Data Mining and Knowledge Discovery (2023) 37:381–408
https://doi.org/10.1007/s10618-022-00890-9

Hierarchical message-passing graph neural networks

Zhiqiang Zhong[1] · **Cheng-Te Li**[2] · **Jun Pang**[1,3]

Received: 10 February 2022 / Accepted: 25 October 2022 / Published online: 17 November 2022

Abstract
Graph Neural Networks (GNNs) have become a prominent approach to machine learning with graphs and have been increasingly applied in a multitude of domains. Nevertheless, since most existing GNN models are based on *flat* message-passing mechanisms, two limitations need to be tackled: (i) they are costly in encoding long-range information spanning the graph structure; (ii) they are failing to encode features in the high-order neighbourhood in the graphs as they only perform information aggregation across the observed edges in the original graph. To deal with these two issues, we propose a novel *Hierarchical Message-passing Graph Neural Networks* framework. The key idea is generating a hierarchical structure that re-organises all nodes in a flat graph into multi-level super graphs, along with innovative intra- and inter-level propagation manners. The derived hierarchy creates shortcuts connecting far-away nodes so that informative long-range interactions can be efficiently accessed via message passing and incorporates meso- and macro-level semantics into the learned node representations. We present the first model to implement this framework, termed *Hierarchical Community-aware Graph Neural Network* (HC-GNN), with the assistance of a hierarchical community detection algorithm. The theoretical analysis illustrates HC-GNN's remarkable capacity in capturing long-range information without introducing heavy additional computation complexity. Empirical experiments conducted

Responsible editor: Albrecht Zimmermann and Peggy Cellier

✉ Cheng-Te Li
chengte@mail.ncku.edu.tw

✉ Jun Pang
jun.pang@uni.lu

Zhiqiang Zhong
zhiqiang.zhong@uni.lu

1 Faculty of Science, Technology and Medicine, University of Luxembourg, Esch-sur-Alzette, Luxembourg

2 Institute of Data Science and the Department of Statistics, National Cheng Kung University, Tainan, Taiwan

3 Interdisciplinary Centre for Security, Reliability and Trust, University of Luxembourg, Esch-sur-Alzette, Luxembourg

on 9 datasets under transductive, inductive, and few-shot settings exhibit that HC-GNN can outperform state-of-the-art GNN models in network analysis tasks, including node classification, link prediction, and community detection. Moreover, the model analysis further demonstrates HC-GNN's robustness facing graph sparsity and the flexibility in incorporating different GNN encoders.

Keywords Graph neural networks · Hierarchical message-passing · Long range communication · Hierarchical structure · Representation learning

1 Introduction

Graphs are a ubiquitous data structure that models objects and their relationships within complex systems, such as social networks, biological networks, recommendation systems, etc Wu et al. (2021). Learning node representation from a large graph has been proved as a useful approach for a wide variety of network analysis tasks, including link prediction Zhang and Chen (2018), node classification Zitnik et al. (2018) and community detection Chen et al. (2019).

Graph Neural Networks (GNNs) are currently one of the most promising paradigms to learn and exploit node representations due to their effective ability to encode node features and graph topology in transductive, inductive, and few-shot settings Zhang et al. (2020). Many existing GNN models follow a similar *flat* message-passing principle where information is iteratively passed between adjacent nodes along observed edges. Such a paradigm is able to incorporate local information surrounded by each node Gilmer et al. (2017). However, it has been proven to suffer from several drawbacks (Xu et al. 2019; Min et al. 2020; Li et al. 2020).

Among these deficiencies of flat message-passing GNNs, the limited ability for information aggregation over long-range has attracted significant attention Li et al. (2018), since most graph-related tasks require the interactions between nodes that are not directly connected Alon and Yahav (2021). That said, flat message-passing GNNs struggle in capturing dependencies between distant node pairs. Inspired by the outstanding effectiveness of very deep neural network models has been demonstrated in computer vision and natural language processing domains LeCun et al. (2015), a natural solution is stacking lots of GNN layers together to directly increase the receptive field of each node. Consequently, deeper models have been proposed by simplifying the aggregation design of GNNs and accompanied by well-designed normalisation units or specific gradient descent method (Chen et al. 2020; Gu et al. 2020). Nevertheless, Alon and Yahav have theoretically shown that flat GNNs are susceptible to being a *bottleneck* when aggregating messages across a long path and lead to severe *over-squashing* issues Alon and Yahav (2021).

On the other hand, in this paper, we further argue another crucial deficiency of flat message-passing GNNs is that they rely on only aggregating messages across the observed topological structure. The hierarchical semantics behind the graph structure provides useful information and should be incorporated into the learning of node representations. Taking the collaboration network in Fig. 1a as an example; author nodes highlighted in light yellow come from the same institutes, and nodes filled with

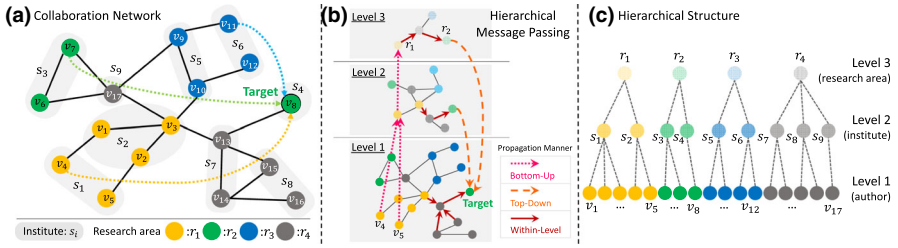

Fig. 1 Elaboration of the proposed hierarchical message passing: **a** a collaboration network, **b** an illustration of hierarchical message-passing mechanism based on (**a**) and (**c**), and **c** an example of the identified hierarchical structure

different colours indicate authors in various research areas. In order to generate the node representation of a given author, existing GNNs mainly capture the co-author level information depending on the explicit graph structure. However, information hidden at *meso* and *macro* levels is neglected. In the example of Fig. 1, meso-level information means authors belong to the same institutes and their connections to adjacent institutes. Macro-level information refers to authors of the same research areas and their relationship with related research areas. Both meso- and macro-level knowledge cannot be directly modelled through flat message passing via observed edges.

In this paper, we investigate the idea of a hierarchical message-passing mechanism to enhance the information aggregation pipeline of GNNs. The ultimate goal is to make the node representation learning process aware of both long-range interactive information and implicit multi-resolution semantics within the graph.

We note that a few graph pooling approaches have recently delivered various attempts to use the hierarchical structure idea (Gao and Ji 2019; Ying et al. 2018; Huang et al. 2019; Ranjan et al. 2020; Li et al. 2020). g- U- Net Gao and Ji (2019) and GXN Li et al. (2020) employ a bottom-up and top-down pooling operation; however, they do not allow long-range message-passing. DiffPool Ying et al. (2018), AttPool Huang et al. (2019) and ASAP Ranjan et al. (2020) target at graph classification tasks instead of enabling node representations to capture long-range dependencies and multi-grained semantics of one graph. Moreover, P-GNNs You et al. (2019) create a different information aggregation mechanism that utilises sampled anchor nodes to impose topological position information into learning node representations. While P-GNNs can capture global information, the hierarchical semantics mentioned above is still overlooked, and the global message-passing is not realised. Besides, the anchor-set sampling process is time-consuming for large graphs, and it cannot work well under the inductive setting.

Specifically, we present a novel framework, *Hierarchical Message-passing Graph Neural Networks* (HMGNNs), elaborated in Fig. 1. In detail, HMGNNs can be organised into the following four phases.

(i) Hierarchical structure generation. To overcome long-distance obstacles in the process of GNN message-passing, we propose to use a hierarchical structure to reduce

the size of graph $\mathcal{G}$ gradually, where nodes at each level t are integrated into different super nodes $(s_1^{t+1}, \ldots, s_n^{t+1})$ at each level $t+1$.

(ii) t-level super graph construction. In order to allow the message passing among generated same-level super nodes, we construct a super graph $\mathcal{G}_t$ based on the connections between nodes at its lower level $t-1$.

(iii) Hierarchical message propagation. With the generated hierarchical structure for a given graph, we develop three propagation manners, including bottom-up, within-level and top-down.

(iv) Model learning. Last, we leverage task-specific loss functions and a gradient descent procedure to train the model.

Designing a feasible hierarchical structure is crucial for HMGNNs, as the hierarchical structure determines how messages can be passed through different levels and what kind of meso- and macro-level information to be encoded in node representations. In this paper, we consider (but are not restricted to) *network communities*. As a natural graph property, the community has been proved very useful for many graph mining tasks (Wang et al. 2014, 2017). Lots of community detection methods can generate hierarchical community structures. Here, we propose an implementation model for the proposed framework, *Hierarchical Community-aware Graph Neural Network* (HC-GNN). HC-GNN exploits a well-known hierarchical community detection method, i.e., the *Louvain* method Blondel et al. (2008) to build up the hierarchical structure, which is then used for the hierarchical message-passing mechanism.

The theoretical analysis illustrates HC-GNN's remarkable capacity in capturing long-range information without introducing heavy additional computation complexity. Extensive empirical experiments are conducted on 9 graph datasets to reveal the performance of HC-GNN on a variety of tasks, i.e., link prediction, node classification, and community detection, under transductive, inductive and few-shot settings. The results show that HC-GNN consistently outperforms a set of state-of-the-art approaches for link prediction and node classification. In the few-shot learning setting, where only 5 samples of each label are used to train the model, HC-GNN achieves a significant performance improvement, up to 16.4%. We also deliver a few empirical insights: (a) the lowest level contributes most to node representations; (b) how to generate the hierarchical structure has a significant impact on the quality of node representations; (c) HC-GNN maintains an outstanding performance for graphs with different levels of sparsity perturbation; (d) HC-GNN possess significant flexibility in incorporating different GNN encoders, which means HC-GNN can achieve superior performance with advanced flat GNN encoders.

Contributions The contribution of this paper is five-fold:

1. We propose a novel *Hierarchical Message-passing Graph Neural Networks* framework, which allows nodes to conveniently capture informative long-range interactions and encode multi-grained semantics hidden behind the given graph.
2. We present the first implementation of our framework, namely HC-GNN[1], by detecting and utilising hierarchical community structures for message passing.

[1] Code and data are available at https://github.com/zhiqiangzhongddu/HC-GNN.

