# Information Augmentation for Few-shot Node Classification

**Zongqian Wu**[1], **Peng Zhou**[1], **Guoqiu Wen**[1*], **Yingying Wan**[2], **Junbo Ma**[1], **Debo Cheng**[1] and **Xiaofeng Zhu**[1,2*]
[1]Guangxi Key Lab of Multi-Source Information Mining and Security, Guangxi Normal University, Guilin 541004, China
[2]School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China
wkzongqianwu@gmail.com

## Abstract

Although meta-learning and metric learning have been widely applied for few-shot node classification (FSNC), some limitations still need to be addressed, such as expensive time costs for the meta-train and difficult of exploring the complex structure inherent the graph data. To address in issues, this paper proposes a new data augmentation method to conduct FSNC on the graph data including parameter initialization and parameter fine-tuning. Specifically, parameter initialization only conducts a multi-classification task on the base classes, resulting in good generalization ability and less time cost. Parameter fine-tuning is designed to have two data augmentation methods (*i.e.*, support augmentation and shot augmentation) on the novel classes to generate sufficient node features so that any traditional supervised classifiers can be used to classify the query set. As a result, the proposed method is the first work of data augmentation for FSNC. Experiment results show the effectiveness and the efficiency of our proposed method, compared to state-of-the-art methods, in terms of different classification tasks.

## 1 Introduction

As the success of deep neural networks is found to significantly depend on the sample number, *i.e.*, large scale amount of samples, there has been growing interest in investigating effective deep models with a few samples. Recently, few-shot learning (FSL) is widely applied to classify unseen samples by training supervised or semi-supervised models with only a few training samples [Huang *et al.*, 2021; Lazarou *et al.*, 2021; Wang *et al.*, 2020b]. Previous FSL methods can be divided into three categories, *i.e.*, meta-learning methods, metric learning methods and data augmentation methods. Meta-learning methods first use the meta-train to learn initialization parameters and then apply them in the meta-test [Song *et al.*, 2019; Oh *et al.*, 2020]. Metric learning methods are designed to learn one task-invariant metric for all the tasks to conduct FSL [Hao *et al.*, 2019; Jiang *et al.*, 2020]. Data augmentation methods augment the number of training samples to increase the shot numbers [Zhou *et al.*, 2021; Yang *et al.*, 2021].

In the literature, either meta-learning methods or metric learning methods are more popular for FSL than data augmentation methods. However, meta-learning methods require each task of the meta-train to have the same class number as the novel classes while metric learning methods are often failed to learn the task-invariant metric due to the issue of task divergence [Cheng *et al.*, 2019]. These issues seriously limit their applications. On the contrary, data augmentation methods are designed to augment the samples and thus be available to conduct traditional supervised or semi-supervised learning as well as avoiding the above issues. However, a few literature have been focused on conducting FSL by data augmentation. For example, [Xian *et al.*, 2019] design a generative model fusing generative adversarial network (GAN) with adversarial autoencoder (VAE) into a unified framework to augment the samples and [Sun *et al.*, 2021] investigate to generate high confidence samples within a ball generator.

Although existing FSL methods have been fast developing, most of these work focus on Euclidean data such as image and text whereas few attention has been paid on non-Euclidean data such as graphs and manifolds [Zhou *et al.*, 2019]. For example, [Zhou *et al.*, 2019] propose a graph meta-learning framework to conduct few-shot node classification (FSNC) based on graph neural networks (GNN). [Huang and Zitnik, 2020] investigate to first represent every node with a local subgraph and then use subgraphs to train GNNs for conducting FSL on graphs. In real applications, non-Euclidean data has shown explosive growth, but many previous FSL methods on non-Euclidea data focused on meta-learning. Hence, there is a great demand of conducting FSL with data augmentation on non-Euclidean data, aiming at dealing with the graph data as well as avoiding the issues of meta-learning.

The challenge of conducting FSNC with data augmentation mainly comes from the complex data structure of the graph data. Specifically, the graph data includes feature information and relation information between two nodes. Usually, it is difficult to simultaneously augment these two kinds of information [Zhou *et al.*, 2019]. Moreover, either a few number of samples or noise makes the data augmentation task more challenging [Ni *et al.*, 2021].

In this paper, we propose a novel data augmentation method (IA-FSNC for shot) to address the above issues for conducting

*Corresponding authors (seanzhuxf@gmail.com).

