Neural Attributed Community Search at Billion Scale

Jianwei Wang[1] Kai Wang[2] Xuemin Lin[2] Wenjie Zhang[1] Ying Zhang[3]
[1]The University of New South Wales, Sydney, Australia
[2]Shanghai Jiao Tong University, Shanghai, China
[3]University of Technology Sydney, Sydney, Australia
jianwei.wang1@unsw.edu.au, w.kai@sjtu.edu.cn, xuemin.lin@sjtu.edu.cn, zhangw@cse.unsw.edu.au, ying.zhang@uts.edu.au

ABSTRACT

Community search has been extensively studied in the past decades. In recent years, there is a growing interest in attributed community search that aims to identify a community based on both the query nodes and query attributes. A set of techniques have been investigated. Though the recent methods based on advanced learning models such as graph neural networks (GNNs) can achieve state-of-the-art performance in terms of accuracy, we notice that 1) they suffer from severe efficiency issues; 2) they directly model community search as a node classification problem and thus cannot make good use of interdependence among different entities in the graph. Motivated by these, in this paper, we propose a new neur**AL** attr**I**buted **C**ommunity s**E**arch model for large-scale graphs, termed *ALICE*. *ALICE* first extracts a candidate subgraph to reduce the search scope and subsequently predicts the community by the Consistency-aware Net, termed *ConNet*. Specifically, in the extraction phase, we introduce the density sketch modularity that uses a unified form to combine the strengths of two existing powerful modularities, i.e., classical modularity and density modularity. Based on the new modularity metric, we first adaptively obtain the candidate subgraph, formed by the k-hop neighbors of the query nodes, with the maximum modularity. Then, we construct a node-attribute bipartite graph to take attributes into consideration. After that, *ConNet* adopts a cross-attention encoder to encode the interaction between the query and the graph. The training of the model is guided by the structure-attribute consistency and the local consistency to achieve better performance. Extensive experiments over 11 real-world datasets including one billion-scale graph demonstrate the superiority of *ALICE* in terms of accuracy, efficiency, and scalability. Notably, *ALICE* can improve the F1-score by 10.18% on average and is more efficient on large datasets in comparison to the state-of-the-art. *ALICE* can finish training on the billion-scale graph within a reasonable time whereas state-of-the-art can not.

KEYWORDS

Attributed Community Search; Graph Neural Networks

1 INTRODUCTION

Graph-structured data has shown particular advantages for modeling relationships and dependencies between objects, making it a powerful tool for data analytics in various fields such as social networks [14, 36, 48, 53], biological networks [22, 55] and financial networks [9, 50]. One of the core tasks of graph analytics is community search (CS)[12, 16, 23, 26, 53] that aims to find a subgraph containing the specific query nodes, with the resulting subgraph (community) being a densely intra-connected structure. In many real-world applications, nodes are often associated with attributes [37, 47]. As such, it is desirable to query using not just query nodes, but also query attributes. Attributed Community Search (ACS) [15, 16], a related but more challenging problem compared to CS, is proposed to deal with such applications. ACS aims to identify a community based on both query nodes and attributes, with the resulting community expected to demonstrate structure cohesiveness and semantic homogeneity. Studying ACS can benefit various applications, e.g., extracting biologically significant clues of protein-protein interaction networks [5, 25], finding the research communities in the collaboration networks [15], detecting fraudulent keywords of the web search [51]. In light of the significance and popularity of ACS, a spectrum of algorithms [15, 18, 25, 28] have been developed, which can be classified into two categories: non-learning-based techniques and learning-based approaches.

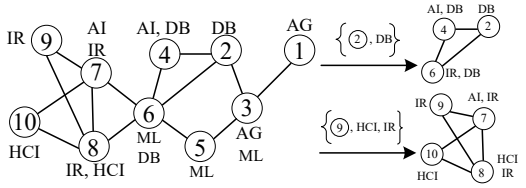


Figure 1: An illustration of attributed community search: The left panel illustrates a citation graph, whereas the right panel displays the retrieved communities corresponding to queries on each arrow.

Existing solutions. Existing non-learning-based attributed community search algorithms [15, 25] use a decoupled scheme that treat structure and attribute separately. They first search for structural-cohesive nodes based on the pre-defined cohesive subgraph models such as k-core [15] and k-truss [25]. Subsequently, the algorithms compute the score of attribute cohesiveness to identify the most relevant communities. These non-learning-based methods, however, are constrained by two primary limitations [28]: 1) Structure inflexibility. The pre-defined subgraph models rely heavily on the hyper-parameter k and the community quality is sensitive to k. In addition, the fixed subgraph models place a highly rigid constraint on the topological structure of communities, making it difficult for real-world communities to meet such priors. 2) Attribute irrelevance. These algorithms consider each attribute independently, which fails to capture the latent correlations between attributes, and thus restricts the exploration capability in the semantic space.

In order to alleviate the above issues, learning-based techniques with Graph Neural Networks (GNNs) have been proposed including ICS-GNN [18] and AQD-GNN [28]. Their frameworks are illustrated in Figure 2(a) and Figure 2(b), respectively. They refrain from imposing constraints on the community structure, and the attributes are propagated through edges to enhance their connection. ICS-GNN is designed for interactive community search that aims to gradually find the community in multiple iterations. In each iteration, a Vanilla Graph Convolutional Network (GCN) model [30] is directly

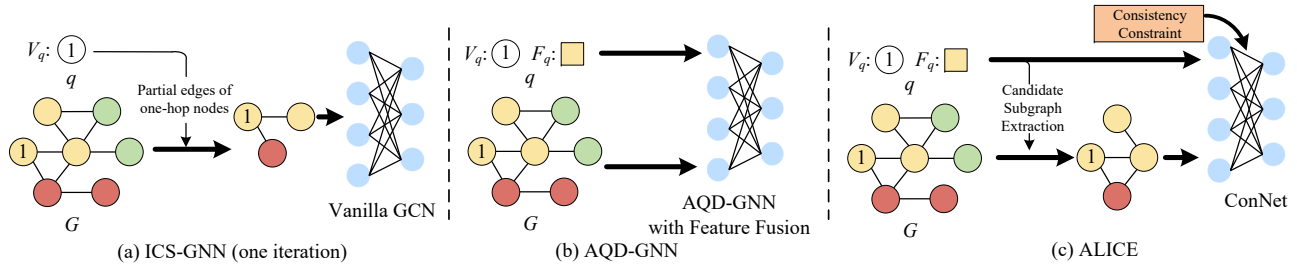


Figure 2: The framework of learning-based (attributed) community search models

applied to the one-hop neighbors of the query nodes. AQD-GNN is proposed to support ACS that inputs both the query q and data graph G, and a feature fusion operator is utilized to combine the representations of the query node, query attribute, and data graph to predict the final community.

Although the above-mentioned learning-based approaches demonstrate remarkable performance, particularly AQD-GNN, which has achieved state-of-the-art accuracy for ACS as evaluated in [28], two main limitations persist for existing learning-based approaches.

Firstly, both ICS-GNN and AQD-GNN encounter significant efficiency problems. ICS-GNN requires the entire model to be retrained when a new query is received. However, training a model is a time-consuming process. In addition, AQD-GNN takes the entire graph as input to learn the representation of each node and searches the entire graph to determine whether nodes belong to the community or not. Both learning and searching the complete graph can be time-consuming, hindering efficiency and scalability.

Secondly, both ICS-GNN and AQD-GNN directly recast the community search as a node classification problem, while the interdependence among different entities remains insufficiently explored: 1) The intricate interaction between query and data graph is overlooked, despite their strong correlation w.r.t the final community. ICS-GNN only inputs the candidate subgraph for GNNs, missing the query. AQD-GNN encodes query nodes, query attributes, and the graph separately, which employs a fusion operator to concatenate them. Nevertheless, the interaction between the query and each node in the graph remains insufficiently explored. 2) The correlation between structure and attribute remains inadequately investigated. Although they encode both graph structure and attribute simultaneously, they still fall short in capturing the correlation between the representations of query nodes and attributes, which is essential for structure and attribute constraints. 3) Both methods typically disregard the connection among nodes within a community. The community is a cohesive component that considers multiple nodes together, whereas node classification focuses on individual node properties. While it is possible to use breath first search (BFS) to select nearby connected nodes after learning, it can still harm accuracy since the optimization signal cannot be backpropagated.

Therefore, to design an efficient and effective learning-based approach for ACS at a large scale, two main challenges exist below.

Challenge 1: How to efficiently perform learning-based ACS at a large scale? A direct way is discarding unpromising nodes/edges at an early stage. However, the size of real-world communities can differ a lot. If we limit our selection to a small portion of nodes as candidates (e.g., ICS-GNN solely depends on one-hop neighbors of the query node), we risk losing many promising nodes; otherwise, an overly broad search scope presents computational difficulties. Thus, how to adaptively select promising candidates while taking both structure and attribute into account is challenging.

Challenge 2: How to effectively exploit the interdependence among different entities to enhance prediction accuracy? There exist abundant entities for ACS including query, data graph, structure, and attribute, while some of them (like structure and attribute) are from heterogeneous spaces [8]. Thus, how to collaboratively utilize these entities, capture their interactions and consistency in the latent space, and improve the overall accuracy is challenging.

Our solutions. To tackle the above challenges, in this paper, we propose a new neur**AL** attr**I**buted **C**ommunity s**E**arch model for large-scale graphs, namely *ALICE* (in Figure 2(c)). *ALICE* is a two-stage framework that first extracts a candidate subgraph and subsequently searches the community over the candidate subgraph by the Consistency-aware Net (*ConNet*).

(1) Candidate Subgraph Extraction. To address Challenge 1, it is crucial to find an effective model for evaluating the cohesiveness of a subgraph. Here we resort to modularity, a parameter-free metric that has been extensively utilized in finding communities [4, 10, 17, 21]. However, existing proposed modularities either select too many loosely connected nodes due to the free-rider effect [49] and the resolution limit problem [17], like the classical modularity [35]; or impose overly stringent requirements on cohesiveness, which may hinder the exploration of promising nodes, like the density modularity [10]. To alleviate this situation, we propose a novel form of modularity called *density sketch modularity* that uses a unified form to balance and combine the strengths of the above two modularities. Then, we adaptively select the structure-based candidate subgraph H induced by the k-hop neighbors of the query nodes s.t. H has the maximum density sketch modularity value. In this way, we do not need to pre-set the value of k. Similarly, to select nodes that possess similar attributes to the query attributes, we construct a node-attribute bipartite graph. With the bipartite graph, we select the attribute-based candidate nodes based on the subgraph that is induced by the k-hop neighbors of the query attributes and has the largest bipartite modularity value.

(2) Consistency-aware Net. To address Challenge 2, we further devise a novel GNN-based consistency-aware net, namely *ConNet* to capture the correlation and consistency. It has three main components: 1) *Cross-attention encoder.* We design a cross-attention encoder that aims to weigh the correlation between each query node (*resp.* attribute) and graph node (*resp.* attribute) and utilize this correlation to learn the structure (*resp.* attribute) representation. In contrast to AQD-GNN, which encodes query locally in one layer, *ConNet* learns

a representation that effectively combines the interaction of the query and each node in the graph. 2) *Structure-attribute consistency*. We devise a structure-attribute consistency module inspired by the recent representation learning methodology that brings related entities closer together in the latent space [34, 38, 39]. In light of the high correlation between structure and attribute, we propose a new approach that aims to minimize the Wasserstein distance between the distribution of structure representation and the distribution of attribute representation. 3) *Local consistency*. We develop a local consistency module, based on the observation that if a node belongs to a community, its neighboring nodes exhibit a high likelihood of being part of the same community and vice versa. It aims to pull closer the prediction results of nodes that are linked together. ACS is then modeled as multi-task learning that signals from the ground-truth labels and signals from the two consistency constraints are then optimized together.