3. Theoretical analysis demonstrate the efficiency and the capacity of HC-GNN in capturing long-range interactions in graphs.
4. Experimental results show that HC-GNN significantly outperforms competing GNN methods on several prediction tasks under transductive, inductive, and few-shot settings.
5. Further empirical analysis is conducted to derive insights into the impact of the hierarchical structure and graph sparsity on HC-GNN and confirm its flexibility in incorporating different GNN encoders.

The rest of this paper is organised as follows. We begin by briefly reviewing additional related work in Sect. 2. Then in Sect. 3, we introduce the preliminaries of this study and state the research problem. In Sect. 4, we introduce our proposed framework *Hierarchical Message-passing Graph Neural Networks* and its first implementation, HC-GNN. Experimental results and empirical analysis are shown in Sect. 5. Finally, we conclude the paper and discuss the future work in Sect. 6.

2 Related work

Flat message-passing GNNs They perform graph convolution, directly aggregate node features from neighbours in the given graph, and stack multiple GNN layers to capture long-range node dependencies (Kipf and Welling 2017; Hamilton et al. 2017; Velickovic et al. 2018; Xu et al. 2019). However, they were observed *not* to benefit from more than a few layers, and recent studies have theoretically expressed this problem as *over-smoothing* (Li et al. 2018; Alon and Yahav 2021), i.e., node representations become indistinguishable when the number of GNN layers increases. On the other hand, GraphRNA (Huang et al. 2019) presents graph recurrent networks to capture interactions between far-away nodes. Still, we cannot apply it to inductive learning settings because they rely on attributed random walks and the recurrent aggregations introduce high computation costs. P-GNNs (You et al. 2019) incorporate a novel global information aggregation mechanism based on the distance of a given target node to each anchor set. However, P-GNNs sacrifice the ability of existing GNNs on inductive node-wise tasks. As shown in their paper, they only support pairwise node classification tasks, i.e., comparing if two nodes have the same class label instead of predicting the class label of each individual node. Additionally, the anchor-set sampling operation brings a high computational cost for large-size graphs. Recently, deeper *flat* GNNs have been proposed by simplifying the aggregation design of GNNs and accompanied by well-designed normalisation units (Chen et al. 2020) or specific gradient descent methods (Gu et al. 2020). Nevertheless, Alon and Yahav (2021) has theoretically shown that flat GNNs are susceptible to being a *bottleneck* when aggregating messages across a long path and lead to severe *over-squashing* issues. Moreover, we will theoretically discuss the advantages of our method compared with flat GNNs in Sect. 4.3, in terms of long-range interactive capability and complexity.

Hierarchical representation GNNs In recent years, some studies generalise the pooling mechanism of computer vision Ronneberger et al. (2015) to GNNs for graph representation learning (Ying et al. 2018; Huang et al. 2019; Gao and Ji 2019; Ranjan

et al. 2020; Li et al. 2020, ?; Rampásek and Wolf 2021). However, most of them, such as DIFFPOOL Ying et al. (2018), ATTPOOL Huang et al. (2019) and ASAP Ranjan et al. (2020), are designed for graph classification tasks rather than learning node representations to capture long-range dependencies and multi-resolution semantics. Thus they cannot be directly applied to node-level tasks. G- U- NET Gao and Ji (2019) defines a similarity-based pooling operator to construct the hierarchical structure, and GXN Li et al. (2020) designs another infomax pooling operator, they implement bottom-up and top-down operations. Despite the success of G- U- NET and GXN in producing graph-level representations, they cannot model the multi-grained semantics and realise long-range message-passing. HARP Chen et al. (2018) and LouvainNE Bhowmick et al. (2020) are two unsupervised network representation approaches that adopt a hierarchical structure, but they do not support the supervised training paradigm to optimise for specific tasks, and they cannot be applied with inductive settings.

More recently, HGNet Rampásek and Wolf (2021) leverages multi-resolution representations of a graph to facilitate capturing long-range interactions. Below, we discuss the main differences between HGNet and HC-GNN. HC-GNN designs different efficient and effective bottom-up and top-down propagation mechanisms to realise elegant hierarchical message-passing rather than directly applying pooling and relational GCN, respectively. We further provide the theoretical analysis to demonstrate the efficiency and capacity of HC-GNN, such analysis has not been performed on HGNet. We also provide a much more careful and comprehensive set of experimental studies to validate the effectiveness of HC-GNN, including comparing learning settings on node classification (transductive, inductive, and few-sot), comparing to more recent competing flat GNN methods, comparing to state-of-the-art hierarchical GNN models, evaluating on the link prediction task, and in-depth analysis on graph sparsity and primary GNN encoders (Sect. 5). Last but not least, in addition to capturing long-range interactions, we further deeply discuss the benefits and the usefulness of the hidden hierarchical structure in a graph.

Table 8 summarises the critical advantages of the proposed HC-GNN and compares it with a number of state-of-the-art methods published recently. We are the first to present the hierarchical message passing to efficiently model long-range informative interaction and multi-grained semantics. In addition, our HC-GNN can utilise the community structures and be applied for transductive, inductive and few-shot inferences.

3 Problem statement

An attributed graph with n nodes can be represented as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, where $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ is the node set, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ denotes the set of edges, and $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \in \mathbb{R}^{n \times \pi}$ is the feature matrix, in which each vector $\mathbf{x}_i \in \mathbf{X}$ is the feature vector associated with node v_i, and π is the dimension of input feature vector of each node. For subsequent discussion, we summarise $\mathcal{V}$ and $\mathcal{E}$ into an adjacency matrix $\mathbf{A} \in \{0, 1\}^{n \times n}$.

Table 1 Summary of main notations

Notation	Description
$\mathcal{G}$	An attributed graph
$\mathcal{V}, \mathcal{E}$	The set of nodes and edges on $\mathcal{G}$, respectively
$\mathbf{A}$	The adjacent matrix of $\mathcal{G}$
$\mathbf{X} \in \mathbb{R}^{n \times \pi}$	The matrix of node features
d	The pre-defined representation dimension
$\mathbf{H} \in \mathbb{R}^{n \times d}$	The hidden node representation matrix
$\mathbf{h}_v \in \mathbb{R}^d$	The hidden node representation of node v
$\mathbf{Z} \in \mathbb{R}^{n \times d}$	The final node representation matrix
$\mathbf{z}_v \in \mathbb{R}^d$	The final node representation of node v
L	The number of layers of *within-level propagation* GNN encoder
T	The number of hierarchy levels
$\mathcal{G}_t$	The super graph at level t
s_n^t	The n-th super node of $\mathcal{G}_t$ at level t
$\mathcal{H}$	The set of constructed super graphs
$\mathcal{N}(v)$	The set of neighbour nodes of node v
γ	A hyper-parameter that used to construct super graph $\mathcal{G}_t$
λ	The pooling ratio

Problem definition Given a graph $\mathcal{G}$ and a pre-defined representation dimension d, the goal is to learn a mapping function $f : \mathcal{G} \rightarrow \mathbf{Z}$, where $\mathbf{Z} \in \mathbb{R}^{n \times d}$ and each row $\mathbf{z}_i \in \mathbf{Z}$ corresponds to the node v_i's representation. The effectiveness of f is evaluated by applying $\mathbf{Z}$ to different tasks, including node classification, link prediction, and community detection. Table 1 lists the mathematical notation used in the paper.

Flat node representation learning Prior to introducing the hierarchical message-passing mechanism, we first give a general review of existing Graph Neural Networks (GNNs) with *flat* message-passing. Let $\hat{\mathbf{A}} = (\hat{\mathbf{A}}_{uv})_{u,v \in \mathcal{V}}$, where $\hat{\mathbf{A}}_{uv}$ is a normalised value of $\mathbf{A}_{uv}$. Thus, we can formally define ℓ-th layer of a flat GNN as:

$$\begin{aligned} \mathbf{m}_a^{(\ell)} &= \textsc{Aggregate}^N(\{\hat{\mathbf{A}}_{uv},\ \mathbf{h}_u^{(\ell-1)} \mid u \in \mathcal{N}(v)\}), \\ \mathbf{m}_v^{(\ell)} &= \textsc{Aggregate}^I(\{\hat{\mathbf{A}}_{uv} \mid u \in \mathcal{N}(v)\})\, \mathbf{h}_v^{(\ell-1)}, \\ \mathbf{h}_v^{(\ell)} &= \textsc{Combine}(\mathbf{m}_a^{(\ell)}, \mathbf{m}_v^{(\ell)}) \end{aligned} \tag{1}$$

where $\textsc{Aggregate}^N(\cdot)$ and $\textsc{Aggregate}^I(\cdot)$ are two possibly differential parameterised functions. $\mathbf{m}_a^{(\ell)}$ is aggregated message from node v's neighbourhood nodes ($\mathcal{N}(v)$) with their structural coefficients, and $\mathbf{m}_v^{(\ell)}$ is the residual message from node v after performing an adjustment operation to account for structural effects from its neighbourhood nodes. After, $\mathbf{h}_v^{(\ell)}$ is the learned as representation vector of node v by with combining $\mathbf{m}_a^{(\ell)}$ and $\mathbf{m}_v^{(\ell)}$, termed as $\textsc{Combine}(\cdot)$, at the ℓ-th iteration/layer.

Note that, we initialise $\mathbf{h}_v^{(0)} = \mathbf{x}_v$ and the final learned representation vector after L iterations/layers $\mathbf{z}_v = \mathbf{h}_v^{(L)}$.

Take the classic Graph Convolutional Network (GCN) Kipf and Welling (2017) as an example, which applies two normalised mean aggregations to aggregate feature vectors node v's neighbourhood nodes $\mathcal{N}(v)$ and combine with itself:

$$\mathbf{h}_v^{(\ell)} = \text{ReLU}\left(\sum_{u \in \mathcal{N}(v)} \frac{\mathbf{W}^{(\ell)}\mathbf{h}_u^{(\ell-1)}}{\sqrt{|\mathcal{N}(u)||\mathcal{N}(v)|}} + \frac{\mathbf{W}^{(\ell)}\mathbf{h}_v^{(\ell-1)}}{\sqrt{|\mathcal{N}(v)||\mathcal{N}(v)|}}\right) \tag{2}$$

where $\sqrt{|\mathcal{N}(u)||\mathcal{N}(v)|}$ is a constant normalisation coefficient for the edge $\mathcal{E}_{uv}$, which is calculated from the normalised adjacent matrix $\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-\mathbf{1/2}}$. $\mathbf{D}$ is the diagonal node degree matrix of $\mathbf{A}$. $\mathbf{W}^{(\ell)} \in \mathbb{R}^{n \times d}$ is a trainable weight matrix of layer ℓ. From Eqs. 1 and 2, we can find that existing GNNs iteratively pass messages between adjacent nodes along observed edges, which will lead to two significant limitations: (a) the limited ability for information aggregation over long-range. They need to stack k layers to capture interactions within k steps for each node; (b) they are infeasible in encoding meso- and macro-level graph semantics.