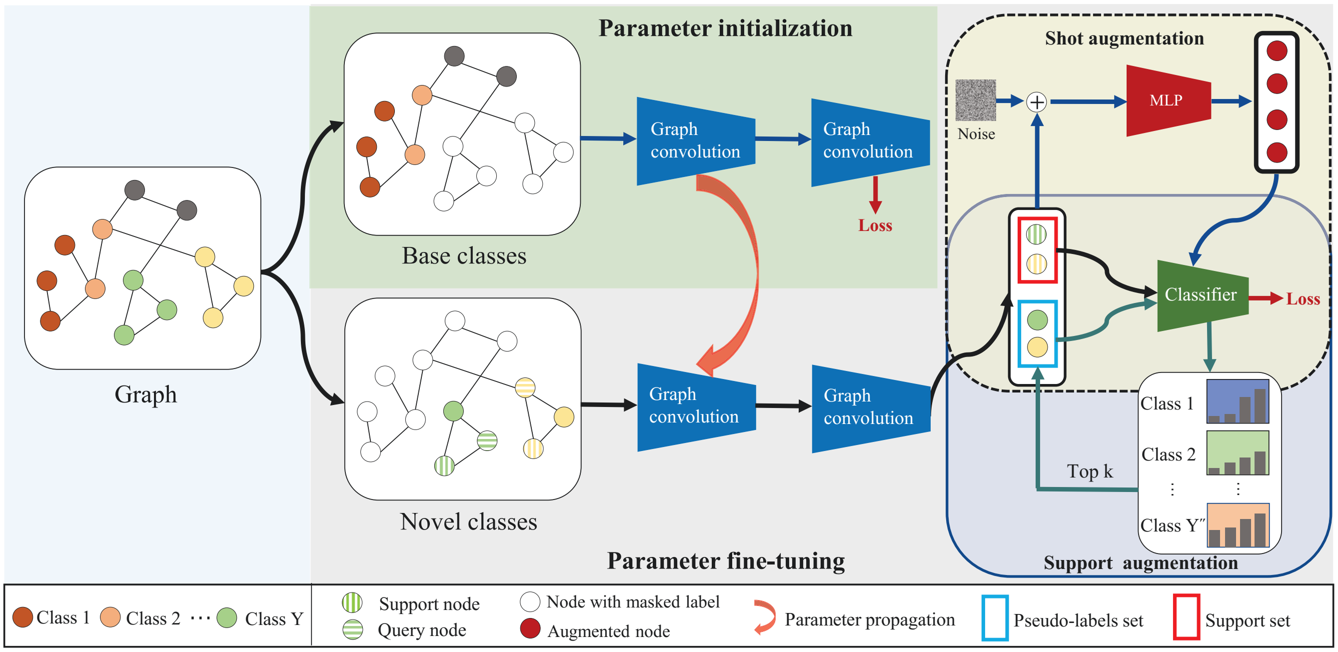



Figure 1: The proposed IA-FSNC includes two sequential modules, *i.e.*, **Parameter initialization** and **Parameter fine-tuning**. Parameter initialization first employs the GCN to conduct a multi-classification task on the base classes and then regards the parameters in the first layer as the initialization parameters of the GCN on the novel classes. Parameter fine-tuning includes support augmentation and shot augmentation. Specifically, support augmentation first employs the GCN to generate node embedding of all nodes in the novel classes and then uses the support features to train a classifier, which further assigns pseudo-labels to the nodes (without selected by either the support set or the query set in novel classes) with low entropy. Support augmentation is obtained by self-training, which involves to update the classifier twice. Shot augmentation employs a multi-layer perception (MLP) on these features with random noise to generate new features.

FSNC. It involves two modules, *i.e.*, parameter initialization and parameter fine-tuning, using the backbone of graph convolutional network (GCN). To do this, we employ a GCN on the based classes to initialize the parameters in the parameter initialization module. In parameter fine-tuning module, the initialized parameters are fine-tuned to fit the support set by two data augmentation methods, *i.e.*, support augmentation and shot augmentation. Specifically, the initialization parameters are inputted to the first graph convolution layer of a GCN model for the novel classes. The outputted node embedding is sequentially conducted support augmentation and shot augmentation to generate sufficient node features so that any supervised classifiers can be used. As a result, two GCNs used in our framework are employed to conduct semi-supervised learning on the graph data by exploring the complex data structure inherent the graph data and requiring less time cost for parameter initialization, while two data augmentation methods are used for achieving the effectiveness as well as making any existing supervised classifiers be available for FSNC.

# 2 Method

## 2.1 Motivation

Given the input graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{v_1, v_2, \cdots, v_n\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ represent the node set and the edge set, respectively, we denote $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]^T$ and $\mathbf{A} \in \{0, 1\}^{n \times n}$, respectively, as the feature matrix of all nodes and the adjacency matrix of $\mathbf{X}$.

Given a graph with node-label pairs $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{n}$ where $\mathbf{y}_i \in \mathbf{Y}$ and $\mathbf{Y}$ denotes the node class set, we divide $\mathbf{Y}$ into two sets, *i.e.*, $\mathbf{Y} = \mathbf{Y}' \cup \mathbf{Y}''$ where $\mathbf{Y}'$ is the base class set and $\mathbf{Y}''$ is the novel class set. The goal of FSNC is to first train a model from the base classes and then to fine-tune it on a few samples within the novel classes. For the evaluation of FSNC models, we usually construct an $N$-way $K$-shot task, where $N$ is the number of novel classes and each class has only $K$ (*e.g.*, 1, 3 and 5) samples. Moreover, the model is trained on the *support set* $\mathcal{S} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N \times K}$ and is evaluated on the *query set* $\mathcal{Q} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=N \times K+1}^{N \times b}$ where the literature including this work usually set $b = 12$.

