# Beyond Laplacian Smoothing for Semi-supervised Community Detection

Guoguo Ai[1], Hui Yan[1,2(✉)], Jian Yang[2], and Xin Li[3]

[1] School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China
yanhui@njust.edu.cn
[2] The Key Laboratory of Intelligent Perception and Systems for High-Dimensional Information of Ministry of Education, Nanjing 210094, China
[3] School of Internet of Things, Nanjing University of Posts and Telecommunications, Nanjing 210003, China

**Abstract.** Graph Convolutional Networks (GCNs) have been proven to be effective in various graph-related tasks, such as community detection. Essentially graph convolution is simply a special form of Laplacian smoothing, acting as a low-pass filter that makes the features of nodes linked to each other similarly. For community detection, however, the similarity of intra-community nodes and the difference of inter-community nodes are equally vital. To bridge the gap between GCNs and community detection, we develop a novel Community-Centric Dual Filter (CCDF) framework for community detection. The central idea is that, besides of low-pass filter in GCN, we define network modularity enhanced high-pass filter to separate the discriminative signals from the raw features. In addition, we design a scheme to jointly optimize low-frequency and high-frequency information extraction on statistical modeling of Markov Random Fields. Extensive experiments demonstrate that the proposed CCDF model can consistently outperform or match state-of-the-art baselines in terms of semi-supervised community detection.

**Keywords:** Graph Convolutional Networks · Community detection · Modularity · Markov Random Field

## 1 Introduction

Many complex systems in various fields (e.g., social science, genetic science, and information science) are generally abstracted as networks, where nodes represent elements, and edges represent mutual interactions between elements in the system, so networks also can be called graphs. One of the significant properties of the network is community detection, which can help us discover objects with the same function in the system, study the relationship between different communities, and so on. Community detection has been successfully used in many applications, e.g., behavior prediction [16] and recommendation system [17].


H. Qiu et al. (Eds.): KSEM 2021, LNAI 12817, pp. 174–187, 2021.
https://doi.org/10.1007/978-3-030-82153-1_15

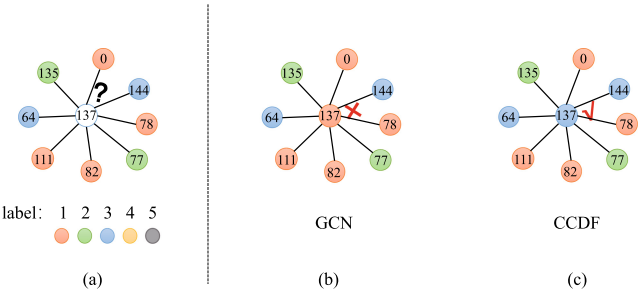



**Fig. 1.** Example of the wrongly-divided node in Texas

A large number of community detection algorithms based on various assumptions and techniques have been proposed, including modularity optimization [18], Markov dynamic algorithms [19], and GCN-based methods [5]. Among these methods, we would like to highlight GCNs, which leverage their representational power to provide state-of-the-art performance on community-finding tasks.

GCNs are constructed by stacking (graph) neural network layers, essentially recursively aggregate information from neighbors, which can be seen as a special form of Laplacian smoothing [11]. The smoothness of signals, i.e., low-frequency information, is the key to the success of graph neural networks (GNNs) [3,4,15].

However, off-the-shelf GCNs framework has two fundamental weaknesses which hinder their performance of community detection. Firstly, most of them seem to be tailor-made to work on assortative (homophilic) graphs [25], where nodes from the same community tend to form edges. In fact, nodes with distinct labels are more likely to link together in disassortative (heterophilic) networks [24]. If we force the representation of connected nodes to be similar by employing low-pass filter, obviously, those direct neighbors but belonging to the different communities inevitably tend to have a similar representation, leading to a blur inter-community distinction. Some researches [8] have further shown that exploring low-frequency signals is insufficient in different scenarios, like community detection. Secondly, it is well known that the node representation will become indistinguishable when we stack many GCN layers, causing over-smoothing [22]. These remind us that low-pass filter of current GCNs is far from optimal for real-world scenarios.