Contributions. Here we summarize our main contributions:

- To enhance the performance of ACS, we propose a novel learning-based method *ALICE* that first extracts promising candidate subgraph and subsequently searches communities by the *ConNet*.
- We design an efficient subgraph extraction algorithm by leveraging a new form of modularity (i.e., density sketch modularity) and node-attribute relationship to adaptively select promising nodes. As evaluated, our approach can significantly reduce the training graph size (e.g., on the *Orkut* dataset with 3.07M nodes, only about <1% of nodes need to be passed to the next stage).
- We propose a GNN-based model *ConNet* to preserve both structure-attribute consistency and local consistency among nodes. It employs a cross-attention encoder to effectively capture the interaction between the query and the data graph.
- Extensive experiments are conducted over 11 popular public datasets, encompassing one billion-scale graph *Friendster*. The results demonstrate that *ALICE* can substantially improve both the search accuracy and the efficiency compared with existing methods. It can elevate the F1-score by 10.18% on average under the setting of query attributes generated from query nodes and is more efficient on large datasets *Google+* and *PubMed* compared with AQD-GNN [28]. Moreover, *ALICE* can finish training on large datasets *Reddit*, *Orkut* and *Friendster* within a reasonable time, whereas ADQ-GNN can not.

Roadmap. Section 2 introduces the preliminaries. Section 3 gives an overview of the whole framework while Section 4 and Section 5 elaborate detailed techniques. Section 6 reports experimental results. Section 7 reviews related work. Section 8 concludes this paper.

2 PRELIMINARIES

Our problem is defined over an undirected attributed graph $G(V, E, F)$ where V is the set of nodes with a cardinality of $|V| = n$ and $E \subseteq V \times V$ is the set of edges. $F = \{F_1, \cdots, F_n\}$ is the set of node attributes and F_i is the attributes of node v_i. Note that each node may have multiple attributes. We use F^d to denote the set of distinct attributes. The community is denoted by $C(V_C, E_C, F_C)$ where $V_C \subseteq V$ and $F_C \subseteq F$. For each $e = (v_i, v_j) \in E_C$, $e \in E$ and $v_i, v_j \in V_C$. The query $q =< V_q, F_q >$ consists of the query nodes V_q and query attributes F_q. C_q is utilized to denote the corresponding community w.r.t. q. When the context is clear, we abbreviate C_q as C. The frequently used notations are summarized in Table 1.

Table 1: Symbols and Descriptions

Notation	Description
$G(V, E, F)$	a graph with attributes in node
$C(V_C, E_C, F_C)$	a community
$q =< V_q, F_q >$	a query with node set V_q and attribute set F_q
$C_q, \tilde{C}_q$	the ground-truth/estimated community of q
$CM(\cdot), DM(\cdot)$	classical modularity and density modularity
$BM(\cdot)$	bipartite modularity for bipartite community
$DSM(\cdot)$	density sketch modularity

Graph Modularity. We use modularity which is a common metric of graph cohesiveness for the candidate subgraph extraction. It is a parameter-free measure [10] and represents the proportion of edges that belong to a particular group minus the expected proportion if the edges are randomly distributed. The higher the graph modularity is, the more cohesive the community is. The classical modularity of a community is defined as:

Definition 1. *(Classical Modularity [35]). Given a graph $G(V, E, F)$ and a community $C(V_C, E_C, F_C)$, the classic modularity is defined as:*

$$CM(G, C) = \frac{1}{2|E|}(2|E_C| - \frac{d_C^2}{2|E|}) \tag{1}$$

where d_C is the sum of degrees of the nodes in C.

When employing classic modularity for CS [10], it suffers from the free-rider effect [49] wherein the resulting community may encompass numerous nodes unrelated to the query nodes, and the resolution limit problem [17] that the resultant community may be too large to highlight some important structures.

Definition 2. *(Free-rider effect [49]). Given a set of query q, let C be a community identified based on a goodness function f, and C^* be the optimal solution (either local or global). The goodness function is said to be affected by the free-rider effect if $f(C \cup C^*) \geq f(C)$.*

Definition 3. *(Resolution limit problem [17]). Given a graph G, query q, the objective function f, a community constraint T, a community C satisfying T and containing all the query q, and any community C' satisfying the constraint T such that $C \cup C'$ is connected and $C \cap C' = \emptyset$, the objective function is said to suffer from the resolution limit problem if there exists a community C' such that $C \cup C'$ satisfies the constraint T and $f(C \cup C') \geq f(C)$.*

Note that the free-rider effect is different from the resolution limit problem. The former pertains to finding the most effective solution for detecting the effect, whereas the latter operates under the assumption of independent communities and ensures connectivity between the given communities. To better alleviate the above issues and incorporate modularity for community search, density modularity is proposed in [10].

Definition 4. *(Density Modularity [10]). Given a graph $G(V, E, F)$ and a community $C(V_C, E_C, F_C)$, the density modularity of C is defined as:*

$$DM(G, C) = \frac{1}{2|V_C|}(2|E_C| - \frac{d_C^2}{2|E|}) \tag{2}$$

where d_C is the sum of degrees of the nodes in C.

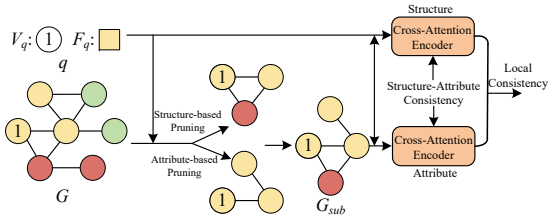


Figure 3: The framework of *ALICE*

While there exist some other definitions of modularity like generalized modularity density [21], recent research [10] claims that density modularity is one of the most effective forms of modularity for CS. Therefore, we concentrate on analyzing and comparing classical modularity and density modularity in this paper.

Graph Neural Networks. Modern GNNs follow a strategy of neighborhood aggregation mechanism, where the representation of a node is iteratively updated by aggregating the representations from its neighbors and its previous layer.

$$h_v^{(k)} = \mathrm{M}\left(h_v^{(k-1)}, \mathrm{AGG}\{h_u^{(k-1)} : u \in N(v)\}\right) \tag{3}$$

where $h_v^{(k)}$ is the representation of node v in layer k, $N(v)$ is the neighbors of node v, AGG is the aggregate function to aggregate messages, and M is the message propagation function that updates the representation of the node by the aggregated representations and its own representation from the previous layer. Different variants of GNNs have been proposed according to their own method of assigning weights to the neighbors and aggregating information.

The framework for Learning-based ACS. The general process for learning-based ACS includes two steps, i.e., the offline model training and the online query steps. The query set which contains both the query nodes and query attributes, the corresponding ground-truth community, and the graph with attribute information are used as inputs. It first trains the model on the training dataset offline and then utilizes the learned model to predict the test queries online. To ensure connectivity, the constrained BFS is used for community identification [28] that selects the nodes with a score larger than the pre-defined threshold and there exists a path from the node to query while scores of nodes in the path are all larger than the pre-defined threshold.

Problem Statement. Given an attributed graph $G(V, E, F)$, and a query $q = <V_q, F_q>$ where $V_q \subseteq V$ is a set of query nodes and $F_q \subseteq F$ is a set of query attributes, the task of Attributed Community Search (ACS) aims to find a query-dependent community C_q, which preserves both structure cohesiveness and attribute homogeneity (i.e., nodes in the community are densely intra-connected, and the attributes of these nodes are similar).

3 OVERVIEW OF *ALICE*

In this paper, we present a novel learning-based approach, named *ALICE*, for solving the problem of ACS. The overall framework is illustrated in Figure 3. Given the query set q and the data graph G, *ALICE* first extracts the candidate subgraph from the data graph using the query. The pruning stage contains two branches to select the promising candidates. The first branch is to extract the candidate subgraph considering the structure cohesiveness, while the second branch is to extract the candidate subgraph considering the semantic homogeneity. Both the structure-based candidates and attribute-based candidates are then combined as one candidate subgraph for the downstream prediction. The candidate subgraph together with the query set are sent to *ConNet* to search for the community. The *ConNet* comprises two branches, one dedicated to structure and the other to attribute, each utilizing the cross-attention encoder to learn the representations. Two consistency constraints including the structure-attribute consistency constraint and the local consistency constraint are used to guide the training. After that, *ConNet* outputs the predicted community. The detailed technique for candidate subgraph extraction is in Section 4 and the detailed architecture of *ConNet* is introduced in Section 5.

4 CANDIDATE SUBGRAPH EXTRACTION

In this section, we introduce details of the candidate subgraph extraction scheme specifically devised for ACS. As previously discussed in Section 1, the current cutting-edge ACS method is trained over the entire graph, thus inherently limiting its efficiency and scalability on large graphs. Here, we outline the desired features that serve as guidelines for developing the candidate subgraph extraction techniques: 1) *Adaptiveness.* An ideal subgraph extraction method ought to be capable of adaptively selecting and determining an appropriate quantity of candidate nodes, considering both the query and the data graph. 2) *Structure-Attribute awareness.* Since ACS aims to identify a structurally cohesive subgraph that upholds attribute homogeneity, the extraction method must pay heed to both the structure and attribute factors to retain as many promising candidate nodes as possible. Driven by these requirements, we present a modularity-based extraction approach. The proposed method involves a twofold process: firstly, detecting structurally cohesive candidate subgraph; and secondly, engaging in attribute-based pruning. The resultant candidate nodes from both phases are subsequently combined to form the candidate subgraph.

4.1 Structure-based Pruning

Our extraction scheme is based on modularity, a popular parameter-free metric of cohesiveness. As outlined in Section 2, there exist multiple types of modularity defined for different scenarios. In this paper, our focus is on classical modularity, which is one of the earliest proposed modularities, and density modularity, which is deemed one of the most powerful forms of modularity for CS, as documented in [10]. However, classical modularity is known to suffer from the free-rider effect and the resolution limit problem, which may result in the selection of too many loosely-connected nodes. On the other hand, density modularity may impose overly stringent requirements on cohesiveness, which may hinder the exploration of additional promising nodes. Hence, both are unsuitable for candidate subgraph extraction for ACS. To strike a balance and harness the benefits of the above two modularities, we propose the density sketch modularity as follows:

Definition 5. *(Density Sketch Modularity). Given a graph $G(V, E, F)$, a community $C(V_C, E_C, F_C)$ and a positive real number $\tau \in \mathbb{R}^+$, the density sketch modularity is defined as:*

$$DSM(G, C) = \frac{1}{2\,|V_C|^{\tau}}\left(2\,|E_C| - \frac{d_C^2}{2\,|E|}\right) \tag{4}$$

where d_C is the sum of degrees of the nodes in C.

By manipulating the value of τ, we can attain varying levels of cohesiveness granularity.

$$\lim_{\tau \to 0} DSM(G, C) = \frac{1}{2}(2\,|E_C| - \frac{d_C^2}{2\,|E|})$$

When τ approximates zero, the difference between classical modularity and density sketch modularity is the $|E|$ in the denominator which is a constant throughout the community within the same data graph. However, we only compare its relative magnitudes within one data graph when using modularity. Hence, density sketch modularity shares the same power as classical modularity when τ approximates zero.

$$\lim_{\tau \to 1} DSM(G, C) = \frac{1}{2\,|V_C|}(2\,|E_C| - \frac{d_C^2}{2\,|E|})$$

When τ approximates one, density sketch modularity is exactly the density modularity.