4 Proposed approach

We propose a framework, *Hierarchical Message-passing Graph Neural Networks* (HMGNNs), whose core idea is to use a hierarchical message-passing structure to enable node representations to receive long-range messages and multi-grained semantics from different levels. Fig. 2 provides an overview of the proposed framework, consisting of four components. First, we create a hierarchical structure to coarsen the input graph $\mathcal{G}$ gradually. Nodes at each level t of the hierarchy are grouped into different super nodes $(s_1^t, \ldots, s_n^t)$. Second, we further organise level t generated super nodes into a super graph $\mathcal{G}_{t+1}$ at level $t+1$ based on the connections between nodes at level t, in order to enable message-passing that encodes the interactions between generated super nodes. Third, we develop three different propagation schemes to propagate messages among nodes within the same level and across different levels. At last, after obtaining node representations, we use the task-specific loss function and a gradient descent procedure to train the model.

4.1 Hierarchical message-passing GNNs

I. Hierarchical structure generation Nodes $\mathcal{V}$ of a graph $\mathcal{G}$ can be naturally organised by super node structures of T different levels, i.e., $\{\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_T\}$, in which densely inter-connected nodes of $\mathcal{V}_{t-1}$ $(2 \leq t \leq T)$ are grouped into a super node of $\mathcal{V}_t$. For example in Fig. 1a, author set $\mathcal{V}_1 = \{v_1, v_2, \ldots, v_{17}\}$ can be grouped into different super nodes $\mathcal{V}_2 = \{s_1, s_2, \ldots, s_9\}$ based on their institutes. Institutes can be further grouped into higher-level super nodes $\mathcal{V}_3 = \{r_1, r_2, \ldots, r_4\}$ according to research areas. Meanwhile, there is a relationship between nodes at different levels, as indicated

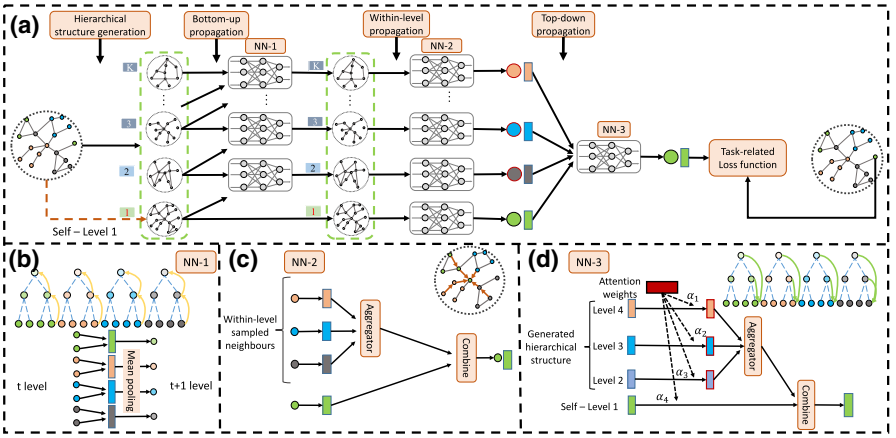

Fig. 2 **a** The architecture of *Hierarchical Message-passing Graph Neural Networks*: we first generate a hierarchical structure, in which each level is formed as a super graph, use the level t graph to update nodes of level $t+1$ graph (bottom-up propagation), apply the typical neighbour aggregation on each level's graph (within-level propagation), use the generated node representations from level $2 \leq t \leq T$ to update node representations at the level 1 (top-down propagation), and optimises the model via a task-specific loss. **b** NN-1: bottom-up propagation. **c** NN-2: within-level propagation. **d** NN-3: top-down propagation

by dashed lines in Fig. 1c. Hence, we can generate a hierarchical structure to depict the inter- and intra-relationships among authors, institutes, and research areas. We will discuss how to implement the hierarchical structure generation in Sect. 4.2.

II. t-Level super graph construction The level t's super graph $\mathcal{G}_t$ is constructed based on level $t-1$ graph $\mathcal{G}_{t-1}$ ($t \geq 2$), where $\mathcal{G}_1$ represents the original graph $\mathcal{G}$. Given nodes at level $t-1$, i.e., $\mathcal{V}_{t-1} = \{s_1^{t-1}, \ldots, s_m^{t-1}\}$, densely inter-connected nodes of $\mathcal{V}_{t-1}$ are grouped into a super node of $\mathcal{V}_t$ according to Sect. 4.1-I. We further create an edge between two super nodes s_i^t and s_j^t if there exist more than γ edges in $\mathcal{G}_{t-1}$ connecting elements in s_i^t and elements in s_j^t, where γ is a hyper-parameter and $\gamma = 1$ by default. In this way, we can have an alternative representation of the hierarchical structure as a list of (super) graphs $\mathcal{H} = \{\mathcal{G}_1, \ldots, \mathcal{G}_T\}$, where $\mathcal{G}_1 = \mathcal{G}$. Moreover, inter-level edges are created to depict the relationships between (super) nodes at different levels t and $t-1$, if a level $t-1$ node has a corresponding super node at level t, see for example Fig. 1c. We initialise the feature vectors of generated super nodes to be zero vectors with the same length as the original node feature vector $\mathbf{x}_i$. Taking the collaboration network in Fig. 1 as an example, at the micro-level (level 1), we have authors and their co-authorship relations; at the meso-level (level 2), we organise authors according to their affiliations and establish relations between institutes; at the macro-level (level 3), institutes are further grouped according to their research areas, and we have the relations among the research areas. In addition, inter-level links are also created to depict the relationships between authors and institutes and between institutes and research areas.

III. Hierarchical message propagation The hierarchical message-passing mechanism works as a supplementary process to enhance the node representations with

long-range interactions and multi-grained semantics. Thus it does not change the flat node representation learning process as described in Sect. 3, to ensure the local information is well maintained. And we adopt the classic GCN, as described in Eq. 2, as our default flat GNN encoder throughout the paper. Particularly, the hierarchical message-passing mechanism consists of ℓ-th layer consisting of 3 steps.

1. *Bottom-up propagation.* After obtaining node representations ($\mathbf{h}_{s^{t-1}}^{(\ell)}$) of $\mathcal{G}_{t-1}$ with ℓ-th flat information aggregation, we perform bottom-up propagation, i.e., NN-1 in Fig. 2b, using node representations in $\mathcal{G}_{t-1}$ to update node representations in $\mathcal{G}_t$ ($t \geq 2$) in the hierarchy $\mathcal{H}$, as follows:

$$\mathbf{a}_{s_i^t}^{(\ell)} = \frac{1}{|s_i^t| + 1} \left(\sum_{s^{t-1} \in s_i^t} \mathbf{h}_{s^{t-1}}^{(\ell)} + \mathbf{h}_{s_i^t}^{(\ell-1)} \right) \tag{3}$$

where s_i^t is a super node in $\mathcal{G}_t$, and s^{t-1} is a node in $\mathcal{G}_{t-1}$ that belongs to s_i^t in $\mathcal{G}_t$. $\mathbf{h}_{s_i^t}^{(\ell-1)}$ is the node representation of s_i^t that generated by layer $\ell-1$ in graph $\mathcal{G}_t$, $|s_i^t|$ is the number of nodes of level $t-1$ that belonging to super node s_i^t, and $\mathbf{a}_{s_i^t}^{(\ell)}$ is the updated representation of s_i^t.

2. *Within-level propagation.* We explore the typical *flat* GNN encoders (Kipf and Welling 2017; Hamilton et al. 2017; Velickovic et al. 2018; Xu et al. 2019; Chen et al. 2020) to propagate information within each level's graph $\{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_T\}$, i.e., NN-2 in Fig. 2c. The aim is to aggregate neighbours' information and update within-level node representations. Specifically, the information aggregation at level t is depicted as follows:

$$\begin{aligned} \mathbf{m}_a^{(\ell)} &= \textsc{Aggregate}^N(\{\hat{\mathbf{A}}_{uv}^t,\ \mathbf{a}_u^{(\ell)} \mid u \in \mathcal{N}^t(v)\}), \\ \mathbf{m}_v^{(\ell)} &= \textsc{Aggregate}^I(\{\hat{\mathbf{A}}_{uv}^t \mid u \in \mathcal{N}^t(v)\})\, \mathbf{a}_v^{(\ell)}, \\ \mathbf{b}_v^{(\ell)} &= \textsc{Combine}(\mathbf{m}_a^{(\ell)}, \mathbf{m}_v^{(\ell)}) \end{aligned} \tag{4}$$

where $\mathbf{a}_u^{(\ell)}$ is the node representation of u after bottom-up propagation at the ℓ-th layer, $\mathcal{N}^t(v)$ is a set of nodes adjacent to v at level t, and $\mathbf{b}_v^{(\ell)}$ is the aggregated node representation of v based on local neighbourhood information. Note that we adopt the classic GCN, as described in Eq. 2, as our default GNN encoder throughout the paper. We will discuss the possibility of incorporating with other advanced GNN encoders in Sect. 5.3.

