets, comparative baselines, variants, and experimental settings. Then, we present the overall experimental results on the semi-supervised node classification task and compare the performance on network visualization. Afterwards, we give an experiment comparing between GEL and GCN [7], GAT [27], and GRAND [38] to validate the advantage of GEL in robustness and oversmoothing issues. Finally, we conduct experiments to study the effects of different hyperparameters.

4.1. Experimental Setup

4.1.1. Datasets

To evaluate the performance of GEL on semi-supervised node classification tasks, we used three benchmark datasets: Cora [39], Citeseer [39], and Pubmed [40]. The statistics of the three datasets are summarized in Table 2.

Table 2. Benchmark dataset statistics.

Dataset	Nodes	Edges	Classes	Features
Cora	2708	5429	7	1433
Citeseer	3327	4732	6	3703
Pubmed	19,717	44,338	3	500

The details for Cora, Citeseer, and Pubmed are as follows:

- Cora [39] is a benchmark dataset related to citations between machine learning papers. It is widely used in the field of graph learning. Each node represents a paper and the edges represent citations between papers. The label of a node indicates the research field of a paper.
- Similar to Cora, Citeseer [39] is another benchmark dataset which represents the citations between computer science papers, keeping a similar configuration to Cora.

- Pubmed [40] is also a citation dataset which is relevant to articles about diabetes. The node features are weighted frequency–inverse document frequencies (TF-IDF). The label of a node denotes the type of diabetes.

4.1.2. Baselines and Variants

We conducted experiments with other comparative algorithms to evaluate the performance of GEL. The details of the comparative baselines are listed as follows:

- GCN [7] is a semi-supervised learning approach which employs novel convolution operators on graph-structured data to learn node representations.
- GAT [27] performs better than GCN by combining an attention mechanism which specifies different weight to a neighbor node.
- DGI [41] is an unsupervised learning approach to study node representations. Maximizing mutual information between patch representations and corresponding high-level summaries were proposed in DGI.
- APPNP [42] improved GCN by connecting GCN with PageRank. A new propagation procedure based on personalized PageRank was proposed to make full use of neighbor information.
- MixHop [30] was proposed to study neighborhood mixing relationships, such as difference operators. Sparsity regularization lets us visualize which neighborhood information will be chosen in priority by the network.
- GraphSAGE [26] was proposed to study node embeddings by sampling and aggregating information which comes from a node’s local neighborhood.
- GRAND [38] first designed random propagation to achieve data augmentation and then proposed consistency loss to optimize the prediction loss of unlabeled nodes through data augmentation.
- RDD [34] was proposed to define node reliability and edge reliability to ensure the quality of a model. Moreover, a new ensemble learning method was proposed to combine the above optimization.

For each dataset, we tested the following variants of our method:

- M-GEL: The variant without the multilayer DropNode propagation mechanism.
- K-GEL: The variant without graph ensemble learning based on knowledge passing mechanism.
- P-GEL: The variant with the multilayer DropNode propagation and graph ensemble learning based on parallel knowledge passing.

4.1.3. Settings

We used PyTorch to conduct our experiments. The preprocessing for the three datasets was accomplished with the reference of Planetoid [43]. The experimental settings of the three basic datasets were exactly the same as works on semi-supervised learning tasks [7]. For Cora, the values of train nodes, valid nodes, and test nodes were, respectively, 140, 500, and 1000. For Citeseer, the values of train nodes, valid nodes, and test nodes were, respectively, 120, 500, and 1000. For Pubmed, the values of train nodes, valid nodes, and test nodes were, respectively, 60, 500, and 1000. In addition, we employed early stopping with a patience of 200 as an indicator of termination in the training process. To evaluate the performance of GEL, the metric for the classification task used in our experiment was accuracy. All experiments were conducted on PyCharm 2020. As for other software versions, we used Python 3.7.3, PyTorch 1.2.0, Numpy 1.16.4, and CUDA 11.2.