As proven below, density sketch modularity also shares two nice properties as density modularity for any varying $\tau \in \mathbb{R}^+$.

Lemma 1. *Whenever density sketch modularity suffers from the free-rider effect, classic modularity suffers from the free-rider effect as well.*

Proof. Assume that C is an identified community, and C^* is the optimal solution. If the density sketch modularity suffers from the free-rider effect, we can obtain from the definition that $DSM(G, C \cup C^*) \geq DSM(G, C)$. Putting the definition of DSM into the inequality, we can get $\frac{1}{2|V_{C\cup C^*}|^\tau}(2\,|E_{C\cup C^*}| - \frac{d^2_{C\cup C^*}}{2|E|}) \geq \frac{1}{2|V_C|^\tau}(2\,|E_C| - \frac{d_C^2}{2|E|})$. As $2\,|V_C|^\tau$ is larger than zero for $\tau \in \mathbb{R}^+$, we can multiply both sides by $2\,|V_C|^\tau$ and get $\{\frac{|V_C|}{|V_{C\cup C^*}|}\}^\tau(2\,|E_{C\cup C^*}| - \frac{d^2_{C\cup C^*}}{2|E|}) \geq 2\,|E_C| - \frac{d_C^2}{2|E|}$. As $\{\frac{|V_C|}{|V_{C\cup C^*}|}\}^\tau$ is always smaller than 1 for $\tau \in \mathbb{R}^+$, hence we get the following inequality: $2\,|E_{C\cup C^*}| - \frac{d^2_{C\cup C^*}}{2|E|} \geq \{\frac{|V_C|}{|V_{C\cup C^*}|}\}^\tau(2\,|E_{C\cup C^*}| - \frac{d^2_{C\cup C^*}}{2|E|}) \geq 2\,|E_C| - \frac{d_C^2}{2|E|}$. We use the first and third items from the inequality and multiply both sides by $\frac{1}{2|E|}$. After that we can get the following expression: $\frac{1}{2|E|}(2\,|E_{C\cup C^*}| - \frac{d^2_{C\cup C^*}}{2|E|}) \geq \frac{1}{2|E|}(2\,|E_C| - \frac{d_C^2}{2|E|})$ which is exactly $CM(G, C \cup C^*) \geq CM(G, C)$. Therefore, if density sketch modularity suffers from the free-rider effect, classic modularity also suffers from the free-rider effect as well. □

Lemma 2. *Whenever density sketch modularity suffers from the resolution limit problem, classic modularity suffers from the resolution limit problem as well.*

Proof. The proof of LEMMA 2 is quite similar to that of LEMMA 1. Assume that C and C' are two communities satisfying $C \cap C' = \emptyset$ and $G[C \cup C']$ being a connected subgraph. And then, we get the proof of LEMMA 2 by replacing the C^* of inequalities in the proof of LEMMA 1 with C'.

□

When identifying the candidate subgraph, existing works either select a small portion of nodes (like ICS-GNN solely depends on one-hop neighbors of the query node) or use the whole graph (like AQD-GNN). Based on density sketch modularity, we choose the subgraph induced by the k-hop neighbors of the query nodes that has the highest modularity value as the candidate subgraph. Note that, in this manner, the candidate subgraph is obtained adaptively, and we do not need to pre-set the value of k. In addition, we use $\tau \in [0, 1]$ to control the granularity of the subgraph, and a higher τ value can produce a more cohesive subgraph. We set τ as 0.8 by default as suggested by our experimental results in Section 6.5.

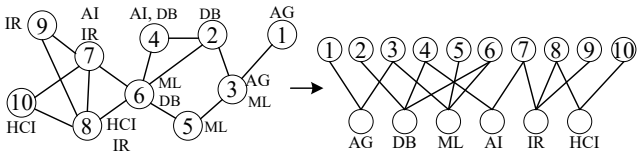


Figure 4: node-attribute bipartite graph

Algorithm 1: Candidate Subgraph Extraction

Input: The query $q =< V_q, F_q >$, the attributed graph $G = (V, E, F)$.
Output: The candidate subgraph G_{sub}

```
// The structure-based pruning.
Initialize set P = V_q, Q = V_q, V_sub = V_q, max_mod = −inf
while |Q| < |V| do
    for each v ∈ Q do
        P ← P ∪ N(v)
    Q ← P; mod = calculate DSM(G, Q)
    if mod > max_mod then
        max_mod ← mod; V_sub = V_sub ∪ Q
// The attribute-based pruning.
BG(V_B = (U, L), E_B) ← construct the bipartite graph.
P, Q ← Query attribute node in BG; max_mod = −inf
while |Q| < |U| + |L| do
    for each v ∈ Q do
        P ← P ∪ N(v)
    Q ← P; mod = calculate BM(BG, Q)
    if mod > max_mod then
        max_mod ← mod; V_sub = V_sub ∪ Q.U
G_sub ←induced subgraph from V_sub
return G_sub
```

Example 1. *For the query node 4 in Figure 1, its 1-hop subgraph contains nodes 2, 4, and 6. Thus, its modularity is* $\frac{1}{2\times 3^{0.8}}(2\times 3 - \frac{10^2}{2\times 14}) = 0.504$ *by DSM with* $\tau = 0.8$. *Similarly, we can get its modularity of 2-hop induced subgraph* $\frac{1}{2\times 7^{0.8}}(2\times 9 - \frac{23^2}{2\times 14}) = -0.094$ *and the modularity of 3-hop induced subgraph is* $\frac{1}{2\times 10^{0.8}}(2\times 14 - \frac{28^2}{2\times 14}) = 0.0$. *Hence, the subgraph induced by its 1-hop neighbors is selected as the structure-based candidate subgraph.*

4.2 Attribute-based Pruning

Attributes play a crucial part in the search for the attributed community. In order to establish the connection between nodes and attributes, we create a node-attribute bipartite graph $BG(V = (U, L), E)$ based on the approach in [28]. The node-attribute bipartite graph contains two types of node sets: the graph node set U

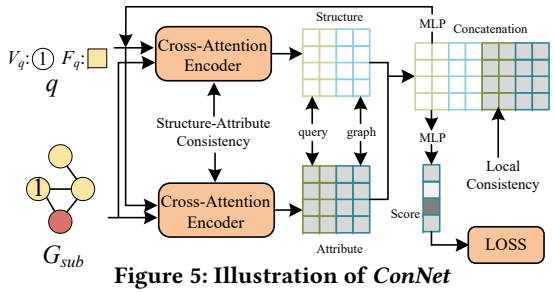


Figure 5: Illustration of *ConNet*

and the attribute node set L. Each distinct attribute is represented by an attribute node in the L side of the node-attribute bipartite graph, and there is a link between $u \in U$ and $l \in L$ if l is the attribute of node u in the original graph.

Example 2. *An example of the node-attribute bipartite graph using the graph in Figure 1 is depicted in Figure 4. The original graph contains 10 nodes with 6 distinct attributes. Therefore, there are 10 nodes on the U side and 6 nodes on the L side. The attributes of node 4 contain AI and DB. Hence, there is an edge between node 4 and node AI, and an edge between node 4 and node DB.*

By utilizing the bipartite graph, nodes in the original graph that are with the same attributes share a common neighbor in the bipartite graph. Additionally, attribute nodes with similar neighborhoods may possess similar semantics. To measure the cohesiveness of the subgraph in the bipartite graph, we resort to bipartite modularity:

Definition 6. *(Bipartite Modularity). Given a bipartite graph $G(V = (U, L), E)$ and a community $C(V_C = (U_C, L_C), E_C)$, the bipartite modularity is defined as follows [29]:*

$$BM(G, C) = \frac{1}{|E|}(2\,|E_C| - \frac{d_C^U d_C^L}{|E|}) \tag{5}$$

where d_C^U is the sum of degrees of the nodes in the U side of C and d_C^L is the sum of degrees of the nodes in the L side of C.

Similar to structure-based pruning, we present an attribute-based pruning approach that leverages the bipartite modularity to identify semantically-similar nodes. Given the query attribute nodes in the bipartite graph, the attribute-based candidate nodes are ascertained based on the subgraph induced by the k-hop neighbors of the query attribute nodes possessing the largest bipartite modularity. Note that, only graph nodes in the induced subgraph are selected since the final output community is a set of graph nodes.

The candidate subgraph extraction algorithm. The overall modularity-based candidate subgraph extraction method is shown in Algorithm 1. The algorithm takes the attributed graph and the query as inputs and produces the candidate subgraph. Initially, it performs structure-based pruning (lines 1 to 7). It first designates the query nodes as the candidate subgraph and the maximum modularity as negative infinity during initialization (line 1). Then, it expands outward hop by hop until all nodes are taken into consideration (lines 2 to 7). For each k, if the density sketch modularity of the induced subgraph formed by the k-hop neighbors of the query nodes exceeds the previous maximal value, the candidate subgraph is expanded to encompass the k-hop neighbors (lines 3 to 7). Next, the algorithm performs attribute-based pruning (lines 8 to 15). It begins by creating the node-attribute bipartite graph (line 8)

Algorithm 2: Forward Propagation of *ConNet*

Input: The qeury $q =< V_q, F_q >$, candidate subgraph G_{sub}.
Output: The predicted community $\tilde{C}_q$.

1 $H_{v_q}^{(0)}, H^{(s,0)}, H_{f_q}^{(0)}, H^{(a,0)} \leftarrow$ feature initialization
2 **for** $k = 0, \cdots, K-1$ **do**
3 $\quad X_q, X_k, X_v = H_{v_q}^{(k)} W_q^{(s,k)}, H^{(s,k)} W_k^{(s,k)}, H^{(s,k)} W_v^{(s,k)}$
4 $\quad H_{v_q}^{(k+1)} = \text{softmax}(\frac{X_q X_k^T}{\sqrt{d}}) X_v$
5 $\quad$ **for** $v \in V(G_{sub})$ **do**
6 $\quad\quad h_v^{(s,k+1)} = \text{MLP}^{(s,k)}\left(\left(1+\epsilon^{(k)}\right) \cdot h_v^{(s,k)} + \sum_{v' \in N(v)} h_{v'}^{(s,k)}\right)$
7 $\quad X_q, X_k, X_v = H_{f_q}^{(k)} W_q^{(a,k)}, H^{(a,k)} W_k^{(a,k)}, H^{(a,k)} W_v^{(a,k)}$
8 $\quad H_{f_q}^{(k+1)} = \text{softmax}(\frac{X_q X_k^T}{\sqrt{d}}) X_v$
9 $\quad$ **for** $v \in V(G_{sub})$ **do**
10 $\quad\quad h_v^{(a,k+1)} = \text{MLP}^{(a,k)}\left(\left(1+\epsilon^{(k)}\right) \cdot h_v^{(a,k)} + \sum_{v' \in N(v)} h_{v'}^{(a,k)}\right)$
11 $\quad H^{(s)} = H_{v_q}^{(k+1)} || H^{(s,k+1)}, H^{(a)} = H_{f_q}^{(k+1)} || H^{(a,k+1)}$
12 $\quad H_{v_q}^{(k+1)} = \text{MLP}^{(v_q,k)}(H^{(s)} || H^{(a)})$
13 $\quad H_{f_q}^{(k+1)} = \text{MLP}^{(f_q,k)}(H^{(s)} || H^{(a)})$
14 $\tilde{C}_q = \text{MLP}(H^{(s)} || H^{(a)})$
15 **return** $\tilde{C}_q$

and maps the query attributes into attribute nodes in the bipartite graph (line 9). Lines 10 to 15 are analogous to lines 2 to 7, with the exception that only the nodes on the U side are selected (line 15). At last, the algorithm outputs the candidate subgraph G_{sub} induced from the selected candidates V_{sub} (lines 16 and 17).

