Trading off Quality for Efficiency of Community Detection: An Inductive Method across Graphs

MENG QIN, Department of CSE, HKUST, Hong Kong SAR
CHAORUI ZHANG, Theory Lab, Huawei, Hong Kong SAR
BO BAI, Theory Lab, Huawei, Hong Kong SAR
GONG ZHANG, Theory Lab, Huawei, Hong Kong SAR
DIT-YAN YEUNG, Department of CSE, HKUST, Hong Kong SAR

Many network applications can be formulated as NP-hard combinatorial optimization problems of community detection (CD). Due to the NP-hardness, to balance the CD quality and efficiency remains a challenge. Most existing CD methods are transductive, which are independently optimized only for the CD on a single graph. Some of these methods use advanced machine learning techniques to obtain high-quality CD results but usually have high complexity. Other approaches use fast heuristic approximation to ensure low runtime but may suffer from quality degradation. In contrast to these transductive methods, we propose an alternative inductive community detection (ICD) method across graphs of a system or scenario to alleviate the NP-hard challenge. ICD first conducts the offline training of an adversarial dual GNN on historical graphs to capture key properties of the system. The trained model is then directly generalized to new unseen graphs for online CD without additional optimization, where a better trade-off between quality and efficiency can be achieved. ICD can also capture the permutation invariant community labels in the offline training and tackle the online CD on new graphs with non-fixed number of nodes and communities. Experiments on a set of benchmarks demonstrate that ICD can achieve a significant trade-off between quality and efficiency over various baselines.

CCS Concepts: • Mathematics of computing → Graph algorithms; • Theory of computation → Inductive inference; Social networks.

Additional Key Words and Phrases: Community detection, graph clustering, inductive graph representation learning

ACM Reference Format:
Meng Qin, Chaorui Zhang, Bo Bai, Gong Zhang, and Dit-Yan Yeung. 2018. Trading off Quality for Efficiency of Community Detection: An Inductive Method across Graphs. Proc. ACM Meas. Anal. Comput. Syst. 37, 4, Article 111 (August 2018), 27 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

For various complex systems, e.g., communication and social networks, graph is a generic model to describe entities and their relations using a set of nodes and edges. Community detection (CD), a.k.a. graph clustering [38], aims to partition the nodes of a graph into several groups (i.e., communities) with dense linkage distinct from other groups [11].

Authors’ addresses: Meng Qin, mengxin_az@foxmail.com, Department of CSE, HKUST, Hong Kong SAR; Chaorui Zhang, chaorui.zhang@gmail.com, Theory Lab, Huawei, Hong Kong SAR; Bo Bai, ee.hobbai@gmail.com, Theory Lab, Huawei, Hong Kong SAR; Gong Zhang, nicholas.zhang@huawei.com, Theory Lab, Huawei, Hong Kong SAR; Dit-Yan Yeung, dyyeung@cse.ust.hk, Department of CSE, HKUST, Hong Kong SAR.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2018 Association for Computing Machinery.
Manuscript submitted to ACM
Since the extracted communities are believed to correspond to several real-world substructures of a system, e.g., cells in wireless networks [7], many network applications can be formulated as CD tasks [25, 29, 35].

Mathematically, CD can be described as some NP-hard combinatorial optimization problems, e.g., modularity maximization [27] and normalized cut (NCut) minimization [41]. Due to the NP-hardness, to balance the quality and efficiency remains a challenge but some real applications have both requirements of high quality and low runtime, e.g., accurate CD on a wireless cellular network with several thousand nodes in a few seconds [7].

**Conventional Transductive CD.** As illustrated in Fig. 1 (a), most existing CD methods are transductive, which independently optimize the CD model on each single graph and can only tackle CD on such a unique graph. Since model parameters are not shared across graphs, the model trained on one graph cannot be directly generalized to other graphs. Usually, one needs to optimize the model from scratch for each new graph. Conventional transductive methods usually focus on either high quality or efficiency.

On the one hand, prior work has demonstrated the ability of advanced machine learning (ML) techniques to achieve high quality, e.g., high accuracy between the CD results and ground-truth. Typical ML-based methods include non-negative matrix factorization [43, 45] and probabilistic graphical models [18, 55]. Graph embedding emerges as a promising technique for CD in recent studies. These methods first learn low-dimensional node representations via random walk [14, 31], matrix factorization [23, 36], or deep learning [42, 51] to capture high-order proximities and nonlinear characteristics of a graph. The CD result is derived by feeding the learned embedding into a downstream clustering module (e.g., KMeans). However, aforementioned ML-based methods usually rely on iterative optimization algorithms (e.g., gradient descent for deep learning and expectation maximization for probabilistic models) to independently learn large-scale parameters for each graph with inevitable high complexity.

On the other hand, to reduce the overall runtime and satisfy real-time constraints of some applications (e.g., decomposing a wireless network with several thousand entities into cells in several seconds [7]) is another major focus. Some transductive methods use fast approximation for classic CD objectives, e.g., greedy modularity maximization [2] and multilevel coarsening for NCut minimization [9]. Several graph embedding approaches also adopt fast approximation to reduce the inference time, e.g., random projection for high-order proximities [57] and randomized SVD to capture local structural information [54]. Despite their high efficiency (e.g., low runtime to derive CD results), they may suffer from quality declines due to the information loss of fast approximation.

---

*Fig. 1. Overview of (a) conventional transductive CD methods for each single graph and (b) our inductive community detection (ICD) framework across graphs, including the (b.1) offline training on historical graphs and (b.2) online generalization to new graphs.*
Trading off Quality for Efficiency of Community Detection: An Inductive Method across Graphs

| Feature Input | Settings | Descriptions | Examples |
|---------------|----------|--------------|----------|
| Standard      |          | Use a constant matrix/one-hot node degree representation as feature inputs. | GraphSAGE [15], GAT [10], GIN [49] |
|               |          | Not informative features to support high-quality CD. | |
|               | Extract additional features via dimension reduction (e.g., PCA) on topology Usually time-consuming. | GapNet [5], ClusNet [49], LGNN [8] |
| How to derive CD results | End-to-End (E2E) | Feed learned embeddings (from GNN) to an output layer to derive CD results. Can tackle inductive CD across graphs with both fixed & non-fixed K. | GAT [26], ClusNet [8], LGNN [4] |
|               | Embedding | Apply a downstream clustering module to the learned embedding (from GNN). Can tackle inductive CD across graphs with both fixed & non-fixed K. | GAT [26], ClusNet [8], LGNN [4] |
| Training Loss | Unsupervised Loss | Use unsupervised loss of existing graph embedding techniques (e.g., skipgram-based loss). Usually lacks robustness for high-quality CD and cannot capture permutation invariant training labels. | GAT [26], ClusNet [49] |
| CD Objectives |          | Use CD objectives (e.g., modularity maximization & NCut minimization) for unsupervised training. Usually combines with the E2E framework and cannot capture permutation invariant training labels. | GAT [26], ClusNet [49] |

**Inductive CD across Graphs.** In contrast to conventional transductive methods, we try to achieve a better trade-off between quality and efficiency from an alternative inductive perspective across multiple graphs of a system or scenario. An inductive community detection (ICD) method is proposed based on the fact that most real-world complex systems independently generate a set of graphs via common knowledge in terms of underlying distributions, e.g., power-law distributions. This hypothesis is also widely adopted in the simulation of various systems [47]. The multiple graphs can be (i) snapshots evolving over time or (ii) independent graphs of a scenario without temporal dependency, e.g., ego-nets in social media. The inductiveness of ICD implies that its model parameters are shared by all the associated graphs.

As illustrated in Fig. 1 (b), ICD includes two phases. (i) We first train a high-quality CD model on historical graphs of a system or scenario in an offline way, aiming to fully capture the underlying distributions of the system or scenario regardless of time. (ii) The trained model can be directly generalized to newly generated graphs of the same system or scenario for online CD without additional optimization, which significantly reduces the runtime on these new graphs and is believed to have high-quality results. In real applications, we usually assume that one has enough time to train a high-quality model using historical data in an offline way, which is also one-time-effort only. Our main focus is to achieve a better trade-off between quality and efficiency of the online CD on new graphs after deploying the trained model to a system with its model parameters fixed.

ICD adopts a graph embedding scheme for CD across graphs using the inductive nature of graph neural networks (GNN). Although existing inductive GNNs (e.g., GraphSAGE [15]) can be generalized to new graphs for online inference, they may suffer from the following limitations for CD as summarized in Table 1.

First, in this study, we consider CD across graphs where topology is the only available information source, i.e., without attributes. As some systems allow the addition and deletion of entities, we also assume the number of nodes $N$ can be non-fixed for each graph. Although most inductive GNNs [15, 40] can be applied to graphs with non-fixed $N$, they are originally designed for attributed graphs with two input sources of topology and node features. In particular, their inductiveness relies on the fixed dimensionality of feature inputs for all the graphs. Our experiments indicate that some standard settings of inductive GNNs for the case without attributes (e.g., using a constant matrix as feature inputs [49]) cannot derive informative node features to support high-quality CD. Other GNN-based methods extract additional features via dimension reduction (e.g., PCA) to map the topology (e.g., adjacency matrix) with non-fixed $N$ to a fixed-dimensional feature space [26], which is usually time-consuming.

Second, graphs from a system or scenario can be assigned with different number of communities $K$. Some GNN-based methods adopt an end-to-end (E2E) framework to approximate some classic CD objectives (e.g., NCut minimization [26]), with the output of GNN fed into a fully-connected output layer to directly derive CD results. However, they can only tackle the inductive CD across graphs with fixed $K$, due to fixed dimensionality of the output layer. These E2E methods adopt an end-to-end (E2E) framework to approximate some classic CD objectives (e.g., NCut minimization [26]), with the output of GNN fed into a fully-connected output layer to directly derive CD results. However, they can only tackle the inductive CD across graphs with fixed $K$, due to fixed dimensionality of the output layer. These E2E...
methods still need to be optimized from scratch for new graphs with non-fixed $K$ (i.e., using conventional transductive settings), which is also time-consuming.

Third, although prior studies have validated the ability of inductive GNNs to tackle (semi-)supervised tasks (e.g., node classification) on new nodes or graphs [15, 40], few of them consider unsupervised node-level tasks (e.g., CD) across graphs. Our experiments indicate that some standard settings of inductive GNNs for unsupervised tasks (e.g., using unsupervised training loss for GNN [15]) lack robustness for the online CD on new graphs.

**Present Work.** ICD is a generic framework that can address the aforementioned limitations of inductive GNNs with several original designs. (i) To enable ICD to tackle inductive CD across graphs with non-fixed $N$, we develop an efficient feature extraction module for inductive GNNs via graph coarsening, which can extract informative node features to support high-quality fast online CD. (ii) In contrast to E2E methods, ICD adopts an inductive graph embedding scheme across graphs with CD results derived via a downstream clustering module, which can tackle the inductive CD with non-fixed $K$. (iii) Note that CD is an unsupervised node-level task, where community labels are permutation invariant. For instance, label assignments $(l_1, l_2, l_3) = (1, 2, 2)$ and $(l_1, l_2, l_3) = (2, 1, 1)$ are the same in terms of CD with $l_i$ as the community label of node $v_i$. Most existing GNN-based methods cannot directly utilize the permutation invariant training labels of CD. In contrast, ICD can further incorporate such label information of historical graphs to the offline training, which enhances the embedding optimization, by combining an adversarial dual GNN structure and a clustering regularization loss based on classic CD objectives, e.g., modularity maximization and NCut minimization.

We summarize our major contributions as follows.

- In contrast to conventional transductive CD methods, we propose a novel ICD method to achieve a better trade-off between quality and efficiency via an alternative inductive graph embedding scheme across graphs.
- Compared with existing inductive GNN based methods, ICD is a generic method that can tackle the inductive CD across graphs with non-fixed $N$ and $K$ while capturing permutation invariant training labels based on several original designs, e.g., graph coarsening based feature extraction, adversarial dual GNN, and clustering regularization loss.
- We compare the quality and efficiency of ICD with 17 baselines on 10 datasets. Experiments demonstrate that ICD can achieve a better trade-off between the quality and efficiency of CD over various baselines.