In the literature, Meta-GNN [Zhou *et al.*, 2019] and its variants [Wang *et al.*, 2020a; Guo *et al.*, 2021; Liu *et al.*, 2021; Ding *et al.*, 2020; Ding *et al.*, 2021] are very popular for FSNC. To do this, in parameter initialization, they first partition the base classes into multiple tasks where each task have the equivalent class number as the novel classes, and then employ model-agnostic meta-learning (MAML) [Finn *et al.*, 2017] to learn the initialization parameters from these tasks. In parameters fine-tuning, these parameters are further fine-tuned by the support set to output a model for evaluating the query set.

However, previous works still have limitations to be addressed. First, in parameter initialization, previous meta-learning methods for FSNC perform many tasks under the backbone of GCNs, resulting in expensive time cost. Moreover, it is unfeasible in the real application for their assumption that each task has the same class number as the novel classes. Second, in parameter fine-tuning, a number of FSL methods ignore to use the unselected node set[1], resulting in insufficient information for parameter fine-tuning due to waiting useful information inherent these nodes. Lastly, to the best of our knowledge, no literature has focused on conducting FSNC with data augmentation as it is challenging to augment the graph data, *i.e.*, simultaneously augmenting nodes and edges.

To address the above issues, we propose a new data augmentation method (namely IA-FSNC) for FSNC, by involving two modules shown in Figure 1. The difference between ours and existing FSNC methods is obvious. First, similar to previous meta-learning methods for FSNC, our method also uses the GCN as the backbone to deal with the complex relation among the graph data, but our method is the first work for FSNC with data augmentation. Second, different from traditional FSNC methods that conduct parameter initialization with multiple tasks and each task has the same class number as the novel classes, our method conducts one multi-class classification task with all classes. Third, compared to previous methods that employ the support set only to conduct parameter fine-tuning on the GCN, our parameter fine-tuning employs the GCN on all node features in the novel classes (including the nodes in the support set and the query set, and the unselected nodes) to generate their node embedding.

## 2.2 Parameter Initialization

In the process of parameter initialization under semi-supervised learning, all nodes are used to learn the node embedding. In particular, parameter initialization in meta-learning is designed to partition all nodes into multiple tasks and each task is further designed to use all nodes to learn their node embedding and the classifiers are for specific classes, *i.e.*, different tasks have different classes.

Obviously, in previous methods, individual tasks have the same input but have different initialization parameters and different outputs including node embedding of all nodes and the classifier. To achieve this, individual tasks are usually using the same network, *e.g.*, a two-layer GCN. As a result, first, the node embedding of all nodes in the first layer (*i.e.*, low-level embedding) is similar for every task since they have the same input. Second, the node embedding of all nodes in the second layer of each task (*i.e.*, high-level embedding) is very different from other tasks as individual tasks have different classes to be distinguished.

Inspired by the above observations, we propose a new method of parameter initialization for FSNC, *i.e.*, propagating the parameters of the first layer in the GCN to the process of parameter fine-tuning as well as only designing a multi-classification task to generate the initialization parameters for the first layer of the GCN in parameter fine-tuning. The motivation is that different tasks have similar parameters in the first layer for learning node embedding so that the resulted parameters are feasible for the initialization parameters to be fine-tuned. In this way, we do not need to use a complex meta-training mechanism for parameter initialization.

To achieve this, we follow the traditional FSNC setting to divide the dataset into base classes and novel classes. The base classes has $M$ classes and novel classes has $N$ classes. We thus conduct a $M$-class classification task by a two-layer GCN to conduct parameter initialization. Specifically, we first define the node embedding of all nodes $\mathbf{Z}^{(i)}$ in the $i$-th layer of the GCN as follows:

$$\begin{cases} \mathbf{Z}^{(i)} = \sigma(\hat{\mathbf{A}}\mathbf{Z}^{(i-1)}\mathbf{W}^{(i-1)}) \\ \mathbf{P} \quad = \mathrm{softmax}(\sigma(\hat{\mathbf{A}}\mathbf{Z}^{(i)}\mathbf{W}^{(i)})) \end{cases} \tag{1}$$

where $\mathbf{W}^{(i)}$ is the trainable parameter of the $i$-th layer and $\sigma$ is the activation function. In particular, we have $\hat{\mathbf{A}} = \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$ where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$, $\tilde{d}_{ii} = \sum_j \tilde{a}_{ij}$ and $\mathbf{I}_N$ is a diagonal matrix. After obtaining the probability matrix $\mathbf{P}$, the cross entropy loss is defined as:

$$\mathcal{L} = -\sum_{i \in \mathcal{S}} \sum_{j=1}^{c} y_{ij} \ln p_{ij} \tag{2}$$

where $c$ represents the number of classes.

