UTCS: Effective Unsupervised Temporal Community Search with Pre-training of Temporal Dynamics and Subgraph Knowledge

Yue Zhang
The Chinese University of Hong
Kong, Shenzhen
Shenzhen, China
yuezhang5@link.cuhk.edu.cn

Yankai Chen
University of Illinois Chicago
Chicago, United States
yankaichen@acm.org

Yingli Zhou
The Chinese University of Hong
Kong, Shenzhen
Shenzhen, China
yinglizhou@link.cuhk.edu.cn

Yucan Guo
Institute of Computing Technology,
Chinese Academy of Sciences
Beijing, China
guoyucan23z@ict.ac.cn

Xiaolin Han
Northwestern Polytechnical
University
Xi'an, China
xiaolinh@nwpu.edu.cn

Chenhao Ma*
The Chinese University of Hong
Kong, Shenzhen
Shenzhen, China
machenhao@cuhk.edu.cn

Abstract

In many real-world applications, the evolving relationships between entities can be modeled as temporal graphs, where each edge has a timestamp representing the interaction time. As a fundamental problem in graph analysis, *community search (CS)* in temporal graphs has received growing attention but exhibits two major limitations: (1) Traditional methods typically require predefined subgraph structures, which are not always known in advance. (2) Learning-based methods struggle to capture temporal interaction information. To fill this research gap, in this paper, we propose an effective **U**nsupervised **T**emporal **C**ommunity **S**earch with pre-training of temporal dynamics and subgraph knowledge model (**UTCS**). UTCS contains two key stages: offline pre-training and online search. In the first stage, we introduce multiple learning objectives to facilitate the pre-training process in the unsupervised learning setting. In the second stage, we identify a candidate subgraph and compute community scores using the pre-trained node representations and a novel scoring mechanism to determine the final community members. Experiments on five real-world datasets demonstrate the effectiveness.

CCS Concepts

• **Computing methodologies** → **Neural networks**; • **Mathematics of computing** → *Graph algorithms*.

Keywords

Community Search, Temporal Graph Mining, Hawkes Process

*Corresponding author

1 Introduction

Graphs are widely utilized in various real-life fields [22, 23, 35, 36], such as social networks, biological graphs, and financial graphs. Community Search (CS), as an important graph analytical problem, has attracted significant attention recently [10]. Given a graph and a set of query nodes, the community search problem aims to find a cohesive and dense subgraph containing the query nodes [28]. CS is widely used in applications like social recommendation, research community detection, and fraud group identification, inspiring numerous specialized approaches. Existing methods can be broadly classified into two categories: traditional CS algorithms [1, 3, 9, 34, 39, 40] and learning-based CS models [11, 14, 17, 31, 32].

Traditional CS algorithms rely on predefined subgraph patterns, such as k-core [4, 7, 28], k-truss [1, 16], and k-clique [6, 34, 41]. These models impose strict topological constraints on the community, which may not always align with the structures of real-world communities. For instance, the k-core-based models assume that each node within the community has a degree of at least k, which may not always hold in reality. In contrast, learning-based CS models like QD-GNN [17] and TransZero [31] adapt flexibly to graph structures without predefined patterns: QD-GNN separately models queries and communities via GNNs, while TransZero uses Graph Transformers to capture node-community proximity.