To remedy these two described weaknesses, we propose a dual graph filtering framework community-centric for community detection simply as CCDF. We first employ the theory of graph Laplacian and network modularity enhancement to formally define a high-pass filter to separate the high-frequency signals from the raw features, equipped to low-frequency signals extracted via GCN. Then we use Markov Random Fields (MRF) to integrate different types of signals adaptively. Figure 1 illustrates a sub-network in Texas, where each node's color denotes its community label. For the target node with id137 in Fig. 1(a), it is

erroneously assigned to community 1 by GCN (Fig. 1(b)). The mistake is that most of its neighbors belong to community 1, which directly affects the target node's predicted label. In comparison, CCDF correctly assigns the target node to its correct community 3 (Fig. 1(c)). This is because CCDF refines the coarse results from GCN by pulling nodes belonging to different communities to be far away from each other in the process of message passing.

The main contributions of this paper are summarized as follows:

- We develop a Community-Centric Dual Filter framework. We cast GCN as a Laplacian smoothing filter to obtain low-frequency information and design network modularity enhanced filter, which can be viewed as Laplacian sharpening as the counterpart of Laplacian smoothing obtain high-frequency information. The learned low-frequency and high-frequency information capture the similarity of intra-community nodes and the difference of inter-community nodes, respectively.
- We propose a novel frequency adaptation method in a complete end-to-end deep network framework. On statistical modeling of MRF, our method simultaneously and adaptively exploits discriminative information from inter-community and intra-community. So it can flexibly enlarge the distance between different communities, while most existing GNNs cannot.
- Extensive experiments have demonstrated that the proposed method can outperform representative baselines on benchmark datasets. We also show that network modularity enhanced filter can significantly boost the performance of community detection.

## 2 Preliminaries

### 2.1 Notations and Problem Definition

A non-directed graph represents as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, X)$, where $\mathcal{V} = \{v_1, v_2, \cdots, v_n\}$ consists of a set of nodes, and $\mathcal{E}$ is a set of edges between nodes. $X \in R^{n \times d}$ denotes the node attribute (feature) matrix, where $X_i$ is the $i$-th row of attribute matrix $X$ and $X_{ij}$ is the $j$-th dimensional attribute of vertex $v_i$, respectively. The topological structure of graph $\mathcal{G}$ is represented by an adjacency matrix $A$, where $A_{ij} = 1$, if $(v_i, v_j) \in \mathcal{E}$, or 0 otherwise. And we define $D$ as the diagonal degree matrix with $D_{ii} = \sum_j A_{ij}$. $\widetilde{A}$ and $\widetilde{D}$ stand for the adjacency matrix and diagonal degree matrix with added self-loops, respectively.

Given a graph $\mathcal{G}$, which partial nodes are labeled. The semi-supervised community detection task is then to label the rest unlabeled nodes in $\mathcal{G}$.

### 2.2 Graph Convolution Networks

Inspired by the successful applications of deep learning to the regular grid data (e.g., images and videos), researchers consider adopting the deep learning technique to process the irregular graph data (e.g. graphs or manifolds). GCNs generalize convolutional neural networks (CNNs) to graph-structured data. The

core operation in GCNs is graph propagation, in which information is propagated from each node to its neighborhoods with some deterministic propagation rules. Spectral approaches apply the convolution operation directly to the spectrum of the graph (i.e., the singular values of graph Laplacian) by treating the node attributes as signals in the graph according to the spectral graph theory. However, it suffers the high computational complexity of singular value decomposition (SVD). Then, ChebNet [9] approximates the spectral filter with Chebyshev polynomials to solve the issue of too complicated calculation. After, [2] proposes GCN via a localized first-order approximation to ChebNet, which simplifies ChebNet. Since then, many methods attempt to advance this architecture, including GAT [12] proposes learning the importance of different neighbors for the central node via a self-attention mechanism, GMNN [20] combines GNNs with probabilistic graphical models, and MixHop [21] employs the mixed-order propagation, etc.