4.2. Node Classification Results

The accuracy of node classification predicted by GEL is shown in Table 3. The results of other baselines are all conducted with the same settings as our algorithm.

From Table 3, we can clearly observe that GEL consistently achieves stable improvements across the three datasets in contrast to other baselines. Specifically, GEL improves upon GCN by a margin of 5%, 5.7%, and 5.2% on Cora, Citeseer, Pubmed. Compared

to GAT, we gain 3.5%, 3.4%, and 5.2% improvements, respectively. When compared to GRAND, GEL achieves 1.1%, 0.4%, and 1.5% improvements, respectively.

Table 3. Overall classification accuracy (%).

Method	Cora	Citeseer	Pubmed
GCN	81.5	70.3	79.0
GAT	83.7	73.2	79.3
DGI	82.9	72.5	77.4
APPNP	84.1	72.1	80.0
MixHop	82.3	72.2	81.4
GraphSAGE	79.7	68.1	78.4
GRAND	85.8	75.8	83.3
RDD	86.1	74.2	81.5
GEL	86.5	76.0	84.2
M-GEL	84.6	74.7	81.1
K-GEL	85.4	74.8	82.9
P-GEL	85.7	75.5	84.2

We observe that P-GEL also outperforms most of the baselines, though still lower than GEL. This indicates serialization ensemble learning method is better than the parallel ensemble learning method. This also suggests that knowledge passing between models is significant. We then conducted an ablation experiment to study the contributions of different components in GEL. From the experimental results of two variants named M-GEL and K-GEL, we have two observations. Firstly, the performance of all GEL variants with some components removed is significantly reduced compared to the full model, demonstrating that every component of the design contributes to GEL's success. Second, GEL without the multilayer DropNode propagation outperforms almost all baselines across the three datasets, illustrating the positive effect of the proposed ensemble learning based on knowledge passing for semi-supervised graph learning.

4.3. Network Visualization

We can explore the network structure in two-dimensional space by network visualization. In this experiment, we visualized the Cora network using GCN, GAT, GRAND, and the proposed method GEL and its variants. Seven visualized networks are illustrated in Figure 4, where each color denotes a class. We summarize the observations as follows:

- GCN tightly confuses red, purple, and green, as well as blue and lake blue. GAT poorly separates the boundary of green and peak green, and strongly confuses red, orange, and purple. GRAND fails to separate purple and red and cannot develop the boundary of green and peak green.
- GEL shows a remarkable ability to visualize the Cora network. It can separate points of different colors and cluster points of the same color, though tightly confuse red and purple. We can also observe that the variants of GEL also show a significant performance in the network visualization compared to GCN, GAT, and GRAND. These results indicate that the multilayer DropNode propagation and ensemble learning on graphs are useful in network visualization.