Despite the success of learning-based CS studies, most are designed for static graphs and overlook crucial **temporal interaction information**. In real-world temporal graphs [5], such as business collaborations, two parties may establish relationships during specific periods. Identifying communities in such graphs is crucial for many real-world applications and has garnered growing research attention. While temporal graphs can be treated as static for input into static models, this often leads to sub-optimal results. For example, a recent learning-based work [14] considers the graph within each time window as static, failing to capture temporal interactions within the time window. Moreover, most learning-based methods heavily rely on costly labeled data. This highlights the urgent need for an **unsupervised** approach that captures temporal information for effective community search. To date, there is no
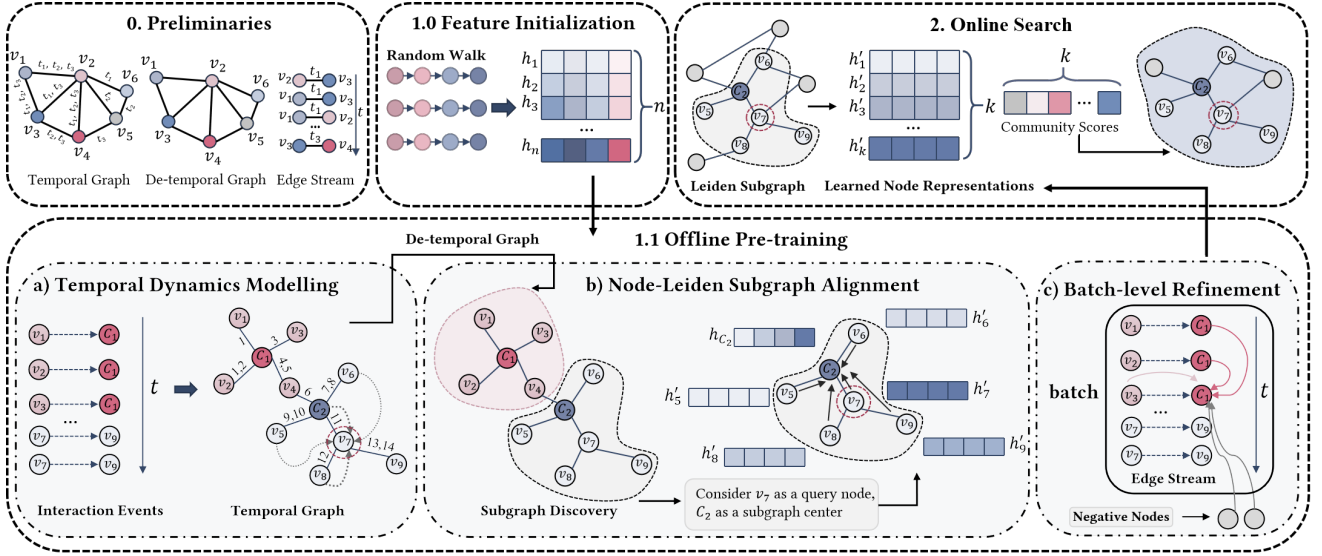
Figure 1: The architecture of our model.

existing learning-based approach for temporal community search in *the unsupervised setting*.

Our solution. To address the aforementioned limitations, we have developed an effective model, namely **U**nsupervised **T**emporal **C**ommunity **S**earch with pre-training of temporal dynamics and subgraph knowledge (**UTCS**). UTCS incorporates temporal and structural information at both local and global scales. As a learning-based method, it operates without requiring a predefined subgraph structure. Our model has two phases: offline pre-training and online search. During pre-training, we introduce the multiple learning objectives through three components: a temporal dynamics learner based on the Hawkes process to model interaction patterns, a subgraph alignment mechanism leveraging Student's t-distribution and KL divergence to capture structural dependencies, and a batch-wise embedding refinement to reconstruct adjacency relations. This phase enables unsupervised and label-free training. In the online search phase, we first locate candidate subgraphs, then leverage learned node representations to compute community scores and employ a dedicated matching mechanism to predict the final community structure. Our main contributions are as follows.

- We propose a learning-based model for the temporal CS problem and, to the best of our knowledge, this is the first unsupervised approach to tackle it.
- Our model follows a two-step framework with modules designed to capture temporal, local, and global structural information. Additionally, it employs a local search approach to significantly reduce time and space complexity.
- Extensive experiments on five real-world datasets demonstrate the effectiveness of the model, achieving an average improvement of 60.44% in F1-score compared to the latest competitors.

2 Problem Formulation

In this paper, we consider an undirected temporal graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ denote the sets of nodes and edges, respectively. Let $|\mathcal{V}| = n$ and $|\mathcal{E}| = m$. Each edge $e \in \mathcal{E}$ is a triplet (u, v, t) with $u, v \in \mathcal{V}$ and $t \in \mathbb{N}$ representing the interaction timestamp between

u and v. Note that (u, v, t_1) and (u, v, t_2) are considered as two edges when $t_1 \neq t_2$. We introduce the concept of temporal neighbors $N_{(u,t)}$, where $N_{(u,t)} = \{w | (u, w, t_w) \in \mathcal{E} \text{ and } t_w < t\}$. Figure 1 preliminaries illustrate a sample temporal graph $\mathcal{G}$ with 6 vertices and 17 temporal edges. In addition, we define the concept of the de-temporal graph, a version of the temporal graph $\mathcal{G}$ with timestamps ignored as $\mathcal{G}'(\mathcal{V}', \mathcal{E}')$, where $\mathcal{E}' = \{(u, v) | \exists (u, v, t) \in \mathcal{E}\}$ and $|\mathcal{E}'| = \tilde{m}$. Figure 1 preliminaries show a de-temporal graph $\mathcal{G}'$.