After the process of parameter initialization, we take the first-layer parameters as the initialization parameters of the first-layer of the GCN in the novel classes. Compared to our parameter initialization method, traditional meta-learning methods (1) propagate the parameters in all layers to parameter fine-tuning but the parameters in the last layers are specified for the classes in the base classes and thus has less useful information for the classes in the novel classes and (2) sequentially conduct multiple GCNs to result in expensive time cost. In particular, traditional meta-learning methods require each task to have the same class number of the novel classes, limiting them in the real applications.

## 2.3 Parameter Fine-tuning

Different from traditional FSNC methods that parameter fine-tuning aims at fine-tuning the initialization parameters to fit the support set, our parameter fine-tuning is to generate sufficient node features so that any supervised classifiers can be used to evaluate the query set. To do this, our parameter fine-tuning includes support augmentation and shot augmentation.

**Support Augmentation by Self-training**

Given initialization parameters from the process of parameter initialization, the two-layer GCN is used to output the node embedding of all node features in the novel classes, as well as train a classifier between the nodes in the support set and their labels. However, this process easily suffers from the over-fitting issue due to limited sample number. To address this, our first solution is to employ self-training [McClosky *et al.*, 2006] to add the number of node features. Specifically, we use the classifier to classify the nodes in the unselected node set. We expect to label some of them to add the node feature number (*i.e.*, support augmentation) based on the uncertainty measurement, *i.e.*, information entropy. That is, the lower the

[1]In the FSNC literature, the novel classes include the nodes in the support sets, the nodes in the query set, and the nodes without selected by either the support set or the query set in the novel classes (unselected node set for shot), *e.g.*, a green node and a yellow node in the novel classes of Figure 1. Moreover, all of nodes in the unselected node set are used for parameter fine-tuning under semi-supervised learning.

information entropy of a node is, the higher the probability of this node is correctly classified. In this way, we select top $k$ nodes with the lowest information entropy for each class and assign them with labels.

Specifically, using Eq. (1) to obtain the parameter matrix $\mathbf{P}'$ of all nodes in the novel class set can be used to calculate the information entropy $\mathbf{h}$. Assuming that we have a set of unselected nodes, *i.e.*, $\mathbf{h} = \{h_1, h_2, \cdots, h_i, \cdots, h_q\}$, and the information entropy of a node $h_i$ is calculated by

$$h_i = -\sum_{j=1}^{c} p'_{ij} \log_2 p'_{ij} \tag{3}$$

After obtaining the entropy of all unselected nodes, we rank them by an increase order to assign reliable node features with pseudo-labels, called them pseudo-label set.

We further update the classifier using both the node features in the support set and the features in the pseudo-label set. As a result, we adaptively update their features and the classifier.

#### Shot Augmentation

After obtaining the node features in the pseudo-label set, we use them and the node features in the support set to generate new node features. This is our second solution to conduct FSNC with a few samples. Specifically, we first add random noise into these features and then apply an MLP to generate new features, *i.e.*, shot augmentation. The goal of adding random noise into the features is generating diverse features. We further use the obtained node features to update the classifier, which is used for evaluating the query set.

We mention that it is difficult to conduct data augmentation for FSNC on the graph data due to the complex data structure. [Ni *et al.*, 2021] partition data augmentation methods to four categories, *i.e.*, support augmentation, shot augmentation, query augmentation and task augmentation. However, many previous methods directly generate new nodes or samples and few literature focused on generating new node features. In this paper, we propose two data augmentation methods to generate the node features as the node embedding contains both the feature information and the relation information between two nodes. As a result, our data augmentation methods can simultaneously augment feature information and relation information to explore the complex data structure on the graph data and their effectiveness is verified in Sec. 3.3.

### 2.4 Objective Function

We conduct two sequential modules to conduct FSNC on the graph data. Two modules employ the same backbone, *i.e.*, GCN, but they have different contributions. Specifically, the GCN in parameter initialization outputs initialization parameters to benefit parameter fine-tuning, *i.e.*, information augmentation. The GCN in parameter fine-tuning design two data augmentation methods (*i.e.*, information augmentation again) to generate sufficient node features so that any supervised classifiers can be used to evaluate the query set.

In parameter fine-tuning, the classifier is updated three times. Different updates have different goals. Specifically, the first update is used to generate the pseudo-label set, and the second update is used to update the features of the nodes in the support set and the pseudo-label set. The third update guarantees generating new features as well as is used to evaluate the query set.

## 3 Experiments

### 3.1 Experimental Settings

#### Datasets

In our experiments, we use four public real-world datasets, including two citation datasets (*i.e.*, Cora and Citeseer [Kipf and Welling, 2016]), one KDD challenge dataset, *i.e.*, Coauthor (CS) [Shchur *et al.*, 2018], and one ecommerce dataset (*i.e.*, Computers [Shchur *et al.*, 2018]).