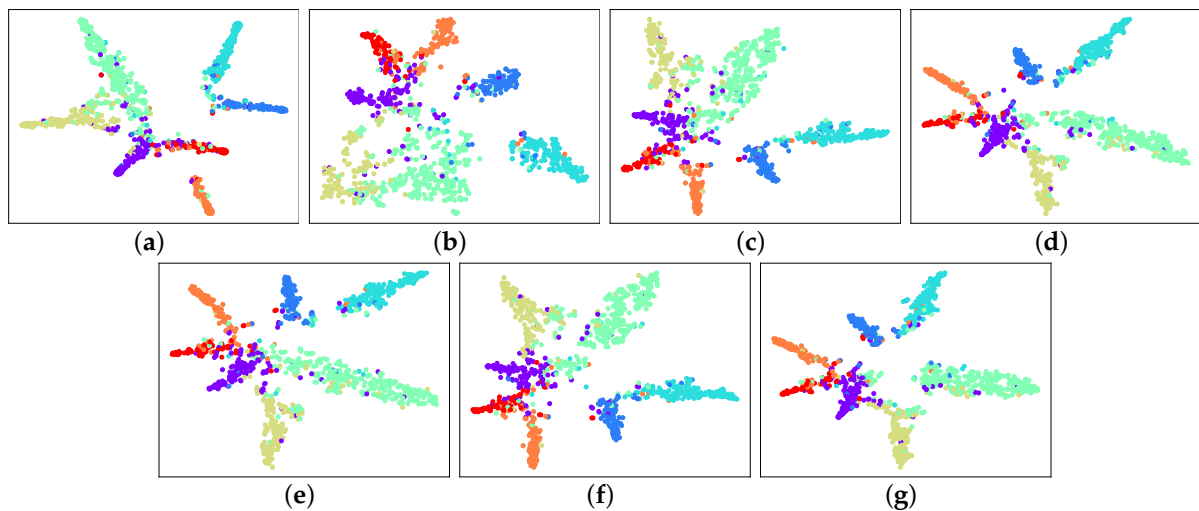


Figure 4. Network visualization on the Cora network using t-SNE. Each color denotes one class. (a) GCN; (b) GAT; (c) GRAND; (d) GEL; (e) P-GEL; (f) K-GEL; (g) M-GEL.

4.4. Robustness Analysis

In this part, we examine the robustness of GEL by perturbing graphs with a random attack method by adding fake edges randomly.

Figure 5 shows the classification accuracy of different methods when perturbing the Cora dataset with different perturbation rates. We can see that GEL outperforms GCN and GAT across all perturbation rates. When compared to the very recent GRAND model, we can observe that although the accuracy of the proposed method is slightly lower than that of GRAND when the perturbing probability is less than 100%, when the perturbing probability is greater than 100%, GEL shows obvious advantages. When adding 200% new random edges into the Cora dataset, we can observe that the classification accuracy for GEL decreases only 17.6%, while it decreases by 26.5% for GRAND, 28.4% for GAT, and 70.8% for GCN. This study indicates the robustness advantage of the GEL model with the increase of the perturbation rate.

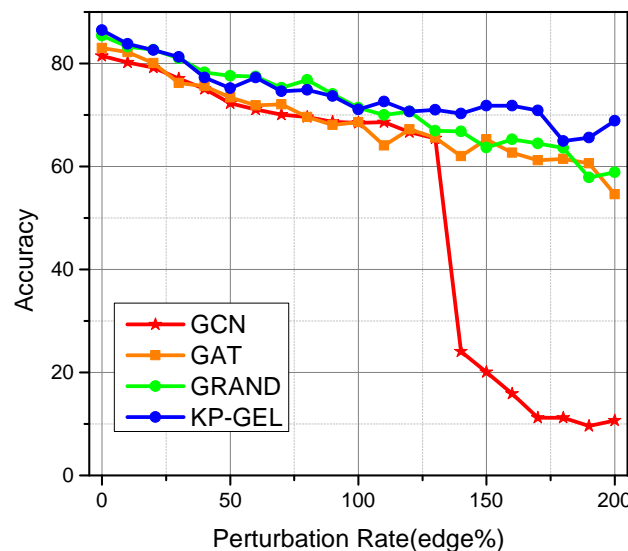


Figure 5. Robustness analysis on the Cora network with random attack edge.

4.5. Oversmoothing Analysis

Many GNNs suffer from oversmoothing problems. When deepening the propagation step, GNNs may make nodes with different labels indistinguishable. In this experiment, we studied how vulnerable GEL was to this problem on the Cora and Citeseer networks by

deepening the propagation step. Higher classification accuracy with a deeper propagation step indicates a less severe oversmoothing issue.

Figure 6 presents the classification accuracy with different propagation steps on the Cora and Citeseer networks. In GEL and GRAND, the propagation step is adjusted with the hyperparameter K , while in GCN and GAT, it is controlled by different hidden layers. As shown in Figure 6, with the propagation step increasing, the classification accuracy of GCN and GAT drops dramatically on both networks because of the oversmoothing problem. However, GEL and GRAND perform completely different. Both GEL and GRAND are not affected by the propagation step. Meanwhile, the performance of GEL is always better than GRAND as the propagation step increases on both networks. This suggests that GEL is much more powerful to alleviate the oversmoothing issue.