Remainder of this paper is organized as follows. In Section 2, we briefly review related work. The formal problem statements and preliminaries of this study are provided in Section 3. Section 4 elaborates on the proposed ICD framework. Experimental settings and evaluation results are described in Section 5. Finally, Section 6 concludes the paper.

## 2 RELATED WORK

In the past few decades, a series of methods have been proposed for CD (a.k.a. graph clustering). As reviewed in [8, 17, 38], CD can be mathematically formulated as several NP-hard combinatorial optimization problems. Overviews of some typical objectives (e.g., NCut minimization and modularity maximization) can be found in [3, 41].

**Transductive CD Methods.** Most existing CD methods are transductive, which usually focus on either high quality or efficiency. Some approaches try to obtain high-quality CD results via advanced ML techniques. Wang et al. [45] used non-negative matrix factorization (NMF) with diffusion kernel similarity to tackle CD. In [43], CD on undirected, directed, and compound graphs was formulated as three NMF objectives. Karrer et al. [18] proposed the degree-corrected stochastic blockmodel and discussed its relations to modularity maximization based on the generative probabilistic model. Graph embedding, which learns distributed representations to capture key properties of graphs, has become
a promising ML technique for various graph inference tasks including CD. Inspired by the skipgram-based word embedding, Perozzi et al. [31] and Grover et al. [14] proposed DeepWalk and node2vec that learn embeddings based on truncated random walks on graphs. Tian et al. [39] and Yang et al. [51] explored the potential of deep learning to derive community-preserved embeddings by reconstructing features of NCut minimization and modularity maximization. In contrast, Wang et al. [46] and Li et al. [22] extracted community-preserved embeddings by incorporating modularity maximization and stochastic block model to NMF objectives. However, aforementioned transductive ML-based methods usually rely on time-consuming iterative optimization algorithms (e.g., gradient descent) to independently learn large-scale parameters for each single graph despite their high quality.

Some other transductive methods try to reduce their optimization or inference time via fast approximation to some objectives. Dhillon et al. [9] developed a multilevel coarsening algorithm for graph-cut based CD using the equivalence between kernel KMeans and some graph-cut objectives, e.g., NCut minimization. Clauset et al. [6] introduced a hierarchical approach with a greedy strategy for modularity maximization, while Wang et al. [44] formulated a relaxed version of modularity maximization as a low-cardinality semidefinite programming objective. In [30], a efficient algorithm based on the Markov chain Monte Carlo sampling was presented to inference the stochastic block model of CD. Some graph embedding methods also try to reduce their inference time using fast approximation. In [57], Zhang et al. learned high-order proximity preserved embedding by Gaussian random projection. Dong et al. [54] adopted sparse randomized SVD to capture local structural properties in the embedding inference. Yang et al. [52] used BKSVD to initialize the learned embedding based on the personalized PageRank measure. However, aforementioned methods usually suffer from quality declines due to the information loss of fast approximation.

In summary, to balance the quality and efficiency of CD remains a challenge for aforementioned transductive methods. In contrast, we consider an alternative inductive graph embedding scheme across graphs to achieve a better trade-off between quality and efficiency using the inductive nature of GNNs.

**Inductive GNNs for CD.** In contrast to transductive methods optimized for a single graph, the learned model parameters of inductive GNNs can be generalized to new unseen nodes and graphs. Most GNNs are originally designed for attributed graphs and rely on the multi-layer aggregation of node features, a.k.a. message passing. Hamilton et al. [15] proposed GraphSAGE, a generic inductive GNN framework based on the graph embedding scheme, with various feature aggregators and both supervised and unsupervised training losses. Veličković et al. [40] introduced GAT that can adaptively adjust the feature aggregation by applying self-attention to node features. You et al. [53] developed a class of message passing GNNs by inductively considering node identities. Although aforementioned inductive GNNs have demonstrated (semi-)supervised node-level tasks (e.g., node classification), few of them consider node-level unsupervised tasks (e.g., CD) across graphs. Our experiments demonstrate that some standard settings of GNN for inductive CD across graphs without attributes (e.g., applying a skipgram-based unsupervised training loss [15] and using a constant matrix as input features of GNNs [49]) lack robustness for high-quality CD.

Some other approaches use inductive GNNs to approximate classic CD objectives in an E2E scheme, where the outputs of GNN are further fed into an output layer to directly derive the CD results. Nazi et al. [26] and Wilder et al. [48] proposed GAP and ClusNet, which are E2E methods approximating NCut minimization and modularity maximization. However, their inductiveness is designed only for CD with fixed number of communities $K$, due to the fixed dimensionality of their output layers. One has to use conventional transductive settings for graphs with non-fixed $K$ (i.e., optimizing them from scratch for each new graph), which is time-consuming. For inductive CD with non-fixed number of nodes $N$, GAP and ClusNet use PCA and node2vec to map the original graph topology into another feature space [26, 48] for input features of GNN, which also have high complexity.
Moreover, the aforementioned GNN-based methods cannot utilize the permutation invariant community labels in their offline training. Although LGNN [4] can capture such label information, it relies on an E2E cross-entropy loss that finds the best mapping from the derived CD result to the training labels among $O(K!)$ cases. Hence, it can only tackle the inductive CD with fixed $K$ and has high complexity to capture training labels, intractable to the case with large $K$. In contrast, our ICD framework can capture permutation invariant training labels to enhance the offline embedding optimization with much lower complexity while supporting the fast online CD with non-fixed $N$ and $K$.

3 PROBLEM STATEMENTS AND PRELIMINARIES

In this study, we consider CD on a set of graphs $\mathcal{S} = \{\mathcal{G}_1, \ldots, \mathcal{G}_T\}$ extracted from a common system or scenario. Each graph $\mathcal{G}_t$ can be represented as $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t)$ with $\mathcal{V}_t = \{v_{t,1}, \ldots, v_{t,N_t}\}$ and $\mathcal{E}_t = \{(v_{t,i}, v_{t,j}) | v_{t,i}, v_{t,j} \in \mathcal{V}_t\}$ denoting the sets of nodes and edges. For each $\mathcal{G}_t$, topology is the only available information source without graph attributes. We use an adjacency matrix $A_t \in \mathbb{R}^{N_t \times N_t}$ to describe its topology with $N_t$ nodes, where $(A_t)_{ij} = (A_t)_{ji} = 1$ if $(v_{t,i}, v_{t,j}) \in \mathcal{E}_t$ and $(A_t)_{ij} = (A_t)_{ji} = 0$ otherwise. Since some systems allow the addition and deletion of entities, we assume different graphs in $\mathcal{S}$ can have different node sets (i.e., $\exists \mathcal{G}_t, \mathcal{G}_s \in \mathcal{S}$ s.t. $\mathcal{V}_t \neq \mathcal{V}_s$) with non-fixed number of nodes $N_t$.

Graphs in $\mathcal{S}$ can be (i) system snapshots evolving over time or (ii) independent graphs of a scenario without temporal dependency and node correspondence, e.g., ego-nets in social media. To ensure the model can tackle both the cases, we assume that the node correspondence among $(\mathcal{V}_1, \ldots, \mathcal{V}_T)$ is unavailable. The proposed ICD framework follows the independent and identically distributed (i.i.d.) hypothesis adopted in the simulation of various network systems [47] that graphs in $\mathcal{S}$ are independently generated via common underlying distributions of a system or scenario.

Combinatorial Optimization of CD. Given a graph $\mathcal{G}_t$ and a pre-set number of communities $K_t$, CD aims to partition the node set $\mathcal{V}_t$ into $K_t$ subsets (i.e., communities) $C_t = \{C_{t,1}, \ldots, C_{t,K_t}\}$ so that (i) within each community the linkage is dense but (ii) between different communities the linkage is relatively loose. $C_t$ also satisfies the disjoint constraint, i.e., $\forall r \neq s \text{ s.t. } C_{t,r} \cap C_{t,s} = \emptyset$. CD can be formulated as several NP-hard combinatorial optimization problems, e.g., NCut minimization [41] and modularity maximization [27].

Given $\mathcal{G}_t$ and $K_t$, NCut minimization aims to get the CD result $C_t$ that minimizes the following NCut metric:

$$\arg \min_{C_t} \text{NCut}(C_t) = \frac{1}{2} \sum_{r=1}^{K_t} \left[ \text{cut}(C_{t,r}, \tilde{C}_{t,r}) / \text{vol}(C_{t,r}) \right]$$

where $\tilde{C}_{t,r} = \mathcal{V}_t - C_{t,r}$ is the complementary set of $C_{t,r}$; $\text{cut}(C_{t,r}, \tilde{C}_{t,r}) = \sum_{v_{t,i} \in C_{t,r}, v_{t,j} \in \tilde{C}_{t,r}} (A_t)_{ij}$ is defined as the cut between $C_{t,r}$ and $\tilde{C}_{t,r}$; $\text{vol}(C_{t,r}) = \sum_{v_{t,i} \in C_{t,r}, v_{t,j} \in \mathcal{V}_t} (A_t)_{ij}$ is the volume of $C_{t,r}$. The objective (1) can be equivalently expressed in the following matrix form:

$$\arg \min_{H_t} \text{tr}(H_t^T L_t H_t) \text{ s.t. } H_t^T H_t = I_{K_t},$$

where $L_t = I_{N_t} - D_t^{-0.5} A_t D_t^{-0.5}$ is the normalized Laplacian matrix of $A_t$; $D_t = \text{diag}(d_{t,1}^2, \ldots, d_{t,N_t}^2)$ is a diagonal matrix with $d_{t,i} = \sum_{j} (A_t)_{ij}$; $I_{N_t}$ is an $N$-dimensional identity matrix. $H_t \in \mathbb{R}^{N_t \times K_t}$ is the membership indicator, where $(H_t)_{ir} = (d_{t,i} \cdot \text{vol}(C_{t,r}^{-1}))^{0.5}$ if $v_{t,i} \in C_{t,r}$ and $(H_t)_{ir} = 0$ otherwise.

Modularity maximization is another classic NP-hard objective of CD. Given $\mathcal{G}_t$ and $K_t$, it aims to find a partition $C_t$ that maximizes the following modularity metric:

$$\arg \max_{C_t} \text{Mod}(C_t) = \frac{1}{2e} \sum_{r=1}^{K_t} \sum_{v_{t,i} \in C_{t,r}} [(A_t)_{ij} - d_{t,i}^2 d_{t,j}^2 / (2e)],$$

where $e = \sum_{i} d_{t,i}^2 / 2$ is the number of edges. The objective (3) can also be rewritten in a matrix form:

$$\arg \min_{H_t} \text{tr}(H_t^T Q_t H_t) \text{ s.t. } \text{tr}(H_t^T H_t) = N_t,$$
where \( Q \in \mathbb{R}^{N \times N} \) is the modularity matrix with \((Q)_{ij} = (A_t)_{ij} - d_i^t d_j^t / (2e)\). \( H_t \in \mathbb{R}^{N_t \times K_t} \) is the membership indicator, where \((H_t)_{ir} = 1\) if \( v_r^t \in C_i^t \) and \((H_t)_{ir} = 0\) otherwise.

**Quality Evaluation Criteria.** The NP-hardness of aforementioned objectives implies that there are so far no polynomial-time algorithms to obtain the optimal solutions. In real applications, we usually use relaxed algorithms (e.g., spectral clustering \[41\] for NCut minimization) to derive feasible CD results and do not expect the optimality. When ground-truth of real applications (e.g., cells in wireless cellular network decomposition) is available, which may not be optimal for a CD objective but is highly related to application requirements, one can evaluate CD quality by measuring the correspondence between CD results and ground-truth, where better correspondence implies higher quality. Moreover, the derived objective values of NCut minimization and modularity maximization can also be unsupervised quality metrics, where smaller NCut and larger modularity implies higher quality.

**Transductive and Inductive Graph Embedding.** For a graph \( G_t \), conventional transductive graph embedding learns a function \( f : \{v_r^t\} \mapsto \{u_r^t \in \mathbb{R}^{1 \times k}\} \) that maps each node \( v_r^t \) to a \( k \)-dimensional vector \( u_r^t \), where \( f \) is optimized only for \( G_t \) without parameter sharing across other graphs. In particular, the learned embedding space is expected to preserve the key properties of \( G_t \), where a node pair \( (v_r^t, v_j^t) \) with similar properties (e.g., in the same community) should have similar representations \( (u_r^t, u_j^t) \). The derived embeddings \( \{u_r^t\} \) can be used as the input of several downstream tasks on \( G_t \) including CD. For each new graph \( G_t' \), transductive methods should optimize \( f \) from scratch.

