Few-Shot Community Detection in Graphs via Strong Triadic Closure and Prompt Learning

Yeqin Zhou and Heng Bao *

Posted Date: 7 August 2025

doi: 10.20944/preprints202508.0468.v1

Keywords: community detection; strong triadic closure; prompt learning; few-shot learning; graph neural networks

Article

Few-Shot Community Detection in Graphs via Strong Triadic Closure and Prompt Learning

Yeqin Zhou [1] and Heng Bao [2,*]

[1] Beijing University of Posts and Telecommunications, Beijing 100876, China, and also with Shandong Police College, Jinan 250200, China
[2] CNCERT/CC
* Correspondence: baoheng20@gmail.com

Abstract

Community detection is a fundamental task for understanding network structures, crucial for identifying groups of nodes with close connections. However, existing methods generally treat all connections in networks as equally important, overlooking the inherent inequality of connection strengths in social networks, and often require large amounts of labeled data. To address these challenges, we propose a few-shot community detection framework, **S**trong **T**riadic **C**losure **C**ommunity **D**etection with **P**rompt (STC-CDP), which combines the Strong Triadic Closure (STC) principle, Graph Neural Networks, and prompt learning. The STC principle, derived from social network theory, states that if two nodes share strong connections with a third node, they are likely to be connected themselves. By incorporating STC constraints during the pre-training phase, STC-CDP can differentiate between strong and weak connections in networks, thereby more accurately capturing community structures. We design an innovative prompt learning mechanism that enables the model to extract key features from a small number of labeled communities and transfer them to the identification of unlabeled communities. Experiments on multiple real-world datasets demonstrate that STC-CDP significantly outperforms existing state-of-the-art methods under few-shot conditions, achieving higher F1 scores and Jaccard similarity particularly on Facebook, Amazon, and DBLP datasets. Our approach not only improves the precision of community detection but also provides new insights into understanding connection inequality in social networks.

Keywords: community detection; Strong Triadic Closure; prompt learning; few-shot learning; Graph Neural Networks

1. Introduction

The community structure within social networks is a pivotal issue in the study of network science, referring to groups of nodes within the network that exhibit a significantly higher density of internal connections compared to connections with external nodes. Community structures are prevalent in real-world networks [1–3], and they hold substantial significance in uncovering network topology, predicting node behavior, and comprehending the dissemination of information. The identification of community structures has broad applications across various domains, ranging from market segmentation [4], recommendation systems [5], to fraud detection [6,7] and event organization [8]. Particularly in large-scale social networks, efficient and accurate community detection can aid platforms in optimizing content distribution, enhancing user engagement, and identifying potential malicious groups. With the rapid expansion of online social networks, traditional community detection methods face increasingly severe challenges. The exponential growth of network scale, dynamic changes in user behavior, and the complexity of community structures all demand continuous innovation in community detection techniques [9]. Especially under conditions where only limited labeled data is available, effectively identifying and extending specific types of communities has become a key research challenge. For instance, in security domains, identifying other potential suspicious groups in the network based on

 doi:10.20944/preprints202508.0468.v1

just a few known suspicious account clusters; or in marketing, discovering more potential customer groups with similar characteristics based on a small number of known high-value user communities.

Despite continuous advancements in community detection techniques, existing methods still face several key challenges. First, most algorithms treat all connections in networks as equally important, failing to consider the inherent inequality of connection strengths in social networks [10]. In real-world social networks, connection strengths vary significantly—some represent intimate relationships while others merely indicate superficial intersections, and these differences have a decisive impact on community structure formation [11,12]. Second, traditional algorithms typically require large amounts of labeled data for training, which is difficult to obtain in practical applications [13]. Although few-shot learning methods have been developed, they are highly sensitive to the quality of seed nodes and often suffer from insufficient flexibility and high computational overhead [14,15]. Third, existing methods generally lack adequate consideration of triangular structures that are common in social networks, despite their fundamental role in community formation [16]. To address these challenges, this paper introduces the Strong Triadic Closure (STC) principle to handle connection inequality in community detection. STC is an important concept in social network theory, stating that if two nodes share strong connections with a third node, they are likely to be connected themselves (either strongly or weakly) [17]. This principle has deep roots in sociology, with research by Granovetter [18] and Burt [19] demonstrating that triangular structures play a crucial role in social cohesion and information propagation. By combining STC with advanced deep learning techniques, we aim to improve the quality and accuracy of community detection, particularly under few-shot conditions.

This paper proposes a community detection framework based on Strong Triadic Closure Community Detection with Prompt—STC-CDP, which combines Graph Neural Networks, prompt learning, and the STC principle to achieve efficient few-shot community detection. Our main contributions include.

1. Innovative STC Injection Mechanism: We propose a method to organically integrate STC properties into the Graph Neural Network training process, enabling the model to learn to distinguish between strong and weak connections, thereby more accurately characterizing community structures.
2. Prompt-based Few-Shot Learning Framework: We design a parameter-efficient prompt learning framework that enables the model to extract key features from a small number of labeled communities and apply this knowledge to identify unlabeled similar communities.
3. End-to-end Community Detection System: We implement a complete pipeline from pre-training and prompt learning to final community prediction, providing a practical solution for real-world applications.

Through experiments on multiple real-world datasets, we demonstrate the superiority of STC-CDP, particularly under few-shot conditions, where it achieves significantly higher F1 scores and Jaccard similarity compared to existing state-of-the-art methods. Our research not only provides a new technical path for community detection but also offers deep insights into understanding connection inequality in social networks.

The remainder of this paper is organized as follows: Section 2 reviews related work; Section 3 introduces problem definition and fundamental concepts; Section 4 elaborates on the STC-CDP method; Section 5 presents experimental settings and results analysis; and Section 6 concludes the paper and discusses future research directions.

2. Related Work

2.1. Community Detection Algorithms

The development of community detection algorithms has evolved from traditional partition-based methods to advanced deep learning approaches. Early methods such as Louvain [20] and Girvan-Newman [21] algorithms primarily relied on modularity metrics to identify communities. With the advancement of deep learning techniques, Graph Neural Network (GNN)-based community detection

 doi:10.20944/preprints202508.0468.v1

methods, such as [22,23], have demonstrated significant advantages in community partitioning tasks by learning low-dimensional embedding representations of nodes.

Community detection research can be broadly categorized into unsupervised and supervised methods. Unsupervised methods include optimization-based approaches, such as graph partitioning through optimizing modularity metrics [24], and matrix factorization methods that learn latent community representations by decomposing adjacency matrices [25]. In recent years, frameworks combining graph representation learning and community detection have made significant progress, as seen in [26–30]. These methods have substantially improved community detection accuracy through co-learning community membership and node representations. However, most of these methods lack precise identification capabilities for specific types of communities and typically require large amounts of labeled data.

Semi-supervised community detection has emerged as a recent research direction [31–33], aiming to discover similar communities in networks using a small number of labeled communities as training data. Zhang [34] proposed a seed expansion-based method that identifies communities by selecting seed nodes and gradually expanding them. However, this approach is highly sensitive to seed node quality, and inappropriate seed node selection may lead to inaccurate community partitioning. Wu et al. [13,35] improved this issue by introducing subgraph inference, further reducing dependence on seed node quality. Additionally, they incorporated prompt learning into community detection, significantly reducing the need for training data. Nevertheless, these methods fail to adequately consider the differences in connection strengths between nodes, assuming all connections have equal importance in community partitioning, which contradicts the inequality of connection strengths in real-world social networks [10]. This may result in insufficient accuracy in identifying certain communities.