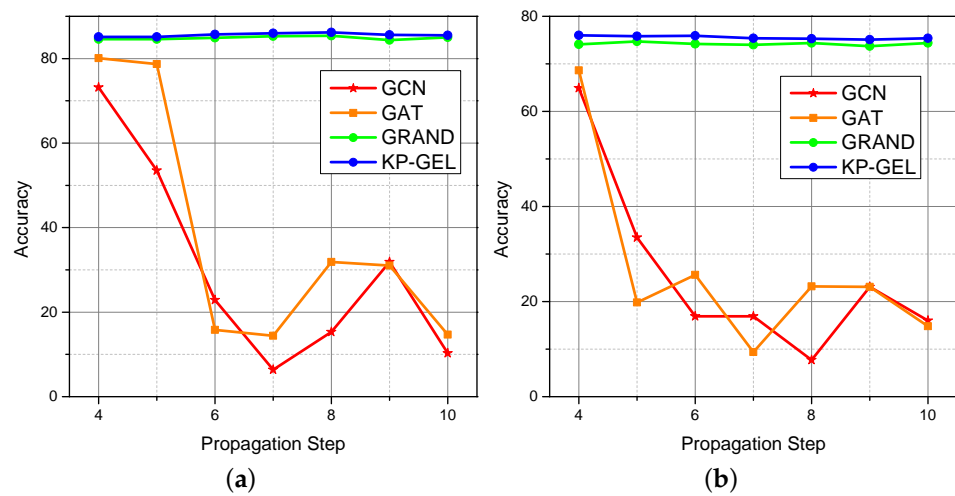


Figure 6. Oversmoothing analysis. (a) Cora Network; (b) Citeseer Network.

4.6. Parameter Analysis

In this subsection, we investigate the parameter sensitivity. For GEL, one of the most crucial parameters is the number of models. As a result, we first studied how it affected the performance of GEL on the three datasets. We set $N \in \{2, 3, 4, 5, 6, 7, 8, 9\}$ with a fixed DropNode probability p and propagation step for each dataset. For the Cora and Citeseer networks, GEL behaved best with six to eight models. For the Pubmed dataset, the GEL performed best with two models.

We also studied some extra additional parameters, that are the propagation step K , DropNode probability p , and KP loss coefficient λ . We performed hyperparameter searching for each dataset. Concretely, we first searched K among $\{2, 3, 4, 5, 6, 7, 8, 9\}$ for each dataset. With the best choice of K , we then searched the DropNode probability of each model, considering that each ensemble learning model should have diversities, and we set the DropNode probability of each model to be different. Afterwards, we studied the number of models for Citeseer and Cora again from six to eight models. Finally, we fixed $\lambda \in \{0.5, 0.7, 1.0\}$. Other parameters included the early stopping patience, hidden layer size, L2 weight decay rate, dropout rate in the input layer, and dropout rate in the hidden layer. These parameters did not cost much time, because the GEL was not sensitive to them. The optimal hyperparameters we used in our experiments are shown in Table 4.

Table 4. Hyperparameters Settings of GEL

Hyperparameter	Cora	Citeseer	Pubmed
Model number N	6	6	2
Propagation step K	8	2	5
DropNode probability p	[0.7, 0.6, 0.5, 0.3, 0.2, 0.15]	[0.7, 0.6, 0.5, 0.4, 0.3, 0.2]	[0.7, 0.6]
KP loss coefficient λ	1.0	0.7	1.0
Learning rate η	0.01	0.01	0.2
Early stopping patience	200	200	200
Hidden layer size	32	32	32
L2 weight decay rate	5×10^{-4}	5×10^{-4}	5×10^{-4}
Dropout rate in input layer	0.5	0.0	0.6
Dropout rate in hidden layer	0.5	0.2	0.8