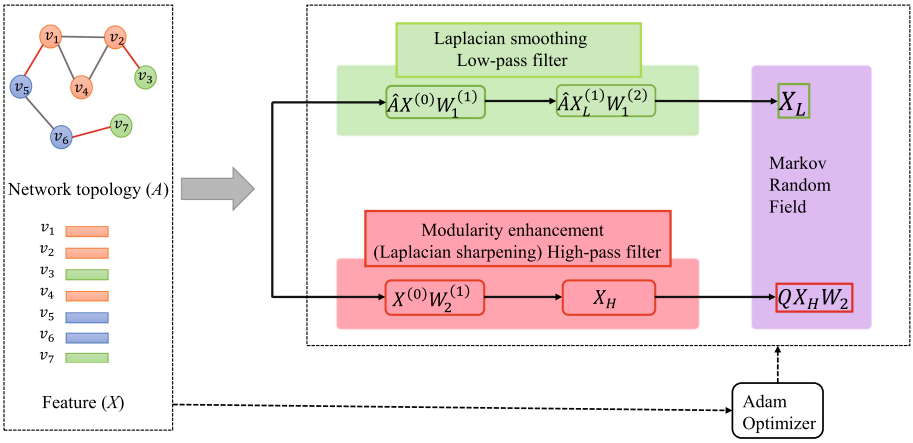



**Fig. 2.** Graphical representation of the proposed framework CCDF (Color figure online)

## 3 Our Proposed Model: CCDF

### 3.1 Overview

In this section, we design a novel Community-Centric Dual Filter (CCDF) framework for community detection which is shown in Fig. 2. We can observe that Fig. 2 consists of two parts: the left is a feature network with adjacency matrix $A$ and node feature matrix $X$, the renormalized adjacent matrix $\hat{A}$ of $A$ expressed as $\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}$, which are used as the inputs of the neural network along with X. The right is CCDF's three major components. In the first component (the

green box), the two convolutional layers of GCN can be interpreted as a special form of Laplacian smoothing that acts as low-pass filter to obtain low-frequency information retains the similarity of the intra-community nodes. The first layer is to learn a deep representation of the attributed network and the second layer is to derive node community membership. In addition, we design a network modularity enhancement that acts as high-pass filter to perform Laplacian sharpening, which learns high-frequency information captures the difference of inter-community nodes as the second component of CCDF (the red box). In the third component (the purple box), we leverage MRF to adaptively integrate the low-frequency and high-frequency information in a complete end-to-end deep network framework. The model CCDF is trained as a whole using the Adam optimizer [6].

### 3.2 Laplacian Smoothing

One of the key components in most GCN models is the graph convolutional layer, which can be described by:

$$X^{(m+1)} = \xi[(I - \widetilde{L})X^{(m)}W^{(m)}] \tag{1}$$

where $X^{(m+1)}$ is the output of the $(m+1)$-th layer, and $X^{(0)}$ is the input nodes' feature matrix $X$. $\xi(\cdot)$ is the nonlinear activation function Softmax or ReLU. $W^{(m)}$ is a trainable weight matrix of the $m$-th layer. The symmetric graph Laplacian $\widetilde{L} = I - \widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}$, where $I$ is the identity matrix. The first term $(I - \widetilde{L})$ is the given graph Laplacian without the parameter.

As shown in Eq. (1), GCN naturally smooths the features in the neighborhoods as each node aggregates information from neighbors. This characteristic is related to Laplacian smoothing. The following is an explanation of Laplacian smoothing.

Regarding $X \in R^{n\times d}$ as a signal defined on graph with normalized Laplacian matrix $\widetilde{L}$, the signal smoothness over the graph can be calculated as:

$$trace(X^T\widetilde{L}X) = \sum_{i,j,k} A_{ij}\left(\frac{X_{ik}}{\sqrt{D_{ii}+1}} - \frac{X_{jk}}{\sqrt{D_{jj}+1}}\right)^2 \tag{2}$$

A smaller value of $trace(X^T\widetilde{L}X)$ indicates smoother graph signal [26], i.e., smaller signal difference between adjacent nodes.

Suppose that we are given a noisy graph signal $X$, we can adopt the following objective function to recover the low-frequency signal:

$$\arg\min_{X^*} g(X^*) = ||X^* - X||^2 + \beta trace(X^{*T}\widetilde{L}X^*) \tag{3}$$

Note that if we set $\beta = 1/2$, one-step gradient descent optimization of $g(X^*)$ in Eq. (3) equals to GCN convolutional operator [7], shown in Eq. (1).