To alleviate the NP-hard challenge of CD, we consider an inductive graph embedding scheme, where parameters of \( f \) are shared by all the graphs in \( S \). We divide \( S \) into a training set \( \Gamma \subset S \) and a test set \( \Gamma' = S - \Gamma \), which represent the sets of historical known and newly generated graphs. We first train \( f \) on \( \Gamma \) in an offline way to fully capture properties (i.e., underlying distributions) of the system or scenario regardless of time. After the offline training, we directly obtain the embedding \( \{u_r^t\} \) from \( f \) for each new graph \( G_t' \in \Gamma' \) with model parameters fixed, which significantly save the inference time. The online CD on \( G_t' \) is the downstream task, where we assume the number of communities \( K_t' \) is given. Finally, we apply a clustering algorithm (e.g., KMeans) to \( \{u_r^t\} \) for the fast high-quality online CD.

**Sources of Training Labels.** Although graph embedding and CD are unsupervised, we assume the permutation invariant CD result \( C_t \) (i.e., label information) of each graph \( G_t \in \Gamma \) is available in the offline training, which can be used to enhance the offline embedding optimization. Note that we do not need to ensure the training labels \( \{C_t|G_t \in \Gamma\} \) are optimal for a specific CD objective. Consistent with our evaluation criteria of CD quality, \( \{C_t|G_t \in \Gamma\} \) can be from (i) ground-truth of real applications (e.g., cells in a wireless cellular network) or (ii) results of a strong (but usually time-consuming) baseline with good objective values (e.g., small NCut or large modularity). Namely, we can also use the CD results of a strong baseline to regularize the offline embedding optimization.

**4 METHODOLOGY**

For a better trade-off between quality and efficiency of CD, we propose a novel ICD method with an overview shown in Fig. 1 (b) including the (i) offline training of an adversarial dual GNN model on historical known graphs and (ii) online generalization to newly generated graphs via an inductive graph embedding scheme. Fig. 2 gives a sketch of the adversarial dual GNN with a running example for the offline training. In the rest of this section, we elaborate on the (i) model architecture as well as the (ii) offline training and online generalization of ICD.

**4.1 Model Architecture**

Inspired by adversarial auto-encoder (AAE) \[24\], we introduce a novel adversarial dual GNN structure. As shown in Fig. 2, it includes (a) a feature extraction module, (b) a generator \( G \), and (c) a discriminator \( D \).

\[ Manuscript submitted to ACM \]
To the best of our knowledge, the adversarial dual GNN is our original design, which is different from existing AAE-based graph embedding methods [28] that regularize the learned embedding using a prior distribution. In contrast, our dual GNN incorporates the permutation invariant label information of historical graphs \( \{G_t \in \Gamma\} \) to the offline training via a novel adversarial process between (i) original training graphs \( \{G_t\} \) and (ii) corresponding auxiliary label-induced graphs \( \{G_t^{(g)}\} \).

(a) Feature Extraction Module. Most GNNs are originally designed for attributed graphs where each node \( v_i^{(t)} \) has a feature vector input described by the \( i \)-th row of a feature matrix \( X_t \). In this study, we adopt GNN as a basic building block of ICD but consider CD without attributes. Instead of using the classic settings of GNN for graphs without attributes (e.g., use a constant matrix as node features [49]), we extract additional structural features \( X_t \) based on the neighbor-induced similarity encoded in classic CD objectives of modularity maximization (4) and NCut minimization (2), resulting in two variants of ICD.

For modularity maximization (4), the modularity matrix \( Q_t \) encodes the neighbor similarity of each graph \( G_t \), so we let \( X_t = Q_t \). For NCut minimization (2), the Laplacian matrix \( L_t \) gives primary characteristics of graph structure, where \( M_t = D_t^{-0.5} A_t D_t^{-0.5} \) is the key component regarding neighbor similarity, so we let \( X_t = M_t \). Both \( Q_t \) and \( M_t \) are the reweighting of \( A_t \), where nodes \( \{v_i^{(t)}, v_j^{(t)}\} \) with similar neighbor-induced features \((X_i)_i, (X_j)_j\) are more likely to be partitioned into the same community. Prior work [39, 51], which uses \( Q_t \) or \( M_t \) to learn graph embeddings based on auto-encoders, has validated our motivation to use \( Q_t \) and \( M_t \) as informative structural features. Our experiments also demonstrate that the extraction of \( X_t \) is essential to supporting the high-quality online CD.

However, the feature dimensionality of \( X_t \in \{Q_t, M_t\} \) is the number of nodes \( N_t \), which may not be fixed over \( t \). Since the inductiveness of GNNs relies on the fixed feature dimensionality of \( X_t \), we introduce an efficient feature extraction module via the heavy-edge matching (HEM) graph coarsening [16], which enable ICD to tackle CD across graphs with non-fixed \( N_t \). It maps \( X_t \in \mathbb{R}^{N_t \times N_t} \) to another feature matrix \( Z_t \in \mathbb{R}^{N_t \times L} \) with fixed feature dimensionality \( L \). Algorithm 1 summarizes the feature extraction procedure.
When $N_t > L$, we first extract reweighted edges $E'_t^w = \{w(v'_t, v'_t) | w(v'_t, v'_t) = (X_t)_{ij}, (A_t)_{ij} = 1\}$ and then apply HEM to $E'_t^w$. For simplicity, we summarize the HEM procedure in Appendix A. Fig. 3 gives running examples of HEM w.r.t. the input graph in Fig. 2. Concretely, HEM merges the original graph $G_t$ with $N_t$ nodes into a supergraph $G'_t$ with $L$ supernodes via a greedy multi-level strategy of continuously merging the node pair with largest weight (in the original graph or induced supergraph) into a supernode. For instance, in Fig. 3, it merges a graph with 8 nodes into 2 supernodes $v'_2 = \{1, 2, 3, 4\}$ and $v'_2 = \{5, 6, 7, 8\}$ via the coarsening with two levels. HEM finally outputs a coarsening matrix $C_t \in \mathbb{R}^{N_t \times L}$, where $(C_t)_{ij} = |v'_i|^{-0.5}$ if node $v'_i$ is merged into supernode $v'_j$ and $(C_t)_{ij} = 0$ otherwise. We then let $Z_t = X_tC_t$ be the reduced features. Since $C_t$ is a sparse matrix, one can obtain $Z_t$ by setting its $j$-th column to $(Z_t)_{ij} = \sum_{i \in v'_j} (C_t)_{ij} (X_t)_{ij}$. When $N_t \leq L$, we set $Z_t = \{X_t, 0_{N_t \times (L-N_t)}\}$ via a padding strategy.

To the best of our knowledge, using HEM to extract feature input $Z_t$ for inductive GNNs is our original design. It is more efficient than the dimension reduction (e.g., PCA of $A_t$) used in existing GNN-based approaches [26, 48] since HEM is widely adopted in some fast transductive methods [9] with low complexity. The derived $Z_t$ is also more informative than the node features in some classic settings of inductive GNNs (e.g., let $Z_t$ be a constant matrix [49]). The extracted $Z_t$ is further fed into the generator $G$.

(b) Generator $G$. As highlighted in Fig. 4, $G$ consists of a feature encoder and an auxiliary label-induced encoder which are multi-layer GNNs with shared parameters $\delta_G$. Both the encoders have the topology and feature inputs described by an adjacency matrix and a feature matrix. Concretely, the feature encoder takes adjacency matrix $A_t$ (w.r.t. original graph $G_t$) and reduced features $Z_t$ as inputs, and derives graph embedding $U_t$. The label-induced encoder takes adjacency matrix $A_t^{(g)}$ (w.r.t. an auxiliary label-induced graph $G_t^{(g)}$) and $Z_t$ as inputs, and outputs label-induced embedding $U_t^{(g)}$. For simplicity, we denote the feature and label-induced encoders as $U_t = G(A_t, Z_t; \delta_G)$ and $U_t^{(g)} = G(A_t^{(g)}, Z_t; \delta_G)$.

Algorithm 1: Feature Extraction of ICD

\begin{algorithm}
\begin{algorithmic}[1]
\caption{Feature Extraction of ICD}
\State \textbf{Input}: input graph $G_t$; neighbor-induced features $X_t$; number of nodes $N_t$; reduced dimensionality $L$
\State \textbf{Output}: reduced features $Z_t \in \mathbb{R}^{N_t \times L}$
\If{$N_t > L$}
\State get supernode membership $V'_t$ & coarsening matrix $C_t$ via HEM (see Algorithm 4 in Appendix A)
\For{each supernode $v'_t \in V'_t$}
\State Initialize $j$-th column of $Z_t$: $(Z_t)_{., j} \leftarrow 0_{N_t \times 1}$
\For{each node $v_t \in v'_t$}
\State $(Z_t)_{ij} \leftarrow (Z_t)_{ij} + (C_t)_{ij} (X_t)_{ij}$
\EndFor
\EndFor
\Else \If{$N_t \leq L$}
\State Pad $X_t$ with zeros: $Z_t \leftarrow [X_t, 0_{N_t \times (L-N_t)}]$
\EndIf
\EndIf
\end{algorithmic}
\end{algorithm}

Fig. 4. Details of the generator $G$ w.r.t. the running example in Fig. 2.
In the offline training, we assume that the topology and CD result (used as training 'ground-truth') of each historical graph \( G_t \in \Gamma \) are available, which are described by an adjacency matrix \( A_t \in \mathbb{R}^{N_t \times N_t} \) and an indicator matrix \( R_t \in \mathbb{R}^{N_t \times K_t} \). We define \((R_t)_ir = 1\) if node \( v'_i \) is in community \( C_t^r \) by the training 'ground-truth' and \((R_t)_ir = 0\) otherwise. Due to the disjoint constraint of CD, only one entry in each row of \( R_t \) is 1 with other entries in the same row set to 0. In addition to the original graph \( G_t \), an auxiliary label-induced graph \( G_t^{(g)} \), with \( A_t^{(g)} = R_t R_t^T \) as the adjacency matrix describing its topology, is introduced to encode the structure of 'ground-truth' based on the following **Fact 1**.

**Fact 1** A for a graph \( G_t \) with \( K_t \) communities, its auxiliary label-induced graph \( G_t^{(g)} \) has \( K_t \) fully connected components with each component corresponding to one unique community of \( G_t \). Namely, there is an edge between nodes \( v'_i \) and \( v'_j \) in \( G_t^{(g)} \) only when they are in the same community.

The proof of **Fact 1** is given in Appendix B. For the example in Fig. 4 with 8 nodes \( \{1, 2, \cdots, 8\} \), \( G_t^{(g)} \) has 2 fully connected components \( \{1, 2, 3, 4\} \) and \( \{5, 6, 7, 8\} \) w.r.t. the 2 communities in the given CD 'ground-truth' of \( G_t \).

Let \( F_t^{(l-1)} \) and \( \hat{F}_t^{(l)} \) be the input and output of the \( l \)-th GNN layer of the feature or label-induced encoder in \( G \) with \( F_t^{(0)} = Z_t \). The \( l \)-th GNN layer is defined as

\[
F_t^{(l)} = \tanh(\hat{D}_t^{-0.5} \hat{A}_t \hat{D}_t^{-0.5} F_t^{(l-1)} W_G^{(l)}),
\]

where \( \hat{A}_t = A'_t + I_{N_t} \) (\( A'_t \in \{A_t, A_t^{(g)}\} \)) is the adjacency matrix with self-edges; \( \hat{D}_t = \text{diag}(d_t^1, \cdots, d_t^{N_t}) \) is the diagonal degree matrix; \( W_G^{(l-1)} \) is the trainable parameter shared by the two encoders; \( (F_t^{(l)})_i \) is the latent feature of node \( v'_i \). In particular, \((F_t^{(l)})_i \) is the nonlinear aggregation of features of \( \{v'_i\} \cup n(v'_i) \) from the previous layer, where \( n(v'_i) \) is the neighbor set of \( v'_i \). It is also known as the message passing of GNN, where neighbors of \( v'_i \) propagate their features to \( v'_i \) for aggregation. By **Fact 1**, \( A_t^{(g)} \) ensures that message passing of the label-induced encoder only occurs in each connected component w.r.t. each community in the 'ground-truth' (e.g., 2 components as highlighted in Fig. 4), while message passing of the feature encoder occurs in the original topology of \( G_t \). Here we use GCN [19] as an example building block of the two encoders. One can easily extend ICD to include other advanced inductive GNNs, e.g., GAT [40] and GIN [49].