5. Conclusions

In this paper, we studied the semi-supervised learning tasks on graphs and presented the graph ensemble learning based on knowledge passing (i.e., GEL) to mitigate the nonrobustness issues faced by most existing GNNs. We proposed the multilayer DropNode propagation, a strategy empowering each node to extract information from diverse subsets of its neighbors. Thus, each node's information depended not only on a single node, but also on multiple subsets of neighbors, alleviating oversmoothing issues. Then, we leveraged ensemble learning based on knowledge passing for considering the interaction among base models to avoid degrading the performance caused by the same perturbations as in prior models, alleviating nonrobustness issues. Experimental results on three datasets demonstrated that GEL outperformed other state-of-the-art baselines on semi-supervised node classification tasks, illustrating the importance of the ensemble learning and multilayer DropNode propagation used in our framework. Additional experiments on random attacks and different numbers of propagation layers showed the advantage of our algorithm with respect to robustness and oversmoothing issues.

In future work, we will improve the adaptability of various attacks, e.g., metattack attacks and adversarial attacks. We will also explore whether our framework is applicable to other graph-based tasks, such as unsupervised learning tasks and some anomaly detection tasks. Another line of research would be to refine our framework by encouraging fewer models (e.g., a distillation model) for better performance.

Author Contributions: Conceptualization, Q.L., W.Z. and F.X.; investigation, Q.L. and K.S.; methodology, Q.L., S.Y. and K.S.; supervision, W.Z. and F.X.; validation, K.S., W.Z., O.A. and A.T.; writing—original draft, Q.L. and S.Y.; writing—review and editing, W.Z., O.A., A.T. and F.X. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the Researchers Supporting Project No. RSP-2021/102 at King Saud University, Riyadh, Saudi Arabia.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The three datasets used in this paper are publicly available.