#### Comparison Methods

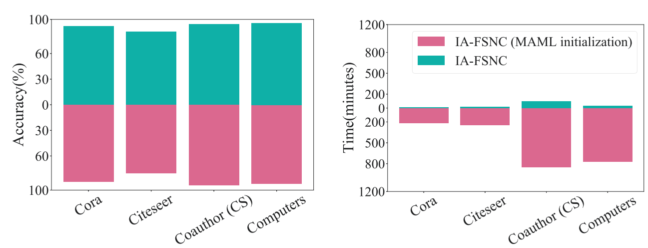
The comparison methods include one GCN method (*i.e.*, GCN [Kipf and Welling, 2016], two meta-learning methods (*i.e.*, Meta-GNN [Zhou *et al.*, 2019] and G-Meta [Huang and Zitnik, 2020]), two metric learning methods (*i.e.*, Proto-GNN [Snell *et al.*, 2017] and GPN [Ding *et al.*, 2020]) and two data augmentation methods, *i.e.*, GCN-ID [Wang *et al.*, 2018] and GCN-DC [Yang *et al.*, 2021][2].

### 3.2 Result Analysis

#### Classification Result Analysis

We report the accuracy of all methods at different shot numbers on all datasets in Table 1. First, our proposed IA-FSNC significantly outperforms all SOTA methods. For example, our method averagely improves by 8.85%, 4.46%, and 3.26%, respectively, compared to GCN, in terms of 1-shot, 3-shot, and 5-shot on all datasets. With similar competition, our method averagely improves by 6.86%, 3.28%, and 2.41%, respectively, compared to the best comparison method *i.e.*, Meta-GNN. This show the effectiveness of our proposed method.

Second, our method achieves different improvements in terms of different shot numbers, compared to the comparison methods. Specifically, our method achieves the improvement by 10.35%, 7.18%, and 5.23% respectively, on all comparison methods and all datasets, in terms of 1-shot, 3-shot and 5-shot. The reason is that the classification model has more room for improvement for less training samples.




(a) Accuracy (b) Time costs

Figure 2: Accuracy and time costs of our parameter initialization and MAML in our framework.

[2]Since no literature focused on data augmentation for FSNC, we adapt FSL methods (*i.e.*, GCN-ID and GCN-DC) to conduct FSNC by replacing their backbone CNN with GCN.

| Datasets | Shot | Baseline | Meta-learning methods | | Metric learning methods | | Data augmentation methods | | |
|---|---|---|---|---|---|---|---|---|---|
| | | GCN | Meta-GNN | G-META | Proto-GNN | GPN | GCN-ID | GCN-DC | IA-FSNC |
| Cora | 1 | 73.31±2.15 | 75.75±1.92 | 69.74±0.01 | 69.67±1.97 | 72.05±0.13 | 74.63±2.15 | 67.26±1.70 | **85.51±1.77** |
| | 3 | 86.21±1.04 | 88.01±1.69 | 83.02±0.01 | 85.42±1.01 | 78.39±0.20 | 87.39±1.20 | 69.81±2.03 | **91.95±0.67** |
| | 5 | 90.14±0.68 | 90.79±0.27 | 85.15±0.01 | 91.33±0.69 | 82.40±0.27 | 90.74±0.78 | 75.63±1.67 | **93.38±0.47** |
| Citeseer | 1 | 65.12±1.86 | 67.36±1.55 | 65.46±0.01 | 63.25±1.50 | 63.53±0.11 | 65.72±2.02 | 64.76±1.37 | **78.44±2.63** |
| | 3 | 77.28±1.61 | 79.01±1.26 | 77.99±0.02 | 74.50±0.99 | 71.67±0.13 | 77.24±1.82 | 69.09±1.45 | **85.28±1.49** |
| | 5 | 80.57±1.49 | 82.98±1.77 | 82.79±0.01 | 77.90±2.01 | 81.28±0.15 | 80.87±1.58 | 71.22±1.33 | **86.86±1.20** |
| Coauthor (CS) | 1 | 86.52±0.74 | 87.39±0.31 | 86.75±0.19 | 76.19±2.04 | 72.79±0.15 | 83.71±1.13 | 72.99±2.89 | **88.24±0.71** |
| | 3 | 92.91±0.11 | 93.34±0.28 | 91.66±0.23 | 88.21±0.32 | 85.36±0.12 | 92.97±0.27 | 84.16±1.20 | **94.35±0.11** |
| | 5 | 93.01±0.28 | 94.01±0.19 | 93.89±0.15 | 91.83±0.55 | 88.29±0.09 | 93.39±0.16 | 85.81±0.80 | **94.79±0.07** |
| Computers | 1 | 81.68±3.02 | 84.09±2.12 | 87.48±0.03 | 79.85±3.33 | 81.55±0.87 | 82.29±3.26 | 84.11±3.01 | **89.86±2.36** |
| | 3 | 93.10±0.82 | 93.88±0.95 | 92.85±0.01 | 81.67±1.98 | 93.92±1.34 | 94.99±0.56 | 86.60±2.45 | **96.01±0.41** |
| | 5 | 95.25±0.46 | 94.37±0.51 | 92.63±0.05 | 88.41±0.81 | 94.74±0.48 | 96.08±0.30 | 90.77±1.06 | **96.75±0.26** |