### 3.3 Network Modularity Enhancement

The low-pass filter in GCNs mainly retains the commonality of node features, which inevitably ignores the difference, so that the learned representations of connected nodes become similar, and always utilizing low-pass filter will lead to the over-smoothing problem [28]. To alleviate the issue, we introduce network modularity, proposed in [23], applied to measure the significance of community structures. We define network modularity enhanced graph convolutional, which makes each node's feature farther away from the centroid of its neighbors from different communities. The new modularity $Q$ of the community partition is expressed as:

$$Q(C) = \frac{1}{m} \sum_{i,j \in \mathcal{V}} (A_{ij} - \frac{\widetilde{D}_{ii}\widetilde{D}_{jj}}{2m})\delta(c_i, c_j) \tag{4}$$

where $C = (c_1, c_2, \ldots, c_n)$ be a partition of network $\mathcal{G}$, and $c_i$ denotes the community label to which node $i$ belongs to. $m$ is the number of edges in the network. $\delta(\cdot, \cdot)$ is an indicator function, which is $-1$, if $c_i \neq c_j$, or 0 otherwise. Each element of modularity $Q$ is shown as:

$$Q = \begin{cases} -(A_{ij} - \frac{(D_{ii}+1)(D_{jj}+1)}{2m}), & c_i \neq c_j \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

From Eq. (5), we can see the modularity $Q$ is defined as the difference between the number of edges with communities and the expected number of such edges over all pairs of nodes [27]. Unlike the definition of Kronecker delta in [23], $Q$ here focuses on the pairwise nodes from different communities, while [23] emphasizes community partition based label consistency. Assume $Q$ approaches the maximum, we have powerful high-frequency information hidden in the inter-community.

**Connection to Laplacian Sharpening**

Each element of Laplacian sharpening is described as:

$$I + \widetilde{L} = \begin{cases} -\frac{A_{ij}}{\sqrt{D_{ii}+1}\sqrt{D_{jj}+1}}, & i \neq j \\ 2, & \text{otherwise} \end{cases} \tag{6}$$

By comparing Eq. (5) with Eq. (6), we can learn that (1) $Q$ has a consistent meaning with Laplacian sharpening. $Q$ adopts the difference of both the degrees of nodes and edges as a discriminant criterion, while Laplacian sharpening adopts the quotient form. Therefore, network modularity enhanced graph convolutional can be viewed as a Laplacian sharpening as the counterpart of Laplacian smoothing. (2) $Q$ captures the distinctive features from different communities by virtue of the known labels ignored by Laplacian sharpening.

### 3.4 Aggregation

Attention mechanism [12] is the most used strategy to learn the importance of different representations from multi-views or sources. For example, we can use two learnable coefficients to measure the proportion of low-frequency and high-frequency signals and aggregate them [8]. However, it would introduce more parameters to learn.

Alternatively, we resort to MRF as our aggregation strategy [5]. The essential ingredient of MRF is an objective (or energy) function consisting of the sum of unary potentials defined as $\phi\{c_i\}$ and pairwise potentials defined as $\theta\{c_i, c_j\}$ :

$$E(C) = -\sum_i \phi\{c_i\} - \alpha \sum_{i,j} \theta\{c_i, c_j\} \tag{7}$$

where $\phi\{c_i\}$ for an individual $i$ measures the cost that it has label $c_i$ and $\theta\{c_i, c_j\}$ for $i$ and $j$ represents the cost that they have labels $c_i$ and $c_j$, respectively. $\alpha$ is a parameter for balancing the unary and pairwise potentials.

In this work, we substitute low-frequency information learned $X$ in Eq. (1) for the unary potential, and network modularity $Q$ in Eq. (5), the sum of pairwise potentials for all node pairs of a given network, finally, we get community detection oriented energy function $E(C|A, X)$:

$$E(C|A, X) = -\sum_i X_i - \alpha \sum_{i,j} Q_{ij} \tag{8}$$

There is the property that the minimum of the energy function corresponds to the best possible community partition [1]. However, minimizing the energy function to yield the most probable community partition for a given network is intractable. The pairwise potentials are defined over a complete network rather than the sparse network. Next, we will use a mean filed approximately for MRF's inference [5] to solve the issue from Eq. (8).

The Gibbs distribution of $E(C|A, X)$ is

$$P(C|A, X) = \frac{1}{N} \exp\{-\sum E(C|A, X)\} \tag{9}$$

where $N$ is a normalized constant.