Acknowledgments: This work was funded by the Researchers Supporting Project No. RSP-2021/102 at King Saud University, Riyadh, Saudi Arabia. The authors would like to thank Lei Wang and Liuwei Fu for their help with the experiments.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Stadtfeld, C.; Vörös, A.; Elmer, T.; Boda, Z.; Raabe, I.J. Integration in emerging social networks explains academic failure and success. *Proc. Natl. Acad. Sci. USA* **2019**, *116*, 792–797. [[CrossRef](#)] [[PubMed](#)]
2. Xu, J.; Yu, S.; Sun, K.; Ren, J.; Lee, I.; Pan, S.; Xia, F. Multivariate relations aggregation learning in social networks. In Proceedings of the ACM/IEEE Joint Conference on Digital Libraries, Virtual Event, China, 1–5 August 2020; pp. 77–86.
3. Xia, F.; Wang, W.; Bekele, T.M.; Liu, H. Big scholarly data: A survey. *IEEE Trans. Big Data* **2017**, *3*, 18–35. [[CrossRef](#)]
4. Ebesu, T.; Fang, Y. Neural citation network for context-aware citation recommendation. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Tokyo, Japan, 7–11 August 2017; pp. 1093–1096.
5. Xia, F.; Wang, L.; Tang, T.; Chen, X.; Kong, X.; Oatley, G.; King, I. CenGCN: Centralized Convolutional Networks with Vertex Imbalance for Scale-Free Graphs. *IEEE Trans. Knowl. Data Eng.* **2022**. [[CrossRef](#)]
6. Lee, K.; Eo, M.; Jung, E.; Yoon, Y.; Rhee, W. Short-term traffic prediction with deep neural networks: A survey. *IEEE Access* **2021**, *9*, 54739–54756. [[CrossRef](#)]
7. Kipf, T.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. In Proceedings of the International Conference on Learning Representations, San Juan, Puerto Rico, 2–4 May 2016.
8. Xiao, Y.; Li, C.; Liu, V. DFM-GCN: A Multi-Task Learning Recommendation Based on a Deep Graph Neural Network. *Mathematics* **2022**, *10*, 721. [[CrossRef](#)]
9. Xia, F.; Sun, K.; Yu, S.; Aziz, A.; Wan, L.; Pan, S.; Liu, H. Graph Learning: A Survey. *IEEE Trans. Artif. Intell.* **2021**, *2*, 109–127. [[CrossRef](#)]
10. Yu, S.; Xia, F.; Sun, Y.; Tang, T.; Yan, X.; Lee, I. Detecting outlier patterns with query-based artificially generated searching conditions. *IEEE Trans. Comput. Soc. Syst.* **2020**, *8*, 134–147. [[CrossRef](#)]
11. Zügner, D.; Akbarnejad, A.; Günnemann, S. Adversarial attacks on neural networks for graph data. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 2847–2856.
12. Zhu, D.; Zhang, Z.; Cui, P.; Zhu, W. Robust graph convolutional networks against adversarial attacks. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 1399–1407.
13. Nagurney, A. Networks in economics and finance in Networks and beyond: A half century retrospective. *Networks* **2021**, *77*, 50–65. [[CrossRef](#)]
14. Kurakin, A.; Goodfellow, I.; Bengio, S.; Dong, Y.; Liao, F.; Liang, M.; Pang, T.; Zhu, J.; Hu, X.; Xie, C.; et al. Adversarial attacks and defences competition. In *The NIPS'17 Competition: Building Intelligent Systems*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 195–231.
15. Kannan, H.; Kurakin, A.; Goodfellow, I. Adversarial logit pairing. *arXiv* **2018**, arXiv:1803.06373.
16. Croce, F.; Hein, M. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In Proceedings of the International Conference on Machine Learning, Virtual, 13–18 July 2020; pp. 2206–2216.
17. Bui, A.; Le, T.; Zhao, H.; Montague, P.; deVel, O.; Abraham, T.; Phung, D. Improving Ensemble Robustness by Collaboratively Promoting and Demoting Adversarial Robustness. In Proceedings of the National Conference on Artificial Intelligence, New York, NY, USA, 7 February 2020.
18. Liu, L.; Wei, W.; Chow, K.H.; Loper, M.; Gursoy, E.; Truex, S.; Wu, Y. Deep Neural Network Ensembles Against Deception: Ensemble Diversity, Accuracy and Robustness. In Proceedings of the Mobile Adhoc and Sensor Systems, Monterey, CA, USA, 4–7 November 2019.
19. Li, Y.; Yosinski, J.; Clune, J.; Lipson, H.; Hopcroft, J.E. Convergent Learning: Do different neural networks learn the same representations? In Proceedings of the International Conference on Learning Representations, San Juan, Puerto Rico, 2–4 May 2016.
20. Weston, J.; Ratle, F.; Mobahi, H.; Collobert, R. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 639–655.
21. Zhu, X.; Ghahramani, Z.; Lafferty, J.D. Semi-supervised learning using gaussian fields and harmonic functions. In Proceedings of the 20th International Conference on Machine Learning (ICML-03), Washington, DC, USA, 21–24 August 2003; pp. 912–919.
22. Sun, K.; Liu, J.; Yu, S.; Xu, B.; Xia, F. Graph force learning. In Proceedings of the 2020 IEEE International Conference on Big Data (Big Data), Atlanta, GA, USA, 10–13 December 2020; pp. 2987–2994.
23. Sun, K.; Wang, L.; Xu, B.; Zhao, W.; Teng, S.W.; Xia, F. Network representation learning: From traditional feature learning to deep learning. *IEEE Access* **2020**, *8*, 205600–205617. [[CrossRef](#)]
24. Yu, S.; Feng, Y.; Zhang, D.; Bedru, H.D.; Xu, B.; Xia, F. Motif discovery in networks: A survey. *Comput. Sci. Rev.* **2020**, *37*, 100267. [[CrossRef](#)]
25. Berrone, S.; Della Santa, F.; Mastropietro, A.; Pieraccini, S.; Vaccarino, F. Graph-Informed Neural Networks for Regressions on Graph-Structured Data. *Mathematics* **2022**, *10*, 786. [[CrossRef](#)]
26. Hamilton, W.L.; Ying, R.; Leskovec, J. Inductive representation learning on large graphs. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 1025–1035.
27. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; Bengio, Y. Graph Attention Networks. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.