Table 1: Classification accuracy (mean and standard deviation) of all methods at different shot numbers on all datasets, where the bold number represents the best results in the whole row.

| C1 | C2 | C3 | Cora | Citeseer | Coauthor (CS) | Computers |
|---|---|---|---|---|---|---|
| ✓ | | | 87.52±1.14 | 77.97±1.86 | 92.06±0.16 | 95.34±0.44 |
| | ✓ | | 90.98±0.81 | 83.49±1.68 | 93.31±0.12 | 93.15±0.75 |
| | | ✓ | 84.81±1.36 | 76.71±1.94 | 93.30±0.25 | 93.79±1.22 |
| ✓ | ✓ | | **92.42±0.61** | 85.20±1.49 | 93.19±0.09 | 95.97±0.32 |
| | ✓ | ✓ | 90.94±0.81 | 83.55±0.16 | 93.91±0.58 | 93.28±0.75 |
| ✓ | | ✓ | 87.43±0.11 | 77.35±1.78 | 92.54±0.34 | 94.91±0.57 |
| ✓ | ✓ | ✓ | 91.95±0.67 | **85.28±1.49** | **94.35±0.11** | **96.01±0.41** |

Table 2: Classification accuracy (mean and standard deviation) of our method with different components at 3-shot on all datasets, where the bold number represents the best results in the whole column.

### 3.3 Ablation Study

#### Effectiveness of Individual Components

Our IA-FSNC includes three key components, *i.e.*, parameter initialization (C1 for short), support augmentation (C2 for short) and shot augmentation (C3 for short). We report the classification results of their combinations on all datasets in Table 2 to demonstrate the effectiveness of individual components. First, each component (*i.e.*, the first three lines in Table 2) is effective as they output significant classification performance, indicating the feasibility of our proposed method. Moreover, among these components, the support augmentation achieves the best results. The reason is that the classifier has serious issue of the overfitting with a few samples, and there is a lot of room for adding pseudo-labels to improve performance. Second, although the shot augmentation achieves the worst improvement, compared to other two components, the improvement is very significant while combing it with other components. The reason is that the shot augmentation is depended on the support augmentation. Hence, it is reasonable for this work to simultaneously taking the support augmentation and the shot augmentation into account.

#### Parameter Initialization Effectiveness

In the last paragraph, we verified the reasonability of our proposed parameter initialization, and further demonstrate its effectiveness and efficiency. To do this, we replace our parameter initialization with MAML [Finn *et al.*, 2017] in our framework and report the results in Figure 2. As a result, our parameter initialization improves by 2.72% in terms of accuracy and nearly 10 times faster in terms of time costs, compared to our framework with MAML. This shows that our proposed parameter initialization has good generalization ability and efficiency.

### 3.4 Parameter Sensitivity Analysis

We investigate the parameter sensitivity of our IA-FSNC, *i.e.*, top $k$ for selecting the pseudo-label set and $\mu$ for the learning rate. To do this, we vary the values of $k$ from 0 to 100 and the values of $\mu$ from 0.01 to 0.2, and then report the results in Figure 3, 4 and 5. Obviously, our method is sensitive to the setting of $k$. Specifically, the accuracy increases with the values of $k$ from 0 to 50. However, the accuracy achieves the peak and becomes stable while the values of $k$ is above 50. The reason is the information entropy has reached a critical point for $k \geq 50$. In particular, the accuracy reduces if the value of $k$ is too small. If the value of $\mu$ is too large (*e.g.*, 10 and 0.2), the accuracy is greatly reduced. Therefore, we experimentally set $k = 50$ and $\mu = 0.02$ in our experiments.

## 4 Conclusion

In this paper, we proposed a new FSL method with data augmentation for FSNC by achieving effectiveness and efficiency. Inspired by observing the reason that traditional parameter initialization is time-consuming, we proposed an effective and efficient solution and then proposed two data augmentation methods (*i.e.*, support augmentation and shot augmentation) to improve the effectiveness of FSNC. As a result, this work is the first work to conduct FSNC with data augmentation on the graph data. Experimental results showed that our method achieves supreme performance, compared to the state-of-the-art methods.