The last layers of the feature and label-induced encoders output the learned graph embedding \( U_t \in \mathbb{R}^{N_t \times k} \) and auxiliary label-induced embedding \( U_t^{(g)} \in \mathbb{R}^{N_t \times k} \), where \( U_t^{(g)} \) preserves more informative label-induced properties than \( U_t \). We further use the non-linear inner product of \( U_t \) to reconstruct the neighbor-induced features \( X_t \):

\[
\hat{X}_t = \tanh(U_t U_t^T) .
\]

The derived embeddings \( \{U_t, U_t^{(g)}\} \) are then in turns fed into the discriminator \( D \) to support the adversarial process between \( G \) and \( D \), while \( \hat{X} \) is further used to regularized \( U_t \) in the offline optimization.

**c) Discriminator** \( D, D \) is an auxiliary classifier to distinguish \( U_t^{(g)} \) from \( U_t \), while \( G \) tries to generate plausible embedding \( U_t \) to fool \( D \). Such an adversarial process helps \( G \) to generate embedding \( U_t \) close to \( U_t^{(g)} \) which encodes the structures of training 'ground-truth'.

For simplicity, we denote \( D \) as \( y_t = D(S_t; \delta_D) \), where \( S_t \in \{U_t, U_t^{(g)}\} \) and \( \delta_D \) are embedding input and set of model parameters; \( y_t \in \mathbb{R}^{N_t} \) is a vector with \( (y_t)_i \) as the probability that \((S_t)_i = (U_t^{(g)})_i \) rather than \((S_t)_i = (U_t)_i \). Let \( P_t^{(l-1)} \) and \( \hat{P}_t^{(l)} \) be input and output of the \( l \)-th layer in \( D \) with \( P_t^{(0)} = S_t \). The \( l \)-th layer of \( D \) is defined as

\[
\hat{P}_t^{(l)} = \text{ReLU}(P_t^{(l-1)} W_D^{(l-1)} + b_D^{(l-1)}),
\]

where \( \{W_D^{(l-1)}, b_D^{(l-1)}\} \) are trainable model parameters. In particular, we use sigmoid as activation function of the last layer instead of ReLU.
4.2 Offline Training and Online Generalization

In this subsection, we elaborate on the offline training and online generalization procedures of ICD across a set of graphs \( S = \{ G_1, \ldots, G_T \} \). For simplicity, we divide \( S \) into a training set \( \Gamma_T \), a validation set \( \Gamma_V \), and a test set \( \Gamma' \), where \( \Gamma = \Gamma_T \cup \Gamma_V \) and \( \Gamma' \) represent the sets of historical known and newly generated graphs, respectively.

The offline training of ICD is based on three objectives of (i) adversarial learning (AL), (ii) feature reconstruction (FR), and (iii) clustering regularization (CR).

The AL objective tries to incorporate the permutation invariant label information of historical graphs to the offline embedding optimization via an adversarial process between \( D \) and \( G \). On the one hand, \( D \) tries to distinguish \( U_t^{(g)} \) from \( U_t \). The objective of \( D \) w.r.t. a graph \( G_t \) is

\[
\min_{\delta_D} L_D(G_t) = -\sum \log(1 - D(U_t)) + \sum \log D(U_t^{(g)})) / N_t
\]

(8)

On the other hand, \( G \) tries to fool \( D \) by minimizing the following loss w.r.t. a graph \( G_t \):

\[
L_{AL}(G_t) = -\sum \log D(U_t)) / N_t.
\]

(9)

Such an adversarial process directs \( G \) to output the embedding \( U_t \) close to \( U_t^{(g)} \), enabling \( G \) to capture the permutation invariant training labels of historical graphs.

The FR loss further forces \( G \) to derive embedding \( U_t \) that preserves key properties of the structural features \( X_t \) by minimizing the reconstruction error between \( X_t \) and \( X_t' \):

\[
L_{FR}(G_t) = \|\tilde{X}_t - X_t\|^2_F = \|\text{tanh}(U_t U_t^T) - X_t\|^2_F.
\]

(10)

In addition to the AL loss, ICD can also capture the permutation invariant training labels via the regularization of some classic CD objectives (e.g., modularity maximization and NCut minimization) corresponding to the two variants (with \( X_t = Q_t \) and \( X_t = M_t \)) as described in Section 4.1. We introduce the CR objective w.r.t. a graph \( G_t \) that minimizes the following loss:

\[
L_{CR}(G_t) = -\text{tr}(H_t^T \tilde{X}_t H_t),
\]

(11)

where \( H_t \) is the membership indicator encoding the training ‘ground-truth’ with same definitions as (4) and (2). Since the constraints on \( H_t \) (i.e., \( \text{tr}(H_t^T H_t) = N_t \) and \( H_t^2 = I_{N_t} \)) are always satisfied for the given ‘ground-truth’, we do not need to consider discrete constraints on \( H_t \). To the best of our knowledge, we are the first to use the CR objective to incorporate permutation invariant label information (encoded by \( H_t \) of training data) to the unsupervised embedding learning, whereas \( H_t \) is the parameter to be optimized or output in existing CD methods [26, 48].

Finally, we derive the objective of \( G \) w.r.t. each \( G_t \in \Gamma_T \) by combining (9), (10), and (11):

\[
\min_{\delta_G} L_G(G_t) = L_{AL}(G_t) + \alpha L_{FR}(G_t) + \beta L_{CR}(G_t).
\]

(12)

where \( \{\alpha, \beta\} \) are parameters to balance \( L_{FR} \) and \( L_{CR} \).

In the joint offline optimization of \( D \) and \( G \), we use the Xavier method [13] to initialize model parameters \( \{\delta_D, \delta_G\} \). The Adam optimizer is applied to iteratively update \( \delta_D \) and \( \delta_G \) based on gradients of (8) and (12). In each epoch, we randomly sample a certain number \( p \) of historical graphs from \( \Gamma_T \) to update \( \{\delta_D, \delta_G\} \). For a selected training sample in each epoch, we can repeatedly updating model parameters \( m \geq 1 \) times. Finally, we save the model parameters \( \{\delta_D^*, \delta_G^*\} \) with the best average CD quality on \( \Gamma_V \) within a pre-set number of epochs \( n \), based on the quality evaluation criteria defined in Section 3.
Algorithm 2: Offline Training of ICD

Input: training set $\Gamma_T$, validation set $\Gamma_V$; feature dimensionality $L$; hyper-parameters $\{\alpha, \beta, \rho, m\}$; learning rates $\{\eta_D, \eta_C\}$; number of epochs $n$
Output: optimized model parameters $\{\delta_U^*, \delta_D^*\}$

1. initialize model parameters $\{\delta_G, \delta_D\}$ via Xavier method
2. for epoch from 1 to $n$
   3. for sample_count from 1 to $p$
      4. randomly sample $G_t \in \Gamma_T$
      5. get structural features $X_t$ & indicator $H_t$ w.r.t. $G_t$
      6. get reduced features $Z_t$ from $(A_t, X_t)$ via Algorithm 1
      7. for update_count from 1 to $m$
         8. update $\delta_D$ via $\delta_D \leftarrow \text{Opt}(\eta_D, \delta_D, \partial L_D / \partial \delta_D)$ with $\delta_G$ fixed
         9. update $\delta_G$ via $\delta_G \leftarrow \text{Opt}(\eta_G, \delta_G, \partial L_G / \partial \delta_G)$ with $\delta_D$ fixed
   10. for each $G_t \in \Gamma_V$
      11. get structural features $X_t$ w.r.t. $G_t$
      12. get reduced features $Z_t$ from $(A_t, X_t)$ via Algorithm 1
      13. get graph embedding $U_t$ from feature encoder
      14. apply KMeans to $U_t$ to get CD result $\tilde{C}_t$
      15. evaluate CD quality $q$ w.r.t. $\tilde{C}_t$
      16. update average quality $q$ on $\Gamma_v$
      17. if have better $q$ then
         18. Update best model param.: $\{\delta_U^*, \delta_D^*\} \leftarrow \{\delta_U, \delta_D\}$

Algorithm 3: Online Generalization of ICD

Input: new graph $G_t \in \Gamma'$, feature dimensionality $L$, number of communities $K'$
Output: online CD result $\tilde{C}_t$

1. get structural features $X_t$ w.r.t. $G_t$
2. get reduced features $Z_t$ from $(A_t, X_t)$ via Algorithm 1
3. get graph embedding $U_t$ from feature encoder
4. apply KMeans to $U_t$ to get CD result $\tilde{C}_t$

After the offline training, we generalize ICD to new graphs for fast online CD. Concretely, for each new graph $G_t \in \Gamma'$, we derive embedding $U_t$ by passing $(A_t, X_t)$ forward the feature encoder of $G$ with its parameters $\delta_U^*$ fixed.

Since we assume the number of communities $K_t$ is given, we apply KMeans to $U_t$, which outputs the CD result w.r.t. the given $K_t$. Hence, the runtime of the online CD on each graph $G_t \in \Gamma'$ includes (i) the feature extraction of $Z_t$, (ii) one feedforward propagation through the feature encoder, and (iii) downstream clustering.

In summary, we conclude the offline training and online generalization procedures of ICD in Algorithm 2 and 3.

For each graph $G_t$, the complexity to get reweighted edges $E_t^{\gamma'}$ is $O(|E_t|)$. When $N_t > L$, there are in total $(N_t - L)$ merging operations for graph coarsening. In each level of graph coarsening, we continuously select the edge with largest weight to merge all the possible node pair, where the number of edges in a new level is much less than that in the previous level, i.e., $|E_t^{(k)}| > |E_t^{(k+1)}|$. The total cost of selecting edges with largest weights is no more than $O(|E_t| + |E_t^{(0)}||E_t^{(0)}| + |E_t^{(1)}||E_t^{(1)}| + \cdots) = O(|E_t||\log|E_t||))$. Hence, the complexity of graph coarsening (i.e., Algorithm 4) is no more than $O((N_t - L) + |E_t||\log|E_t||))$. Moreover, the complexity to derive $Z_t$ via $X_t$ and $C_t$ is no more than $O(N_t L n^*)$, with $n^*$ as the maximum number of nodes merged into a supernode. In summary, the total
complexity to derive the reduced feature matrix $Z_t$ based on $X_t$ is no more than $O((N_t - L) + |E_t| \log |E_t| + N_t L n^*) = O(|E_t| \log |E_t| + N_t L)$. We extract $M_t$ and $Q_t$ from $A_t$, we first derive node degrees $d_{t} = \{d_{1}^{t}, \ldots, d_{N_t}^{t}\}$ with cost of $O(|E_t|)$ by utilizing the sparsity of $A_t$. Since $M_t$ is also a sparse matrix, the complexity to derive $M_t$ is no more than $O(|E_t|)$. Although the complexity to derive $Q_t$ is $O(N_t^2)$, we use the efficient matrix multiplication $Q_t = A_t - d_{t}^2 d_{t} / (2w)$, which can be easily paralleled via GPUs with significant speedup rate.

Let $L^{t-1}$ be the feature dimensionality of the $t$-th GNN layer of encoder in $G$, with $L^{t} = L$. We used the efficient sparse-dense matrix multiplication to implement the graph convolutional operation described in (5). The complexity of one feedforward propagation through the feature encoder is no more than $O(|E_t| \log |E_t| + |E_t| L^{t-1}) = O(|E_t| L^{t-1})$, with $L' = L^{t-1}$ and $L' < L^{t-1}$. Hence, the complexity of one feedforward propagation is linear in $|E_t|$ w.r.t. $G_t$. One can further speed up the feedforward propagation using GPUs.