The exact distribution $P(C|A, X)$ is difficult to compute. Thereby we replace the exact probability distribution with an approximate distribution $\hat{P}(C|A, X)$ which is decomposable, i.e.,

$$\hat{P}(C|A, X) = \prod \hat{P}_i(c_i|A, X) \tag{10}$$

**Lemma 1** [5]**.** *Given $P(C|A, X)$ in Eq. (9) and its approximation $\hat{P}(C|A, X)$ in Eq. (10), we can derive the update equation of $\hat{P}_i(c_i|A, X)$ by minimizing the KL-divergence $D(P||\hat{P})$, we can derive*

$$\hat{P}_i(C|A, X) \leftarrow \frac{1}{N_i} \exp\{X + QX\} \tag{11}$$

### 3.5 The Whole Architecture

Based on the update Eq. (11), we transform the MRF's inference into a convolution process and formulate it as convolution operations to finalize the last component of the integrated end-to-end deep neural network. Here, we formally define the whole framework of CCDF, which has four steps:

$$X_L^{(1)} = \mathrm{ReLU}(\hat{A}X^{(0)}W_1^{(1)}) \tag{12}$$

$$X_L = \mathrm{Softmax}(\hat{A}X_L^{(1)}W_1^{(2)}) \tag{13}$$

$$X_H = \mathrm{Softmax}(X^{(0)}W_2^{(1)}) \tag{14}$$

$$Z = \mathrm{Softmax}(X_L + QX_HW_2) \tag{15}$$

where $X_L^{(1)}$, $X_L$ and $Z$ are defined as the outputs of the first, second and third convolution layer, $X_H$ is a layer of nonlinear feature extraction. $W_1^{(1)}$, $W_1^{(2)}$, $W_2^{(1)}$ and $W_2$ are the corresponding weights, respectively.

CCDF can be trained by minimizing the cross entropy between the predicted and the (partial) ground-truth community labels under parameters $\theta = \{W_1^{(1)}, W_1^{(2)}, W_2^{(1)}, W_2\}$, i.e.,

$$\arg\min_{\theta} \mathscr{L}(Z, Y) = \arg\min_{\theta} \sum_{i \in V_l} \sum_{l=1}^{m} Y_{il} \ln Z_{il} \tag{16}$$

where $V_l$ is the set of labeled nodes and $m$ is the number of labels, $Y_{il}$ is 1 if node $v_i$ has label $l$, otherwise 0.

From Eq. (15), we can learn our method is to introduce a transformation of MRF model [5] and its inference to a convolutional layer to be added as the last layer of the whole framework.

**Placing CCDF in the Context of Related Prior Work**
It can be easily seen that GCN is a special case of our model. Specifically, when we ignore Eq. (14) and Eq. (15), the model becomes GCN. And in Eq. (15), we use the result of two-layer GCN ($X_L$) instead of nonlinear feature extraction ($X_H$) and then change the indicator function $\delta$. Meanwhile, add the similarity matrix that derives from attributed space and force the balance coefficient to be positive. This model becomes MRFasGCN (MasG) which has been proposed by [5].

Experimental results (Fig. 4) show that GCN obtains a relatively coarse community result for both our method and MasG [5]. Then MasG further reinforces similar or nearby nodes to have compatible community labels. It offers smooth labeling among nearby nodes by shortening the distance between nodes from the same communities. The difference is that our model refines coarsely labeled communities by enlarging the distance between nodes from different communities. Compared with them, we find that most of GCNs prefer to aggregate the low-frequency information, which makes them inadequate for learning

on disassortative graphs and suffers from over-smoothing seriously. Among the methods that differ from utilizing low-frequency information only, our method uses the MRF bridges the low-frequency and high-frequency information so able to integrate and train parameters in low-pass and high-pass extractors together to develop an end-to-end deep learning framework for community detection.

**Table 1.** The statistics of datasets

| Datasets | #Nodes | #Edges | #Attributes | #Communities |
|---|---|---|---|---|
| Cora | 2708 | 5429 | 1433 | 7 |
| Citeseer | 3327 | 4732 | 3703 | 6 |
| Pubmed | 19717 | 44338 | 500 | 3 |
| Texas | 183 | 328 | 1703 | 5 |
| Wisconsin | 262 | 530 | 1703 | 5 |
| Cornell | 195 | 304 | 1703 | 5 |
| Washington | 217 | 446 | 1703 | 5 |

# 4 Experiments

## 4.1 Datasets

We conduct experiments on seven widely-used benchmark datasets. We choose the commonly used citation graphs Cora, Citeseer and Pubmed for assortative datasets. We consider webpage graphs Texas, Wisconsin, Cornell, and Washington for disassortative datasets. We summarize the dataset statistics in Table 1.