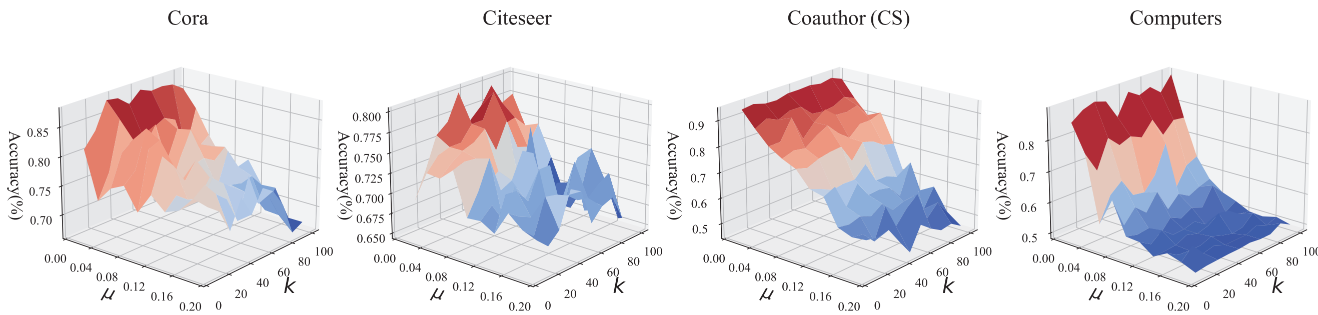



Figure 3: Results of IA-FSNC at different parameter settings (*i.e.*, $\mu$ and $k$) in terms of 1-shot.

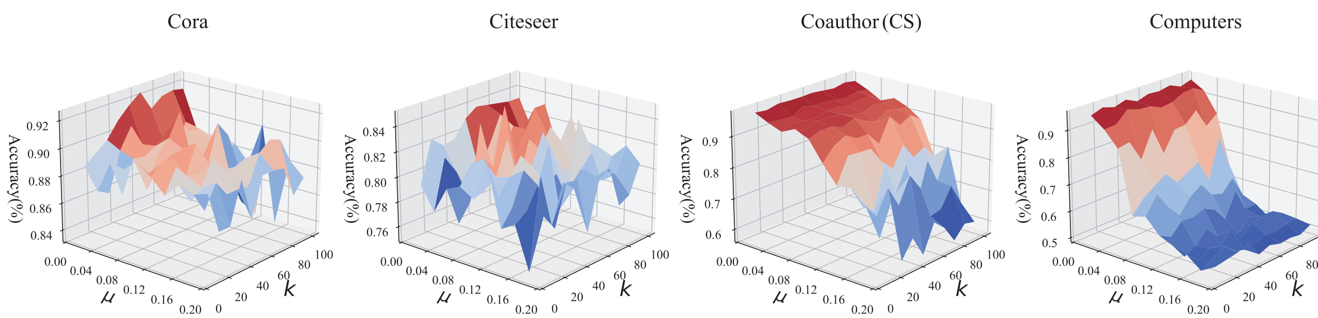



Figure 4: Results of IA-FSNC at different parameter settings (*i.e.*, $\mu$ and $k$) in terms of 3-shot.

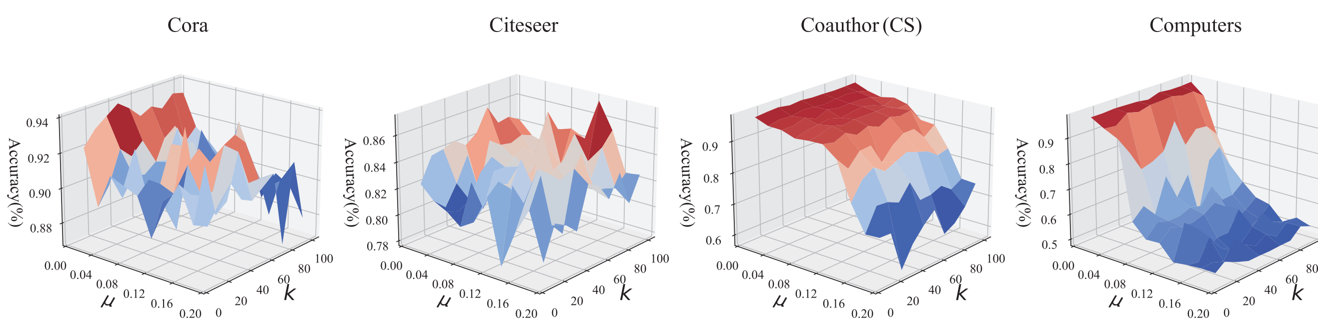



Figure 5: Results of IA-FSNC at different parameter settings (*i.e.*, $\mu$ and $k$) in terms of 5-shot.

## Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 61876046, and in part by the Guangxi "Bagui" Teams for Innovation and Research.

## References

[Cheng *et al.*, 2019] Yu Cheng, Mo Yu, Xiaoxiao Guo, and Bowen Zhou. Few-shot learning with meta metric learners. *arXiv preprint arXiv:1901.09890*, 2019.

[Ding *et al.*, 2020] Kaize Ding, Jianling Wang, Jundong Li, Kai Shu, Chenghao Liu, and Huan Liu. Graph prototypical networks for few-shot learning on attributed networks. In *CIKM*, pages 295–304, 2020.