28. Xu, K.; Li, C.; Tian, Y.; Sonobe, T.; Kawarabayashi, K.i.; Jegelka, S. Representation learning on graphs with jumping knowledge networks. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; pp. 5453–5462.
29. Wu, F.; Souza, A.; Zhang, T.; Fifty, C.; Yu, T.; Weinberger, K. Simplifying graph convolutional networks. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 6861–6871.
30. Abu-El-Haija, S.; Perozzi, B.; Kapoor, A.; Alipourfard, N.; Lerman, K.; Harutyunyan, H.; Ver Steeg, G.; Galstyan, A. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 21–29.
31. Tu, K.; Ma, J.; Cui, P.; Pei, J.; Zhu, W. Autone: Hyperparameter optimization for massive network embedding. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 216–225.
32. Dietterich, T.G. Ensemble learning. In *The Handbook of Brain Theory and Neural Networks*; MIT Press: Cambridge, MA, USA, 2002; Volume 2, pp. 110–125.
33. Hou, X.; Qi, P.; Wang, G.; Ying, R.; Huang, J.; He, X.; Zhou, B. Graph Ensemble Learning over Multiple Dependency Trees for Aspect-level Sentiment Classification. *arXiv* **2021**, arXiv:2103.11794.
34. Zhang, W.; Miao, X.; Shao, Y.; Jiang, J.; Chen, L.; Ruas, O.; Cui, B. Reliable data distillation on graph convolutional network. In Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data, Portland, OR, USA, 14–19 June 2020; pp. 1399–1414.
35. Zou, J.; Fu, L.; Zheng, J.; Yang, S.; Yu, G.; Hu, Y. A many-objective evolutionary algorithm based on rotated grid. *Appl. Soft Comput.* **2018**, *67*, 596–609. [[CrossRef](#)]
36. Zou, J.; Fu, L.; Yang, S.; Zheng, J.; Ruan, G.; Pei, T.; Wang, L. An adaptation reference-point-based multiobjective evolutionary algorithm. *Inf. Sci.* **2019**, *488*, 41–57. [[CrossRef](#)]
37. Mun, Y.J.; Kang, J.W. Ensemble of random binary output encoding for adversarial robustness. *IEEE Access* **2019**, *7*, 124632–124640. [[CrossRef](#)]
38. Feng, W.; Zhang, J.; Dong, Y.; Han, Y.; Luan, H.; Xu, Q.; Yang, Q.; Kharlamov, E.; Tang, J. Graph Random Neural Networks for Semi-Supervised Learning on Graphs. In Proceedings of the Neural Information Processing Systems, Vancouver, BC, Canada, 6–12 December 2020.
39. Sen, P.; Namata, G.; Bilgic, M.; Getoor, L.; Galligher, B.; Eliassi-Rad, T. Collective classification in network data. *AI Mag.* **2008**, *29*, 93–93. [[CrossRef](#)]
40. Namata, G.; London, B.; Getoor, L.; Huang, B.; EDU, U. Query-driven active surveying for collective classification. In Proceedings of the 10th International Workshop on Mining and Learning with Graphs, Washington, DC, USA, 24–25 July 2012; Volume 8.
41. Veličković, P.; Fedus, W.; Hamilton, W.L.; Liò, P.; Bengio, Y.; Hjelm, R.D. Deep Graph Infomax. *ICLR (Poster)* **2019**, *2*, 4.
42. Klicpera, J.; Bojchevski, A.; Günnemann, S. Predict then Propagate: Graph Neural Networks meet Personalized PageRank. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
43. Yang, Z.; Cohen, W.; Salakhudinov, R. Revisiting semi-supervised learning with graph embeddings. In Proceedings of the International Conference on Machine Learning, New York, NY, USA, 20–22 June 2016; pp. 40–48.