2.2. Triadic Closure Principle and Its Applications in Network Analysis

The Triadic Closure principle is a core concept in social network analysis, describing the phenomenon of "friends of friends are friends." Research by Bianconi et al. [16] and Granovetter [18] demonstrates that this principle not only helps understand social network structures but also predicts the formation of potential connections in networks. In graph theory, Triadic Closure is used to quantify the clustering coefficient of networks, where a high clustering coefficient typically indicates the presence of tight community structures in the network.

Kleinberg and Easley [17] further distinguished the different impacts of strong and weak ties in Triadic Closure, introducing the concept of Strong Triadic Closure (STC). As a significant work in this field, Sintos and Tsaparas [36] proposed the MINSTC problem, an optimization problem that minimizes the number of weak edges while satisfying STC properties. They proved that MINSTC is equivalent to solving the minimum vertex cover problem in the wedge graph Z(G). Tsourakakis et al. [37] extended this work, arguing that triangles (or other higher-order subgraph structures, i.e., motifs) in graphs are stronger signals of community structure, and thus these motifs can be leveraged to improve clustering effectiveness. Recent research, such as the work by Chakraborty et al. [38], combines graph mining with the Triadic Closure principle, applying STC to dense subgraph discovery. Shang et al. [39] proposed TriHetGCN, an extension of traditional Graph Convolutional Networks (GCN) that incorporates explicit topological metrics—triadic closure and degree heterogeneity—to address the issue of GCNs ignoring node attributes and intrinsic structural relationships between node pairs.

However, research on applying methods that combine graph representation learning with the Triadic Closure principle to targeted community detection tasks remains limited. First, existing graph representation learning methods often lack explicit modeling of community structures during the pre-training phase, making the generated node representations difficult to directly reflect community information. Second, most existing prompt learning methods are designed for tasks such as node classification or link prediction, relying on direct manipulation of node features while failing to adequately consider structural characteristics at the community level. Therefore, there exists a gap between these methods and the actual requirements when applied to community detection tasks.

 doi:10.20944/preprints202508.0468.v1

3. Problem Definition and Preliminaries

3.1. Community Detection

Definition 1 (Community Detection): Given a graph $\mathcal{G}$, with m labeled communities $\dot{\mathcal{C}} = \{\dot{\mathcal{C}}^1, \dot{\mathcal{C}}^2, \ldots, \dot{\mathcal{C}}^m\}$ (where $\forall_{i=1}^{m} \dot{\mathcal{C}}^i \subset \mathcal{G}$) as training data, the goal is to find a set of other similar communities $\hat{\mathcal{C}}$ such that $|\hat{\mathcal{C}}| \gg |\dot{\mathcal{C}}|$ in $\mathcal{G}$.

This definition describes the few-shot community detection problem, where the objective is to identify other communities in the graph that share similar structural characteristics based on a small number of known community examples. This setting corresponds to real-world scenarios where manually labeling a large number of communities is costly, while automatically discovering communities with similar properties is highly valuable.

3.2. Strong Triadic Closure (STC)

Let $\mathcal{G} \equiv (\mathcal{V}, \mathcal{E})$ be an undirected graph representing a social network, where the vertex set $\mathcal{V}$ corresponds to individuals and the edge set $\mathcal{E}$ corresponds to connections (relationships) between these individuals. Our goal is to label the relationships in the social network, classifying them as either strong or weak relationships.

We represent this labeling as a function $\ell : \mathcal{E} \to \{W, S\}$, which maps each edge $e \in \mathcal{E}$ to a label W (weak relationship) or S (strong relationship). A pair of edges $e_1 \equiv \{u, v\} \in \mathcal{E}$ and $e_2 \equiv \{v, w\} \in \mathcal{E}$ is called a wedge if $\{u, w\} \notin \mathcal{E}$, denoted as $e_1 \wedge e_2$ to represent the wedge between edges e_1 and e_2, where $u, v, w \in \mathcal{V}$.

Definition 2 (Strong Triadic Closure STC): Given a graph $\mathcal{G}$, if the labeling ℓ in the graph satisfies the Strong Triadic Closure (STC) property, then there does not exist a pair of edges (u, v) and (v, w) such that $\ell(u, v) \equiv S$ and $\ell(v, w) \equiv S$, but $(u, w) \notin \mathcal{E}$.

The Strong Triadic Closure (STC) property reveals an important phenomenon in social networks: if a vertex v has strong connections with vertices u and w, i.e., if $\ell(u, v) \neq S$ and $\ell(v, w) \equiv S$, then u and w are more likely to form an edge in $\mathcal{E}$, which can be either a weak or strong connection [17], as shown in Figure 1. This property reflects the transitivity and cohesion of community structures in social networks.

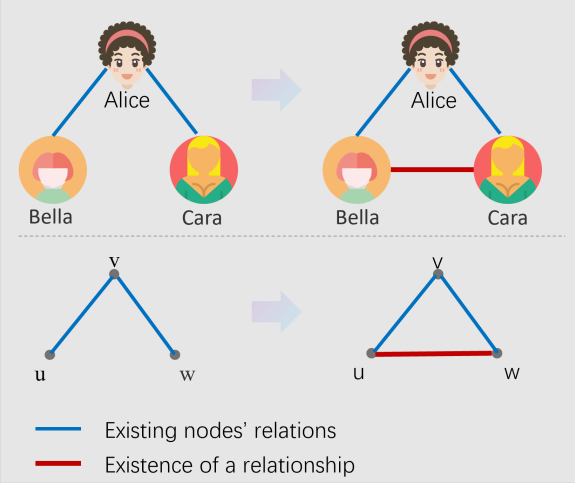

Figure 1. Strong Triadic Closure Property. The figure illustrates the STC principle with concrete examples: if Alice has strong connections with both Bella and Cara, then Bella and Cara are likely to be connected. The bottom part shows the abstract representation where if vertex v has strong connections with vertices u and w, then u and w should be connected to satisfy the STC property.

Corollary 1 (Strong Triadic Closure Violation): Given a graph $\mathcal{G}$ and an edge labeling function ℓ, if $\ell(u, v) \equiv S$, $\ell(v, w) \equiv S$, and $(u, w) \notin \mathcal{E}$, then the vertex triplet $u, v, w \in \mathcal{V}$ constitutes an STC violation. Let $B(\ell, \mathcal{G})$ denote the total number of violations induced by labeling ℓ on graph $\mathcal{G}$.

 doi:10.20944/preprints202508.0468.v1

This corollary defines the violation of STC constraints, providing a theoretical foundation for subsequent community detection using STC properties. The violation count $B(\ell, \mathcal{G})$ can be used as a metric to evaluate the quality of strong and weak relationship labeling in the graph, and can also serve as an optimization objective to minimize STC violations to obtain edge labeling that better conforms to the structural characteristics of social networks.

4. STC-CDP: The Proposed Approach

The STC-CDP method consists of three main steps: edge labeling, pre-training, and fine-tuning, aiming to perform efficient community detection by combining the Strong Triadic Closure (STC) principle with Graph Neural Network (GNN) techniques. The framework consists of three main components: (1) STC-based edge labeling to distinguish strong and weak connections, (2) STC-enhanced contrastive learning pre-training to learn graph representations, and (3) prompt-based fine-tuning for few-shot community detection.