[Ding *et al.*, 2021] Kaize Ding, Jianling Wang, Jundong Li, James Caverlee, and Huan Liu. Weakly-supervised graph meta-learning for few-shot node classification. *arXiv preprint arXiv:2106.06873*, 2021.

[Finn *et al.*, 2017] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, pages 1126–1135, 2017.

[Guo *et al.*, 2021] Zhichun Guo, Chuxu Zhang, Wenhao Yu, John Herr, Olaf Wiest, Meng Jiang, and Nitesh V Chawla. Few-shot graph learning for molecular property prediction. In *Web Conference*, pages 2559–2567, 2021.

[Hao *et al.*, 2019] Fusheng Hao, Fengxiang He, Jun Cheng, Lei Wang, Jianzhong Cao, and Dacheng Tao. Collect and select: Semantic alignment metric learning for few-shot learning. In *ICCV*, pages 8460–8469, 2019.

[Huang and Zitnik, 2020] Kexin Huang and Marinka Zitnik. Graph meta learning via local subgraphs. In *NeurIPS*, 2020.

[Huang *et al.*, 2021] Kai Huang, Jie Geng, Wen Jiang, Xinyang Deng, and Zhe Xu. Pseudo-loss confidence metric for semi-supervised few-shot learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8671–8680, 2021.

[Jiang *et al.*, 2020] Wen Jiang, Kai Huang, Jie Geng, and Xinyang Deng. Multi-scale metric learning for few-shot learning. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(3):1091–1102, 2020.

[Kipf and Welling, 2016] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[Lazarou *et al.*, 2021] Michalis Lazarou, Tania Stathaki, and Yannis Avrithis. Iterative label cleaning for transductive and semi-supervised few-shot learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8751–8760, 2021.

[Liu *et al.*, 2021] Zemin Liu, Yuan Fang, Chenghao Liu, and Steven CH Hoi. Relative and absolute location embedding for few-shot node classification on graph. In *AAAI*, pages 4267–4275, 2021.

[McClosky *et al.*, 2006] David McClosky, Eugene Charniak, and Mark Johnson. Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 152–159, 2006.

[Ni *et al.*, 2021] Renkun Ni, Micah Goldblum, Amr Sharaf, Kezhi Kong, and Tom Goldstein. Data augmentation for meta-learning. In *ICML*, pages 8152–8161, 2021.

[Oh *et al.*, 2020] Jaehoon Oh, Hyungjun Yoo, ChangHwan Kim, and Se-Young Yun. Boil: Towards representation change for few-shot learning. *arXiv preprint arXiv:2008.08882*, 2020.

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

[Snell *et al.*, 2017] Jake Snell, Kevin Swersky, and Richard S Zemel. Prototypical networks for few-shot learning. *arXiv preprint arXiv:1703.05175*, 2017.

[Song *et al.*, 2019] Xingyou Song, Wenbo Gao, Yuxiang Yang, Krzysztof Choromanski, Aldo Pacchiano, and Yunhao Tang. Es-maml: Simple hessian-free meta learning. *arXiv preprint arXiv:1910.01215*, 2019.

[Sun *et al.*, 2021] Pengfei Sun, Yawen Ouyang, Wenming Zhang, and Xinyu Dai. MEDA: meta-learning with data augmentation for few-shot text classification. In *IJCAI*, pages 3929–3935, 2021.

[Wang *et al.*, 2018] Yu-Xiong Wang, Ross Girshick, Martial Hebert, and Bharath Hariharan. Low-shot learning from imaginary data. In *CVPR*, pages 7278–7286, 2018.

[Wang *et al.*, 2020a] Ning Wang, Minnan Luo, Kaize Ding, Lingling Zhang, Jundong Li, and Qinghua Zheng. Graph few-shot learning with attribute matching. In *CIKM*, pages 1545–1554, 2020.

[Wang *et al.*, 2020b] Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys*, 53(3):1–34, 2020.

[Xian *et al.*, 2019] Yongqin Xian, Saurabh Sharma, Bernt Schiele, and Zeynep Akata. f-vaegan-d2: A feature generating framework for any-shot learning. In *SVPR*, pages 10275–10284, 2019.

[Yang *et al.*, 2021] Shuo Yang, Lu Liu, and Min Xu. Free lunch for few-shot learning: Distribution calibration. *arXiv preprint arXiv:2101.06395*, 2021.

[Zhou *et al.*, 2019] Fan Zhou, Chengtai Cao, Kunpeng Zhang, Goce Trajcevski, Ting Zhong, and Ji Geng. Meta-gnn: On few-shot node classification in graph meta-learning. In *CIKM*, pages 2357–2360, 2019.

[Zhou *et al.*, 2021] Jing Zhou, Yanan Zheng, Jie Tang, Jian Li, and Zhilin Yang. Flipda: Effective and robust data augmentation for few-shot learning. *arXiv preprint arXiv:2108.06332*, 2021.