3. *Top-down propagation.* The top-down propagation is illustrated by NN-3 in Fig. 2d. We use node representations in $\{\mathcal{G}_2, \ldots, \mathcal{G}_T\}$ to update the representations of original nodes in $\mathcal{G}$. The importance of messages at different levels can be different for other tasks. Hence, we adopt the attention mechanism Velickovic et al. (2018) to adaptively learn the contribution weights of different levels during top-down integration, given by:

$$\mathbf{h}_v^{(\ell)} = \text{ReLU}(\mathbf{W} \cdot \text{MEAN}\{\alpha_{uv} \mathbf{b}_u^{(\ell)}\}), \forall u \in \mathcal{C}(v) \cup \{v\} \tag{5}$$

where α_{uv} is a trainable normalised attention coefficient between node v to super node u or itself, MEAN is an element-wise mean operation, $\mathcal{C}(v)$ denotes the set of different-level super nodes from level $\{2, \ldots, K\}$ that node v belongs to ($|\mathcal{C}(v)| = K - 1$), and ReLU is the activation function. $\mathbf{H}^{(\ell)}$ is the generated node representation of layer ℓ with $\mathbf{h}_v^{(\ell)} \in \mathbf{H}^{(\ell)}$. We generate the output node representations of the last layer (L) via:

$$\mathbf{z}_v = \sigma(\mathbf{W} \cdot \text{MEAN}\{\alpha_{uv}\mathbf{b}_u^{(L)}\}), \forall u \in \mathcal{C}(v) \cup \{v\} \tag{6}$$

where σ is the Euclidean normalisation function to reshape values into $[0, 1]$. $\mathbf{Z} \in \mathbb{R}^{n \times d}$ is the final generated node representation with each row vector $\mathbf{z}_v \in \mathbf{Z}$.

IV. Model learning The proposed HMGNNs could be trained in unsupervised, semi-supervised, or supervised settings. Here, we only discuss the supervised setting used for node classification in our experiments. We define the loss function based on cross entropy, as follows:

$$\mathcal{L} = -\sum_{v \in \mathcal{V}} \mathbf{y}_v^\top \log(\text{Softmax}(\mathbf{z}_v)) \tag{7}$$

where $\mathbf{y}_v$ is a one-hot vector denoting the label of node v. We allow $\mathcal{L}$ to be customised for other task-specific objective functions, e.g., the negative log-likelihood loss Velickovic et al. (2018).

Algorithm 1 *Hierarchical Message-passing Graph Neural Networks*

Input: Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$
Output: Node representations $\mathbf{Z} \in \mathbb{R}^{n \times d}$

$\mathbf{h}_v^{(0)} \leftarrow \mathbf{x}_v$
Generate hierarchical structure: $\mathcal{H} = \{\mathcal{G}_t \mid t = 1, 2, \ldots, T\}$
for $\ell \leftarrow \{1, 2, \ldots, L\}$ **do**
  $\mathbf{h}_v^{(\ell)} = \text{ReLU}(\sum_{u \in \mathcal{N}(v)} \frac{\mathbf{W}^{(\ell)}\mathbf{h}_u^{(\ell-1)}}{\sqrt{|\mathcal{N}(u)||\mathcal{N}(v)|}} + \frac{\mathbf{W}^{(\ell)}\mathbf{h}_v^{(\ell-1)}}{\sqrt{|\mathcal{N}(v)||\mathcal{N}(v)|}}), \forall v \in \mathcal{G}$
  for $t \leftarrow \{2, \ldots, T\}$ **do**
    $\mathbf{a}_{s_i^t}^{(\ell)} = \frac{1}{|s_i^t|+1}\left(\sum_{s^{t-1} \in s_i^t} \mathbf{h}_{s^{t-1}}^{(\ell)} + \mathbf{h}_{s_i^t}^{(\ell-1)}\right), \forall s_i^t \in \mathcal{G}_t$
    $\mathbf{b}_v^{(\ell)} = \text{ReLU}(\sum_{u \in \mathcal{N}(v)} \frac{\mathbf{W}^{(\ell)}\mathbf{a}_u^{(\ell)}}{\sqrt{|\mathcal{N}(u)||\mathcal{N}(v)|}} + \frac{\mathbf{W}^{(\ell)}\mathbf{a}_v^{(\ell)}}{\sqrt{|\mathcal{N}(v)||\mathcal{N}(v)|}}), \forall v \in \mathcal{G}_t$
  end for
  for $v \in \mathcal{G}$ **do**
    if $\ell < L$ **then**
      $\mathbf{h}_v^{(\ell)} = \text{ReLU}(\mathbf{W} \cdot \text{MEAN}\{\alpha_{uv}\mathbf{b}_u^{(\ell)}\}), \forall u \in \mathcal{C}(v) \cup \{v\}$
    else
      $\mathbf{z}_v = \sigma(\mathbf{W} \cdot \text{MEAN}\{\alpha_{uv}\mathbf{b}_u^{(L)}\}), \forall u \in \mathcal{C}(v) \cup \{v\}$
    end if
  end for
end for

We summarise the process of *Hierarchical Message-passing Graph Neural Networks* in Algorithm 1. Given a graph $\mathcal{G}$, we first generate the hierarchical structure and combine it with the original graph $\mathcal{G}$, to obtain $\mathcal{H} = \{\mathcal{G}_t \mid t = 1, 2, \ldots, T\}$, where

$\mathcal{G}_1 = \mathcal{G}$ (line 2). For each node, including original and generated super nodes, in each NN layer, we perform three primary operations in order: (1) bottom-up propagation (line 6), (2) within-level propagation (line 7), and (3) top-down propagation (line $9-15$). After getting the representation vector of each node that is enhanced with informative long-range interactions and multi-grained semantics, and we train the model with the loss function $\mathcal{L}$ in Eq. 7.

4.2 Hierarchical community-aware GNN

Identifying hierarchical super nodes for the proposed HMGNNs is the most crucial step as it determines how the information will be propagated within and between levels. We consider *hierarchical network communities* to construct the hierarchy. The network community has been proved helpful for assisting typical network analysis tasks, including node classification (Wang et al. 2014, 2017) and link prediction (Sun and Han 2012; Rossetti et al. 2015). Taking the algorithm efficiency into account and avoiding introducing additional hyper-parameters, i.e., the number of hierarchy levels, we adopt the well-known *Louvain* algorithm Blondel et al. (2008) to build the first implementation of HMGNNs, termed as *Hierarchical Community-aware Graph Neural Network* (HC-GNN). The *Louvain* algorithm returns us a hierarchical structure as described in Sect. 4.1 without the need for a pre-defined number of hierarchies, based on which we can learn node representations involving long-range interactive information and multi-grained semantics. Due to page limit, we include more details about community detection algorithms in App. A.

4.3 Theoretical analysis and model comparison

Long-range interactive capability We now theoretically analyse the asymptotic complexity of different GNN models to capture long-range interaction. We first analyse flat GNN models, that they need to stack $\mathcal{O}(\text{diam}(\mathcal{G}))$ layers to ensure the communication between any pair of nodes in $\mathcal{G}$. For HMGNNs, let us assume the pooling ratio $\lambda = |\mathcal{V}_{t+1}|/|\mathcal{V}_t|$. Thus, the potentially total number of nodes in HMGNNs over $\mathcal{G}$ with n nodes is $\sum_{t=1}^{\infty} n\lambda^t = \mathcal{O}(n)$, while the number of possible levels is $\log_{\lambda^{-1}} n = \mathcal{O}(\log n)$. That said, the shortest path between any two nodes of $\mathcal{G}$ is upper-bounded by $\mathcal{O}(\log n)$. Compared to $\mathcal{O}(\text{diam}(\mathcal{G}))$ with flat GNNs, HMGNNs leads to significant improvement over the capability in capturing long-range interactions.

Model complexity For the vanilla flat GNN model, i.e., GCN, its computational complexity of one layer is $\mathcal{O}(n^3)$ Kipf and Welling (2017), and the computational complexity of a GCN model contains ℓ is $\mathcal{O}(\ell n^3)$. For another attention-enhanced flat GNN model, i.e., Graph Attention Network (GAT) Velickovic et al. (2018), except for the same convolutional operation as GCN, the additional masked attention over all nodes requires $\mathcal{O}(\ell n^2)$ computational complexity Velickovic et al. (2018). Thus, overall it takes $\mathcal{O}(\ell(n^3 + n^2))$ complexity. For the hierarchical representation model, graph U-Net (G- U- NET) Gao and Ji (2019), its computational complexity of one

Table 2 Summary of dataset statistics. LP: Link Prediction, NC: Node Classification, CD: Community Detection, N.A. means a dataset does not contain node features or node labels

Dataset	Task	#Nodes	#Edges	#Features	#Classes
Grid	LP	400	760	N.A.	N.A.
Cora	LP&NC	2708	5278	1433	7
Power	LP	4941	6594	N.A.	N.A.
Citeseer	NC	3312	4660	3703	6
Pubmed	NC	19,717	44,327	500	3
Emails	CD	799	10,182	N.A.	18
PPI	NC	56,658	818,435	50	121
Protein	NC	42,576	79,482	29	3
Ogbn-arxiv	NC	169,343	1,166,243	128	40

hierarchy is $\mathcal{O}(2\ell n^3)$, because its unpooling operation introduces another $\mathcal{O}(\ell n^3)$ complexity, in addition to the convolutional operations as GCN. Thus the complexity of g- U- Net with T levels is $\sum_{t=1}^{T} 2\ell(n\lambda^{t-1})^3 = \mathcal{O}(2\ell n^3)$, since the pooled graphs are supposed have much smaller number of nodes than $\mathcal{G}$. For HC-GNN, take GCN as an example GNN encoder and the *Louvain* algorithm as an example hierarchical structure construction method, which has optimal $O(n \log c)$ computational complexity Traag (2015), where c is the average degree. The top-down propagation allows each node of $\mathcal{G}$ to receive T different messages from T levels with different weights, this introduces $\mathcal{O}(Tn)$ computational complexity, where T is the number of levels, and we assume $T \ll n$. Altogether, the complexity of HC-GNN is $\sum_{t=1}^{T} \ell(n\lambda^{t-1})^3 + \mathcal{O}(n \log c + Tn) = \mathcal{O}(\ell n^3 + n \log c + Tn)$, which is more efficient than GAT and g- U- Net.

5 Experiments

We conduct extensive experiments to answer 6 research questions (RQ):

- **RQ1:** How does HC-GNN performs *vs.* state-of-the-art methods for node classification (**RQ1-1**), community detection (**RQ1-2**), and link prediction (**RQ1-3**)?
- **RQ2:** Can HC-GNN leads to satisfying performance under settings of transductive, inductive, and few-shot learning?
- **RQ3:** How do different levels in the hierarchical structure contribute to the effectiveness of node representations?
- **RQ4:** How do various hierarchical structure generation methods affect the performance of HC-GNN?
- **RQ5:** Does HC-GNN survive from low sparsity of graphs?
- **RQ6:** Does HC-GNN available with different encoders?