4.1. Edge Labeling Using STC

4.1.1. Graph-Theoretic Modeling of the STC Problem

We first define a wedge as a pair of edges sharing a common vertex, formally represented as a wedge triplet $W \equiv \{(u, v), (v, w)\}$, where v is the shared vertex. If $(u, w) \notin \mathcal{E}$, this wedge is called an open wedge (or open triangle). The Strong Triadic Closure (STC) property requires that at least one edge in each open wedge must be labeled as a weak relationship.

To handle the STC problem, we transform the original graph $\mathcal{G}$ into a dual graph $\mathcal{G}_W \equiv (\mathcal{V}_W, \mathcal{E}_W)$, called the wedge graph:

- $\mathcal{V}_W \equiv \{v_e | e \in \mathcal{E}\}$, where each edge e in the original graph is mapped to a vertex v_e in the wedge graph
- $\mathcal{E}_W \equiv \{(v_{e_1}, v_{e_2}) | \exists \text{ open wedge } W, e_1, e_2 \in W\}$

Specifically, for each pair of edges $e_1 \equiv (u, v)$ and $e_2 \equiv (v, w)$ in the original graph $\mathcal{G} \equiv (\mathcal{V}, \mathcal{E})$, if they form an open wedge (i.e., $(u, w) \notin \mathcal{E}$), we add an edge (v_{e_1}, v_{e_2}) to the wedge graph $\mathcal{G}_W$. Through this transformation, we can convert the STC problem into finding a minimum edge set in the wedge graph $\mathcal{G}_W$ that satisfies the STC property constraints for all open wedges. This is closely related to the minimum vertex cover problem in graph theory.

In a graph $\mathcal{G}$, if the Strong Triadic Closure (STC) property is satisfied, there should be no open triangles that violate this property. Specifically, for any open triangle $\langle (u, v), (v, w) \rangle$, edges (u, v) and (v, w) cannot be simultaneously labeled as strong. This implies that in each open triangle, at least one edge must be labeled as weak to cover the triangle. We assume that the goal of social network construction is to establish strong relationships with others, therefore, our objective is to maximize the number of strong relationships while satisfying the STC property. This is equivalent to finding the minimum edge set to cover all open triangles in the graph and labeling these edges as weak relationships.

Following the approach in [36], we transform the problem into a Minimum Weak Edge Cover Problem, which aims to find the minimum edge set to cover all open triangles in the graph and label these edges as weak relationships. To solve the Minimum Weak Edge Cover Problem, we convert it into a Minimum Vertex Cover problem. Specifically, for a graph $\mathcal{G} \equiv (\mathcal{V}, \mathcal{E})$, a vertex set $\mathcal{V}_C \subseteq \mathcal{V}$ is a vertex cover of graph $\mathcal{G}$ if for each edge $(u, v) \in \mathcal{E}$, either vertex u or v belongs to the vertex set $\mathcal{V}_C$. The goal of the Minimum Vertex Cover problem is to find the vertex set $\mathcal{V}_C$ with the minimum number of vertices. By selecting these vertices, we can cover all relevant edges, thereby indirectly covering all open triangles. This method effectively transforms the edge cover problem into a vertex cover problem, allowing us to utilize existing minimum vertex cover algorithms for solution.

 doi:10.20944/preprints202508.0468.v1

4.1.2. STC Solution Based on Minimum Vertex Cover

After constructing the wedge graph $\mathcal{G}_W$, the STC problem is transformed into solving the minimum vertex cover problem on $\mathcal{G}_W$. The minimum vertex cover $C \subseteq \mathcal{V}_W$ corresponds to the minimum edge set in the original graph $\mathcal{G}$ that should be labeled as weak to satisfy the STC property.

Theorem 1 (Minimum Vertex Cover): Given a graph $\mathcal{G} \equiv (\mathcal{V}, \mathcal{E})$ and its corresponding wedge graph $\mathcal{G}_W \equiv (\mathcal{V}_W, \mathcal{E}_W)$, there exists a natural correspondence between the minimum vertex cover C^* of $\mathcal{G}_W$ and the minimum weak edge set in $\mathcal{G}$ that satisfies the STC property.

Formally, given the minimum vertex cover C^* of the wedge graph $\mathcal{G}_W$, we define the edge labeling function $\ell : \mathcal{E} \to \{\text{weak}, \text{strong}\}$ for graph $\mathcal{G}$ as:

$$\ell(e) \equiv \begin{cases} \text{weak}, & \text{if } v_e \in C^* \\ \text{strong}, & \text{otherwise} \end{cases} \tag{1}$$

Since the minimum vertex cover problem is NP-hard, we employ a greedy algorithm for approximate solution. The specific steps are shown in Algorithm 1. Algorithm 1 provides a greedy approximation with an approximation ratio of 2, meaning the number of weak edges in the resulting solution does not exceed twice the optimal solution.

Algorithm 1 Wedge Graph-based STC Edge Labeling Algorithm

Require: Graph $\mathcal{G} \equiv (\mathcal{V}, \mathcal{E})$
Ensure: Edge labeling function ℓ satisfying STC property
Construct wedge graph $\mathcal{G}_W \equiv (\mathcal{V}_W, \mathcal{E}_W)$:
  $\mathcal{V}_W \leftarrow \{v_e | e \in \mathcal{E}\}$
  $\mathcal{E}_W \leftarrow \emptyset$
for all $v \in \mathcal{V}$ and all pairs $u, w \in N(v)$ **do**
  if $(u, w) \notin \mathcal{E}$ **then**
    $\mathcal{E}_W \leftarrow \mathcal{E}_W \cup \{(v_{(u,v)}, v_{(v,w)})\}$
  end if
end for
Compute approximate minimum vertex cover C of $\mathcal{G}_W$:
  $C \leftarrow \emptyset$
while $\mathcal{E}_W \neq \emptyset$ **do**
  Select vertex $v_{max} \in \mathcal{V}_W$ with maximum degree
  $C \leftarrow C \cup \{v_{max}\}$
  Remove v_{max} and all its adjacent edges
end while
for all $e \in \mathcal{E}$ **do**
  if $v_e \in C$ **then**
    $\ell(e) \leftarrow$ weak
  else
    $\ell(e) \leftarrow$ strong
  end if
end for
return labeling function ℓ

4.2. STC-Enhanced Contrastive Learning Pre-training

After completing the STC edge labeling algorithm described in Section 4.1.2, we input the labeling results as edge attributes into the Graph Neural Network (GNN) for pre-training. The pre-training framework employs a multi-level contrastive learning strategy that fully leverages edge label information to enhance the model's ability to understand graph structures. The pre-training phase is designed with three key objectives: node-level contrastive loss, subgraph-level contrastive loss, and edge prediction loss. These objectives work together to enable the model to learn structural features in graphs and identify potential community patterns.

 doi:10.20944/preprints202508.0468.v1

Specifically, the Graph Neural Network encoder $\mathrm{GNN}_\Theta(\cdot)$ receives the STC-labeled graph as input, where edge labels (strong/weak) are converted into edge attributes. By learning these special structural information, the model can more accurately capture community structures in networks. The goal of the pre-training phase is to make the GNN model learn core features of graph structures, particularly considering the impact of edges with different strengths on community structures. To this end, we design a contrastive learning framework specifically for STC characteristics, enabling the model to understand the different roles of weak and strong edges in community formation.