**Table 2.** Comparison of prediction accuracy

| Method | Cora | Citeseer | Pubmed | Texas | Wisconsin | Cornell | Washington |
|---|---|---|---|---|---|---|---|
| ChebNet [9] | 81.20 | 69.80 | 74.40 | 64.13 | 53.43 | 38.77 | 47.70 |
| JKNet [10] | 80.20 | 68.70 | 78.00 | 64.21 | 55.72 | 51.02 | *62.38* |
| IncepGCN [13] | 77.60 | 69.30 | 77.70 | 63.04 | 54.96 | *54.08* | 60.55 |
| SGC [11] | 81.00 | 71.90 | 78.90 | 57.61 | *56.49* | 53.06 | 60.55 |
| GAT [12] | 83.00 | *72.50* | 79.00 | 64.18 | 48.85 | 47.96 | 46.79 |
| DropEdge [13] | 82.80 | 72.30 | 79.60 | 65.21 | 56.48 | 45.91 | 58.71 |
| ALaGCN [14] | 82.90 | 70.90 | 79.60 | *68.48* | *56.49* | 47.96 | 56.88 |
| GraphHeat [15] | *83.70* | *72.50* | **80.50** | 64.13 | 51.90 | 35.70 | 48.62 |
| GCN [2] | 81.50 | 70.30 | 79.00 | 63.04 | 54.96 | 47.96 | 56.88 |
| CCDF | **84.10** | **73.70** | *80.10* | **77.17** | **62.59** | **61.22** | **70.64** |

### 4.2 Experimental Setup

We aim to provide a rigorous and fair comparison between different methods on each dataset by using the same dataset splits and training procedure. For Cora, Citeseer and Pubmed, we use 20 nodes per class for training, 500 nodes for validation, and 1000 nodes for testing. There is no standard division for Texas, Wisconsin, Cornell and Washington. In order to verify the effectiveness and robustness, we use 20% for training, 30% for validation, and 50% for testing. We tune hyperparameters for all methods individually and the baseline results are essentially the same as their original reports. For CCDF, We tune the following hyper-parameters, in assortative datasets, the hyper-parameter set is: learning rate = 0.04, dropout = 0.8, weight decay in {2e−4, 6e−4, 4e−6}. In disassortative datasets, the hyper-parameter set is: learning rate = 0.05, dropout in {0.5, 0.6}, weight decay in {1e−3, 2e−3, 2e−4}. In training, we run 240 epochs, and the random seed is fixed. We adopt the widely-used Adam optimizer [6] and run experiments on Pytorch. Besides, we use Accuracy (ACC) as the metric to evaluate the performance of all methods. Our implementation is available online.[1]

### 4.3 Comparison with the Existing Methods

To comprehensively evaluate our method, we compare it with the following nine state-of-the-art semi-supervised methods, including ChebNet [9], JKNet [10], IncepGCN [13], SGC [11], GAT [12], DropEdge [13], ALaGCN [14], GraphHeat [15] and GCN [2].

As shown in Table 2, we can see our method CCDF improves upon GCN by a margin of 3.4%, 14.13% in Citeseer and Texas. CCDF improves 12.99% and 19.56% more accurately than GAT and SGC in Texas. In disassortative networks, CCDF much higher than other methods. Because nodes with distinct labels are more likely to link together in disassortative networks [24]. Only using low-pass filters is not suitable for disassortative networks. The experimental results show that our method CCDF performs the best on 8 of 9 methods, which demonstrates the effectiveness of CCDF.

### 4.4 Ablation Study

To validate the effectiveness of individual component in the proposed method CCDF, we compare CCDF with its five variants: 1) In our method, we ignore Eq. (14) and Eq. (15), which is equivalent to GCN. 2) The weight parameters $W_2$ in CCDF is set to 1 without training (as the most general way), namely CCDF without $W_2$. 3) $Q$ in Eq. (5) is replaced by community structure embedding matrix, $max\{\log\frac{\widetilde{A}_{ij}D}{d_i d_j} - \log\mathfrak{k}, 0\}(\mathfrak{k} = 2)$, defined in [22], namely CCDF with $C_{emb}$. 4) $Q$ is replaced by $I + D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$, which is the initial first-order Chebyshev filter derived in GCN, namely CCDF with $S_{1-order}$. 5) In order to compare

[1] https://github.com/KSEM2021/CCDF.

network modularity enhancement and Laplacian sharpening, we replace Eq. (5) with Eq. (6), when $c_i \neq c_j$, namely CCDF with $\hat{A}$. The last configuration is the full proposed method CCDF.