5.1 Evaluation setup

Datasets We perform experiments on both synthetic and real-world datasets. For the link prediction task, we adopt 3 datasets:

- Grid You et al. (2019). A synthetic 2D grid graph representing a 20×20 grid with $|\mathcal{V}| = 400$ and no node features.
- Cora Sen et al. (2008). A citation network consists of 2, 708 scientific publications and 5, 429 links. A 1, 433 dimensional word vector describes each publication as a node feature.
- Power Watts and Strogatz (1998). An electrical grid of western US with 4, 941 nodes and 6, 594 edges and no node features.

For node classification, we use 6 datasets: including Cora, Citeseer Kipf and Welling (2017) and Pubmed Kipf and Welling (2017) and a large-scale benchmark dataset Ogbn-arxiv Hu et al. (2020) for transductive settings, and 2 protein interaction networks Protein and PPI Ying et al. (2018) for inductive settings.

- Cora. The same above-mentioned Cora dataset contains 7 classes of nodes. Each node is labelled with the class it belongs to.
- Citeseer Sen et al. (2008). Each node comes with 3, 703-dimensional node features.
- Pubmed Namata et al. (2012). A dataset consists of 19, 717 scientific publications from PubMed database about diabetes classified into one of 3 classes. Each node is described by a TF/IDF weighted word vector from a dictionary which consists of 500 unique words.
- PPI Zitnik and Leskovec (2017). 24 protein-protein interaction networks and nodes of each graph comes with 50 dimensional feature vector.
- Protein Borgwardt et al. (2005). 1113 protein graphs and nodes of each graph comes with 29 dimensional feature vector. Each node is labelled with a functional role of the protein.
- Ogbn-arxiv Hu et al. (2020). A large-scale citation graph between 169, 343 computer science arXiv papers. Each node is an arXiv paper, and each directed edge indicates that one paper cites another one. Each paper comes with a 128-dimensional feature vector obtained by averaging the embeddings of words in its title and abstract. The task is to predict the 40 subject areas of these papers.

For node community detection, we use an email communication dataset:

- Emails Leskovec and Krevl (2014). 7 real-world email communication graphs from SNAP with no node features. Each graph has 6 communities, and each node is labelled with the community it belongs to.

The data statistics of datasets is summarised in Table 2 and they are available for download with our published code.

Experimental settings We evaluate HC-GNN under the settings of transductive and inductive learning. For node classification, we additionally conduct experiments with the few-shot setting.

- *Transductive learning* For link prediction, we follow the experimental settings of You et al. (2019) to use 10% existing links and an equal number of non-existent

links as validation and test sets. The remaining 80% existing links and a dual number of non-existent links are used as the training set. For node classification, we follow the semi-supervised settings of Kipf and Welling (2017): if there are enough nodes, for each class, we randomly sample 20 nodes for training, 500 nodes for validation, and 1000 nodes for testing. For the Emails dataset, we follow the supervised learning settings of Huang et al. (2019) to randomly select 80% nodes as the training set, and use the two halves of remaining as the validation and test set, respectively. We report the test performance when the best validation performance is achieved.

- *Inductive learning* This aims at examining a model's ability to transfer the learned knowledge from existing nodes to future ones that are newly connected to existing nodes in a graph. Hence, we hide the validation and testing graphs during training. We conduct the experiments for inductive learning using PPI and Protein datasets. We train models on 80% graphs to learn an embedding function f and apply it on the remaining 20% graphs to generate the representation of new-coming nodes.
- *Few-shot learning* Since the cost of collecting massive labelled datasets is high, having a few-shot learning model would be pretty valuable for practical applications. Few-shot learning can also be considered as an indicator to evaluate the robustness of a deep learning model. We perform few-shot node classification, in which only 5 samples of each class are used for training. The sampling strategies for testing and validation sets follow those in transductive learning.

Evaluation metrics We adopt AUC to measure the performance of link prediction. For node classification, we use micro- and macro-average F1 scores and accuracy. NMI score is utilised for community detection evaluation.

Competing methods To validate the effectiveness of HC-GNN, we compare it with 10 competing methods which include 6 flat message-passing GNN models, (GCN Kipf and Welling (2017), GraphSAGE Hamilton et al. (2017), GAT Velickovic et al. (2018), GIN Xu et al. (2019), P-GNNs You et al. (2019), GCNII Chen et al. (2020)), 3 hierarchical GNN models (HARP Chen et al. (2018), g- U- Net Gao and Ji (2019), GXN Li et al. (2020)) and another state-of-the-art model. (GraphRNA Huang et al. (2019)). For more details about competing methods, refer to App. B.

Reproducibility For fair comparison, all methods adopt the same representation dimension ($d = 32$), learning rate ($= 1e-3$), Adam optimiser and the number of iterations ($= 200$) with early stop (50). In terms of the neural network layers, we report the one with better performance of GCNII with better performance among $\{8, 16, 32, 64, 128\}$; for other models, we report the one with better performance between $2-4$; For all models with hierarchical structure (including g- U- Net and HC-GNN), we use GCN as the default GNN encoder for fair comparision. Note that for the strong competitor, P-GNNs, since its representation dimension is related to the number of nodes in a graph, we add a linear regression layer at the end of P-GNNs for node classification tasks to ensure its end-to-end structure is the same as other models Huang et al. (2019). For HC-GNN, the number of HC-GNN layers is varied and denoted as 1L, 2L or 3L. In Sect. 5.3, HC-GNN adopts the number of layers leading to the best performance for model analysis i.e., 2L for the Cora dataset, 1L for the

Citeseer and Pubmed datasets. For *Louvain* community detection, we use the implementation of a given package,[2] which does not require any hyper-parameters. We use PyTorch Geometric to implement all models mentioned in this paper. More details are referred to our code file.[3] The experiments are repeated 10 times, and average results are reported. Note that we use only node features with unique one-hot identifiers to differentiate different nodes if there are no given node features from the datasets and use the original node features if they are available. We employ Pytorch to implement all models. Experiments were conducted with GPU (NVIDIA Tesla V100) machines.

5.2 Experimental results

Transductive node classification (RQ1-1&RQ2) We present the results of transductive node classification in Table 3. We can see that HC-GNN consistently outperforms all of the competing methods in the 5 datasets, and even the shallow HC-GNN model with only one layer may lead to better results. We think the outstanding performance of HC-GNN results from two aspects: (a) the hierarchical structure allows the model to capture informative long-range interactions of graphs, i.e., propagating messages from and to distant nodes in the graph; and (b) the meso- and macro-level semantics reflected by the hierarchy is encoded through bottom-up, within-level, and top-down propagations. On the other hand, P-GNNs, HARP, and GraphRNA perform worse in semi-supervised node classification. The possible reason is they need more training samples, such as using 80% of existing nodes as the training set, as described in their papers (You et al. 2019; Huang et al. 2019), but we have only 20 nodes for training in the semi-supervised setting.

Inductive node classification (RQ1-1&RQ2) The results are reported in Table 4.[4] We can find that HC-GNN is still able to show some performance improvement over existing GNN models. But the improvement gain is not so significant and inconsistent in different layers of HC-GNN compared to the results in transductive learning. The possible reason is that different graphs may have other hierarchical community structures. Nevertheless, the results lead to one observation: the effect of transferring hierarchical semantics between graphs for inductive node classification is somewhat limited. Therefore, exploring an ameliorated model that can adaptively exploit hierarchical structure for different graphs for different tasks would be interesting. We further discuss it in Sect. 6 as one concluding remark.

Few-shot node classification (RQ1-1&RQ2) Table 5 demonstrates better performance in few-shot learning than all competing methods across 3 datasets. Such results indicate that the hierarchical message passing is able to transfer supervised information through inter- and intra-level propagations. In addition, the hierarchical message-passing pipeline further enlarges the influence range of supervision information from

[2] https://python-louvain.readthedocs.io/en/latest/api.html.

[3] Code and data are available at https://github.com/zhiqiangzhongddu/HC-GNN.

[4] Since HARP, P-GNNs and GraphRNA cannot be applied in the inductive setting, we do not present their results in Table 4.

Table 3 Results in Micro-F1 and Macro-F1 for transductive semi-supervised node classification task

	Cora		Citeseer		Pubmed		Emails	Ogbn-arxiv
	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	NMI	Acc (%)
GCN	0.802 ± 0.019	0.786 ± 0.020	0.648 ± 0.019	0.612 ± 0.012	0.779 ± 0.027	0.777 ± 0.026	0.944 ± 0.010	$71.74 \pm 0.29^{\ddagger}$
GraphSAGE	0.805 ± 0.013	0.792 ± 0.009	0.650 ± 0.027	0.611 ± 0.020	0.768 ± 0.031	0.763 ± 0.030	0.925 ± 0.014	$71.49 \pm 0.27^{\ddagger}$
GAT	0.772 ± 0.019	0.761 ± 0.023	0.620 ± 0.024	0.594 ± 0.015	0.775 ± 0.036	0.770 ± 0.022	$\underline{0.947} \pm 0.009$	$72.06 \pm 0.31^{\ddagger}$
GIN	0.762 ± 0.020	0.759 ± 0.018	0.615 ± 0.023	0.591 ± 0.020	0.744 ± 0.036	0.733 ± 0.041	0.640 ± 0.047	$71.76 \pm 0.33^{\ddagger}$
P-GNNs	0.438 ± 0.044	0.431 ± 0.040	0.331 ± 0.019	0.314 ± 0.018	0.558 ± 0.033	0.551 ± 0.036	0.598 ± 0.020	OOM
GCNII	$\underline{0.823} \pm 0.017$	$\underline{0.801} \pm 0.022$	$\underline{0.722} \pm 0.011$	$\underline{0.677} \pm 0.010$	$\underline{0.791} \pm 0.009$	$\underline{0.790} \pm 0.016$	0.947 ± 0.010	$\underline{72.74} \pm 0.16$
HARP	0.363 ± 0.020	0.350 ± 0.021	0.343 ± 0.023	0.317 ± 0.017	0.441 ± 0.024	0.329 ± 0.019	0.371 ± 0.014	OOM
GraphRNA	0.354 ± 0.070	0.244 ± 0.040	0.352 ± 0.050	0.259 ± 0.047	0.476 ± 0.054	0.355 ± 0.089	0.434 ± 0.047	OOM
g- U- Net	0.805 ± 0.017	0.796 ± 0.018	0.673 ± 0.015	0.628 ± 0.012	0.782 ± 0.018	0.781 ± 0.019	0.939 ± 0.015	71.78 ± 0.37
GXN	0.811 ± 0.025	$\underline{0.801} \pm 0.031$	0.698 ± 0.017	0.619 ± 0.021	0.784 ± 0.014	0.782 ± 0.012	0.943 ± 0.011	70.99 ± 0.27
HC-GNN-1L	0.819 ± 0.002	$\mathbf{0.816} \pm 0.005$	$\mathbf{0.728} \pm 0.005$	$\mathbf{0.686} \pm 0.003$	$\mathbf{0.812} \pm 0.009$	$\mathbf{0.806} \pm 0.009$	$\mathbf{0.961} \pm 0.005$	72.69 ± 0.25
HC-GNN-2L	$\mathbf{0.834} \pm 0.007$	$\mathbf{0.816} \pm 0.006$	0.696 ± 0.002	0.652 ± 0.006	$\mathbf{0.809} \pm 0.004$	$\mathbf{0.804} \pm 0.005$	$\mathbf{0.962} \pm 0.005$	$\mathbf{72.79} \pm 0.31$
HC-GNN-3L	0.813 ± 0.008	$\mathbf{0.806} \pm 0.006$	0.686 ± 0.006	0.633 ± 0.008	$\mathbf{0.804} \pm 0.004$	0.780 ± 0.020	0.935 ± 0.014	72.58 ± 0.27