PROBLEM 1 (TEMPORAL COMMUNITY SEARCH (TCS) [14]). *Given a temporal graph $\mathcal{G}$ and a set of query nodes $Q \subseteq \mathcal{V}$, the TCS problem aims to return a subgraph (i.e., community) $\mathcal{T}_Q$ containing Q.*

In this paper, we focus on the learning-based approach to solve the TCS problem. For a set of query nodes $Q \subseteq \mathcal{V}$, and its corresponding community $\mathcal{T}_Q$, the nodes in $\mathcal{V}$ can be classified into two categories based on the indicator function f below, i.e., $\forall v \in \mathcal{V}$: $f(v) = 1$ if $v \in \mathcal{T}_Q$; otherwise, $f(v) = 0$. Specifically, our objective is to train a neural network $\mathcal{M}$ to identify communities in a temporal graph.

3 Our Methodology

3.1 Overview

In this section, we explain the details of UTCS. As depicted in Figure 1, UTCS consists of two phases: offline pre-training and online search.

3.2 Offline Pre-training

To better align with real-world communities, we address the TCS problem by training a neural network to model both temporal interaction information and structural relationships.

*3.2.1 **Temporal Dynamics Modeling.*** To establish a good starting point, we initialize node representations using node2vec [12]. To incorporate temporal information, we model the temporal graph as a chronological order of edge streams, as shown in Figure 1 preliminaries. Then, we resort to the Hawkes process [15, 42], where past interactions temporarily boost the likelihood of future events

with exponentially decaying influence. Inspired by [21, 42], we construct a temporal loss function based on the Hawkes process. This approach allows us to capture the influence of past events on current interactions, providing a comprehensive representation of temporal information. The temporal loss term can be formulated as:

$$L_{tmp} = -\log \sigma(\lambda_{(u,v,t)}) - \sum_{i=1}^{l} \mathbb{E}_{\bar{h}_i \sim P_n(v)} \log \sigma(-\lambda_{(u,\bar{h}_i,t)}), \quad (1)$$

where σ is the sigmoid activation function and $\lambda_{(u,v,t)}$ is the conditional intensity function of v joining source node u's neighborhood. We sample negative nodes $\bar{h}$ with probability proportional to their degree $P_n(v) \propto d_v^{3/4}$ [25], using a fixed sample size of l.

3.2.2 **Node-Leiden Subgraph Alignment**. To capture the relationship between nodes and communities, we apply the Leiden algorithm [29] to obtain a set of Leiden subgraphs $\{\mathcal{L}_i\}$, and refine the node representations by incorporating the community structure captured by these subgraphs. The probability $q_{(v,i,t)}$ of node v belonging to a Leiden subgraph $\mathcal{L}_i$ at time t is computed using Student's t-distribution [30] based on embedding proximity. We derive the target distribution $p_{(v,i,t)}$ by squaring and normalizing the assignment probabilities to reinforce high-confidence predictions. Finally, we introduce the KL divergence [18] loss term to align the real-time assignment distribution with the target distribution:

$$L_{node} = \sum_{\mathcal{L}_i} p_{(v,i,t)} \log p_{(v,i,t)} / q'_{(v,i,t)}. \quad (2)$$

Probability $q'_{(v,i,t)}$ is adjusted by updated node embeddings. This loss ensures proper node alignment with their respective subgraphs.

3.2.3 **Batch-Level Embedding Refinement**. Existing temporal graph methods often process edge streams in batches, overlooking structural connectivity. Inspired by [2, 38], we propose a batch-level refinement that reconstructs adjacency relationships via node embeddings. Specifically, similar to the computation of temporal conditional intensity in Eq. (1), we retrieve the historical neighbors $N_{(u,t)}$ as positive nodes and sample some non-neighbors $\overline{N}_{(u,t)} \sim P_n(u)$ as negative nodes. By maximizing the cosine similarity between positive pairs and minimizing it for negative pairs, we capture local structural information. The loss term is thus defined as follows:

$$L_u = -\log \frac{\exp\left(\cos(z_u^t, z_h^t)/T\right)}{\exp\left(\cos(z_u^t, z_h^t)/T\right) + \sum_{\bar{h} \in \overline{N}_{(u,t)}} \exp\left(\cos(z_u^t, z_{\bar{h}}^t)/T\right)}$$

$$L_{batch} = \frac{1}{|N_{(u,t)}| + 1} \sum_{h \in N_{(u,t)} \cup \{v\}} L_u, \quad (3)$$

where z_u^t and z_v^t are the temporal embeddings of source node u and target node v at current time t, respectively. $\cos(z_u^t, z_h^t)$ measures the cosine similarity between the anchor z_u^t and a positive sample z_h^t, with temperature $T = 0.5$ controlling the distribution sharpness.