Let $k$ be the dimensionality of graph embedding. For a graph $G_t$ with $K_t$ communities, the complexity of downstream $K$Means clustering is no more than $O(N_t K_t kI)$ for $I$ iterations. The runtime of $K$Means can also be significantly reduced via parallel implementations.

5 EXPERIMENTS

In this section, we elaborate on our experiments. Concretely, we first introduce the experiment setup (e.g., datasets, baselines, and metrics) in Section 5.1. Evaluation results of CD quality and efficiency are analyzed in Section 5.2. Based on the evaluation results, we quantitatively compare the trade-off between quality and efficiency via a novel trade-off score in Section 5.3. Further experiments of ablation study and convergence analysis are given in Section 5.4 and 5.5.

5.1 Experiment Setup

Datasets. We evaluate ICD on 2 synthetic benchmarks (with 6 settings) and 4 real datasets. Statistics of the datasets are depicted in Table 2, where $T$, $N$, $|E|$, and $K$ are the number of graphs, nodes, edges, and communities.

| T          | N         | $|E|$ | $K$ | Avg $|E|$ | Avg $K$ | Avg $E$ | Avg $K$ | Description                  |
|------------|-----------|------|-----|----------|---------|---------|---------|-------------------------------|
| GN-Net     | 2,000     | 5,000| 5,000| 5,000    | 48,355  | 250     | 250     | Fixed N, Fixed K, w/ ground-truth |
| GN-Net     | 2,000     | 5,000| 5,000| 5,000    | 48,468  | 49,981  | 49,246  | Fixed N, Non-fixed K, w/ ground-truth |
| Random     | 2,000     | 5,000| 5,000| 5,000    | 22,559  | 25,707  | 23,923  | Fixed N, Non-fixed K, w/ ground-truth |
| Random     | 2,000     | 5,000| 5,000| 5,000    | 22,585  | 25,639  | 23,923  | Fixed N, Non-fixed K, w/ ground-truth |
| Random     | 2,000     | 5,000| 5,000| 5,000    | 22,999  | 29,016  | 28,311  | Fixed N, Non-fixed K, w/ ground-truth |
| Real       | 3,000     | 1,279| 1,279| 1,279    | 18,584  | 39,713  | 39,291  | Fixed N, Non-fixed K, w/o ground-truth |
| Enron      | 410       | 1,031| 1,048| 1,048    | 314     | 5,097   | 1,176   | Non-fixed N, Non-fixed K, w/o ground-truth |
| AS         | 733       | 103  | 1,647| 4,183    | 487     | 26,469  | 16,324  | Non-fixed N, Non-fixed K, w/o ground-truth |

$\text{LFR-Net}$ [20] is a more challenging synthetic benchmark for CD that can simulate the power-law properties of real-world network systems. It uses $(d, d_{\text{max}}, c_{\text{min}}, c_{\text{max}}, \mu)$ to independently generate each graph, where $d$ and $d_{\text{max}}$
are the average and maximum node degree; $c_{\text{min}}$ and $c_{\text{max}}$ are the minimum and maximum community size; $\mu$ is the mixing ratio between the external degree and total degree of each node $v_i$ w.r.t. the community $v_i$ belongs to. We fixed $(d, d_{\text{max}}, c_{\text{min}}, c_{\text{max}}) = (10, 100, 10, 200)$ and set $\mu \in \{0.3, 0.6\}$. With the increase of $\mu$, the community structures are increasingly difficult to identify. We consider both cases with fixed and non-fixed $N$, where we respectively fixed $N = 5,000$ and randomly set $N \in [5000, 6000]$ to independently generate 2,000 graphs for each case. For simplicity, we use $L(c, \mu)$ ($c \in \{f, n\}$) to denote the dataset with a setting of $\mu$, where $f$ and $n$ represent the case with fixed and non-fixed $N$.

**Taxi**\textsuperscript{1} [32] is a real GPS dataset including trajectories of 1,279 taxis in Beijing, which forms a mobile ad hoc network system. We independently sampled 3,000 graphs from the dataset. In each graph, we treated each taxi as a node and constructed the topology based on the top-50 neighbors with the closest distance for each node.

**Reddit**\textsuperscript{2} [50] is a social network dataset from Reddit. We extracted 1,870 graphs with $N \geq 500$ from the dataset, where each graph corresponds to an online discussion thread. The comment interactions between users in each thread are described as the topology of each graph.

**Enron**\textsuperscript{3} [37] is a real interaction network extracted from the email system of Enron company, which contains email records from 1980-01-01 to 2004-02-04. We extracted 410 graphs with $N \geq 500$ from the records where we treated each user as a unique node and constructed the topology based on email interactions.

**AS**\textsuperscript{4} [21] is a dataset from the border gateway protocol (BGP) logs of a communication network, which describe the topology between a set of autonomous systems. It contains 733 daily instances from 1997-08-11 to 2000-01-02. Each BGP router is abstracted as a unique node with the communication between routers extracted as graph topology.

In summary, the 2 datasets of GN-Net are with fixed $N$ and $K$. The LFR-Net covers cases of (i) fixed $N$ but non-fixed $K$ and (ii) non-fixed $N$ and $K$. Taxi is with fixed $N$ but non-fixed $K$, while Reddit, Enron, and AS are with non-fixed $N$ and $K$. GN-Net and LFR-Net are synthetic benchmarks that can simultaneously generate graph topology and CD ground-truth of each graph, while Taxi, Reddit, Enron, and AS are real datasets without ground-truth.

**Baselines.** We compared ICD over 17 baselines which can be divided into 2 categories of transductive and inductive approaches. Table 3 gives an overview of these approaches.

| Trans | SNMF | SC | GraClus | MC-SBM | Local | LocNet | N2V | M-NMF | DNR | LFR-Net | RandNE | NRP | ProNE | MNMF | DNR | NECS | RandNE | MCSBM | Locale | N2V | ClusNet | ICD |
|-------|------|----|---------|--------|-------|--------|-----|-------|-----|---------|--------|-----|-------|------|-----|------|--------|-------|--------|-----|--------|-----|
| Ind   | V    | V  | V       | V      | V     | V      | V   | V     | V   | V       | V      | V  | V     | V    | V   | V     | V      | V    | V      | V   | V      | V   |
| Emb   | V    | V  | V       | V      | V     | V      | V   | V     | V   | V       | V      | V  | V     | V    | V   | V     | V      | V    | V      | V   | V      | V   |

\textsuperscript{1}https://www.microsoft.com/en-us/research/publication/t-drive-driving-directions-based-on-taxi-trajectories/

\textsuperscript{2}https://github.com/weihua916/powerful-gnns/blob/master/dataset.zip

\textsuperscript{3}http://networkrepository.com/ia-enron-email-dynamic.php

\textsuperscript{4}http://snap.stanford.edu/data/as-733.html

Manuscript submitted to ACM
Both variants of ICD were also evaluated. For simplicity, we use ICD-M and ICD-C to denote the variants adopting objectives of modularity maximization and NCut minimization, respectively.

**Evaluation Metrics.** In our evaluation, we consider CD with given $K$. By the quality evaluation criteria in Section 3, we adopted quality metrics of modularity (3) and NCut (1) for all the datasets. For datasets with ground-truth (i.e., GN-Net and LFR-Net), we also used metrics of normalized mutual information (NMI) and accuracy (AC) to measure the correspondence between CD results and ground-truth. For a given graph $G_t$ with $K_t$ communities, we can use $H_t = \{H^t_1, \ldots, H^t_K_t\}$ and $C_t = \{C^t_1, \ldots, C^t_{K_t}\}$ to denote the ground-truth and CD result with $H^t_i \subseteq V_t$ and $C^t_j \subseteq V_t$ as node member sets w.r.t. the $r$-th community. Given $H_t$ and $C_t$, NMI is defined as

$$
NMI(H_t, C_t) = \frac{-2 \sum_{H^t_i \in H_t} \sum_{C^t_j \in C_t} \frac{n_{r,s}}{|n|} \log \frac{n_{r,s} n_{r,s}}{|n| |n|} + \sum_{C^t_j \in C_t} \frac{n_{s}}{|n|} \log \frac{n_{s} n_{s}}{|n|} \cdot \frac{1}{n_t},
$$

(13)

where $n_t = |H^t_i|$, $n_s = |C^t_j|$, $n_{r,s} = |H^t_i \cap C^t_j|$, and $n = N_t$. Let $L_t = (l^t_1, \ldots, l^t_{N_t})$ and $R_t = (r^t_1, \ldots, r^t_{N_t})$ be label sequences of CD result and ground-truth w.r.t. node sequence $(v^t_1, \ldots, v^t_{N_t})$, where $l^t_i$ and $r^t_i$ are community labels of node $v^t_i$. AC is defined as

$$
AC(R_t, L_t) = \frac{\sum_{i=1}^{N_t} \delta(r^t_i, \text{map}(l^t_i))}{N_t},
$$

(14)

where $\text{map}(\cdot)$ is the Kuhn-Munkres mapping that gives the best map from $L_t$ to $R_t$; $\delta(a, b)$ is the Kronecker delta function with $\delta(a, b) = 1$ if $a = b$ and $\delta(a, b) = 0$ otherwise.

Usually, larger modularity, NMI and AC, as well as smaller NCut imply better quality. Moreover, the total runtime of a method to derive its final CD result was used as the efficiency metric with lower runtime indicating higher efficiency.

**Evaluation Settings.** All the transductive baselines are independently optimized to tackle CD on each single graph, while ICD ties to alleviate the NP-hard challenge via an inductive framework across graphs. To illustrate the superiority of ICD beyond transductive methods, for each dataset, we used the first 80% graphs as training set $\Gamma_t$ for the remaining 10% and 10% deployed as validation set $\Gamma_v$ and test set $\Gamma'_t$. In the offline training, we saved model parameters of ICD with the best average quality on $\Gamma_v$, where NMI and modularity are used as the validation quality metrics for datasets with and without ground-truth. Since we focus on the trade-off between quality and efficiency on new test graphs, the evaluation was conducted on $\Gamma'_t$, where we had to train transductive baselines from scratch for each graph in $\Gamma'_t$.

For inductive embedding baselines (i.e., GSAGE, GAT, and GIN), we use the same training and test settings with ICD, i.e., offline training on $\Gamma_t$ and online generalization to $\Gamma'_t$. Since we consider CD on graphs without attributes, we use a constant matrix $I_{N_t \times L}$, where each entry is set to 1, as the input features of these GNN-based methods, which is a widely-used setting of inductive GNNs for the case without attributes [49]. Since GAT [40] and GIN [49] do not give the recommended unsupervised training loss for node-level embedding, we used the unsupervised loss of GSAGE [15] to train GSAGE, GAT, and GIN on $\Gamma_t$ of each dataset. Note that most existing GNN-based embedding methods cannot directly integrate the permutation invariant training labels even using a supervised loss, e.g., cross-entropy.

For the E2E GNN-based methods (i.e., GAP and ClusNet), we could only use the same inductive settings as ICD on datasets with fixed $K$ (i.e., GN-Net) but had to train them from scratch on $\Gamma'_t$ of other datasets with non-fixed $K$, due to the fixed dimensionality of their E2E output layers. As recommended in [26] and [48], we used PCA and node2vec to map adjacency matrices $\{A_t\}$ to a feature space with fixed dimensionality for the GNN input features of GAP and ClusNet. Moreover, we used their own unsupervised training losses defined in [26] and [48].
For all the embedding methods, we used KMeans as the downstream clustering algorithm. We ran KMeans 10 times to avoid its limitation of getting locally optimal solutions, which was included in the total runtime. The embedding dimensionality of all the embedding methods was set to be the same on each dataset. For other E2E baselines, we got CD results directly from their outputs.

For datasets with ground-truth (i.e., GN-Net and LFR-Net), we directly used their CD ground-truth to derive the membership indicator $H_f$ for ICD, i.e., the first source of training labels defined in Section 3. Since Taxi, Reddit, Enron, and AS do not provide ground-truth of each training graph, we used CD results of the baseline with best average modularity on $\Gamma_V$ to derive $H_f$ for the offline training of ICD, i.e., the second source of training labels in Section 3. Note that we do not need to derive $H_f$ in the evaluation phase when generalizing the trained model to test set $\Gamma'$.

Some other details of experiment setups (e.g., parameter settings and layer configurations) are given in Appendix C.1.