Top-2 performances of each dataset are marked in bold and underline, respectively

Results in Acc for node classification of Ogbn-arxiv follows the default settings of OGB dataset Hu et al. (2020), and results in NMI for community detection (i.e., on the Emails data in the last column). Standard deviation errors are given. ‡ indicates the results from OGB leaderboard Hu et al. (2020). OOM: out-of-memory. 1L: model with 1-layer GNN encoder for *within-level propagation*

Table 4 Micro-F1 results for inductive node classification. Standard deviation errors are given

	PPI	Protein
GCN	0.444 ± 0.004	0.542 ± 0.018
GraphSAGE	0.409 ± 0.014	$\underline{0.637} \pm 0.018$
GAT	0.469 ± 0.062	0.608 ± 0.077
GIN	$\underline{0.571} \pm 0.008$	0.631 ± 0.016
GCNII	0.507 ± 0.008	0.614 ± 0.011
g- U- Net	0.433 ± 0.012	0.547 ± 0.011
GXN	0.510 ± 0.094	0.578 ± 0.014
HC-GNN-1L	0.48 ± 0.091	$\mathbf{0.638} \pm 0.027$
HC-GNN-2L	$\mathbf{0.584} \pm 0.087$	0.622 ± 0.031
HC-GNN-3L	$\mathbf{0.584} \pm 0.002$	0.582 ± 0.025

Top-2 performances of each dataset are marked in bold and underline, respectively
1L: model with 1-layer GNN encoder for *within-level propagation*

Table 5 Micro-F1 results for few-shot node classification

	Cora	Citeseer	Pubmed
GCN	0.695 ± 0.049	0.561 ± 0.054	0.699 ± 0.059
GraphSAGE	0.719 ± 0.024	0.559 ± 0.049	0.707 ± 0.051
GAT	0.630 ± 0.030	0.520 ± 0.054	0.664 ± 0.046
GIN	0.691 ± 0.038	0.509 ± 0.060	0.714 ± 0.036
P-GNNs	0.316 ± 0.040	0.332 ± 0.011	0.547 ± 0.037
GCNII	0.701 ± 0.022	0.564 ± 0.015	$\underline{0.717} \pm 0.047$
HARP	0.224 ± 0.033	0.260 ± 0.035	0.415 ± 0.039
GraphRNA	0.274 ± 0.063	0.206 ± 0.019	0.429 ± 0.042
g- U- Net	0.706 ± 0.054	$\underline{0.567} \pm 0.044$	0.693 ± 0.036
GXN	$\underline{0.721} \pm 0.035$	0.564 ± 0.21	0.706 ± 0.043
HC-GNN-1L	0.681 ± 0.023	$\mathbf{0.639} \pm 0.019$	0.704 ± 0.043
HC-GNN-2L	$\mathbf{0.759} \pm 0.015$	$\mathbf{0.660} \pm 0.024$	$\mathbf{0.724} \pm 0.052$
HC-GNN-3L	$\mathbf{0.752} \pm 0.017$	$\mathbf{0.642} \pm 0.016$	$\mathbf{0.742} \pm 0.045$

Top-2 performances of each dataset are marked in bold and underline, respectively
Standard deviation errors are given. 1L: model with 1-layer GNN encoder for *within-level propagation*

a small number of training samples. With effective and efficient pathways to broadcast information, HC-GNN is proven to be quite promising in few-shot learning.

Community detection (RQ1-2) The community detection results conducted on the Emails dataset are also shown in Table 3. It can be seen that HC-GNN again outperforms all competing methods. We believe this is because the communities identified by Louvain are further exploited by learning their hierarchical interactions in HC-GNN. In other words, HC-GNN is able to reinforce the intra- and inter-community effect and encode it into node representations.

Table 6 Results in AUC for link prediction. Standard deviation errors are given

	Grid	Cora-Feat	Cora-NoFeat	Power-Feat	Power-NoFeat
GCN	0.763 ± 0.036	0.869 ± 0.006	0.785 ± 0.007	0.624 ± 0.013	0.562 ± 0.012
GraphSAGE	0.775 ± 0.018	0.870 ± 0.006	0.741 ± 0.017	0.569 ± 0.012	0.510 ± 0.009
GAT	0.782 ± 0.028	0.874 ± 0.010	0.789 ± 0.012	0.621 ± 0.013	0.551 ± 0.019
GIN	0.756 ± 0.025	0.862 ± 0.009	0.782 ± 0.010	0.620 ± 0.011	0.549 ± 0.006
P-GNNs	<u>0.867</u> ± 0.034	0.818 ± 0.013	<u>0.792</u> ± 0.012	<u>0.704</u> ± 0.006	<u>0.668</u> ± 0.021
GCNII	0.807 ± 0.024	0.889 ± 0.019	0.770 ± 0.011	0.695 ± 0.014	0.577 ± 0.015
HARP	0.687 ± 0.021	0.837 ± 0.033	0.721 ± 0.017	0.529 ± 0.004	0.502 ± 0.004
g- U- Net	0.701 ± 0.032	<u>0.909</u> ± 0.006	0.772 ± 0.007	0.628 ± 0.024	0.584 ± 0.019
GXN	0.642 ± 0.089	0.889 ± 0.003	0.781 ± 0.011	0.645 ± 0.013	0.562 ± 0.015
HC-GNN-1*L*	0.823 ± 0.035	0.884 ± 0.006	**0.795** ± 0.012	0.682 ± 0.016	0.654 ± 0.017
HC-GNN-2*L*	**0.913** ± 0.011	0.895 ± 0.007	**0.837** ± 0.006	**0.767** ± 0.020	**0.722** ± 0.020
HC-GNN-3*L*	**0.914** ± 0.011	0.891 ± 0.007	**0.839** ± 0.004	**0.784** ± 0.017	**0.746** ± 0.021

Top-2 performances of each dataset are marked in bold and underline, respectively
1*L*: model with 1-layer GNN encoder for *within-level propagation*

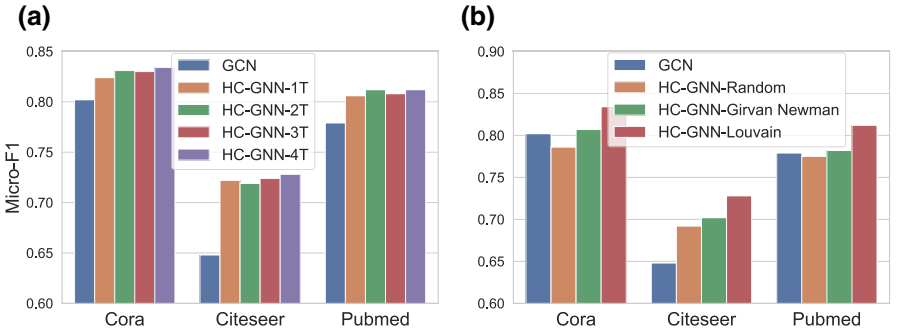

Fig. 3 Results in Micro-F1 for semi-supervised node classification using HC-GNN by varying: **a** the number of hierarchy levels adopted for message passing, and **b** the approaches to generate the hierarchical structure. $2T$ means model with first 2 hierarchy levels

Link prediction (RQ1-3) Here, we motivate our idea by considering pairwise relation prediction between nodes. Suppose a pair of nodes u, v are labelled with label y, and our goal is to predict y for unseen pairs. From the perspective of representation learning, we can solve the problem via learning an embedding function f that computes the node representation $\mathbf{z}_v$, where the objective is to maximise the likelihood of distribution $p(y|\mathbf{z}_u, \mathbf{z}_v)$. The results in Table 6 indicate that the HC-GNN leads to competitive performance compared to all competing methods, with up to 11.7% AUC improvement, demonstrating its effectiveness on link prediction tasks. When node features are accessible (i.e., Cora-Feat and Power-Feat), all models perform relatively well, and G- U- NET has slightly better performance on Cora-Feat dataset. Because node features provide meaningful information to predict pairwise relations. Another interesting perspective is investigating the models' performance without contextual node features (e.g., Grid, Cora-NoFeat and Power-NoFeat). It is surprising that HC-GNN variants show great superiority in these three datasets. We argue that when only topological information is available, the hierarchical semantics introduced by HC-GNN helps find missing links.