By integrating the three modules, the complete community search loss function is formulated as: $L = L_{tmp} + L_{node} + L_{batch}$.

3.3 Online Search Phase

After offline pre-training, we move to an online search phase where nodes are embedded into a shared space, and proximity indicates

similarity. A candidate v's community score s_v reflects its likelihood of belonging to the target community based on its distance to the query nodes Q. The Expected Community Score Gain (ECSG) measures the gap between the scores of community nodes and the expected scores of random nodes [31]. To avoid costly global scoring, we adopt a local search strategy that reduces computation while maintaining comparable performance.

Algorithm 1: Online Search

Input: The query nodes Q, de-temporal graph $\mathcal{G}'$ and offline pre-training network $f(\cdot)$.
Output: The predicted community $\mathcal{T}_Q$.
1 Initialize $\mathcal{D} \leftarrow Q, \{C_i\} \leftarrow \text{Leiden}(\mathcal{G}')$;
2 **foreach** C_i *with* $C_i \cap Q \neq \emptyset$ **do**
3     $\{C_{i,j} | 1 \le j \le k\} \leftarrow \text{TopKSimC}(C_i)$;
4     $\mathcal{D} = \mathcal{D} \cup C_i \cup C_{i,1} \cup \cdots \cup C_{i,k}$
5 $z_Q \leftarrow \frac{\sum_{q \in Q} f(q)}{|Q|}$;
6 **for** $v \in \mathcal{D}$ **do** $s_v \leftarrow \frac{z_Q \cdot f(v)}{\|z_Q\| \cdot \|f(v)\|}$;
7 Initialize $\mathcal{T}_Q \leftarrow Q, max\_score \leftarrow -\infty$;
8 $\mathcal{S} = \{s_v | v \in \mathcal{D}\}$;
9 **while** $|\mathcal{T}_Q| < |\mathcal{D}|$ **do**
10     $u \leftarrow \arg\max_{v \in \mathcal{D} \setminus \mathcal{T}_Q} s_v$;
11   **if** $\text{ECSG}(\mathcal{S}, \mathcal{T}_Q \cup \{u\}, \mathcal{G}') > max\_score$ **then**
12         $max\_score \leftarrow \text{ECSG}(\mathcal{S}, \mathcal{T}_Q \cup \{u\}, \mathcal{G}')$;
13         $\mathcal{T}_Q = \mathcal{T}_Q \cup \{u\}$
14   **else** Break ;
15 **return** $\mathcal{T}_Q$;

The details are outlined in Algorithm 1. Given query nodes Q, the de-temporal graph $\mathcal{G}'$, and the pre-trained neural network, it outputs the predicted community $\mathcal{T}_Q$ containing Q. We first partition $\mathcal{G}'$ via the Leiden algorithm to identify subgraphs C_i containing Q (lines 1-2), retrieve k similar subgraphs with TopKSimC, and merge them into the search space $\mathcal{D}$ (lines 3-4). Next, the community scores are computed for nodes in $\mathcal{D}$ based on cosine similarity with z_Q (lines 5-6). Starting with $\mathcal{T}_Q$ as Q and max_score as negative infinity, we iteratively add the node with the highest score in $\mathcal{D} \setminus \mathcal{T}_Q$ that increases the ECSG (lines 7-13). The search concludes by returning the final community $\mathcal{T}_Q$.

4 Experiments
4.1 Setup

Datasets. We use five real-world temporal graphs across various

Table 1: Datasets statistics.

Datasets	$|\mathcal{V}|$	$|\mathcal{E}|$	$|\mathcal{E}'|$	t_{max}	#C
School [24]	327	188,508	5,802	7,375	9
Brain [26]	5,000	1,955,488	1,751,910	12	10
Patent [13]	12,214	41,916	41,915	891	6
arXivAI [33]	69,854	699,206	699,198	27	5
arXivCS [33]	169,343	1,166,243	1,166,237	29	40

domains [21], each with ground-truth communities, e.g., academic collaboration networks and biological networks. The datasets statistics are summarized in Table 1. $|\mathcal{V}|$ is the number of nodes, and $|\mathcal{E}|$ denotes the number of temporal edges, $|\mathcal{E}'|$ represents the number of edges in the de-temporal graph by ignoring the temporal information, and t_{max} denotes the number of different timestamps. #C denotes the number of communities for each graph. Our code is available at https://github.com/zyxxxx2/UTCS.