### 5.2 Quantitative Evaluation

In our evaluation, both the mean and standard deviation of all the quality and efficiency metrics on $\Gamma'$ of each dataset were recorded. Evaluation results regarding the trade-off between each quality metric (e.g., average NMI, AC, modularity, and NCut) and the unique efficiency metric of average runtime are visualized in Fig. 5, 6, and 7. Concretely, we map each method to a data point in a 2D space with its $x$-axis and $y$-axis corresponding to the unique efficiency metric (i.e., runtime) and a quality metric. Number records of the mean and standard deviation w.r.t. each metric are given in the Appendix C.2. According to Fig. 5, 6, and 7, we have the following observations.

For all the datasets, ICD-M and ICD-C are in the top groups with best quality. In some cases, they even have the best or second-best quality metrics. Moreover, the runtime of ICD-M and ICD-C is also competitive to the transductive baselines with fast approximation (e.g., Graclus and RandNE) and other inductive GNNs with fast online generalization (e.g., GSAGE). From the view of alleviating the NP-hard challenge, ICD can achieve a significant trade-off between quality and efficiency of online CD. We further quantitatively evaluate the trade-off achieved by all the methods based on the visualization results of Fig. 5, 6, and 7 in Section 5.3.
Fig. 6. Evaluation results on LFR-Net w.r.t. quality metrics of NMI↑, AC↑, modularity↑, & NCut↓ (y-axis) and efficiency metric of runtime↓ (x-axis).

Although the inductive GNN based embedding baselines (i.e., GSAGE, GAT, and GIN) have slightly faster runtime than ICD, they suffer from poor quality on all the datasets. The results validate that the extracted features $Z_t$ and hybrid training loss of ICD, which incorporates the permutation invariant training labels, are essential to ensure high CD quality. On datasets with fixed $K$ (i.e., GN-Net), we used the same inductive settings for the E2E GNN-based methods (i.e., GAP and ClusNet). However, the online CD of GAP and ClusNet on GN-Net is still with poor quality, indicating that the graph embedding scheme in ICD is more robust than inductive E2E baselines. Moreover, we had to use the time-consuming transductive settings for GAP and ClusNet on datasets with non-fixed $K$, training them from scratch for each test graph. In contrast, ICD can still tackle the fast online CD with non-fixed $K$ via its graph embedding scheme.

In most cases, ICD-M and ICD-C have similar CD quality. ICD-M has better quality than ICD-C on datasets where community structures are difficult to identify (e.g., GN-0.3, $L(f, 0.6)$ and $L(n, 0.6)$). A possible reason is that the modularity matrix $Q_t$ is more informative than the normalized adjacency matrix $M_t$ to be structural features $X_t$ of ICD.
Fig. 7. Evaluation results on datasets without ground-truth w.r.t. quality metrics of modularity↑, & NCut↓ (y-axis) and efficiency metric of runtime↓ (x-axis).

Fig. 8. Intuitions of the trade-off score (TOS) based on the visualization results of CD quality (i.e., y-axis) and efficiency (i.e., x-axis).

5.3 Trade-Off Analysis

In this study, we focus on the trade-off between quality and efficiency of CD. However, to evaluate the trade-off between two aspects that conflict with each other using a comprehensive metric remains an open issue. We propose a novel trade-off score (TOS) to quantitatively evaluate the trade-off achieved by a method, based on the normalized area covered by an induced rectangle of a data point in the efficiency-quality visualization results like Fig. 5, 6, and 7. Intuitions of TOS are shown in Fig. 8. Our goal is to let larger TOS indicate a better trade-off between quality and efficiency.

In our evaluation, we used NMI, AC, modularity, and NCut as quality metrics, while we adopted the runtime of a method as the unique efficiency metric. Let $Q$ and $E$ be values of the quality and efficiency metrics achieved by a method. Given the tuple $(E, Q)$, each method can be mapped to a point in a 2D space as shown in Fig. 5, 6, and 7.

Since there is magnitude difference between $E$ and $Q$, we first normalize them into the range of $[0, 1]$. When using NMI or AC as the quality metric $Q$, we derive the normalized metric via $\tilde{Q} = Q/1$, since $Q \in [0, 1]$ and larger $Q$ indicates better quality. When using modularity as $Q$, we have $Q \in [-1, 1]$, where larger $Q$ implies better quality. We can normalize $Q$ via $\tilde{Q} = (Q + 1)/2$. When using NCut as $Q$, we have $Q \in [0, +\infty)$ with smaller $Q$ indicating better quality, so we normalize $Q$ via $\tilde{Q} = (Q_m - Q)/Q_m$, with $Q_m$ as the maximum NCut among all the methods. For the efficiency
Trading off Quality for Efficiency of Community Detection: An Inductive Method across Graphs

Table 4. TOS between quality metrics of NMI and AC as well as efficiency metric of runtime.

| NMI       | AC       | NMI | AC       | NMI | AC       | NMI | AC       | NMI | AC       |
|-----------|----------|-----|----------|-----|----------|-----|----------|-----|----------|
| Google    | 0.91     | 0.63 | 0.63     | 0.76 | 0.63     | 0.76 | 0.63     | 0.76 | 0.63     |
| SC        | 0.96     | 0.64 | 0.41     | 0.41 | 0.64     | 0.41 | 0.41     | 0.41 | 0.64     |
| Graclus   | 0.98     | 0.64 | 0.65     | 0.65 | 0.98     | 0.65 | 0.65     | 0.65 | 0.98     |
| GC-SBM    | 0.96     | 0.64 | 0.66     | 0.66 | 0.96     | 0.66 | 0.66     | 0.66 | 0.96     |
| NEC       | 0.99     | 0.64 | 0.50     | 0.50 | 0.99     | 0.50 | 0.50     | 0.50 | 0.99     |
| N-MNF     | 0.69     | 0.64 | 0.39     | 0.39 | 0.69     | 0.39 | 0.39     | 0.39 | 0.69     |
| DNR       | 0.87     | 0.64 | 0.45     | 0.45 | 0.87     | 0.45 | 0.45     | 0.45 | 0.87     |
| RandNE    | 0.78     | 0.64 | 0.58     | 0.58 | 0.78     | 0.58 | 0.58     | 0.58 | 0.78     |
| NRP       | 0.91     | 0.64 | 0.80     | 0.80 | 0.91     | 0.80 | 0.80     | 0.80 | 0.91     |
| ProNE     | 0.95     | 0.64 | 0.92     | 0.92 | 0.95     | 0.92 | 0.92     | 0.92 | 0.95     |

Table 5. TOS between quality metrics of modularity and NCut as well as efficiency metric of runtime.

| NMI       | AC       | NMI | AC       | NMI | AC       | NMI | AC       | NMI | AC       |
|-----------|----------|-----|----------|-----|----------|-----|----------|-----|----------|
| Google    | 0.51     | 0.54 | 0.54     | 0.54 | 0.51     | 0.54 | 0.51     | 0.54 | 0.51     |
| SC        | 0.61     | 0.64 | 0.61     | 0.64 | 0.61     | 0.64 | 0.61     | 0.64 | 0.61     |
| Graclus   | 0.62     | 0.65 | 0.62     | 0.65 | 0.62     | 0.65 | 0.62     | 0.65 | 0.62     |
| GC-SBM    | 0.63     | 0.65 | 0.63     | 0.65 | 0.63     | 0.65 | 0.63     | 0.65 | 0.63     |
| NEC       | 0.60     | 0.65 | 0.60     | 0.65 | 0.60     | 0.65 | 0.60     | 0.65 | 0.60     |
| N-MNF     | 0.46     | 0.65 | 0.46     | 0.65 | 0.46     | 0.65 | 0.46     | 0.65 | 0.46     |
| DNR       | 0.61     | 0.65 | 0.61     | 0.65 | 0.61     | 0.65 | 0.61     | 0.65 | 0.61     |
| RandNE    | 0.60     | 0.65 | 0.60     | 0.65 | 0.60     | 0.65 | 0.60     | 0.65 | 0.60     |
| NRP       | 0.62     | 0.65 | 0.62     | 0.65 | 0.62     | 0.65 | 0.62     | 0.65 | 0.62     |
| ProNE     | 0.63     | 0.65 | 0.63     | 0.65 | 0.63     | 0.65 | 0.63     | 0.65 | 0.63     |

The TOS values w.r.t. the trade-off between four quality metrics (i.e., NMI, AC, modularity, and NCut) and efficiency metric of runtime are depicted in Table 4 and 5. As described in Section 5.1, we used NMI and modularity as validation quality metrics for datasets with and without ground-truth, respectively. For the TOS w.r.t. NMI and modularity on datasets with and without ground-truth, ICD-M and ICD-C have the best or second-best TOS in most cases. For the rest

where $E$ is the metric $E$ (i.e., runtime), smaller $E$ implies higher efficiency with $E \in [0, +\infty)$, so we normalize $E$ via $\tilde{E} = (E_m - E) / E_m$, where $E_m$ is the maximum runtime among all the methods.

As highlighted in Fig. 8, when using NMI, AC, and modularity as quality metrics, a data point close to the top left corner in a visualization result indicates the corresponding method can achieve a better trade-off between quality and efficiency. For NCut, a data point close to the bottom left corner implies a better trade-off. Based on this intuition, we define TOS w.r.t. the given $(Q, E)$ as

$$\text{TOS}(Q, E) = \hat{Q} \times \tilde{E},$$

(15) which is the area covered by the induced rectangle illustrated in Fig. 8 with larger TOS indicating a better trade-off between quality and efficiency.

Manuscript submitted to ACM
cases (i.e., modularity) on datasets with ground-truth as well as AC and NCut on all the datasets), ICD-M and ICD-C also have the top-3 TOS values. In summary, ICD can achieve a significant trade-off between quality and efficiency of the online CD over various baselines.

5.4 Ablation Study

For ICD-M and ICD-C, we also validated the effectiveness of (i) AL loss, (ii) FR loss, (iii) CR loss, (iv) feature input $Z_t$, and (v) GNN by respectively excluding the corresponding components from the original model. In case (iv), we used a constant matrix $1_{N_t \times L}$ with all entries set to 1 to replace $Z_t$ (i.e., a standard setting of GNNs for graphs without attributes [49]), while we use a fully-connected network that only takes $Z_t$ as input to replace GNNs in $G$ with the same layer configurations. We conducted ablation studies on $L(f_{0.6}, L(n_{0.6})$, and $\text{GN-0.3}$ whose community structures are more difficult to identify than other datasets. In each case, the mean value $\mu$ and standard deviation $\sigma$ of all the quality metrics (on test set $\Gamma$) were recorded. Results in the format of $\mu(\sigma)$ are shown in Table 6.

According to Table 6, the FR loss, feature input $Z_t$, and GNN are key components to ensure the high quality of ICD because there are significant quality declines in cases without these three components. The AL and CR losses are components to further enhance the CD quality, as they can incorporate the permutation invariant training labels to the offline optimization of ICD. In summary, all the components are essential to ensure the high quality of ICD.

5.5 Convergence Analysis

We also analyzed the convergence of the offline training of ICD. In each epoch, we recorded the average CD quality (in terms of NMI, AC, modularity, and NCut) w.r.t. the learned embedding $\{U_t\}$ on the training set $\Gamma_t$ and validation set $\Gamma_V$ as well as the generalized embedding $\{U_t\}$ on the test set $\Gamma'$. To further validate the effectiveness of label-induced embedding $\{U_t^{(g)}\}$, the average CD quality of $\{U_t^{(g)}\}$ on $\Gamma_t$ was also recorded. We use results on $L(n_{0.6})$, whose community structures are difficult to identify, as an example. The convergence curves of the average quality w.r.t. (i) $\{U_t\}$ on $\Gamma_t$, (ii) $\{U_t\}$ on $\Gamma_V$, (iii) $\{U_t\}$ on $\Gamma_V$, and (iv) $\{U_t\}$ on $\Gamma'$ for ICD-M and ICD-C are shown in Fig. 9.