5 CONSISTENCY-AWARE NET

Based on the obtained substructure G_{sub} and the input query $q =< V_q, F_q >$, a consistency-aware net, namely *ConNet*, is designed to predict the community. The overall architecture is illustrated in Figure 5 and Algorithm 2. *ConNet* incorporates three main components, including 1) Cross-attention encoder, 2) Structure-attribute consistency, and 3) Local consistency. *ConNet* runs for K layers and uses two cross-attention encoders to encode the structure and attribute information of both the query and data graph into the latent space (lines 2 to 13). Both structure and attribute representations are concatenated and fed into an MLP for predicting the community (line 14). While learning the representation, two consistency constraints are employed to guide the training. The structure-attribute consistency constraint aims to obtain consistent representations for the structure and for the attribute, while the local consistency constraint aims to achieve an aligned prediction result for neighboring nodes. In the following subsections, we describe these main components in detail.

5.1 Feature Initialization

As the GNN model needs vectorized inputs, we introduce a vectorization technique for the structure iuput and attribute input.

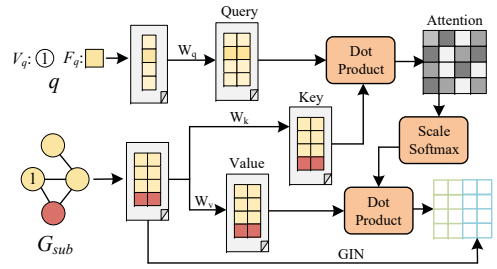


Figure 6: Illustration of Cross Attention Encoder

Structure Input. The query node set V_q is encoded as a one-hot vector $H_{v_q}^{(0)} \in \{0,1\}^{|V_{sub}|}$ where the i-th bit equals to one if $v_i \subseteq V_q$. For example, the query node v_2 is encoded as $[0,1,0,0,0,0,0,0,0,0]^T$ for the graph in Figure 1. As the size of the candidate subgraph is small, the length of the one-hot vector is also small.

Attribute Input. The query attribute set F_q is encoded as a one-hot vector $H_{f_q}^{(0)} \in \{0,1\}^{|V_{sub}|}$ where the i-th bit equals to one if there exist an attribute f_j while $f_j \subseteq F_q$ and $f_j \subseteq F_i$. Each bit indicates the relevance of node attributes and query attributes. For example, given the query attribute set $\{DB\}$ and graph in Figure 1, the query attribute set is encoded as $[0,1,0,1,0,1,0,0,0,0]^T$. The input of the candidate subgraph is the stack of the feature of each node. Note that although different subgraphs of equal length may share an initial query node/attribute representation, their final representation will differ since the GNN model propagates the representation through different edges in different subgraphs.

5.2 Cross-Attention Encoder

The query and data graph are not isolated entities, but instead, have a high correlation w.r.t. the resulting community. This interaction plays a crucial role in learning query-dependent embeddings. Encoding the two inputs separately would lead to a lack of information exchange between them, resulting in indistinct representations and diminishing the accuracy. To capture such interaction, we design a cross-attention encoder that leverages a cross-attention mechanism to learn the embeddings. The overall illustration of the cross-attention encoder is shown in Figure 6.

In the retrieval system, elements are stored in a "key-value" pair format where the key serves as the identifier for the corresponding value which can be a large file. When a query is submitted, the system compares the query with each key stored in the system. If a key matches the query, the corresponding value is retrieved and returned by the system. The design of cross-attention follows this architecture of the *query-key-value* retrieval.

The cross-attention encoder is utilized to encode both structure and attribute information. Specifically, we showcase the utilization of the cross-attention encoder for the encoding of structure information in layer k. The structure inputs consist of the query nodes $H_{v_q}^{(k)}$ and the graph $H^{(s,k)}$. The query goes through a linear transformation layer which is parameterized by a weight matrix $W_q^{(s,k)} \in \mathbb{R}^{d_k \times d_{k+1}}$ where d_k is the dimension of the hidden vector of layer k. Similarly, we project the graph into the latent space by two weight matrices $W_k^{(s,k)}, W_v^{(s,k)} \in \mathbb{R}^{d_k \times d_{k+1}}$ for key and value. We use superscript "s" for structure-related components and "a" for attribute-related components.

$$X_q = H_{v_q}^{(k)} W_q^{(s,k)},\ X_k = H^{(s,k)} W_k^{(s,k)},\ X_v = H^{(s,k)} W_v^{(s,k)} \tag{6}$$

Next, to calculate the similarity between the query and key, we perform a dot-product operation, followed by scaling the result by the square root of the dimension, and applying a softmax function to normalize the resulting vector. This produces an attention matrix $X \in \mathbb{R}^{|V_{sub}| \times |V_{sub}|}$, where each element $x_{ij} \in X$ represents the correlation between nodes v_i of query and v_j of data graph. Next, we use the dot product of X and X_v to obtain the query representation that combines both information from the query and data graph.

$$X = \text{softmax}(\frac{X_q X_k^T}{\sqrt{d_{k+1}}}),\ H_{v_q}^{(k+1)} = X X_v \tag{7}$$

In addition to encoding the query representation, we also employ GNN to obtain the data graph representation and subsequently concatenate the two representations for community prediction. In this paper, we use Graph Isomorphism Network (GIN) [11] as the backbone, which has an excellent structure-preserving ability and has been utilized in various graph analytic tasks such as subgraph counting [42, 43] and graph classification [33]. The formula of GIN of cross-attention encoder for structure encoding is as follows:

$$h_v^{(s,k+1)} = \text{MLP}^{(s,k)}\left(\left(1+\epsilon^{(k)}\right) \cdot h_v^{(s,k)} + \textstyle\sum_{v' \in N(v)} h_{v'}^{(s,k)}\right) \tag{8}$$

where $h_v^{(s,k)} \in H^{(s,k)}$ is the latent representation of node v in layer k, $N(v)$ is the neighbor set of v and $\epsilon^{(k)}$ is a learnable parameter in layer k. An MLP is utilized to learn to combine the information from the neighborhood and the previous layer. The result is concatenated with the query representation to obtain the structure representation:

$$H^{(s)} = H_{v_q}^{(k+1)} || H^{(s,k+1)} \tag{9}$$

where $||$ is the concatenate operation. Similarly, we can get the attribute representation $H^{(a)}$. The encoder runs for K layers, and each layer uses an MLP to pass query information to the next layer:

$$H_{v_q}^{(k+1)} = \text{MLP}^{(v_q,k)}(H^{(s)} || H^{(a)}) \tag{10}$$

And similarly, we can get the $H_{f_q}^{(k+1)}$.

The cross-attention encoder is based on the formula of the scaled dot-product attention from the self-attention mechanism [40], but with two key differences: 1) There are two different sequences of vectors are used for the cross-attention encoder, while only one sequence of vectors is used for the self-attention mechanism; 2) The self-attention aims to capture the relationship between itself and data in the datasets, while the cross-attention encoder focuses on capturing the relationship between the query and the data graph.

5.3 Structure-Attribute Consistency

The current non-learning-based approaches for ACS consider the structure and attribute separately, whereas AQD-GNN, a learning-based method, fuses these representations using a feature fusion operator. However, AQD-GNN falls short in considering the correlation between query nodes and query attributes, as it only concatenates these representations. Although structures and attributes

come from two heterogeneous spaces, they are a paired sample to describe a node, and thus they are close-related to each other and should be in close proximity to each other in the latent space. Each node's representation is a point in the latent space, and representations of multiple nodes form a distribution [19]. In this paper, we introduce a structure-attribute consistency constraint that aims to minimize the discrepancy between the distribution of structure representation and the distribution of attribute representation from the perspective of the whole graph.

There have been many metrics to measure the discrepancy between two distributions, such as the Kullback-Leibler (KL) divergence [31] and the Jensen-Shannon (JS) divergence [32]. In this paper, we use the Wasserstein distance [41]. It has a better property since two distributions converge under Wasserstein distance while failing to exhibit convergence under KL and JS divergences in some cases [1, 2, 42]. The Wasserstein distance is defined:

Definition 7. *(Wasserstein Distance). Given random variables μ and ν that are subject to probability distributions $\mathbb{P}_s$ and $\mathbb{P}_a$, the Wasserstein-1 distance W_1 between distributions $\mathbb{P}_s$ and $\mathbb{P}_a$ is defined:*

$$W_1(\mathbb{P}_s, \mathbb{P}_a) = \inf_{\gamma \in \pi(\mathbb{P}_s, \mathbb{P}_a)} \mathbb{E}_{(\mu,\nu)\sim\gamma}[||\mu - \nu||] \tag{11}$$

where $\pi(\mathbb{P}_s, \mathbb{P}_a)$ denotes the set of all joint distributions whose marginals are $\mathbb{P}_s$ and $\mathbb{P}_a$ respectively.

In this definition, γ is the "mass" needed to be transported from μ to ν to transform the distributions $\mathbb{P}_s$ into the distribution $\mathbb{P}_a$. And in this case, the element $\gamma_{i,j}$ is the probability that $h_{v_i}^{(s)} \in H^{(s)}$ matches $h_{v_j}^{(a)} \in H^{(a)}$. As the infimum in Equation 11 is highly intractable. The Wasserstein-1 distances can be reformulated with the Kantorovich-Rubinstein duality [20]:

$$W_1(\mathbb{P}_s, \mathbb{P}_a) = \sup_{||f_w||_L \le 1} \mathbb{E}_{\mu\sim\mathbb{P}_s}[f_w(\mu)] - \mathbb{E}_{\nu\sim\mathbb{P}_a}[f_w(\nu)] \tag{12}$$

where f_w satisfies the 1-Lipschitz condition that map the μ, ν in variable space to the real space $\mathbb{R}$.

By clamping the weights of f_w to a fixed box [2, 42], the Wasserstein distance is minimized when f_w is optimized to minimize:

$$\mathcal{L}_w(H^{(s)}, H^{(a)}) = \sum_{h_v^{(a)} \in H^{(a)}} f_w(h_v^{(a)}) - \sum_{h_u^{(s)} \in H^{(s)}} f_w(h_u^{(s)}) \tag{13}$$

5.4 Local Consistency

Existing learning-based community search and attributed community search models have been implemented by modeling the problem as a binary node classification task over the entire graph, which can be flawed as nodes within a real-world community are not isolated but rather cohesively connected as a unified module. Hence, solely relying on the gradient signal from the node classification task may not be sufficient. To alleviate this issue, we introduce the local consistency for the ACS to enhance the link between nodes in the predicted community in this section.

In particular, the structure representation and the attribute representation are first concatenated before sending to an MLP for computing the final community score:

$$H = H^{(s)}||H^{(a)} \tag{14}$$

We then minimize the loss between the self-multiplication of the concatenated matrix and the adjacency matrix as follows:

$$\mathcal{L}_m(H, A) = \left\| A - HH^T \right\|_F \tag{15}$$

where A is the adjacency matrix, and $||\cdot||_F$ denotes the Frobenius norm of the matrix [6]. The auxiliary link prediction objective $\mathcal{L}_m(\cdot)$ captures the idea that neighboring nodes should be predicted together, based on the intuition that if one node belongs to a community, its neighbors are also likely to belong to the same community, and vice versa.