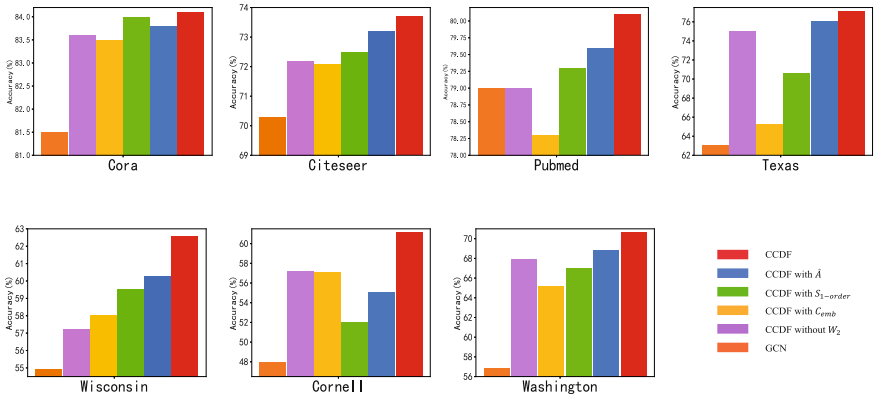



**Fig. 3.** Comparison of CCDF with its five variants

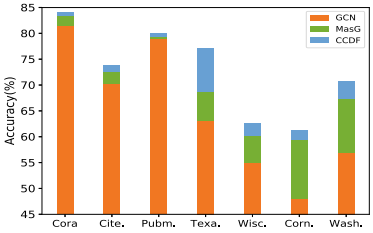


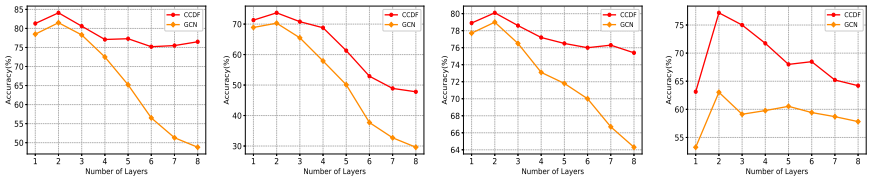

**Fig. 4.** Comparison experiment with GCN and MasG [5]

The results of the ablation study are shown in Fig. 3, from which we have the following two observations. First, all CCDF's variants with component changes witness clear performance drops compared to the full propose method, which indicates that the existence of $W_2$ and the $Q$ in Eq. (5) have a certain influence on our method. Second, the experimental results of CCDF and its variants are significantly higher than GCN on six of seven datasets.

In order to compare with GCN and [5]'s method (MasG) which is stated in Sect. 3.5, we add one comparative experiment between GCN, MasG, and our method. The results show in Fig. 4.

## 5 Alleviating Over-Smoothing Problem

To verify whether CCDF can alleviate the over-smoothing problem, we compare the performance of GCN and CCDF at different model depths in Cora, Citeseer, Pubmed, and Texas. The results show in Fig. 5. We can see that GCN achieves the best performance at two layers. As the number of layers increases, the performance of GCN drops rapidly, which indicates that GCN suffers from over-smoothing seriously. On the contrary, the results of CCDF are relatively stable and higher than GCN, which indicates that CCDF has the ability to suppress over-smoothing. CCDF contains both low-frequency and high-frequency information, which can keep node representations from becoming indistinguishable.




(a) Cora (b) Citeseer (c) Pubmed (d) Texas

**Fig. 5.** Test accuracy with different model depth

## 6 Conclusion

In this paper, we develop a Community-Centric Dual Filter (CCDF) framework for semi-supervised community detection task. CCDF can simultaneously and adaptively exploit low-frequency and high-frequency information from intra-community and inter-community, respectively. Furthermore, we leverage MRF to adaptively integrate the low-frequency and high-frequency information in a complete end-to-end deep network framework. Extensive experiments validate the effectiveness of our proposed method and demonstrate that CCDF can outperform or match baseline on various datasets.

**Acknowledgments.** This work was supported by the National Natural Science Foundation of China (No. 61773215, 61802206).