According to Fig. 9, the CD quality of $\{U_t\}$ continuously improves in the first several epochs on $\Gamma_t$, $\Gamma_V$, and $\Gamma'$. In particular, the average NMI and AC of $\{U_t^{(g)}\}$ on $\Gamma_t$ are always 1 for ICD-M and ICD-C. It indicates that the CD results derived from the auxiliary label-induced embedding $\{U_t^{(g)}\}$ have perfect mapping to the ground-truth. Therefore, $\{U_t^{(g)}\}$ can capture key information of the permutation invariant training ‘ground-truth’ of $\Gamma_t$ and further enhance the learned embedding $\{U_t\}$ by incorporating the training labels via an adversarial process. In summary, Fig. 9 validates the effectiveness of the offline training and adversarial dual GNN structure of ICD.
We intend to consider an topology 'mismatches' with attributes, we need to control the contribution of attributes to avoid quality declines. Attributes carry consistent characteristics with topology, we can fully utilize attributes to improve CD quality. When attributes may also carry noise and mismatched characteristics, resulting in unexpected quality declines. On the other hand, attributes may provide complementary information beyond topology to further improve the CD quality.

In this study, we considered inductive CD without attributes. Although ICD can be easily extended to include node attributes, prior research [33, 34] has demonstrated the complicated correlation between graph topology and attributes. On the one hand, attributes may provide complementary information beyond topology to further improve the CD quality. On the other hand, attributes may also carry noise and mismatched characteristics, resulting in unexpected quality declines. We intend to consider an adaptive incorporation scheme between these two heterogeneous sources for ICD. When attributes carry consistent characteristics with topology, we can fully utilize attributes to improve CD quality. When topology ‘mismatches’ with attributes, we need to control the contribution of attributes to avoid quality declines.

Fig. 9. Convergence curves of ICD-M and ICD-C on $L(n,6)$ w.r.t. NMI, AC, modularity, and NCut in the first 50 iterations (x-axis).

6 CONCLUSION

In this paper, we proposed an ICD method to alleviate the NP-hard challenge of CD, obtaining a better trade-off between quality and efficiency. We first conduct the offline training of an adversarial dual GNN on historical known graphs and generalize the model to newly generated graphs for fast high-quality online CD. ICD is a generic method that can tackle the inductive CD across graphs with non-fixed number of nodes and communities based on its inductive graph embedding scheme and efficient feature extraction module. It can also incorporate the permutation invariant training labels to the offline training by combining unsupervised CD objectives (e.g., modularity maximization and NCut minimization) with the adversarial dual GNN. Extensive experiments validate that ICD can achieve a significant trade-off between quality and efficiency over various baselines. In the rest of this section, we conclude this paper by highlighting some possible future research directions.

In this study, we considered inductive CD without attributes. Although ICD can be easily extended to include node attributes, prior research [33, 34] has demonstrated the complicated correlation between graph topology and attributes. On the one hand, attributes may provide complementary information beyond topology to further improve the CD quality. On the other hand, attributes may also carry noise and mismatched characteristics, resulting in unexpected quality declines. We intend to consider an adaptive incorporation scheme between these two heterogeneous sources for ICD. When attributes carry consistent characteristics with topology, we can fully utilize attributes to improve CD quality. When topology ‘mismatches’ with attributes, we need to control the contribution of attributes to avoid quality declines.
Scaling GNNs up to ultra-large graphs (in terms of the number of nodes) is a significant direction in recent research, where some sampling strategies are used to split a large graph into multiple subgraphs \cite{5,10,56} (i.e., mini-batches with small number of nodes). Note that ICD is an inductive framework across graphs, which can also be trained on and generalized to multiple subgraphs sampled from a single large graph. Existing GNNs that can be scaled up to large graphs usually focus on (semi-)supervised node-level tasks (e.g., node classification) with available node attributes, while this study considers inductive unsupervised node-level tasks (i.e., CD) across graphs without attributes. In our future work, we intend to explore efficient (i) feature extraction and (ii) subgraph sampling strategies for ICD on ultra-large graphs without attributes. A possible challenge is to ensure the consistency of embeddings \( \{ U_t \} \) of multiple subgraphs sampled from a common large graph (e.g., embeddings of all the subgraphs are mapped into a common latent space) without the auxiliary information of node attributes.

REFERENCES

\begin{thebibliography}{10}
\footnotesize
\setlength{\itemsep}{-1em}
\item [1] Emmanuel Abbe. 2017. Community detection and stochastic block models: recent developments. *Journal of Machine Learning Research* 18, 1 (2017), 6446–6531.
\item [2] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory & Experiment* 2008, 10 (2008), P10008.
\item [3] Mingming Chen, Konstantin Kuzmin, and Boledaw K Szymanski. 2014. Community detection via maximization of modularity and its variants. *IEEE Transactions on Computational Social Systems* 1, 1 (2014), 46–65.
\item [4] Zhengdao Chen, Linsha Li, and Joan Bruna. 2020. Supervised Community Detection with Line Graph Neural Networks. In *Proceedings of the 8th International Conference on Learning Representations*. 1705.08415.
\item [5] Wei-Lin Chiang, Xuangming Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. 2019. Cluster-gen: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 257–266.
\item [6] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. 2004. Finding community structure in very large networks. *Physical Review E* 70, 6 (2004), 066111.
\item [7] Lin Dai and Bo Bai. 2017. Optimal decomposition for large-scale infrastructure-based wireless networks. *IEEE Transactions on Wireless Communications* 16, 8 (2017), 4956–4969.
\item [8] Asim K Dey, Yahui Tian, and Yulia R Gel. 2022. Community detection in complex networks: From statistical foundations to data science applications. *Wiley Interdisciplinary Reviews: Computational Statistics* 14, 2 (2022), e1566.
\item [9] Inderjit S Dhillon, Yuqiang Guan, and Brian Kulis. 2007. Weighted graph cuts without eigenvectors: a multilevel approach. *IEEE Transactions on Pattern Analysis & Machine Intelligence* 29, 11 (2007), 1944–1957.
\item [10] Wenzheng Feng, Yuanzhe Dong, Tinglin Huang, Ziqi Yin, Xu Cheng, Evgeny Kharlamov, and Jie Tang. 2022. GRAND+: Scalable Graph Random Neural Networks. In *Proceedings of the ACM Web Conference 2022*. 3248–3258.
\item [11] Santo Fortunato and Mark EJ Newman. 2022. 20 years of network community detection. *Nature Physics* 18, 8 (2022), 848–850.
\item [12] Michelle Girvan and Mark EJ Newman. 2000. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences* 99, 12 (2002), 7821–7826.
\item [13] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 30th International Conference on Artificial Intelligence & Statistics*. 249–256.
\item [14] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 855–864.
\item [15] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Proceedings of Advances in Neural Information Processing Systems*. 1024–1034.
\item [16] Bruce Hendrickson and Robert W Leland. 1995. A Multi-Level Algorithm For Partitioning Graphs. *Proceedings Supercomputing* 95, 28 (1995), 1–14.
\item [17] Di Jin, Zhiyi Yu, Pengfei Jiao, Shurui Fan, Dongxiao He, Jia Wu, Philip Yu, and Weinong Zhang. 2021. A survey of community detection approaches: From statistical modeling to deep learning. *IEEE Transactions on Knowledge & Data Engineering* (2021).
\item [18] Brian Karrer and Mark EJ Newman. 2011. Stochastic blockmodels and community structure in networks. *Physical Review E* 83, 1 (2011), 016107.
\item [19] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of the 5th International Conference on Learning Representations*. 1609.02907.
\item [20] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. 2008. Benchmark graphs for testing community detection algorithms. *Physical Review E* 78, 4 (2008), 046110.
\item [21] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. 2005. Graphs over time: densification laws, shrinking diameters and possible explanations. In *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*. 177–187.
\end{thebibliography}

Manuscript submitted to ACM
Algorithm 4: HEM Graph Coarsening

Input: input graph $G_t$; neighbor-induced features $X_t$; number of nodes $N_t$; reduced dimensionality $L$
Output: supernode set (i.e., merging membership) $V_t'$, coarsening matrix $C_t$
1. get node set $V_t$ and reweighted edges $E_t^w$ from \{ $G_t$, $X_t$ \}
2. initialize level index: $k ← 0$
3. initialize 1st level’s node and edge sets: $V_t^1 ← V_t$, $E_t^1 ← E_t^w$
4. initialize number of (super)nodes: $N_t' ← |V_t^1|$
5. while $N_t' > L$ do
   6. initialize next level’s node and edge set: $V_t^{(k+1)} ← \emptyset$, $E_t^{(k+1)} ← \emptyset$
   7. sort $E_t^{(k)}$ based on edge weights in descending order
   8. arrange the sorted result as a queue $q(E_t^{(k)})$
   9. while $q(E_t^{(k)}) \neq \emptyset$ do
      10. pop $(v_i^{(k)}, v_j^{(k)})$ from $q(E_t^{(k)})$
      11. if $v_i^{(k)} \in V_t^{(k)}$ and $v_j^{(k)} \in V_t^{(k)}$ then
         12. delete $v_i^{(k)}$ and $v_j^{(k)}$ from $V_t^{(k)}$
         13. merge $v_i^{(k)}$ and $v_j^{(k)}$ into supernode $v_{ij}^{(k+1)}$
         14. add $v_{ij}^{(k+1)}$ to $V_t^{(k+1)}$
         15. adjust $E_t^{(k)}$ and $E_t^{(k+1)}$ w.r.t. $V_t^{(k)}$ and $V_t^{(k+1)}$
         16. update number of (super)nodes: $N_t' ← N_t' - 1$
      17. if $N_t' = L$ then
         18. Get supernode set output: $V_t' ← V_t^{(k+1)} \cup V_t^{(k)}$
         19. derive coarsening matrix $C_t$ based on $V_t'$
         20. return $V_t'$ and $C_t$
      21. $V_t^{(k+1)} ← V_t^{(k+1)} \cup V_t^{(k)}$, $E_t^{(k+1)} ← E_t^{(k+1)} \cup E_t^{(k)}$
      22. update level index: $k ← k + 1$
23. get supernode set output: $V_t' ← V_t^{(k)}$
24. derive coarsening matrix $C_t$ based on $V_t'$
25. return $V_t'$ and $C_t$

[54] Jie Zhang, Yuxiao Dong, Yan Wang, Jie Tang, and Ming Ding. 2019. ProNE: Fast and Scalable Network Representation Learning. In Proceedings of the 28th International Joint Conference on Artificial Intelligence, Vol. 19. 4278–4284.
[55] Pan Zhang and Cristopher Moore. 2014. Scalable detection of statistically significant communities and hierarchies, using message passing for modularity. Proceedings of the National Academy of Sciences 111, 51 (2014), 18144–18149.
[56] Wentao Zhang, Yu Shen, Zheyu Lin, Yang Li, Xiaosen Li, Wen Ouyang, Yangyu Tao, Zhi Yang, and Bin Cui. 2022. Pasca: A graph neural architecture search system under the scalable paradigm. In Proceedings of the ACM Web Conference 2022. 1817–1828.
[57] Ziwei Zhang, Peng Cui, Haoyang Li, Xiao Wang, and Wenwu Zhu. 2018. Billion-scale network embedding with iterative random projection. In Proceedings of the 2018 IEEE International Conference on Data Mining. IEEE, 787–796.

A HEM GRAPH COARSENING ALGORITHM

The HEM graph coarsening procedure is summarized in Algorithm 4.

B PROOF OF FACT 1

By the disjoint constraint of CD that each node can only belong to one unique community (i.e., $\forall r \neq s$ s.t. $C_r^{(i)} \cap C_s^{(i)} \neq \emptyset$), only one entry in each row of $R_t$ must be 1 with rest entries in the same row set to 0. For adjacency matrix $A^{(g)}_t = R_t R_t^T$, consider each entry $(A^{(g)}_t)_{ij} = \sum_{r=1}^{K_t} ( (R_t)_{ir} (R_t)_{jr} )$. 