5.5 Learning Objectives

There are three learning objectives in *ConNet*. The first one is in Equation 13 that aims to preserve the structure-attribute consistency via minimizing the Wasserstein distance between the distribution of structure and the distribution of attribute. The second one is in Equation 15 which aims to maintain local consistency to enhance the link between nodes in the community. And the third loss uses the binary cross entropy (BCE) which aims to minimize the difference between the predicted community and the ground-truth community. For a query q_i, the predicted community score is denoted as $\tilde{C}_{q_i} \in [0,1]^{|V_{sub}|}$ and the ground-truth community is denoted as $C_{q_i} \in \{0,1\}^{|V_{sub}|}$, the BCE loss is defined as:

$$\mathcal{L}_b(\tilde{C}_{q_i}, C_{q_i}) = \sum_{j=1}^{|V_{sub}|} -\Big(C_{q_i,j} log(\tilde{C}_{q_i,j}) + (1 - C_{q_i,j}) log(1 - \tilde{C}_{q_i,j})\Big) \tag{16}$$

where $C_{q_i,j}$ is the j-th bit of C_{q_i}.

The task of ACS is then modeled as multi-task learning to take the above three losses into account together. The overall loss function is defined as:

$$\mathcal{L} = \mathcal{L}_b + \alpha\mathcal{L}_w + \beta\mathcal{L}_m \tag{17}$$

where $\alpha, \beta \in [0, 1]$ are the coefficients to balance the above three loss functions. Note that $\mathcal{L}_w$ and $\mathcal{L}_m$ are two unsupervised losses that do not require the ground-truth labels. This property allows $\mathcal{L}_w$ and $\mathcal{L}_m$ to be easily adaptable to various scenarios, such as incomplete or poor ground-truth data, thereby enhancing the generalization ability and robustness of *ALICE*.

5.6 Analysis and Discussion

We now analyze the expressive power and the structure-preserving ability of the cross-attention encoder. We then further discuss the time complexity of *ALICE*.

Expressive Power and Structure-Preserving Ability. Here we prove that the proposed graph neural network of *ConNet* is as powerful as the Weisfeiler-Lehman (WL) isomorphism test.

Lemma 3. *There exist parameters for K-layered GINs such that, for any positive integer K, if the degrees of nodes are bounded by a constant and the size of node features is finite, and for any graphs g_1*

Table 2: Statistics of the datasets

Dataset	$\|V\|$	$\|E\|$	$\|F^d\|$	N_c
Texas	187	279	1703	5
Cornell	195	285	1703	5
Washt	230	392	1703	5
Wiscs	265	469	1703	5
Cora	2708	5429	1433	7
Citeseer	3312	4715	3703	6
Google+	7856	321,268	2024	91
PubMed	19,717	44,324	500	3
Reddit	232,965	47,396,905	602	41
Orkut	3,072,627	117,185,083	1000	5000
Friendster	65,608,366	1,806,067,135	1000	5000

and g_2, if the 1-WL algorithm outputs that g_1 and g_2 are not isomorphic within K rounds, then the embeddings of g_1 and g_2 computed by the GIN are distinct.

The proof of LEMMA 3 can be found in [11].

Lemma 4. *If the 1-WL algorithm outputs that g_1 and g_2 are not isomorphic within K rounds, the embeddings of g_1 and g_2 computed by the cross-attention encoder are different.*

Proof. The output of the cross-attention encoder concatenates two sources of embedding,i.e., the query H_q and data graph H_g, and the GIN is used to encode the data graph. Given g_1 and g_2, and its outputs are $H_1 = H_{q_1}||H_{g_1}$ and $H_2 = H_{q_2}||H_{g_2}$. If g_1 and g_2 are "non-isomorphic" within K round, H_{g_1} and H_{g_2} which are the output of g_1 and g_2 by GIN should be different by LEMMA 3. Hence H_1 and H_2 must be different since two different vectors concatenating any vectors will result in two different vectors. Therefore, the cross-attention encoder is as powerful as 1-WL test. □

Complexity Analysis. The time complexity of *ALICE* consists of the time cost of candidate subgraph extraction and the time cost for *ConNet*. The time complexity of candidate subgraph extraction is $O(|E| + 2 \times |V| \times |F^d|)$ since it needs to construct the node-attribute bipartite graph and propagates through each edge to find the k-hop neighbors in the data graph and the bipartite graph. *ConNet* needs to run for K layers and is trained for t epochs, and *ConNet* is applied in the candidate subgraph $G_{sub} = (V_{sub}, E_{sub})$. The time complexity of the projection of three matrices is $O(3 \times |V_{sub}| \times d^2)$ where d is the maximum latent dimension. The dot product of query and key takes $O(|V_{sub}|^2 \times d)$ and the dot product of attention and value also takes $O(|V_{sub}|^2 \times d)$. The time complexity of GIN applied in the candidate subgraph is $O(|E_{sub}|)$ [42]. Generally, $|F_d| \approx |V_{sub}|$, hence we assume the time expended for attribute encoding closely approximates that of structure encoding, and the time required for each layer is close. Therefore, the overall time complexity of *ConNet* is $O(t \times K \times 2 \times (3 \times |V_{sub}| \times d^2 + 2 \times |V_{sub}|^2 \times d + |E_{sub}|))$.

6 EXPERIMENTAL EVALUATION

In this section, we evaluate the performance of *ALICE* compared with a variety of existing solutions over 11 real-world benchmark datasets with a maximum of 65 million nodes and 1.8 billion edges.

6.1 Dataset Description

We use 11 public datasets following the previous research [15, 18, 25, 28] to conduct the experiments. The statistics information is summarized in Table 2 where $|V|$ is the number of nodes, $|E|$ denotes the number of edges, $|F^d|$ is the number of distinct attributes, and N_c is the number of communities in the graph. The first nine datasets, including *Texas*, *Cornell*, *Washington (Washt)*, *Wisconsin (Wiscs)*, *Cora*, *Citeseer*, *Google+*, *Pubmed*, and *Reddit*, are attributed graphs with ground-truth communities. To further test the scalability and efficiency of the approaches over large graphs, we add two non-attributed datasets, *Orkut* and *Friendster*, that contain 5000 top-quality ground-truth communities. We generate an attribute pool consisting of $|F^d| = 1000$ different attributes for these two graphs following the processing phase in [25].

6.2 Experimental Setup

Baseline: We use three baselines for ACS including: 1) ACQ [15], which is a non-learning k-core-based model; 2) ATC [25], which is a non-learning k-truss-based model; 3) AQD-GNN [28], which is a learning-based model with feature fusion. We also test the performance of *ALICE* compared with ICS-GNN [18], which is a GNN-based interactive community search model. Two baselines are also used for the comparison of non-attributed community search including 1) CTC [27], which is a k-truss-based community search model; 2) k-ECC [7], which models communities as k-edge connected components.

Query Setting: We categorize all ground-truth communities into three distinct groups: training communities, validation communities and test communities. The ratio of these groups is approximately 5:1:4. This process serves to evaluate the performance when the model is exposed to previously unseen communities. And then, we generate 150 pairs of training data as $\mathcal{D}_{train} = \{q_i, C_{q_i}\}_{i=1}^{150}$. Each query $q_i =< V_{q_i}, F_{q_i} >$ contains the query node set V_{q_i} and the query attribute set F_{q_i}. C_{q_i} is the corresponding ground-truth community of q_i in the training communities. We then generate 100 pairs of validation queries and 100 pairs of test queries. We only split those datasets that have larger than 10 ground-truth communities (i.e., *Google+, Reddit, Orkut, and Friendster*) to avoid insufficient information on ground-truth communities. The training data is utilized to train our model, the validation data is used to select the optimal threshold of the predicted score to determine the community, and the test data is employed to evaluate the performance. We randomly select $1 \sim 3$ nodes from the ground-truth community as the query nodes of each query. In addition, we use the following three mechanisms to generate the query attributes.

- **Empty attribute query (EmA).** We set the attribute query set empty $F_{q_i} = \emptyset$, and query in the EmA is $q_i =< V_{q_i}, \emptyset >$.
- **Attribute from communities (AFC).** We first select the 5 most common attributes in ground-truth communities and then randomly select one of these attributes as the query attribute. Query in the AFC query set is $q_i =< V_{q_i}, F^c_{q_i} >$.
- **Attribute from the query node (AFN).** We directly use the attribute from the query nodes as the query attribute. Query in the AFN query set is $q_i =< V_{q_i}, F^n_{q_i} >$.

Metrics: Combining metrics used in existing works[15, 25, 28], we use three widely used metrics to evaluate the quality of the found communities, including F1-score [25, 28], average degree (Avg.d) [15], and the Community pair-wise Jaccard (CPJ) [15]. We evaluate the found communities within the candidate subgraph. As the primary objective of ACS is to identify a community that really matches the expectations, we mainly focus on the metric of F1-score

which measures the alignment between the found communities and the ground-truth communities. Given the ground-truth community set denoted as $C = \{C_{q_1}, \cdots, C_{q_t}\}$ and the predicted community sets denoted as $\tilde{C} = \{\tilde{C}_{q_1}, \cdots, \tilde{C}_{q_t}\}$, the F1-score which is based on precision and recall is defined as follows. Here, C_{q_i} and $\tilde{C}_{q_i}$ are the ground-truth and predicted community vectors for query q_i.

$$pre(C, \tilde{C}) = \frac{\sum_{i=1}^{t} \sum_j C_{q_i,j} \cdot \tilde{C}_{q_i,j}}{\sum_{i=1}^{t} \sum_j \tilde{C}_{q_i,j}}, rec(C, \tilde{C}) = \frac{\sum_{i=1}^{t} \sum_j C_{q_i,j} \cdot \tilde{C}_{q_i,j}}{\sum_{i=1}^{t} \sum_j C_{q_i,j}}$$

$$F1(\tilde{C}, C) = \frac{2 \cdot pre(C, \tilde{C}) \cdot rec(C, \tilde{C})}{pre(C, \tilde{C}) + rec(C, \tilde{C})}$$

Moreover, we use Avg.d of nodes in the community to measure the structure cohesiveness and the CPJ to measure the attribute cohesiveness as in [15]. The definitions of Avg.d and CPJ are given:

$$Avg.d(\tilde{C}) = \frac{1}{t} \sum_{i=1}^{t} \frac{1}{|\tilde{C}_{q_i}|} \sum_{v \in \tilde{C}_{q_i}} d(v)$$

$$CPJ(\tilde{C}) = \frac{1}{t} \sum_{i=1}^{t} \frac{1}{|\tilde{C}_{q_i}|^2} \sum_{u \in \tilde{C}_{q_i}} \sum_{v \in \tilde{C}_{q_i}} \frac{|F_u \cap F_v|}{|F_u \cup F_v|}$$

Where $d(v)$ is the degree of node v in $\tilde{C}_{q_i}$ and F_v is the attribute set of node v. Attributes that share the same label are considered equivalent. Note that, a higher value of F1-score or Avg.d or CPJ indicates the higher quality of the identified community.

Implementation Details: The latent dimension is set as 128. The model is trained for 300 epochs with early stopping. The loss balance coefficients α, β in Equation 17 are both set as 0.1. The γ of the density sketch modularity is set as 0.8. The candidate subgraph size is set as 1000 and runs for 20 rounds in ICS-GNN. We conduct experiments using $1 \sim 3$ query nodes and AFN as the query attribute by default. The dropout rate is set as 0.45. We set k of the baselines to 4 by default. Adam optimizer with a decaying learning rate is employed to train the model. We conduct our experiments on a machine with Intel(R) Xeon(R) Gold 6248R CPU, 512GB memory, and Nvidia A5000 (GPU).