4.2.1. STC-Based Representation Learning Framework

Given a graph $\mathcal{G} \equiv (\mathcal{V}, \mathcal{E})$ with edges $\mathcal{E}$ labeled by the STC algorithm, where the edge labeling is $\ell : \mathcal{E} \to \{\text{weak}, \text{strong}\}$, our goal is to learn a graph encoder $\mathrm{GNN}_\Theta(\cdot)$ that can capture community structures.

The representation learning framework consists of two key components:

- Node-level Contrastive Learning: Learns the consistency between node representations and their corresponding community representations.
- Community-level Contrastive Learning: Learns the consistency between the original community structure and the perturbed community structure, where the perturbation retains strong edges and preferentially removes weak edges.

Formally, for a node $v \in \mathcal{V}$, its representation is defined as:

$$z(v) \equiv \mathrm{GNN}_\Theta(X, \mathcal{E})[v] \tag{2}$$

where $X \in \mathbb{R}^{|\mathcal{V}| \times d}$ is the node feature matrix, $\mathcal{E} \in \mathbb{R}^{2 \times |\mathcal{E}|}$ is the edge index matrix, Θ denotes the parameters of the GNN, and d is the dimension of node features. $\mathrm{GNN}_\Theta(X, \mathcal{E})[v]$ denotes the embedding of node v produced by the GNN, with dimension $\mathbb{R}^h$, where h is the hidden layer dimension.

For a subgraph $\mathcal{G}_S \equiv (\mathcal{V}_S, \mathcal{E}_S)$, its representation is defined as:

$$z(\mathcal{G}_S) \equiv \mathrm{READOUT}(\{z(v) | v \in \mathcal{V}_S\}) \tag{3}$$

where $\mathcal{V}_S \subseteq \mathcal{V}$ is the set of nodes in the subgraph, $\mathcal{E}_S \subseteq \mathcal{E}$ is the set of edges in the subgraph, and $\mathrm{READOUT} : \mathbb{R}^{|\mathcal{V}_S| \times h} \to \mathbb{R}^h$ is a pooling function that aggregates the set of node representations $\{z(v) | v \in \mathcal{V}_S\}$ into a subgraph representation vector. Common pooling functions include mean pooling, max pooling, or attention pooling.

4.2.2. STC-Guided Contrastive Learning

Our contrastive learning framework leverages the STC property and learns representations through the following two key loss functions:

Node-Community Contrastive Loss: Encourages the alignment between node representations and their corresponding community representations

$$\mathcal{L}_{\text{node}}(\Theta) \equiv -\sum_{v \in \mathcal{V}_B} \log \frac{\exp(z(v) \cdot z(\mathcal{G}_v)/\tau)}{\sum_{\mathcal{G}' \in \mathcal{B}} \exp(z(v) \cdot z(\mathcal{G}')/\tau)} \tag{4}$$

where $\mathcal{V}_B$ is the set of nodes in the batch, $\mathcal{G}_v$ is the subgraph containing node v, $\mathcal{B}$ is the set of subgraphs in the batch, and τ is the temperature parameter.

STC-Guided Community Contrastive Loss: Encourages the alignment between the original community representation and the perturbed community representation that retains strong edges

$$\mathcal{L}_{\text{subg}}(\Theta) \equiv -\sum_{\mathcal{G}_S \in \mathcal{B}} \log \frac{\exp(z(\mathcal{G}_S) \cdot z(\tilde{\mathcal{G}}_S)/\tau)}{\sum_{\mathcal{G}' \in \mathcal{B}} \exp(z(\mathcal{G}_S) \cdot z(\mathcal{G}')/\tau)} \tag{5}$$

 doi:10.20944/preprints202508.0468.v1

where $\tilde{\mathcal{G}}_S$ is the perturbed version of $\mathcal{G}_S$, and the perturbation strategy is based on STC labeling, retaining edges with higher strength.

Formal Definition of the Perturbation Strategy: Given a subgraph $\mathcal{G}_S \equiv (\mathcal{V}_S, \mathcal{E}_S)$ and its STC edge labeling, the perturbation operation $\mathcal{P}$ is defined as:

$$\mathcal{P}(\mathcal{G}_S) \equiv (\mathcal{V}_S, \mathcal{E}'_S) \tag{6}$$

where $\mathcal{E}'_S \subseteq \mathcal{E}_S$, and

$$P(e \in \mathcal{E}'_S | e \in \mathcal{E}_S) \equiv \begin{cases} p_{\text{strong}}, & \text{if } \ell(e) \equiv \text{strong} \\ p_{\text{weak}}, & \text{if } \ell(e) \equiv \text{weak} \end{cases} \tag{7}$$

Typically, $p_{\text{strong}} > p_{\text{weak}}$ is set to ensure that strong edges are preferentially retained, which is consistent with the STC assumption that strong edges are more important for community structure.

4.2.3. Edge Prediction Auxiliary Task

To further leverage the edge labeling information provided by STC, we introduce edge prediction as an auxiliary task. This task requires the model to predict the type of each edge (no edge, weak edge, or strong edge), which is formalized as:

$$\mathcal{L}_{\text{edge}}(\Theta) \equiv -\sum_{(u,v)\in\mathcal{E}_B} \log P_\Theta(\ell(u,v)|z(u), z(v)) \tag{8}$$

where $\mathcal{E}_B$ is the set of edges in the batch, P_Θ denotes the probability distribution for predicting the edge label based on the node representations, and $\ell(u,v)$ is the ground-truth label of edge (u,v) (0 for weak edge, 1 for strong edge), while $z(u)$ and $z(v)$ are the representation vectors of nodes u and v, respectively.

The final training objective is a weighted combination of these losses:

$$\mathcal{L}(\Theta) \equiv \lambda_{\text{node}}\mathcal{L}_{\text{node}}(\Theta) + \lambda_{\text{subg}}\mathcal{L}_{\text{subg}}(\Theta) + \lambda_{\text{edge}}\mathcal{L}_{\text{edge}}(\Theta) \tag{9}$$

where λ_{node}, λ_{subg}, and λ_{edge} are hyperparameters that balance the contributions of each loss term.

In this way, the learned representations not only capture the relationships between nodes and their communities but also encode the edge strength information provided by STC, thereby enabling a better understanding of community structures.

The detailed workflow of the pre-training process is summarized in Algorithm 2.

Algorithm 2 STC-based Graph Pre-training Algorithm

Require: Graph $\mathcal{G} \equiv (\mathcal{V}, \mathcal{E}, \mathcal{A})$ with STC edge labels, where $\mathcal{A}$ denotes edge attributes
Ensure: Pre-trained GNN model
Initialize GNN parameters Θ
for epoch = 1 to epochs **do**
  Randomly sample a batch of nodes $\mathcal{B} \subset \mathcal{V}$
  for each node $v \in \mathcal{B}$ **do**
    Extract the k-hop subgraph $\mathcal{N}_v$
  end for
  for each subgraph $\mathcal{N}_v$ **do**
    Create a perturbed version $\tilde{\mathcal{N}}_v$
  end for
  Use the GNN to process subgraphs and obtain node embeddings $\mathbf{z}_i$ and summary vectors $\mathbf{s}_i$
  Compute $\mathcal{L}_{\text{node}}$, $\mathcal{L}_{\text{subg}}$, and $\mathcal{L}_{\text{edge}}$
  Calculate the total loss $\mathcal{L}_{\text{total}} \equiv \alpha\mathcal{L}_{\text{node}} + \beta\mathcal{L}_{\text{subg}} + \gamma\mathcal{L}_{\text{edge}}$
  Update parameters Θ to minimize $\mathcal{L}_{\text{total}}$
end for
return the trained GNN model

 doi:10.20944/preprints202508.0468.v1

4.3. Prompt Learning and Knowledge Transfer

After the pre-training phase, the GNN encoder has acquired an understanding of the underlying community structures in the network. To apply this knowledge to downstream tasks, we adopt a prompt learning framework, using a small number of labeled samples to guide the model in community discovery.