5.3 Empirical model analysis

Contribution of different levels (RQ3) Since HC-GNN highly relies on the generated hierarchical structure, we aim to examine how different levels in the hierarchy contribute to the prediction. We report the transductive semi-supervised node classification performance by varying the number of levels (from $1T$ to $4T$). GCN is also selected for comparison because it considers no hierarchy, i.e., only within-level propagation in the original graph. The results are shown in Fig. 3a, in which $1T$ and $2T$ indicate only the first hierarchy level and the first 2 hierarchy levels are adopted, respectively. We can find that HC-GNN using more levels for hierarchy construction lead to better results. The flat message passing of GCN cannot work well. Such results provide strong evidence that GNNs can significantly benefit from the hierar-

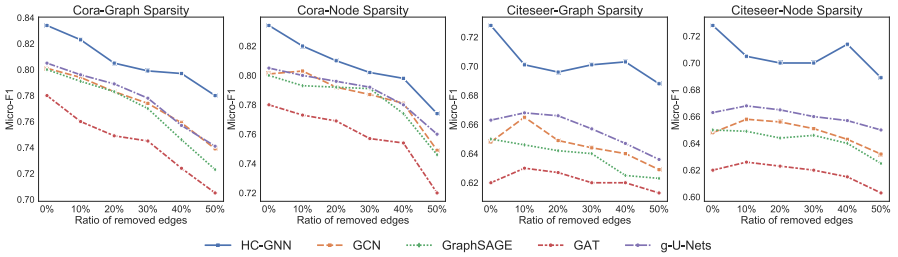

Fig. 4 Results on semi-supervised node classification in graphs by varying the percentage of removed edges

chical message-passing mechanism. In addition, more hierarchical semantics can be encoded if more levels are adopted.

Influence of hierarchy generation approaches (RQ4) HC-GNN implements the proposed *Hierarchical Message-passing Graph Neural Networks* based on the *Louvain* community detection algorithm, that is termed HC-GNN-*Louvain* in this paragraph. We aim to validate (A) whether the community information truly benefits the classification tasks, and (B) how different approaches to generate the hierarchical structure affect the performance. To answer (A), we construct a random hierarchical structure to generate randomised HC-GNN, termed HC-GNN-Random, in which *Louvain* detects hierarchical communities, and nodes are randomly swapped among the same-level communities. In other words, the hierarchy structure is maintained, but community memberships are perturbed. The results on semi-supervised node classification are exhibited in Fig. 3b. We can see that HC-GNN-Random works worse than GCN in Cora and Pudmed, and much worse than HC-GNN-*Louvain*. It implies that hierarchical communities generated from the graph topology genuinely lead to a positive effect on information propagation. Meanwhile, it is surprisingly found that HC-GNN-Random achieves better performance than GCN on Citeseer. We argue this is because HC-GNN-Random has the ability to spread supervision information in the hierarchy structure, leading to the occasional improvement. To answer (B), we utilise *Girvan Newman* Girvan and Newman (2002) to produce the hierarchical structure by following the same way described in Sect. 4.1, and have a model named HC-GNN-*Girvan Newman*. The results are shown in Fig. 3b. Although HC-GNN-*Girvan Newman* is not as effective as HC-GNN-*Louvain*, they still outperform GCN. Such a result indicates that the approaches to generate the hierarchical structure will influence the capability of HC-GNN. While HC-GNN-*Louvain* leads to promising performance, one can search for a proper hierarchical community detection method to perform better on different tasks.

Influence of graph sparsity (RQ5) Since community detection algorithms are sensitive to the sparsity of the graph Nadakuditi and Newman (2012), we aim at studying how HC-GNN perform under graphs with low sparsity values in the task of semi-supervised node classification. We consider two kinds of sparsity: one is graph sparsity by randomly removing a percentage of edges from all edges in the graph, i.e., $10\% - 50\%$; the other is node sparsity by randomly drawing a portion of edges

Table 7 Comparison of HC-GNN with different primary GNN encoders (*within-level propagation*), follow the transductive node classification settings

Models	Cora	Citeseer	Pubmed
GCN	0.802	0.648	0.779
HC-GNN w/ GCN	**0.834**	**0.728**	**0.812**
GAT	0.772	0.629	0.775
HC-GNN w/ GAT	**0.801**	**0.712**	**0.819**
GCNII	0.823	0.722	0.791
HC-GNN w/ GCNII	**0.841**	**0.734**	**0.816**

The performances of HC-GNN are marked in bold
Reported results in Micro-F1

incident to every node in the graph. The random removal of edges can be considered that users hide partial connections due to privacy concerns. The results for Cora and Citeseer are presented in Fig. 4. HC-GNN significantly outperforms the competing methods on graph sparsity and node sparsity under different edge-removal percentages. Such results prove that even though communities are subject to sparse graphs, but it will not damage HC-GNN's performance making it worse than other competing models.

Ablation study of different primary GNN encoders (RQ6) We adopted GCN as the default primary GNN encoder in model presentation (Sect. 4) and previous experiments. Here, we present more experimental results by endowing HC-GNN with advanced GNN encoders in Table 7. The table demonstrates that advanced GNN encoders can still benefit from the multi-grained semantics of HC-GNN. For instance, GCNII can stack lots of layers to capture long-range information; however, it still follows a *flat* message-passing mechanism hence naturally ignoring the multi-grained semantics. HC-GNN further ameliorates this problem for better performance.

6 Conclusion and future work

This paper has presented a novel *Hierarchical Message-passing Graph Neural Networks* (HMGNNs) framework, which deals with two critical deficiencies of the *flat* message passing mechanism in existing GNN models, i.e., the limited ability for information aggregation over long-range and infeasible in encoding meso- and macro-level graph semantics. Following this innovative idea, we further presented the first implementation, *Hierarchical Community-aware Graph Neural Network* (HC-GNN), with the assistance of a hierarchical communities detection algorithm. The theoretical analysis confirms HC-GNN's significant ability in capturing long-range interactions without introducing heavy computation complexity. Extensive experiments conducted on 9 datasets show that HC-GNN can consistently outperform state-of-the-art GNN models in 3 tasks, including node classification, link prediction, and community detection, under settings of transductive, inductive, and few-shot learning. Furthermore, the proposed hierarchical message-passing GNN provides model flexibility. For instance, it friendly allows different choices and customised designs of the hierarchical structure,

and it incorporates well with advanced flat GNN encoders to obtain more impressive results. That said, the HMGNNs could be easily applied to work as a general practical framework to boost downstream tasks with arbitrary hierarchical structure and encoder.

The proposed hierarchical message-passing GNNs provide a good starting point for exploiting graph hierarchy with GNN models. In the future, we aim to incorporate the learning of the hierarchical structure into the model optimisation of GNNs such that a better hierarchy can be searched on the fly. Moreover, it is also interesting to extend our framework for heterogeneous networks.

Acknowledgements This work is supported by the Luxembourg National Research Fund through grant PRIDE15/10621687/SPsquared, and supported by Ministry of Science and Technology (MOST) of Taiwan under grants 110-2221-E-006-136-MY3, 110-2221-E-006-001, and 110-2634-F-002-051.

Appendix A: Introduction of community detection algorithms

A.1 *Louvain* community detection algorithm

This section gives necessary background knowledge about the *Louvain* Blondel et al. (2008) community detection algorithm we used in this paper. Generally, this is a method to extract communities from large scale graphs by optimising modularity.

Modularity The problem of community detection requires the partition of a network into communities of densely connected nodes, with the nodes belonging to different communities being only sparsely connected. The so-called modularity of the partition often measures the modularity of the partitions resulting from these methods. The modularity of a partition is a scalar value between -1 and 1 that measures the density of links inside communities as compared to links between communities and can be defined as Blondel et al. (2008):

$$Q = \frac{1}{2m} \sum_{i,j} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j), \tag{8}$$

where c_i is the community to which node v_i is assigned, k_i and k_j are the sum of weights of the edges attached to nodes v_i and v_j, respectively. The δ-function $\delta(u, v)$ is 1 if $u = v$ and 0 otherwise and $m = \frac{1}{2} \sum_{ij} A_{ij}$.

In order to maximise this value (Q) efficiently, the *Louvain* community detection algorithm has two main phases that are repeated iteratively: *(i)* each node in the graph is assigned to its own community; *(ii)* for each node, the change in modularity is

calculated by removing v from its own community and moving it into the community of each neighbour u of v. This value is easily calculated by two steps: (1) removing v from its original community and (2) inserting v into the community of u. This is a typical greedy optimisation, some following work proposed solutions to optimise its efficiency significantly Blondel et al. (2008).

A.2 *Girvan Newman* community detection algorithm

The *Girvan-Newman* algorithm Girvan and Newman (2002) for the detection and analysis of community structure relies on the iterative elimination of edges that have the highest number of shortest paths between nodes passing through them. By removing edges from the graph one by one, the network breaks down into smaller pieces, so-called communities.

The **betweenness centrality** Freeman (1977) of a node v is defined as the number of shortest paths between pairs of other nodes that run through v. It is a measure of the influence of a node over the flow of information between other nodes, especially in cases where information flow over a network primarily follows the shortest available path. Based on the definition of betweenness centrality, the *Girvan-Newman* algorithm can be generally divided into four main steps:

1. For every edge in a graph, calculate the edge betweenness centrality.
2. Remove the edge with the highest betweenness centrality.
3. Calculate the betweenness centrality for every remaining edge.
4. Repeat steps $2-3$ until there are no more edges left.

Appendix B: Competing methods

Competing methods To validate the effectiveness of HC-GNN, we compare it with 9 competing methods which include 6 flat message-passing GNN models, 2 hierarchical GNN models and another state-of-the-art model.

- GCN[5] Kipf and Welling (2017) is the first deep learning model which generalises the convolutional operation on graph data and introduces the semi-supervised train paradigm.
- GraphSAGE[6] Hamilton et al. (2017) extends the convolutional operation of GCN to mean/ max/ LSTM convolutions and introduces a sampling strategy before employing convolutional operations on neighbour nodes.
- GAT[7] Velickovic et al. (2018) employs trainable attention weight during message aggregation from neighbours, which makes the information received by each node different and provides interpretable results.

[5] https://github.com/tkipf/pygcn.

[6] https://github.com/williamleif/GraphSAGE.

[7] https://github.com/PetarV-/GAT.

- GIN[8] Xu et al. (2019) summarises previous existing GNN layers as two components, AGGREGATE and COMBINE, and models injective multiset functions for the neighbour aggregation.
- HARP[9] Chen et al. (2018) is a hierarchical structure by various collapsing methods for unsupervised node representation learning.
- P-GNNs[10] You et al. (2019) introduces anchor-set sampling to generate node representation with global position-aware.
- G- U- NET[11] Gao and Ji (2019) generalises the U-nets architecture of convolutional neural networks for graph data to get better node representation. It constructs a hierarchical structure with the help of pooling and unpooling operators.
- GraphRNA[12] Huang et al. (2019) proposes using recurrent neural networks to capture the long-range node dependencies to assist GNN to obtain better node representation.
- GXN Li et al. (2020)[13] proposes an infomax pooling operator for graph data to get the hierarchy structure.
- GCNII[14] Chen et al. (2020) simplifies the aggregation design of flat GNNs, joined with well-designed normalisation units to get much deeper GNN models.