6.3 Effectiveness Evaluation

Due to AQD-GNN running out of memory during the training on *Reddit/Orkut/Friendster* and ATC/CTC needing more than 7 days on *Friendster*, we omit these results.

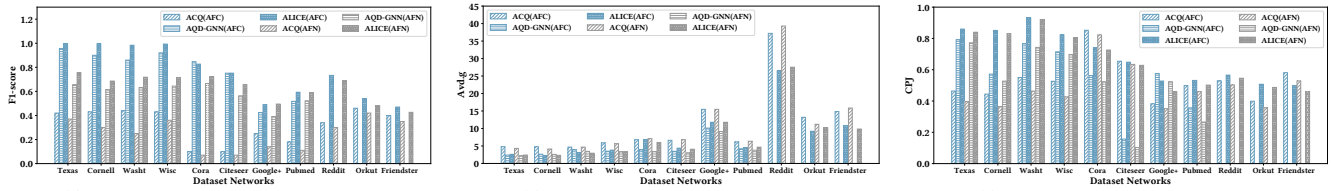
Exp-1: Attributed Community Search. We first evaluate our algorithm for the ACS. Figure 7 reports the result. We present two query scenarios: the one-node query, which employs a single query node, and the multi-node query, which utilizes 1 to 3 nodes as query nodes. Note that ACQ can only use one query node as input while ATC, AQD-GNN, and *ALICE* can support multiple query nodes. Therefore, we use ACQ as the non-learning-based baseline and AQD-GNN as the learning-based baseline for the one-node query in Figure 7 (a-c). We use ATC as the non-learning-based baseline and AQD-GNN as the learning-based baseline for the multi-node query in Figure 7 (d-f). When measured by F1-score, we observe that learning-based methods, including AQD-GNN and *ALICE*, exhibit superior performance compared with non-learning methods under both the one-node query and the multi-node query with an average improvement of 41.75% and 54.50% in F1-score, respectively. Overall, *ALICE* shows the best performance under both cases with an average F1-score 10.18% improvement compared with AQD-GNN using AFN as the query attributes.

In terms of Avg.d, non-learning methods possess an inherent advantage since they adopt structure cohesiveness as the objective. Across various queries, non-learning methods consistently tend to incorporate high-degree nodes into the community to promote structure cohesiveness, which is also a contributing factor to the low F1-score. In the context of learning-based methods, *ALICE* outperforms AQD-GNN. The performance stems from the local consistency that aligns the predictions of adjacent nodes, thereby enhancing the connection of nodes within a community. Additionally, the modularity-based pruning prioritizes cohesive subgraphs. When considering CPJ, we find that learning-based approaches exhibit superior performance as they can better approximate the relevance of attributes. Among 11 datasets, *ALICE* outperforms baselines in 8 datasets, which validates the effectiveness of *ALICE*.

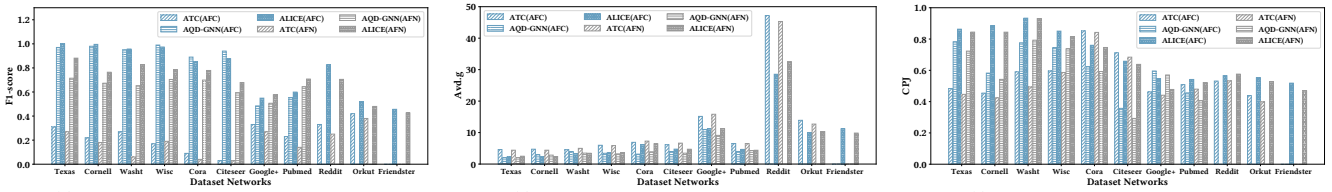
Exp-2: Non-Attributed Community Search. We further test the performance of *ALICE* for non-attributed community search. k-ECC [7] and CTC [27] are used as non-learning-based baselines. Both AQD-GNN and *ALICE* use the EmA setting for non-attributed community search where the query attribute is set as empty. The overall result is illustrated in Figure 8 (a). As depicted in the figures, the learning-based approach outperforms non-learning methods significantly. Among all the methods, *ALICE* exhibits the highest performance across the datasets with an average improvement of 40.03% when compared with k-ECC in F1-score and of 6.74% when compared with AQD-GNN in F1-score.

Exp-3: Interactive Community Search. Interactive community search (ICS) is recently proposed in ICS-GNN [18]. ICS-GNN generates an answer community in response to a query in multiple rounds, and the community can be refined through user feedback. ICS-GNN follows a process of identifying a candidate subgraph, learning node embeddings through a Vanilla GCN model, and using a BFS-based algorithm to select a community of size k with the highest GNN scores. In this part, we replace the Vanilla GCN model with AQD-GNN and *ALICE* to evaluate the performance of ICS. The results of three models are depicted in Figure 8 (b). From the figure, we find that *ALICE* consistently shows the best performance.

Exp-4: ACS under incomplete gound-truth. This section investigates the robustness of learning-based techniques when dealing with incomplete ground-truth information. Figure 8 (c) illustrates the results. Specifically, for each training pair $\{q_i, C_{q_i}\}$, we randomly mask 50% of the nodes in C_{q_i} and rely solely on the remaining half as the ground-truth to train the model. We use AQD-GNN and ATC as baselines as they support multi-node query. The figure shows that *ALICE* exhibits a notable level of effectiveness, with an average F1-score improvement of 16.05% and 30.34% compared to AQD-GNN and ATC, respectively. This is attributable to the modularity-based pruning and the local consistency that put structure cohesiveness prior to the community. Unlike AQD-GNN, which relies solely on the supervised signal $\mathcal{L}_b$, the proposed *ALICE* incorporates two new unsupervised signals, $\mathcal{L}_w$ and $\mathcal{L}_b$, to enhance the robustness of the model under incomplete ground-truth data.

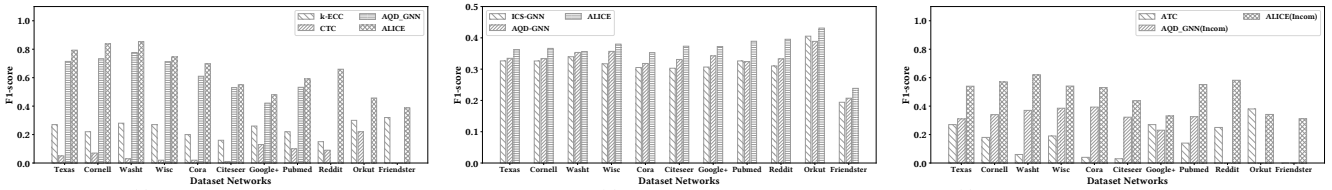


(a) F1-score of ACS over one-node query (b) Avg.d of ACS over one-node query (c) CPJ of ACS over one-node query



(d) F1-score of ACS over multi-node query (e) Avg.d of ACS over multi-node query (f) CPJ of ACS over multi-node query

Figure 7: Results on attributed community search



(a) Results on non-acs (b) Results on ics (c) Results with incomplete ground-truth

Figure 8: Community search with other settings

6.4 Efficiency Evaluation

Exp-5: Evaluation of Different Stages. Since the learning-based methods have demonstrated superior performance compared to other baseline methods. Hence, we further assess the efficiency of ICS-GNN, AQD-GNN and *ALICE*. Table 3 presents two phases of time, including train time and query time, both of which are crucial for efficiency. We report data preparation time plus the model train (query) time for each phase. Data preparation involves loading the data, initializing features, and extracting candidate subgraphs. Note that, ICS-GNN needs to train one model for one query, thus we report the total time needed for one query in the query time phase. The algorithms are trained using a dataset comprising 150 samples and are subsequently queried using one single data sample. "—" indicates that the algorithm is out of memory or needs more than 7 days during the evaluation. "***" indicates that one cell is not applicable for one model. The table shows that AQD-GNN is faster than *ALICE* in the extremely small graph with only hundreds of nodes while *ALICE* performs much more efficiently for large-scale graphs. Moreover, *ALICE* can finish training on *Reddit*, *Orkut* and *Friendster*, whereas ADQ-GNN cannot. The reason is that AQD-GNN uses the whole graph as the input and trains the model. For the medium and large graphs, the pruning process can significantly reduce the training size and avoid the issue of being out of memory. The query time of both AQD-GNN and *ALICE* are consistent and are much smaller than the preparation and training time. Moreover, ICS-GNN needs much more query time as it needs to train one model for one input query while training a model is a time-intensive task.

Exp-6: Scalability Evaluation. In this part, we evaluate the scalability of AQD-GNN and *ALICE* in Figure 9. The total time consists of the data preparation time, the model training time, and the query time. The result of the relation between total time and node number

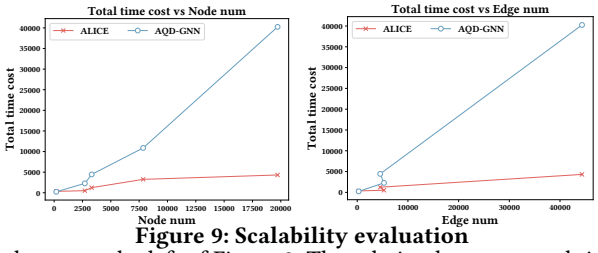


Figure 9: Scalability evaluation

is shown on the left of Figure 9. The relation between total time and edge number is shown on the right of Figure 9. From the figure, we find that the time cost of *ALICE* grows at a much slower rate than AQD-GNN as the number of nodes or edges increases, which confirms the high scalability of our design.

6.5 Ablation Study

Exp-7: Different Components. In this part, we conduct experiments over different variants of *ALICE* to verify the effectiveness of each component in the proposed model. The overall results are demonstrated in Figure 10. The first variant (denoted by w/o Pruning) is trained on the whole graph to test the influence of candidate subgraph extraction. The second variant (denoted by w/o $\mathcal{L}_b$) is trained without the loss $\mathcal{L}_b$. The third variant (denoted by w/o $\mathcal{L}_w$) is trained without the loss $\mathcal{L}_w$ to check the effectiveness of the structure-attribute consistency. The fourth variant (denoted by w/o $\mathcal{L}_m$) is trained without the loss $\mathcal{L}_m$ to confirm the effectiveness of the local consistency. We also test the effectiveness of these components under incomplete ground-truth (denoted by Incom) as Exp-4. As shown in the figure, we can find that all four components can improve the accuracy of ACS. The prediction of the community is most heavily influenced by $\mathcal{L}_b$ since it supplies ground-truth information. The pruning technique can effectively reduce the search space, and hence be helpful to increase both the effectiveness and efficiency of the model. Additionally, it is observed that $\mathcal{L}_w$ and $\mathcal{L}_m$

Table 3: Efficiency evaluation on different datasets (in seconds)

Method	*Texas*	*Cornell*	*Washt*	*Wisc*	*Cora*	*Citeseer*	*Google+*	*Pubmed*	*Reddit*	*Orkut*	*Frienster*
ICS-GNN (Train)	***	***	***	***	***	***	***	***	***	***	***
AQD-GNN (Train)	2.2+233	2.1+234	2.5+239	2.9+232	64.1+2214	59.3+4390	834.6+10035	3171.8+37059	—	—	—
ALICE (Train)	2.6+344	2.5+381	3.8+332	1.8+324	16.32+509	59.8+1239	189.8+3256	123.5+4317	8681+1107	2594.8+2224	65415.6+1244
ICS-GNN (Query)	20.5	25.1	27.4	28.6	167.7	124.3	627.6	112.3	1034.7	1540.8	24253.7
AQD-GNN (Query)	0.015+0.0021	0.014+0.0020	0.017+0.0022	0.019+0.0020	0.427+0.0026	0.395+0.0019	5.564+0.0019	21.14+0.0019	—	—	—
ALICE (Query)	0.017+0.0053	0.017+0.0045	0.025 + 0.0044	0.014 +0.0050	0.104+0.0041	0.398+0.0047	1.26+0.0053	0.823+0.0058	5.78+0.0052	17.29+0.0045	436.1+0.0048

(1) : We report preparation time + train (query) time; (2) : — indicates out of memory or not finished within 7 days; (3) : *** indicates this cell not applicable to this model.