4.3.1. Prompt Function Design

We design a simple yet efficient prompt function $\mathrm{PT}_\Phi(\cdot)$, which takes node embeddings as input and predicts, via a multi-layer perceptron (MLP), whether nodes belong to the same community:

$$\hat{\mathcal{C}}_v \equiv \mathrm{PT}_\Phi(\mathcal{N}_v) \tag{10}$$

where $\hat{\mathcal{C}}_v$ denotes the candidate community centered at node v, and $\mathcal{N}_v$ represents its k-hop neighborhood (K-EGO network).

The prompt function is implemented by comparing the embedding relationships between the central node and each node in its K-EGO network, performing a binary classification prediction:

$$\hat{\mathcal{C}}_v \equiv \{u \in \mathcal{N}_v \mid \sigma(\mathrm{PT}_\Phi(z(u), z(v))) \geq \tau\} \tag{11}$$

where $z(u)$ and $z(v)$ are node embeddings provided by the pre-trained GNN, PT_Φ is the parameterized prompt function, σ is the sigmoid function that converts the output to a probability between 0 and 1, and τ is the threshold parameter (default value is 0.2).

4.3.2. Edge Weight-Aware K-EGO Network Construction

During the prompt learning phase, we introduce an edge weight-based K-EGO network construction strategy. Given a graph $\mathcal{G} \equiv (\mathcal{V}, \mathcal{E}, \mathcal{A})$, where $\mathcal{A}$ is the set of edge attributes, each edge $e_{i,j} \in \mathcal{E}$ is assigned a weight according to its attribute $a_{i,j} \in \mathcal{A}$:

$$w(e_{i,j}) \equiv \begin{cases} w_{\text{strong}}, & \text{if } a_{i,j} \equiv 1 \text{ (strong edge)} \\ w_{\text{weak}}, & \text{if } a_{i,j} \equiv 0 \text{ (weak edge)} \end{cases} \tag{12}$$

where $w_{\text{strong}} > w_{\text{weak}}$ (typically, $w_{\text{strong}} \equiv 5.0$, $w_{\text{weak}} \equiv 1.0$).

When constructing the K-EGO network for node v, the probability of selecting an edge is proportional to its weight:

$$P(e_{i,j} \in \mathcal{N}_v) \propto w(e_{i,j}) \tag{13}$$

In this way, strong edges are more likely to be retained, thereby better preserving community structure information in the K-EGO network.

4.3.3. Training Strategy with Positive-Negative Sample Balancing

During the training of the prompt function, the selection of positive and negative samples is crucial for model performance. Given a node v in community $\mathcal{C}_i$, the nodes in its K-EGO network $\mathcal{N}_v$ can be divided into two categories:

- Positive samples: $\mathcal{P}_v \equiv \{u \in \mathcal{N}_v | u \in \mathcal{C}_i\}$
- Negative samples: $\mathcal{N}_v \setminus \mathcal{P}_v$

Since negative samples usually far outnumber positive samples, we adopt a weighted sampling strategy to balance the ratio of positive and negative samples. For each negative sample u, its probability of being selected is:

$$P(u \in \mathcal{N}_v^{\text{sampled}}) \propto \frac{1}{w(e_{v,u})} \tag{14}$$

 doi:10.20944/preprints202508.0468.v1

This means that negative samples connected by weak edges are more likely to be selected, while those connected by strong edges are less likely to be chosen. The intuition behind this strategy is that nodes connected by strong edges are more likely to belong to the same community and thus are less reliable as negative samples, while nodes connected by weak edges are more likely to belong to different communities and thus are more reliable as negative samples.

Finally, the prompt function is trained by optimizing the following loss function:

$$\mathcal{L}_{\mathrm{PT}}(\Phi) \equiv \sum_{\mathcal{C}_i \in \mathcal{C}_{\mathrm{train}}} \sum_{v \in \mathcal{C}_i} \left(\sum_{u \in \mathcal{P}_v} \mathcal{L}_{\mathrm{BCE}}(\sigma(\mathrm{PT}_\Phi(z(u), z(v))), 1) + \sum_{u \in \mathcal{N}_v^{\mathrm{sampled}}} \mathcal{L}_{\mathrm{BCE}}(\sigma(\mathrm{PT}_\Phi(z(u), z(v))), 0) \right) \tag{15}$$

4.3.4. Community Prediction Process

Based on the pre-trained GNN and prompt function, we achieve large-scale community discovery from a small number of labeled communities through knowledge transfer. Specifically, for each node $v \in \mathcal{V}$, we generate a candidate community $\hat{\mathcal{C}}_v \equiv \{u \in \mathcal{N}_v^w | \sigma(\mathrm{PT}_\Phi(z(u), z(v))) \geq \tau\}$ based on its weighted K-EGO network $\mathcal{N}_v^w$. By computing the community embedding $z(\hat{\mathcal{C}}_v) \equiv \mathrm{READOUT}(\{z(u) | u \in \hat{\mathcal{C}}_v\})$ and $z(\mathcal{C}_i) \equiv \mathrm{READOUT}(\{z(u) | u \in \mathcal{C}_i\})$, and using the Euclidean distance $d(\hat{\mathcal{C}}_v, \mathcal{C}_i) \equiv ||z(\hat{\mathcal{C}}_v) - z(\mathcal{C}_i)||_2$ to evaluate similarity, we select the k most similar candidate communities for each training community. This enables knowledge transfer from training communities to target communities, ultimately yielding the predicted community set $\mathcal{C}_{\mathrm{pred}} \equiv \bigcup_{i=1}^{|\mathcal{C}_{\mathrm{train}}|} \mathcal{S}_i$.

5. Experiments

In this section, we conduct a comprehensive evaluation of the STC-CDP method to verify the effectiveness of combining the Strong Triadic Closure (STC) principle with prompt learning for few-shot community detection. All experiments are conducted on an NVIDIA 3090 GPU, implemented using PyTorch and PyTorch-Geometric frameworks. We adopt widely recognized community detection evaluation metrics and report the average results and standard deviations over multiple runs to ensure the reliability and stability of the experimental results.

Our experimental design aims to answer the following key research questions:

RQ1 (Overall Performance): How does STC-CDP perform on few-shot community detection tasks compared to existing state-of-the-art methods?

RQ2 (Ablation Study): What are the specific contributions of the STC module and the prompt learning mechanism to the performance of STC-CDP?

RQ3 (Parameter Sensitivity): How do key hyperparameters (such as STC weight and the number of labeled communities) affect the performance of STC-CDP?

RQ4 (Computational Efficiency): How does the computational efficiency and resource consumption of STC-CDP compare to baseline methods?