## References

1. Nowozin, S., Lampert, C.H.: Structured Learning and Prediction in Computer Vision. Now publishers Inc. (2011)
2. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)

3. Nt, H., Maehara, T.: Revisiting graph neural networks: all we have is low-pass filters. arXiv preprint arXiv:1905.09550 (2019)
4. Li, Q., Wu, X.M., Liu, H., et al.: Label efficient semi-supervised learning via graph filtering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9582–9591 (2019)
5. Jin, D., Liu, Z., Li, W., et al.: Graph convolutional networks meet Markov random fields: semi-supervised community detection in attribute networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, no. 01, pp. 152–159 (2019)
6. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
7. Jin, W., Derr, T., Wang, Y., et al.: Node similarity preserving graph convolutional networks. Proceedings of the 14th ACM International Conference on Web Search and Data Mining, pp. 148–156 (2021)
8. Bo, D., Wang, X., Shi, C., Shen, H.: Beyond low-frequency information in graph convolutional networks. arXiv preprint arXiv:2101.00797 (2021)
9. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. arXiv preprint arXiv:1606.09375 (2016)
10. Xu, K., Li, C., Tian, Y., et al.: Representation learning on graphs with jumping knowledge networks. In: International Conference on Machine Learning, pp. 5453–5462. PMLR (2018)
11. Wu, F., Souza, A., Zhang, T., et al.: Simplifying graph convolutional networks. International International Conference on Machine Learning, pp. 6861–6871. PMLR (2019)
12. Veličković, P., Cucurull, G., Casanova, A., et al.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
13. Rong, Y., Huang, W., Xu, T., et al.: DropEdge: towards deep graph convolutional networks on node classification. arXiv preprint arXiv:1907.10903 (2019)
14. Xie, Y., Li, S., Yang, C., et al.: When Do GNNs work: understanding and improving neighborhood aggregation. In: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI (2020)
15. Xu, B., Shen, H., Cao, Q., et al.: Graph convolutional networks using heat kernel for semi-supervised learning. arXiv preprint arXiv:2007.16002 (2020)
16. Yin, H., Hu, Z., Zhou, X., et al.: Discovering interpretable geosocial communities for user behavior prediction. In: IEEE 32nd International Conference on Data Engineering (ICDE), pp. 942–953 (2016)
17. Bernardes, D., Diaby, M., Fournier, R., et al.: A social formalism and survey for recommender systems. ACM SIGKDD Explor. Newsl. **16**(2), 20–37 (2015)
18. Blondel, V.D., Guillaume, J.L., Lambiotte, R., et al.: Fast unfolding of communities in large networks. J. Stat. Mech.: Theory Exp. P10008 (2008)
19. Jin, D., Yang, B., Baquero, C., et al.: A Markov random walk under constraint for discovering overlapping communities in complex networks. J. Stat. Mech.: Theory Exp. P05031 (2011)
20. Qu, M., Bengio, Y., Tang, J.: GMNN: Graph Markov neural networks. In: International Conference on Machine Learning. pp. 5241–5250. PMLR (2019)
21. Abu-El-Haija, S., Perozzi, B., Kapoor, A., et al.: MixHop: higher-order graph convolutional architectures via sparsified neighborhood mixing. In: International Conference on Machine Learning, pp. 21–29. PMLR (2019)
22. Li, Y., Sha, C., Huang, X., et al.: Community detection in attributed graphs: an embedding approach. In: Proceedings of the AAAI Conference on Artificial Intelligence (2018)

23. Newman, M.E., Girvan, M.: Finding and evaluating community structure in networks. Phys. Rev. E **69**(2), 026113 (2004)
24. Zhu, J., Yan, Y., Zhao, L., et al.: Generalizing graph neural networks beyond homophily. arXiv preprint arXiv:2006.11468 (2020)
25. Chien, E., Peng, J., Li, P., et al.: Adaptive universal generalized PageRank graph neural network. arXiv preprint arXiv:2006.07988 (2020)
26. Cui, G., Zhou, J., Yang, C., et al.: Adaptive graph encoder for attributed graph embedding. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2020)
27. Newman, M.E.: Modularity and community structure in networks. Proc. Natl. Acad. Sci. U.S.A. **103**(23), 8577–8582 (2006)
28. Oono, K., Suzuki, T.: Graph neural networks exponentially lose expressive power for node classification. arXiv preprint arXiv:1905.10947 (2019)