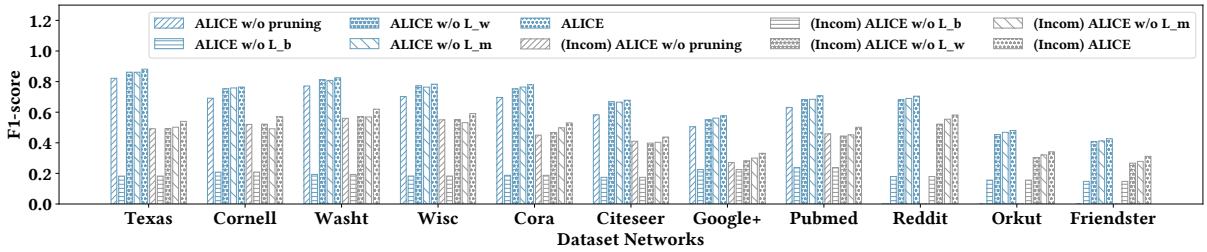
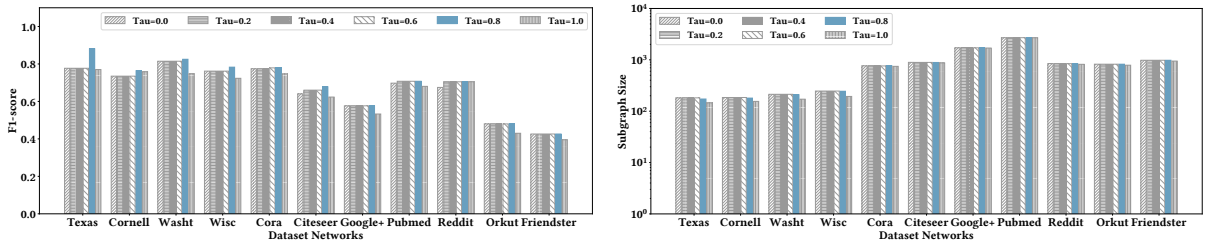


Figure 10: Ablation study



(a) Accuracy (b) Subgraph size

Figure 11: Comparison of different modularities

can enhance the F1-score by 2.71% and 2.16% in average respectively under complete ground-truth labels and improve performance by 4.57% and 4.03% respectively under incomplete ground-truth information. The results validate the effectiveness of $\mathcal{L}_w$ and $\mathcal{L}_m$, especially under incomplete ground-truth labels.

Exp-8: Different Modularity Definitions. In this section, we test the model performance with density sketch modularity of varying τ. Note that different τ will lead to different modularity definitions, e.g, $\tau = 0$ leads to classical modularity and $\tau = 1$ leads to density modularity as analyzed in Section 4. The result is reported in Figure 11, with the accuracy in Figure 11(a) and the subgraph size comparison in Figure 11(b). The figure shows that (1) density sketch modularity with $\tau = 0.8$ has a better F1-score than other choices on the test datasets; (2) the subgraph size remains the same or decreases as τ increases, which aligns with our analysis in Section 4.

6.6 Hyper-parameter Sensitivity

Exp-9: Varying α and β. In this section, we explore the effect of α and β to the performance of *ALICE*. We use two datasets including *Texas* and *Cornell*. The results are illustrated in Figure 12. The figure demonstrates that parameters of small but positive values (i.e., 0.1~0.5) outperform other settings. Specifically, we set both α and β varying from 0 to 1 and report the F1-score at an interval of 0.1. The results show that (1) α and β in the range from 0.1 to 0.5 outperform other settings on the test datasets; (2) α and β of small but positive values outperform those of zero. This further validates the effectiveness of $\mathcal{L}_w$ and $\mathcal{L}_m$.

Exp-10: Training Epoch. In this part, we explore the impact of the training epoch on the accuracy of ACS. We train the model from 1 to 300 epochs and evaluate the accuracy every 20 epochs. Figure 13(a) presents the results. The figure shows that the performance of *ALICE* improves considerably as the training epoch increases. After 200 epochs, the rate of improvement decelerates, eventually leading the model to attain a stable state.

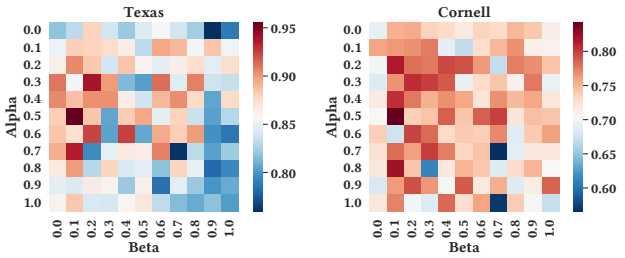
Exp-11: Training Loss. In this part, we investigate the loss during training. Figure 13 (b) reports the loss of *ALICE* over different datasets. In the figure, it is evident that the loss diminishes significantly at the onset, decreasing by approximately 95% following 20 epochs. Thereafter, the loss progressively converges towards 0.

Exp-12: Training Set Size. We also study the sensitivity of the number of samples used in the training phase. We increase the size from 50 to 300 and test the performance of the model every 50 samples. The result is reported in Figure 13 (c). It can be observed that the accuracy increases with the sample size ranging from 50 to 150. After the sample size reaches 150, the accuracy stabilizes.

Exp-13: Validation Set Size. We test the accuracy with varying numbers of samples used in the validation phase. The sample size was increased from 50 to 300, and the accuracy is reported every 50 samples. The results are presented in Figure 13(d). The figure demonstrates that there is a consistent level of accuracy across different sample sizes of validation.

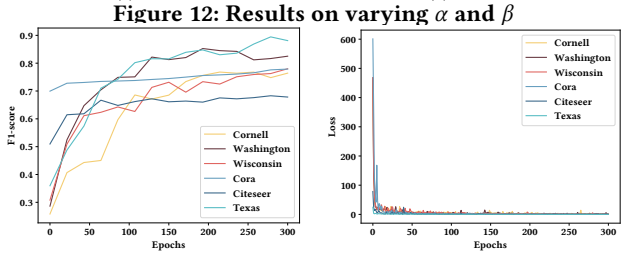
6.7 Case Study

In this part, we conduct a case study using arXiv [3] which is a co-author network comprising a diverse collection of scholarly articles from fields such as physics, mathematics, and computer science. We select articles from the field of computer science published between 2011 and 2015, and build a co-author network consisting of 42,065 nodes and 102,165 edges. The network encompasses 40

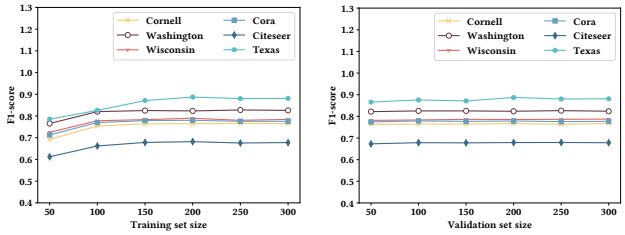


(a) Texas (b) Cornell

Figure 12: Results on varying α and β

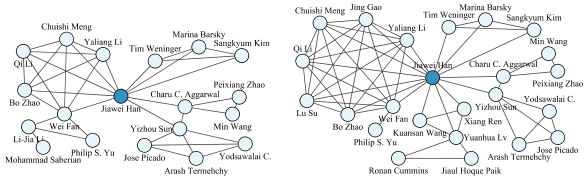


(a) Varying epoch number (b) Train loss

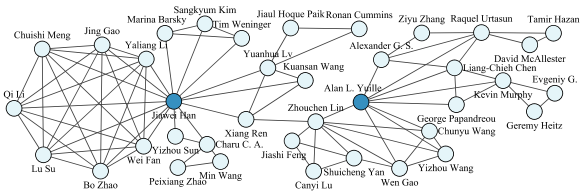


(c) Varying training set size (d) Varying validation set size

Figure 13: Experiments on hyper-parameter sensitivity



(a) AQD-GNN (b) ALICE



(c) ALICE for query {Jiawei Han, Alan L. Yuille, DB, IR, CV}

Figure 14: Case study

subdisciplines, e.g., database (DB), and information retrieval (IR). We consider Prof Jiawei Han, a renowned researcher in database and data mining. We use Jiawei Han as a query node and use 2 query attributes, including DB and IR. The return communities are shown in Figure 14. Specifically, Figure 14 (a) illustrates the community discovered by AQD-GNN and Figure 14 (b) demonstrates the community discovered by *ALICE*. As AQD-GNN is out of memory, we input the candidate subgraph generated by *ALICE* for both methods, with a cap of 400 nodes. We delete unconnected nodes from the result communities. The figure shows that both methods can effectively find the research collaborators of Jiawei Han. Moreover, we find that AQD-GNN misses some promising candidates, e.g., Xiang Ren who is a former student of Jiawei Han and was in close cooperation with Jiawei Han. In contrast, *ALICE* can discover such an important candidate. Additionally, we further consider Prof Alan L. Yuille, a renowned researcher in computer vision (CV). We use Jiawei Han and Alan L. Yuille as query nodes and use {DB, IR, CV} as query attributes. The result is illustrated in Figure 14 (c). We can find that although Jiawei Han and Alan L. Yuille belong to two different communities under the arXiv network, nodes in the returned community are closely related to the query.

7 RELATED WORK

In this section, we review related studies regarding classical community search and ML/DL for community search.

Classical Community Search. Community search aims to find cohesive subgraphs in a graph that contain query nodes and satisfy given constraints. Some well-known classical methods use k-related measurements to model the community such as k-core [13, 46, 54], k-truss [24, 44], k-clique [45, 52], and k-edge connected component (k-ECC) [7, 23]. However, these methods suffer from structure inflexibility (i.e., the real-world community may always dissatisfy with the constraints). Recently, a graph modularity-based method [10] is proposed, which aims to find a subgraph that has the maximum graph modularity. When targeting attributed graphs, several algorithms have been proposed that consider both structure and keyword cohesiveness, e.g. ACQ [15] and ATC [25]. Both algorithms adopt a two-stage procedure that first considers the structure constraint and then selects the community with the highest attribute score. However, they fail to capture the correlation between structure and attribute that are closely related.

ML/DL for Community Search. With the powerful approximation ability of the neural network, learning-based techniques (like GNNs) have been adopted for community search. The general approach of using GNNs for community search is to model the problem as a binary node classification task, where the goal is to predict the probability of each vertex belonging to a community. ICS-GNN [18] leverages a GNN model to search community in an iterative manner where the community is modeled as a k-sized subgraph with maximum GNN scores. AQD-GNN [28] is proposed to support the attributed graph that takes both the cohesive structure and homogeneous attributes into account. However, these methods suffer from severe efficiency issues.