5.1. Experimental Setup

5.1.1. Datasets

We use five widely adopted real-world social network datasets, each containing overlapping communities of different scales and characteristics: Facebook, Amazon, DBLP, Livejournal, and Twitter. Table 1 presents the basic statistics of these datasets.

 doi:10.20944/preprints202508.0468.v1

Table 1. Dataset Statistics.

Datasets	Nodes	Edges	$\#\mathcal{C}$	$\lvert\bar{\mathcal{C}}\rvert$
Amazon	13,178	33,767	4,517	9.31
DBLP	114,095	466,761	4,559	8.4
Twitter	87,760	1,293,985	2,838	10.88
Youtube	216,544	1,393,206	2,865	7.67
Livejournal	316,606	4,945,140	4,510	17.65

5.1.2. Baseline Methods

We compare STC-CDP with the following representative methods:

- SEAL [34]: A method that learns heuristic rules for target communities based on generative adversarial networks.
- CLARE [35]: Proposes a subgraph-based inference framework, including a locator and a rewriter.
- ProCom [13]: A few-shot community detection method that adopts a prompt learning strategy.

5.1.3. Evaluation Metrics

We use bidirectional matching F1 score and Jaccard similarity as the main evaluation metrics, which are widely recognized standard measures in the field of community detection. Given M ground-truth communities $\dot{\mathcal{C}}^{(i)}$ and N predicted communities $\hat{\mathcal{C}}^{(j)}$, the score is calculated as follows:

$$\frac{1}{2}\left(\frac{1}{N}\sum_{j}\max_{i}\delta\left(\hat{\mathcal{C}}^{(j)},\dot{\mathcal{C}}^{(i)}\right)+\frac{1}{M}\sum_{i}\max_{j}\delta\left(\hat{\mathcal{C}}^{(j)},\dot{\mathcal{C}}^{(i)}\right)\right) \tag{16}$$

where δ can be either the F1 function or the Jaccard function.

5.1.4. Implementation Details

The hyperparameter settings for STC-CDP are shown in Table 2.

Table 2. Hyper-parameters in STC-CDP.

Component	Hyper-parameter	Value
Encoding	Batch size	256
	Number of epochs	30
	Learning rate	1e-3
	Implementation of $\mathrm{GNN}_{\Theta}(\cdot)$	2 layers GCN
	k-ego subgraph	2
	Embedding dimension	128
	Temperature τ	0.1
	Ratio ρ for corruption	0.85
	Loss weight λ	1
Sampling	Batch size	32
	Number of epochs	100
	Embedding dimension	64
	MLP layers	3
	LGPNs layers	3
	Learning rate	1e-2
	Discount factor γ	1
Fine-tuning	Implementation of $\mathrm{PF}_{\Phi}(\cdot)$	2 layers MLP
	Number of epochs	30
	Learning rate	1e-3
	k-ego subgraph	3
	Number of prompts m	20
	Threshold value α	0.2

 doi:10.20944/preprints202508.0468.v1

5.2. Experimental Results and Analysis

5.2.1. Overall Performance Comparison (RQ1)

Table 3 presents the overall performance comparison of STC-CDP and baseline methods on five datasets. Most of the comparative results are sourced from [13].

Table 3. Performance Comparison of Different Methods on Five Datasets (F1 Score / Jaccard Similarity).

Method	Facebook	Amazon	DBLP	Livejournal	Twitter
SEAL	$31.10_{\pm 3.84}/23.02_{\pm 2.98}$	$82.26_{\pm 4.04}/75.44_{\pm 4.69}$	$41.74_{\pm 6.35}/33.25_{\pm 7.05}$	$42.85_{\pm 2.60}/35.03_{\pm 7.05}$	$16.97_{\pm 1.32}/10.55_{\pm 0.98}$
CLARE	$28.53_{\pm 1.36}/19.64_{\pm 1.16}$	$78.89_{\pm 2.10}/68.50_{\pm 2.90}$	$48.75_{\pm 2.51}/38.30_{\pm 2.21}$	$45.38_{\pm 4.07}/36.38_{\pm 3.69}$	$20.05_{\pm 0.88}/12.52_{\pm 0.63}$
ProCom	$38.57_{\pm 2.02}/28.05_{\pm 1.85}$	$84.36_{\pm 0.23}/75.84_{\pm 0.24}$	$50.96_{\pm 1.57}/39.47_{\pm 1.64}$	$54.35_{\pm 3.04}/44.93_{\pm 2.96}$	$31.09_{\pm 0.35}/20.78_{\pm 0.29}$
STC-CDP (Ours)	$\mathbf{39.51}_{\pm 0.85}/\mathbf{29.15}_{\pm 0.65}$	$\mathbf{85.05}_{\pm 0.15}/\mathbf{76.54}_{\pm 0.20}$	$\mathbf{57.35}_{\pm 0.07}/\mathbf{46.33}_{\pm 0.08}$	$\mathbf{55.01}_{\pm 1.39}/\mathbf{44.66}_{\pm 1.3}$	$\mathbf{31.87}_{\pm 0.04}/\mathbf{21.45}_{\pm 0.03}$

It is evident from Table 3 that STC-CDP achieves the best performance across all datasets. This demonstrates the effectiveness of combining the STC principle with prompt learning in capturing community structures. The results clearly show the consistent superiority of our proposed method across different evaluation metrics and datasets.

5.2.2. Ablation Study (RQ2)

To verify the contribution of each component in STC-CDP, we conducted a detailed ablation study, and the results are shown in Table 4.

Table 4. Ablation Study Results of STC-CDP.

Variant	Facebook		Amazon		DBLP		Livejournal		Twitter	
	F1	Jaccard	F1	Jaccard	F1	Jaccard	F1	Jaccard	F1	Jaccard
Basic GNN	33.8	24.1	83.3	74.8	45.7	35.3	41.5	33.6	27.0	18.0
GNN+STC	34.3	24.7	83.5	74.9	47.3	36.8	47.0	38.6	29.5	19.7
GNN + Prompt Learning	38.8	28.2	84.3	75.9	51.4	40.2	54.0	44.6	31.1	20.8
STC-CDP (Full Model)	**39.5**	**29.1**	**85.1**	**76.5**	**57.4**	**46.3**	**55.0**	**44.7**	**31.9**	**21.5**

The study shows that both the STC principle and the prompt learning mechanism significantly improve model performance. Specifically, adding only the STC principle increases the average F1 score by 2.07

5.2.3. Parameter Sensitivity Analysis (RQ3)

To gain deeper insight into the robustness and performance characteristics of the STC-CDP model, we conducted a sensitivity analysis of key hyperparameters, focusing on the STC loss weight and the number of labeled communities. Using the Facebook dataset as a benchmark, we systematically tuned these parameters to investigate their impact on model performance. The results are shown in Figures 2 and 3.

 doi:10.20944/preprints202508.0468.v1

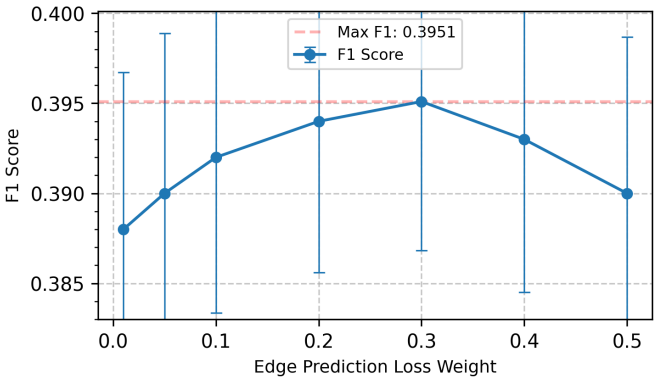

Figure 2. Impact of Edge Prediction Loss Weight on Model Performance. The F1 score initially increases with higher weight values, reaching peak performance at 0.3, then gradually decreases when the weight exceeds this optimal value.