Appendix C: Model comparison

In Sect. 2, we have systematically discussed related work and highlighted the differences between HC-GNN and them. Here, we further present Table 8 to summarise the critical advantages of the proposed HC-GNN and compare it with a number of state-of-the-art methods published recently. We are the first to present the hierarchical message passing to efficiently model long-range informative interaction and multi-grained semantics. In addition, our HC-GNN can utilise the community structures and be applied for transductive, inductive and few-shot inferences.

References

Alon U, Yahav E (2021) On the bottleneck of graph neural networks and its practical implications. In: Proceedings of the 2021 international conference on learning representations (ICLR)

Bhowmick AK, Meneni K, Danisch M, Guillaume J, Mitra B (2020) Louvainne: Hierarchical louvain method for high quality and scalable network embedding. In: Proceedings of the 2020 ACM international conference on web search and data mining (WSDM), pp 43–51

Blondel VD, Guillaume J-L, Lambiotte R, Lefebvre E (2008) Fast unfolding of communities in large networks. J Stat Mech Theory Exp 2008(10):10008

[8] https://github.com/weihua916/powerful-gnns.

[9] https://github.com/GTmac/HARP.

[10] https://github.com/JiaxuanYou/P-GNN.

[11] https://github.com/HongyangGao/Graph-U-Nets.

[12] https://github.com/xhuang31/GraphRNA_KDD19.

[13] https://github.com/limaosen0/GXN.

[14] https://github.com/chennnM/GCNII.

Table 8 Model comparison in aspects of Node-wise Task (NT), SUPervised training paradigm (SUP), Transductive Inference (TI), Inductive Inference (II), Long-range Information (LI), and Hierarchical Semantics for Node Representations (HSNR)

	NT	SUP	TI	II	LI	HSNR
GCN Kipf and Welling (2017)	√	√	√	√		
GraphSAGE Hamilton et al. (2017)	√	√	√	√		
GAT Velickovic et al. (2018)	√	√	√	√		
GIN Xu et al. (2019)	√	√	√	√		
P-GNNs You et al. (2019)	√	√	√		√	
GCNII Chen et al. (2020)	√	√	√	√	√	
DIFFPOOL Ying et al. (2018)		√	√	√		
G- U- NET Gao and Ji (2019)	√	√	√	√		√
ATTPOOL Huang et al. (2019)		√	√	√		
ASAP Ranjan et al. (2020)		√	√	√		
GXN Li et al. (2020)	√	√	√	√		√
GraphRNA Huang et al. (2019)	√	√	√			
HARP Chen et al. (2018)	√		√			
LouvainNE Bhowmick et al. (2020)	√		√			
HC-GNN	√	√	√	√	√	√

Borgwardt KM, Ong CS, Schönauer S, Vishwanathan SVN, Smola AJ, Kriegel H (2005) Protein function prediction via graph kernels. Bioinformatics 21(suppl–1):47–56
Chen Z, Li L, Bruna J (2019) Supervised community detection with line graph neural networks. In: Proceedings of the 2019 international conference on learning representations (ICLR)
Chen H, Perozzi B, Hu Y, Skiena S (2018) HARP: Hierarchical representation learning for networks. In: Proceedings of the 2018 AAAI conference on artificial intelligence (AAAI), pp 2127–2134
Chen M, Wei Z, Huang Z, Ding B, Li Y (2020) Simple and deep graph convolutional networks. In: Proceedings of the 2020 international conference on machine learning (ICML)
Freeman LC (1977) A set of measures of centrality based on betweenness. Sociometry, 35–41
Gao H, Ji S (2019) Graph u-nets. In: Proceedings of the 2019 international conference on machine learning (ICML)
Gilmer J, Schoenholz SS, Riley PF, Vinyals O, Dahl GE (2017) Neural message passing for quantum chemistry. In: Proceedings of the 2017 international conference on machine learning (ICML)
Girvan M, Newman MEJ (2002) Community structure in social and biological networks. Proc Natl Acad Sci 99:7821–7826
Gu F, Chang H, Zhu W, Sojoudi S, El Ghaoui L (2020) Implicit graph neural networks. In: Proceedings of the 2020 annual conference on neural information processing systems (NeurIPS)
Hamilton WL, Ying Z, Leskovec J (2017) Inductive representation learning on large graphs. In: Proceedings of the 2017 annual conference on neural information processing systems (NIPS), pp 1025–1035
Huang J, Li Z, Li N, Liu S, Li G (2019) Attpool: Towards hierarchical feature representation in graph convolutional networks via attention mechanism. In: Proceedings of the 2019 IEEE international conference on computer vision (ICCV), pp 6480–6489
Huang X, Song Q, Li Y, Hu X (2019) Graph recurrent networks with attributed random walks. In: Proceedings of the 2019 ACM conference on knowledge discovery and data mining (KDD), pp 732–740
Hu W, Fey M, Zitnik M, Dong Y, Ren H, Liu B, Catasta M, Leskovec J (2020) Open graph benchmark: Datasets for machine learning on graphs. In: Proceedings of the 2020 annual conference on neural information processing systems (NeurIPS)
Kipf TN, Welling M (2017) Semi-supervised classification with graph convolutional networks. In: Proceedings of the 2017 international conference on learning representations (ICLR)

LeCun Y, Bengio Y, Hinton GE (2015) Deep learning. Nature 521(7553):436–444
Leskovec J, Krevl A (2014) SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data
Li M, Chen S, Zhang Y, Tsang IW (2020) Graph cross networks with vertex infomax pooling. In: Proceedings of the 2020 annual conference on neural information processing systems (NeurIPS)
Li Q, Han Z, Wu X (2018) Deeper insights into graph convolutional networks for semi-supervised learning. In: Proceedings of the 2018 AAAI conference on artificial intelligence (AAAI), pp. 3538–3545
Li Z, Kovachki NB, Azizzadenesheli K, Liu B, Stuart AM, Bhattacharya K, Anandkumar A (2020) Multipole graph neural operator for parametric partial differential equations. In: Proceedings of the 2020 annual conference on neural information processing systems (NeurIPS)
Li P, Wang Y, Wang H, Leskovec J (2020) Distance encoding - design provably more powerful graph neural networks for structural representation learning. In: Proceedings of the 2020 annual conference on neural information processing systems (NeurIPS)
Min Y, Wenkel F, Wolf G (2020) Scattering GCN: overcoming oversmoothness in graph convolutional networks. In: Proceedings of the 2020 annual conference on neural information processing systems (NeurIPS)
Nadakuditi RR, Newman MEJ (2012) Graph spectra and the detectability of community structure in networks. Phys Rev Lett 108(18):188701
Namata G, London B, Getoor L, Huang B (2012) Query-driven active surveying for collective classification. In: Proceedings of the 2012 international workshop on mining and learning with graphs, p 8
Rampásek L, Wolf G (2021) Hierarchical graph neural nets can capture long-range interactions. In: IEEE international workshop on machine learning for signal processing (MLSP), pp 1–6
Ranjan E, Sanyal S, Talukdar PP (2020) ASAP: adaptive structure aware pooling for learning hierarchical graph representations. In: Proceedings of the 2020 AAAI conference on artificial intelligence (AAAI), pp. 5470–5477
Ronneberger O, Fischer P, Brox T (2015) U-net: convolutional networks for biomedical image segmentation. In: Proceedings of the 2015 medical image computing and computer-assisted intervention (MICCAI). Lecture notes in computer science, vol 9351, pp 234–241
Rossetti G, Guidotti R, Pennacchioli D, Pedreschi D, Giannotti F (2015) Interaction prediction in dynamic networks exploiting community discovery. In: Proceedings of the 2015 IEEE/ACM international conference on advances in social networks analysis and mining (ASONAM), pp 553–558
Sen P, Namata G, Bilgic M, Getoor L, Galligher B, Eliassi-Rad T (2008) Collective classification in network data. AI Mag 29(3):93–93
Sun Y, Han J (2012) Mining heterogeneous information networks: a structural analysis approach. ACM SIGKDD explorations newsletter
Traag VA (2015) Faster unfolding of communities: Speeding up the louvain algorithm. Phys Rev E 92(3):032801
Velickovic P, Cucurull G, Casanova A, Romero A, Lio P, Bengio Y (2018) Graph attention networks. In: Proceedings of the 2018 international conference on learning representations (ICLR)
Wang X, Cui P, Wang J, Pei J, Zhu W, Yang S (2017) Community preserving network embedding. In: Proceedings of the 2017 AAAI conference on artificial intelligence (AAAI), pp 203–209
Wang J, Peng J, Liu O (2014) An approach for hesitant node classification in overlapping community detection. In: Processings of the 2014 Pacific Asia conference on information systems (PACIS), p 47
Watts DJ, Strogatz SH (1998) Collective dynamics of small-world networks. Nature 393:440
Wu Z, Pan S, Chen F, Long G, Zhang C, Yu PS (2021) A comprehensive survey on graph neural networks. IEEE Trans Neural Netw Learn Syst 32(1):4–24
Xu K, Hu W, Leskovec J, Jegelka S (2019) How powerful are graph neural networks? In: Proceedings of the 2019 international conference on machine learning (ICML)
Ying R, You J, Morris C, Ren X, Hamilton WL, Leskovec J (2018) Hierarchical graph representation learning with differentiable pooling. In: Proceedings of the 2018 annual conference on neural information processing systems (NeurIPS), pp 4805–4815
You J, Ying R, Leskovec J (2019) Position-aware graph neural networks. In: Proceedings of the 2019 international conference on machine learning (ICML)
Zhang Z, Cui P, Zhu W (2020) Deep learning on graphs: a survey. IEEE Trans Knowl Data Eng. https://doi.org/10.1109/TKDE.2020.2981333
Zhang M, Chen Y (2018) Link prediction based on graph neural networks. In: Proceedings of the 2018 annual conference on neural information processing systems (NeurIPS), pp 5171–5181

Zitnik M, Leskovec J (2017) Predicting multicellular function through multi-layer tissue networks. Bioinformatics 33:190–198
Zitnik M, Agrawal M, Leskovec J (2018) Modeling polypharmacy side effects with graph convolutional networks. Bioinformatics 34(13):457–466