8 CONCLUSION

In this paper, we explore the problem of attributed community search. To improve the accuracy and efficiency, we propose a novel model *ALICE* that first extracts the candidate subgraph and then predicts the community based on the query and candidate subgraph. In the candidate subgraph extraction phase, we design a new modularity named density sketch modularity and adaptively select a reasonable amount of neighbors considering both structure and attribute. In the prediction phase, we devise *ConNet* to integrate consistency constraints for the prediction of the attributed community. It utilizes a cross-attention encoder to encode the interaction information between the query and the data graph. Structure-attribute consistency and local consistency are utilized to guide the training of the model. We conduct various experiments over 11 real-world benchmark datasets from multiple aspects. The results demonstrate that *ALICE* shows a good performance in terms of prediction accuracy, efficiency, robustness, and scalability.

REFERENCES

[1] Martín Arjovsky and Léon Bottou. 2017. Towards Principled Methods for Training Generative Adversarial Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=Hk4_qw5xe

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein generative adversarial networks. In *International conference on machine learning*. PMLR, 214–223.

[3] arXiv.org submitters. 2023. arXiv Dataset. https://doi.org/10.34740/KAGGLE/DSV/6101996

[4] Michael J Barber. 2007. Modularity and community detection in bipartite networks. *Physical Review E* 76, 6 (2007), 066102.

[5] Sourav S Bhowmick and Boon Siew Seah. 2015. Clustering and summarizing protein-protein interaction networks: A survey. *IEEE Transactions on Knowledge and Data Engineering* 28, 3 (2015), 638–658.

[6] Albrecht Böttcher and David Wenzel. 2008. The Frobenius norm and the commutator. *Linear algebra and its applications* 429, 8-9 (2008), 1864–1885.

[7] Lijun Chang, Xuemin Lin, Lu Qin, Jeffrey Xu Yu, and Wenjie Zhang. 2015. Index-based optimal algorithms for computing steiner components with maximum connectivity. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*. 459–474.

[8] Xu Chen, Siheng Chen, Jiangchao Yao, Huangjie Zheng, Ya Zhang, and Ivor W Tsang. 2020. Learning on attribute-missing graphs. *IEEE transactions on pattern analysis and machine intelligence* 44, 2 (2020), 740–757.

[9] Dawei Cheng, Fangzhou Yang, Xiaoyang Wang, Ying Zhang, and Liqing Zhang. 2020. Knowledge graph-based event embedding framework for financial quantitative investments. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2221–2230.

[10] Junghoon Kim, Siqiang Luo, Gao Cong, and Wenyuan Yu. 2022. DMCS : Density Modularity based Community Search. In *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*. ACM, 889–903.

[11] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How Powerful are Graph Neural Networks?. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

[12] Wanyun Cui, Yanghua Xiao, Haixun Wang, Yiqi Lu, and Wei Wang. 2013. Online search of overlapping communities. In *Proceedings of the 2013 ACM SIGMOD international conference on Management of data*. 277–288.

[13] Wanyun Cui, Yanghua Xiao, Haixun Wang, and Wei Wang. 2014. Local search of communities in large graphs. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. 991–1002.

[14] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph neural networks for social recommendation. In *The world wide web conference*. 417–426.

[15] Yixiang Fang, Reynold Cheng, Siqiang Luo, and Jiafeng Hu. 2016. Effective community search for large attributed graphs. *Proceedings of the VLDB Endowment* 9, 12 (2016), 1233–1244.

[16] Yixiang Fang, Xin Huang, Lu Qin, Ying Zhang, Wenjie Zhang, Reynold Cheng, and Xuemin Lin. 2020. A survey of community search over big graphs. *The VLDB Journal* 29 (2020), 353–392.

[17] Santo Fortunato and Marc Barthelemy. 2007. Resolution limit in community detection. *Proceedings of the national academy of sciences* 104, 1 (2007), 36–41.

[18] Jun Gao, Jiazun Chen, Zhao Li, and Ji Zhang. 2021. ICS-GNN: lightweight interactive community search via graph neural network. *Proceedings of the VLDB Endowment* 14, 6 (2021), 1006–1018.

[19] Ji Gao, Xiao Huang, and Jundong Li. 2021. Unsupervised graph alignment with wasserstein distance discriminator. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 426–435.

[20] Nathael Gozlan, Cyril Roberto, Paul-Marie Samson, and Prasad Tetali. 2017. Kantorovich duality for general transport costs and applications. *Journal of Functional Analysis* 273, 11 (2017), 3327–3405.

[21] Jiahao Guo, Pramesh Singh, and Kevin E Bassler. 2023. Resolution limit revisited: community detection using generalized modularity density. *Journal of Physics: Complexity* 4, 2 (2023), 025001.

[22] Leon Hetzel, David S Fischer, Stephan Günnemann, and Fabian J Theis. 2021. Graph representation learning for single-cell biology. *Current Opinion in Systems Biology* 28 (2021), 100347.

[23] Jiafeng Hu, Xiaowei Wu, Reynold Cheng, Siqiang Luo, and Yixiang Fang. 2016. Querying minimal steiner maximum-connected subgraphs in large graphs. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. 1241–1250.

[24] Xin Huang, Hong Cheng, Lu Qin, Wentao Tian, and Jeffrey Xu Yu. 2014. Querying k-truss community in large and dynamic graphs. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. 1311–1322.

[25] Xin Huang and Laks VS Lakshmanan. 2017. Attribute-driven community search. *Proceedings of the VLDB Endowment* 10, 9 (2017), 949–960.

[26] Xin Huang, Laks VS Lakshmanan, and Jianliang Xu. 2019. Community search over big graphs. *Synthesis Lectures on Data Management* 14, 6 (2019), 1–206.

[27] Xin Huang, Laks V. S. Lakshmanan, Jeffrey Xu Yu, and Hong Cheng. 2015. Approximate Closest Community Search in Networks. *Proc. VLDB Endow.* 9, 4 (2015), 276–287. https://doi.org/10.14778/2856318.2856323

[28] Yuli Jiang, Yu Rong, Hong Cheng, Xin Huang, Kangfei Zhao, and Junzhou Huang. 2022. Query driven-graph neural networks for community search: from non-attributed, attributed, to interactive attributed. *Proceedings of the VLDB Endowment* 15, 6 (2022), 1243–1255.

[29] Junghoon Kim, Kaiyu Feng, Gao Cong, Diwen Zhu, Wenyuan Yu, and Chunyan Miao. 2022. ABC: attributed bipartite co-clustering. *Proceedings of the VLDB Endowment* 15, 10 (2022), 2134–2147.

[30] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=SJU4ayYgl

[31] Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics* 22, 1 (1951), 79–86.

[32] Christopher Manning and Hinrich Schutze. 1999. *Foundations of statistical natural language processing*. MIT press.

[33] Haggai Maron, Heli Ben-Hamu, Hadar Serviansky, and Yaron Lipman. 2019. Provably powerful graph networks. *Advances in neural information processing systems* 32 (2019).

[34] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems* 26 (2013).

[35] Mark EJ Newman and Michelle Girvan. 2004. Finding and evaluating community structure in networks. *Physical review E* 69, 2 (2004), 026113.

[36] Mark EJ Newman, Duncan J Watts, and Steven H Strogatz. 2002. Random graph models of social networks. *Proceedings of the national academy of sciences* 99, suppl_1 (2002), 2566–2572.

[37] Joseph J Pfeiffer III, Sebastian Moreno, Timothy La Fond, Jennifer Neville, and Brian Gallagher. 2014. Attributed graph models: Modeling network structure with correlated attributes. In *Proceedings of the 23rd international conference on World wide web*. 831–842.

[38] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*. PMLR, 8748–8763.

[39] William Shiao, Zhichun Guo, Tong Zhao, Evangelos E. Papalexakis, Yozen Liu, and Neil Shah. 2023. Link Prediction with Non-Contrastive Learning. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net. https://openreview.net/pdf?id=9Jaz4APHtWD

[40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[41] Cédric Villani et al. 2009. *Optimal transport: old and new*. Vol. 338. Springer.

[42] Hanchen Wang, Rong Hu, Ying Zhang, Lu Qin, Wei Wang, and Wenjie Zhang. 2022. Neural Subgraph Counting with Wasserstein Estimator. In *Proceedings of the 2022 International Conference on Management of Data*. 160–175.

[43] Hanchen Wang, Ying Zhang, Lu Qin, Wei Wang, Wenjie Zhang, and Xuemin Lin. 2022. Reinforcement Learning Based Query Vertex Ordering Model for Subgraph Matching. In *38th IEEE International Conference on Data Engineering, ICDE 2022, Kuala Lumpur, Malaysia, May 9-12, 2022*. IEEE, 245–258.

[44] Kai Wang, Xuemin Lin, Lu Qin, Wenjie Zhang, and Ying Zhang. 2020. Efficient bitruss decomposition for large-scale bipartite graphs. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE, 661–672.

[45] Kai Wang, Wenjie Zhang, Xuemin Lin, Lu Qin, and Alexander Zhou. 2022. Efficient personalized maximum biclique search. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 498–511.

[46] Kai Wang, Gengda Zhao, Wenjie Zhang, Xuemin Lin, Ying Zhang, Yizhang He, and Chunxiao Li. 2023. Cohesive Subgraph Discovery over Uncertain Bipartite Graphs. *IEEE Transactions on Knowledge and Data Engineering* (2023).

[47] Yanhao Wang, Yuchen Li, Ju Fan, Chang Ye, and Mingke Chai. 2021. A survey of typical attributed graph queries. *World Wide Web* 24 (2021), 297–346.

[48] Yiqi Wang, Long Yuan, Zi Chen, Wenjie Zhang, Xuemin Lin, and Qing Liu. 2023. Towards efficient shortest path counting on billion-scale graphs. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*. IEEE, 2579–2592.

[49] Yubao Wu, Ruoming Jin, Jing Li, and Xiang Zhang. 2015. Robust local community detection: on free rider effect and its elimination. *Proceedings of the VLDB Endowment* 8, 7 (2015), 798–809.

[50] Sheng Xiang, Dawei Cheng, Chencheng Shang, Ying Zhang, and Yuqi Liang. 2022. Temporal and Heterogeneous Graph Neural Network for Financial Time Series Prediction. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 3584–3593.

[51] Dong-Hui Yang, Zhen-Yu Li, Xiao-Hui Wang, Kavé Salamatian, and Gao-Gang Xie. 2021. Exploiting the Community Structure of Fraudulent Keywords for Fraud Detection in Web Search. *Journal of Computer Science and Technology* 36 (2021), 1167–1183.

[52] Long Yuan, Lu Qin, Wenjie Zhang, Lijun Chang, and Jianye Yang. 2017. Index-based densest clique percolation community search in networks. *IEEE Transactions on Knowledge and Data Engineering* 30, 5 (2017), 922–935.

[53] Fan Zhang, Conggai Li, Ying Zhang, Lu Qin, and Wenjie Zhang. 2018. Finding critical users in social communities: The collapsed core and truss problems. *IEEE Transactions on Knowledge and Data Engineering* 32, 1 (2018), 78–91.

[54] Fan Zhang, Ying Zhang, Lu Qin, Wenjie Zhang, and Xuemin Lin. 2016. When engagement meets similarity: efficient (k, r)-core computation on social networks. *arXiv preprint arXiv:1611.03254* (2016).

[55] Xiao-Meng Zhang, Li Liang, Lin Liu, and Ming-Jing Tang. 2021. Graph neural networks and their current applications in bioinformatics. *Frontiers in genetics* 12 (2021), 690049.