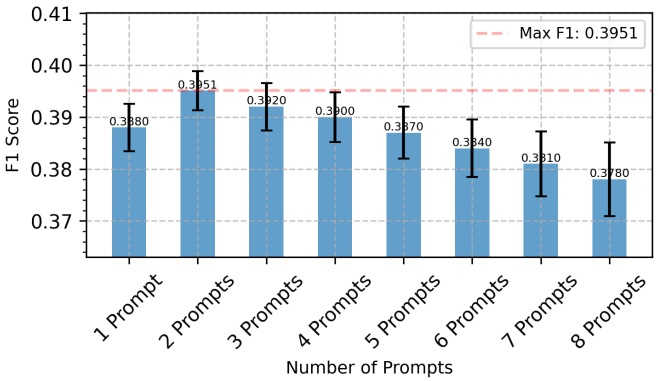

Figure 3. Impact of Number of Prompts on Model Performance. The model achieves optimal performance with 2 prompts, with diminishing returns when using more prompts, demonstrating the efficiency of few-shot learning.

Impact of Edge Prediction Loss Weight: As shown in Figure 2, the edge prediction loss weight has a significant impact on model performance. As the weight increases from 0 to 0.3, the model performance exhibits a clear upward trend, which fully validates the effectiveness of the STC principle in community detection tasks. However, when the weight exceeds 0.3, model performance begins to decline. This phenomenon may be attributed to two factors: first, an excessively high STC constraint may cause the model to focus too much on triadic closure structures, thereby neglecting other important community features; second, overly strong constraints may introduce additional noise, affecting the model's generalization ability. This finding provides important guidance for model tuning, suggesting that the edge prediction loss weight should be set at around 0.3 to achieve optimal performance.

Impact of the Number of Prompts: As demonstrated in Figure 3, the number of prompts significantly affects model performance. We observed that model performance initially increases and then plateaus as the number of prompts changes. Specifically, when the number of prompts increases from 1 to 2, the model performance reaches its peak at approximately 0.3951, and as the number continues to increase (up to 8 prompts), the performance improvement gradually levels off. This phenomenon has important practical implications: first, it confirms that STC-CDP can effectively learn from a small number of labeled samples, which is highly consistent with our original intention in designing a few-shot learning framework; second, the results indicate that only 2 prompts are required to achieve near-optimal performance, greatly reducing the annotation cost in real-world applications. This finding not only validates the efficiency of the model but also provides important guidance for parameter configuration in practical deployment.

Overall, the parameter sensitivity analysis reveals the response characteristics of the STC-CDP model to key hyperparameters, providing reliable recommendations for parameter configuration

 doi:10.20944/preprints202508.0468.v1

in real-world applications. At the same time, these findings further confirm the rationality and effectiveness of the model design.

5.2.4. Computational Efficiency Analysis (RQ4)

Table 5 compares the computational efficiency of different methods across multiple datasets. The results show that STC-CDP achieves significantly better training times than mainstream methods such as SEAL and CLARE on most datasets. For example, on the Amazon dataset, the training time of STC-CDP is 275 seconds, which is much lower than SEAL (1 hour 3 minutes) and CLARE (529 seconds), but slightly higher than ProCom (144 seconds). On datasets such as DBLP, Livejournal, and Twitter, STC-CDP also demonstrates high efficiency, with training times only slightly higher than ProCom but significantly better than SEAL and CLARE. The efficiency of STC-CDP mainly benefits from the parameter efficiency of the prompt learning framework and the effective design of the STC module. Although its training time is slightly longer than ProCom, STC-CDP offers advantages in model expressiveness and scalability, enabling efficient training while maintaining accuracy. In summary, STC-CDP outperforms most mainstream methods in computational efficiency and can efficiently handle large-scale social networks under limited resource conditions, making it suitable for real-world applications.

Table 5. Efficiency Study in Terms of Total Running Time.

Method	Facebook	Amazon	DBLP	Livejournal	Twitter
SEAL	2h27m	1h3m	50m	3h35m	2h28m
CLARE	275s	529s	22m	832s	36m
ProCom	30s	144s	367s	260s	446s
STC-CDP (Ours)	**82s**	**275s**	**706s**	**818s**	**635s**

6. Conclusions

This study proposes STC-CDP, a few-shot community detection framework integrating Strong Triadic Closure (STC) with prompt learning. The main contributions are: First, we introduce the STC principle to community detection, addressing limitations in handling connection strength inequality. STC provides theoretical foundation for distinguishing strong and weak ties, enabling more accurate identification of community structures. Experiments show STC-based modeling significantly improves detection accuracy across real-world datasets.

Second, we design a parameter-efficient prompt learning framework that alleviates few-shot detection challenges. By combining pre-training with prompt adaptation, STC-CDP extracts key features from limited labeled communities and transfers knowledge to unlabeled ones, reducing data dependency and computational costs.

Third, ablation studies verify the synergistic effect between STC and prompt learning. STC enhances network structure understanding while prompt learning improves few-shot generalization. Their combination outperforms individual methods, demonstrating framework effectiveness.

Despite these advances, limitations remain. Future directions include extending to dynamic networks, exploring continuous connection strength representations, and improving theoretical frameworks. STC-CDP provides new theoretical perspectives for community detection, advancing social network analysis with broad applications in social media analysis, market segmentation, public health, and information dissemination.

Author Contributions: Conceptualization, Y.Z. and H.B.; methodology, Y.Z.; software, Y.Z.; validation, Y.Z. and H.B.; formal analysis, Y.Z.; investigation, Y.Z.; resources, H.B.; data curation, Y.Z.; writing—original draft preparation, Y.Z.; writing—review and editing, H.B.; visualization, Y.Z.; supervision, H.B.; project administration, H.B.; funding acquisition, H.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

 doi:10.20944/preprints202508.0468.v1

Institutional Review Board Statement: Not applicable for studies not involving humans or animals.

Informed Consent Statement: Not applicable for studies not involving humans.

Data Availability Statement: The data is all usable and can be provided in its original form.

Acknowledgments: The authors would like to thank the anonymous reviewers for their valuable comments and suggestions.