Manuscript submitted to ACM
Table 7. Parameter settings and layer configurations of ICD on all the datasets.

| Dataset | ICD-M | ICD-C | Layer Configurations |
|---------|-------|-------|----------------------|
| GN-0.3 | 0.3   | 0.3   | (1,0.1,5,500)       |
| GN-0.4 | 0.4   | 0.4   | (1,0.1,5,500)       |
| 100    | 0.1   | 0.1   | (1,0.1,5,500)       |
| 100    | 0.1   | 0.1   | (1,0.1,5,500)       |
| 100    | 0.1   | 0.1   | (1,0.1,5,500)       |
| 100    | 0.1   | 0.1   | (1,0.1,5,500)       |
| Reddit | 0.01  | 0.01  | (1,0.1,5,500)       |
| AS     | 0.01  | 0.01  | (1,0.1,5,500)       |
| Koon   | 0.01  | 0.01  | (1,0.1,5,500)       |

- If nodes $v_i', v_j'$ are in the same community $C_t'$, we have $R_{ij} = R_{js} = 1$ and $(A^{(g)}_t)_{ij} = (R_t)_{ij} = 1$. Hence, there is an edge between $(v_i', v_j')$ with weight 1 in $G_t^{(g)}$ when they are in the same community.

- If nodes $v_i', v_j'$ are in different communities, i.e., $v_i' \in C_t'$ and $v_j' \in C_r'$ ($r \neq t$), we have $R_{ij} = 1$ but $(A^{(g)}_t)_{ij} = 0$. Thus, there is no edge between $(v_i', v_j')$ in $G_t^{(g)}$ when they are in different communities.

In summary, there is an edge between nodes $(v_i', v_j')$ in $G_t^{(g)}$ only if they are partitioned into the same community according to the given CD result $C_t = C_t', \cdots, C_K'$.

C.1 Detailed Experiment Setups

Experiment Environment. We used PyTorch to implement ICD while we adopted the official or widely-used open-source implementations of other baselines. All the experiments were conducted on a server with Intel Xeon CPU (E5-2650v4@2.20GHz, 48 cores), 1 Tesla V100 GPU, 61GB memory, as well as the Ubuntu Linux OS. In this setting, methods implemented by PyTorch or TensorFlow were speeded up via GPUs.

Parameter Settings and Layer Configurations. On validation set $\Gamma_v$ of each dataset, we tuned parameters and determined layer configurations for ICD-M and ICD-C. Concretely, we adjusted $\alpha \in \{1, 5, 10\}$ and $m \in \{1, 2, \cdots, 5\}$ for both the variants, while we tuned $\beta \in \{0.1, 1, 2, \cdots, 5\}$ and $\beta' \in \{0.1, 1, 100, 1000\}$ for ICD-M and ICD-C, respectively. The recommended parameter settings and layer configurations for each dataset are depicted in Table 7, where dimensionality of the first and last layer of $G$ is set to the dimensionality of the reduced feature and graph embedding. Note that we use the same layer configurations for both ICD-M and ICD-C on each dataset. Moreover, we set the number of epochs $n = 100$ for all the datasets.

C.2 Detailed Quantitative Evaluation Results

In our evaluation, we recorded mean $\mu$ and standard deviation $\sigma$ of all the metrics on the test set $\Gamma_t$ of each dataset. Number records in the format $\mu(\sigma)$ w.r.t. NMI, AC, modularity, NCut and runtime on each dataset are depicted in Table 8, 9, 10, and 11. For ICD, we also recorded the runtime of (i) feature extraction of $Z_t$, (ii) one feedforward propagation through the GNN encoder, and (iii) downstream $K$Means clustering (with 10 independent runs), which are denoted as ‘Feat’, ‘Prop’, and ‘Clus’ in Table 11.
Table 8. Quantitative evaluation of CD quality in terms of NMI (%) and AC ↑ (%) for the total runtime of ICD on all the datasets (see Table 11), the downstream clustering is a major bottleneck, but it is also the key component that enables ICD to tackle online CD with non-fixed K. In our future work, we intend to

| Dataset | NMI (%) | AC (%) |
|---------|---------|--------|
| Taxi | 0.76 (0.47) | 1.91 (1.33) |
| Reddit | 0.72 (0.32) | 1.31 (0.83) |
| AS | 0.68 (0.42) | 1.31 (0.83) |
| Enron | 0.68 (0.42) | 1.31 (0.83) |
| | 0.76 (0.47) | 1.91 (1.33) |
| | 0.72 (0.32) | 1.31 (0.83) |
| | 0.68 (0.42) | 1.31 (0.83) |
| | 0.68 (0.42) | 1.31 (0.83) |
| | 0.76 (0.47) | 1.91 (1.33) |
| | 0.72 (0.32) | 1.31 (0.83) |
| | 0.68 (0.42) | 1.31 (0.83) |
| | 0.68 (0.42) | 1.31 (0.83) |
| | 0.76 (0.47) | 1.91 (1.33) |
| | 0.72 (0.32) | 1.31 (0.83) |
| | 0.68 (0.42) | 1.31 (0.83) |
| | 0.68 (0.42) | 1.31 (0.83) |
| | 0.76 (0.47) | 1.91 (1.33) |
| | 0.72 (0.32) | 1.31 (0.83) |
| | 0.68 (0.42) | 1.31 (0.83) |
| | 0.68 (0.42) | 1.31 (0.83) |
| | 0.76 (0.47) | 1.91 (1.33) |
| | 0.72 (0.32) | 1.31 (0.83) |
| | 0.68 (0.42) | 1.31 (0.83) |
| | 0.68 (0.42) | 1.31 (0.83) |
| | 0.76 (0.47) | 1.91 (1.33) |
| | 0.72 (0.32) | 1.31 (0.83) |
| | 0.68 (0.42) | 1.31 (0.83) |
| | 0.68 (0.42) | 1.31 (0.83) |
| | 0.76 (0.47) | 1.91 (1.33) |
| | 0.72 (0.32) | 1.31 (0.83) |
| | 0.68 (0.42) | 1.31 (0.83) |
| | 0.68 (0.42) | 1.31 (0.83) |

For the total runtime of ICD on all the datasets (see Table 11), the downstream clustering is a major bottleneck, but it is also the key component that enables ICD to tackle online CD with non-fixed K. In our future work, we intend to Manuscript submitted to ACM.
Table 11. Quantitative evaluation of CD quality in terms of runtime (sec).

| Method | GN | GN | L(f,0.3) | L(n,0.3) | Taxi | Reddit | AS | Enron |
|--------|----|----|----------|----------|------|--------|----|-------|
| SNMF   | 326.90 (63.67) | 232.32 (51.10) | 129.15 (32.03) | 136.21 (13.41) | 194.13 (54.34) | 2.13 (1.40) | 6.01 (8.97) | 112.31 (83.53) | 1.68 (1.25) |
| SC     | 62.14 (1.05) | 46.73 (0.69) | 21.04 (1.28) | 42.34 (2.21) | 129.15 (3.35) | 3.23 (0.17) | 1.90 (0.17) | 1.51 (1.50) | 0.54 (0.17) |
| MC-SBM | 30.10 (23.77) | 25.66 (25.48) | 7.29 (0.34) | 32.40 (16.63) | 102.42 (26.12) | 24.29 (2.30) | 1.36 (1.67) | 0.40 (0.57) | 1.65 (0.72) |
| Graclus| 4.39 (0.14) | 4.23 (0.17) | 2.89 (1.15) | 3.04 (1.35) | 5.29 (1.61) | 3.23 (1.37) | 2.90 (0.17) | 0.62 (0.19) | 0.54 (0.17) |
| M-MMF | 81.34 (1.14) | 79.07 (0.99) | 52.21 (0.81) | 56.34 (14.63) | 102.42 (14.66) | 24.29 (2.30) | 11.36 (16.72) | 1.65 (0.72) | 4.56 (2.19) |
| LDR   | 42.94 (0.30) | 40.92 (0.21) | 31.19 (0.23) | 31.19 (0.30) | 31.19 (0.30) | 39.23 (3.87) | 40.02 (3.91) | 4.39 (0.80) | 4.70 (0.71) |
| NECS  | 1670.29 (57.72) | 1470.47 (103.41) | 741.24 (32.72) | 822.94 (79.06) | 929.72 (79.06) | 46.95 (1.22) | 4.36 (1.49) | 266.28 (166.50) | 1.60 (1.26) |
| RandINE | 14.73 (0.22) | 12.48 (0.14) | 2.12 (0.18) | 2.49 (0.22) | 2.89 (0.30) | 3.91 (0.33) | 0.18 (0.02) | 0.48 (0.17) | 0.26 (0.07) |
| NBP   | 29.24 (1.87) | 19.26 (0.85) | 6.72 (1.13) | 8.59 (0.39) | 9.95 (0.91) | 6.57 (1.07) | 0.18 (0.02) | 0.48 (0.32) | 0.26 (0.07) |
| ProNE | 21.17 (0.33) | 18.85 (0.13) | 6.67 (0.23) | 7.49 (0.29) | 8.75 (0.70) | 8.77 (0.73) | 0.99 (0.03) | 0.48 (0.31) | 0.42 (0.13) |
| GAGE  | 8.18 (0.30) | 8.24 (0.31) | 2.58 (0.17) | 2.50 (0.16) | 2.81 (0.92) | 2.90 (0.92) | 2.90 (0.17) | 0.62 (0.17) | 0.29 (0.15) |
| GAT   | 5.37 (0.12) | 5.56 (0.15) | 2.03 (0.20) | 2.37 (0.19) | 2.77 (0.31) | 2.81 (0.29) | 0.62 (0.02) | 0.33 (0.16) | 0.69 (0.30) |
| GIN   | 4.56 (0.06) | 4.58 (0.10) | 1.86 (0.14) | 1.93 (0.15) | 1.83 (0.22) | 1.85 (0.23) | 0.16 (0.01) | 0.24 (0.10) | 0.67 (0.29) |
| GAP   | 19.59 (0.61) | 19.59 (0.59) | 66.45 (1.96) | 66.94 (1.75) | 76.79 (6.73) | 76.79 (6.73) | 8.66 (2.59) | 6.32 (4.48) | 70.53 (45.17) |
| ClusNet | 40.16 (0.24) | 39.21 (0.21) | 73.28 (4.97) | 76.96 (4.27) | 88.50 (7.63) | 90.07 (8.14) | 19.10 (0.28) | 6.55 (3.92) | 49.93 (29.66) |
| ICD-M | 13.63 (0.55) | 11.17 (0.32) | 2.48 (0.18) | 3.56 (0.28) | 2.93 (0.32) | 4.19 (0.46) | 0.43 (0.02) | 0.29 (0.14) | 2.37 (1.20) |
| Feat  | 0.42 (0.01) | 0.42 (0.01) | 0.20 (0.01) | 0.20 (0.01) | 0.20 (0.01) | 0.23 (0.02) | 0.23 (0.02) | 0.24 (0.0036) | 0.007 (0.0077) |
| Prop  | 0.02 (0.005) | 0.03 (0.006) | 0.03 (0.005) | 0.03 (0.0023) | 0.03 (0.0023) | 0.03 (0.0001) | 0.01 (0.0055) | 0.13 (0.05) | 0.03 (0.001) |
| Clus  | 13.18 (0.55) | 10.76 (0.32) | 2.50 (0.18) | 3.33 (0.26) | 2.67 (0.31) | 3.91 (0.43) | 0.19 (0.02) | 0.27 (0.12) | 0.03 (0.001) |
| ICD-C | 14.11 (0.50) | 11.42 (0.30) | 2.84 (0.20) | 3.25 (0.25) | 3.36 (0.37) | 4.24 (0.44) | 0.43 (0.02) | 0.30 (0.14) | 2.38 (1.28) |
| Feat  | 0.47 (0.01) | 0.46 (0.01) | 0.24 (0.09) | 0.25 (0.09) | 0.28 (0.03) | 0.28 (0.03) | 0.28 (0.03) | 0.24 (0.005) | 0.008 (0.0093) |
| Prop  | 0.02 (0.003) | 0.01 (0.003) | 0.03 (0.004) | 0.03 (0.004) | 0.03 (0.0024) | 0.03 (0.0002) | 0.01 (0.0087) | 0.13 (0.05) | 0.03 (0.0011) |
| Clus  | 13.62 (0.50) | 10.95 (0.30) | 2.57 (0.20) | 3.28 (0.25) | 3.05 (0.33) | 3.93 (0.42) | 0.19 (0.01) | 0.28 (0.12) | 0.10 (0.49) |

further reduce the runtime of ICD by replacing the downstream clustering module with a generic E2E module that can directly derive the CD result w.r.t. a specific (non-fixed) K via an output layer.