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

Abbreviations

The following abbreviations are used in this manuscript:

STC	Strong Triadic Closure
CDP	Community Detection with Prompt
GNN	Graph Neural Network
MLP	Multi-Layer Perceptron

References

1. Yang, H.; Li, B.; Cheng, F.; Zhou, P.; Cao, R.; Zhang, L. A Node Classification-Based Multiobjective Evolutionary Algorithm for Community Detection in Complex Networks. *IEEE Trans. Comput. Soc. Syst.* **2024**, *11*, 292–306.
2. Li, B.; Kamuhanda, D.; He, K. Centroid-Based Multiple Local Community Detection. *IEEE Trans. Comput. Soc. Syst.* **2024**, *11*, 455–464.
3. Ni, L.; Li, Q.; Zhang, Y.; Luo, W.; Sheng, V.S. Local Community Detection in Multi-Attributed Road-Social Networks. *IEEE Trans. Knowl. Data Eng.* **2025**, *37*, 3514–3527.
4. Ding, X.; Mittal, A.; Gopal, A. DELPHYNE: A Pre-Trained Model for General and Financial Time Series. *arXiv preprint arXiv:2506.06288* **2025**.
5. Wu, X.; Xiong, Y.; Zhang, Y.; Jiao, Y.; Zhang, J.; Zhu, Y.; Yu, P.S. ConsRec: Learning Consensus Behind Interactions for Group Recommendation. *Proceedings of the ACM Web Conference 2023* **2023**, 240–250.
6. Fionda, V.; Pirrò, G. Community deception in attributed networks. *IEEE Trans. Comput. Soc. Syst.* **2022**, *11*, 228–237.
7. Madi, S.A.; Pirrò, G. Node-centric community deception based on safeness. *IEEE Trans. Comput. Soc. Syst.* **2023**, *11*, 2955–2965.
8. Li, Q.; Zhu, Y.; Ye, J.; Yu, J.X. Skyline group queries in large road-social networks revisited. *IEEE Transactions on Knowledge and Data Engineering* **2021**, *35*, 3115–3129.
9. Xin, X.; Wang, C.; Ying, X.; Wang, B. Deep community detection in topologically incomplete networks. *Physica A* **2017**, *469*, 342–352.
10. Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; Yu, P.S. A comprehensive survey on graph neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 4–24.
11. Onnela, J.P.; Saramäki, J.; Hyvönen, J.; Szabó, G.; Lazer, D.; Kaski, K.; Kertész, J.; Barabási, A.L. Structure and tie strengths in mobile communication networks. *Proc. Natl. Acad. Sci. USA* **2007**, *104*, 7332–7336.
12. Dunbar, R.I.M. Do online social media cut through the constraints that limit the size of offline social networks? *R. Soc. Open Sci.* **2016**, *3*, 150292.
13. Wu, X.; Xiong, K.; Xiong, Y.; He, X.; Zhang, Y.; Jiao, Y.; Zhang, J. ProCom: A Few-shot Targeted Community Detection Algorithm. *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* **2024**, 3414–3424.
14. Perozzi, B.; Al-Rfou, R.; Skiena, S. Deepwalk: Online learning of social representations. *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* **2014**, 701–710.
15. Li, Y.; King, I. Autograph: Automated graph neural network. *Neural Information Processing: 27th International Conference, ICONIP 2020* **2020**, 189–201.
16. Bianconi, G.; Darst, R.K.; Iacovacci, J.; Fortunato, S. Triadic closure as a basic generating mechanism of communities in complex networks. *Phys. Rev. E* **2014**, *90*, 042806.
17. Easley, D.; Kleinberg, J. *Networks, crowds, and markets: Reasoning about a highly connected world*; Cambridge University Press, 2010.
18. Granovetter, M.S. The strength of weak ties. *Am. J. Sociol.* **1973**, *78*, 1360–1380.

 doi:10.20944/preprints202508.0468.v1

19. Burt, R.S. Structural holes and good ideas. *Am. J. Sociol.* **2004**, *110*, 349–399.
20. Blondel, V.D.; Guillaume, J.L.; Lambiotte, R.; Lefebvre, E. Fast unfolding of communities in large networks. *J. Stat. Mech. Theory Exp.* **2008**, *2008*, P10008.
21. Girvan, M.; Newman, M.E.J. Community structure in social and biological networks. *Proc. Natl. Acad. Sci. USA* **2002**, *99*, 7821–7826.
22. Rashnodi, O.; Rastegarpour, M.; Moradi, P.; Zamanifar, A. Community detection in attributed social networks using deep learning. *J. Supercomput.* **2024**, *80*, 25933–25973.
23. Xiong, K.; Jin, Y.; Xiong, Y.; Zhang, J. GetCom: An Efficient and Generalizable Framework for Community Detection. *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management* **2024**, 2650–2659.
24. Fortunato, S.; Hric, D. Community detection in networks: A user guide. *Phys. Rep.* **2016**, *659*, 1–44.
25. Ji, J.; Feng, S.; Li, Y. Tensorized Unaligned Multi-view Clustering with Multi-scale Representation Learning. *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* **2024**, 1246–1256.
26. Li, J.; Lai, S.; Shuai, Z.; Tan, Y.; Jia, Y.; Yu, M.; Lu, Y. A comprehensive review of community detection in graphs. *Neurocomputing* **2024**, 128169.
27. Wu, X.; Lu, W.; Quan, Y.; Miao, Q.; Sun, P.G. Deep dual graph attention auto-encoder for community detection. *Expert Syst. Appl.* **2024**, *238*, 122182.
28. Kojaku, S.; Radicchi, F.; Ahn, Y.Y.; Fortunato, S. Network community detection via neural embeddings. *Nat. Commun.* **2024**, *15*, 9446.
29. Huang, D.; Song, J.; He, Y. Community detection algorithm for social network based on node intimacy and graph embedding model. *Eng. Appl. Artif. Intell.* **2024**, *132*, 107947.
30. Gmati, H.; Mouakher, A.; Gonzalez-Pardo, A.; Camacho, D. A new algorithm for communities detection in social networks with node attributes. *J. Ambient Intell. Humaniz. Comput.* **2024**, 1–13.
31. Zhao, Y.; Li, W.; Liu, F.; Wang, J.; Luvembe, A.M. Integrating heterogeneous structures and community semantics for unsupervised community detection in heterogeneous networks. *Expert Syst. Appl.* **2024**, *238*, 121821.
32. Liu, X.; Zhang, M.; Liu, Y.; Liu, C.; Li, C.; Wang, W.; Bouyer, A. Semi-supervised community detection method based on generative adversarial networks. *J. King Saud Univ. Comput. Inf. Sci.* **2024**, *36*, 102008.
33. Momenzadeh, S.; Mohammadiani, R.P. Community Detection by ELPMeans: An Unsupervised Approach That Uses Laplacian Centrality and Clustering. *arXiv preprint arXiv:2502.19895* **2025**.
34. Zhang, Y.; Xiong, Y.; Ye, Y.; Liu, T.; Wang, W.; Zhu, Y.; Yu, P.S. SEAL: Learning heuristics for community detection with generative adversarial networks. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* **2020**, 1103–1113.
35. Wu, X.; Xiong, Y.; Zhang, Y.; Jiao, Y.; Shan, C.; Sun, Y.; Yu, P.S. CLARE: A semi-supervised community detection algorithm. *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* **2022**, 2059–2069.
36. Sintos, S.; Tsaparas, P. Using strong triadic closure to characterize ties in social networks. *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* **2014**, 1466–1475.
37. Tsourakakis, C.E.; Pachocki, J.; Mitzenmacher, M. Scalable motif-aware graph clustering. *Proceedings of the 26th International Conference on World Wide Web* **2017**, 1451–1460.
38. Arachchi, C.W.; Kumpulainen, I.; Tatti, N. Dense Subgraph Discovery Meets Strong Triadic Closure. *arXiv preprint arXiv:2502.01435* **2025**.
39. Shang, K.; Yi, J.; Small, M.; Zhou, Y. Triadic Closure-Heterogeneity-Harmony GCN for Link Prediction. *arXiv preprint arXiv:2504.20492* **2025**.