Baselines. Existing CS methods are categorized into traditional and learning-based. We compare two traditional methods, MPC [27] and QTCS [20]. For the learning-based approaches, we compared

Table 2: Temporal Community Search. (The best and second best results are marked in bold and underlined respectively; "OOM" denotes the cases out-of-memory; "-" denotes the result is not available; "< 0.01" denotes the value is less than 0.01).

| Algorithms	School			Brain			Patent			arXivAI			arXivCS			Rank
	Jaccard	F1-Score	NMI	Jaccard	F1-Score	NMI	Jaccard	F1-Score	NMI	Jaccard	F1-Score	NMI	Jaccard	F1-Score	NMI	
MPC	0.0676	0.1267	0.0602	-	-	-	0.0205	0.0402	0.0108	-	-	-	-	-	-	7/7/7
QTCS	0.7118	0.8316	0.6198	0.1590	0.2743	0.0529				< 0.01	< 0.01	< 0.01	< 0.01	< 0.01	< 0.01	5/5/4
QD-GNN	0.75 ± 0.31	0.80 ± 0.29	0.68 ± 0.34	0.20 ± 0.07	0.33 ± 0.08	0.06 ± 0.05	0.24 ± 0.01	0.38 ± 0.02	0.05 ± 0.01	OOM	OOM	OOM	OOM	OOM	OOM	3/3/3
COCLEP	0.90 ± 0.02	0.94 ± 0.01	0.87 ± 0.02	0.14 ± 0.04	0.24 ± 0.05	0.05 ± 0.04	0.19 ± 0.00	0.30 ± 0.00	0.07 ± 0.00	0.12 ± 0.00	0.19 ± 0.00	0.04 ± 0.00	0.06 ± 0.00	0.10 ± 0.00	0.04 ± 0.00	2/2/2
TransZero	0.14 ± 0.00	0.25 ± 0.00	0.01 ± 0.00	0.24 ± 0.00	0.38 ± 0.00	0.11 ± 0.00	0.14 ± 0.00	0.24 ± 0.00	< 0.01	0.09 ± 0.00	0.17 ± 0.00	< 0.01	0.02 ± 0.00	0.03 ± 0.00	< 0.01	4/4/5
CS-TGN	0.69 ± 0.01	0.78 ± 0.01	0.63 ± 0.01	0.12 ± 0.01	0.21 ± 0.01	0.04 ± 0.01	0.08 ± 0.02	0.14 ± 0.04	0.04 ± 0.01	OOM	OOM	OOM	OOM	OOM	OOM	6/6/6
UTCS	**0.93 ± 0.01**	**0.95 ± 0.01**	0.83 ± 0.00	**0.32 ± 0.00**	**0.49 ± 0.00**	**0.18 ± 0.00**	0.18 ± 0.01	0.31 ± 0.00	0.06 ± 0.00	**0.20 ± 0.00**	**0.33 ± 0.00**	**0.10 ± 0.00**	**0.12 ± 0.00**	**0.22 ± 0.00**	**0.07 ± 0.00**	1/1/1

three representative static CS methods and one temporal CS method, i.e., TransZero [31], COCLEP [19], QD-GNN [17] and CS-TGN [14].

Effectiveness Metrics and Implementation Details. In this paper, we mainly focus on the F1-score, Normalized Mutual Information (NMI) [8], and Jaccard similarity (JAC) [37], with higher values indicating better results. Models are trained for 200 epochs with embedding dimension 128, $T = 0.5$, historical neighbors $h = 3$, negative samples $\bar{h} = 3$, learning rate 0.01, $k = 2$, and batch size 1024. The final results are averaged over five runs. The baseline hyperparameters are set according to their respective original papers.

4.2 Detailed Analysis of UTCS

• **Effectiveness Evaluation.** We compare UTCS with baselines across five datasets by generating 100 random queries per dataset, averaging F1, JAC, and NMI over five runs. As shown in Table 2: (1) Traditional CS methods struggle with temporal communities. Specifically, MPC often returns empty results, and QTCS performs poorly on large datasets like arXivAI and arXivCS; (2) Learning-based methods generally outperform traditional ones, suggesting real-world communities differ from predefined structures; (3) Among learning-based models, UTCS consistently achieves the best results, improving F1-score by 60.44% over the second-best model, COCLEP.

• **Online Query Time.** Figure 2 compares the running time of all
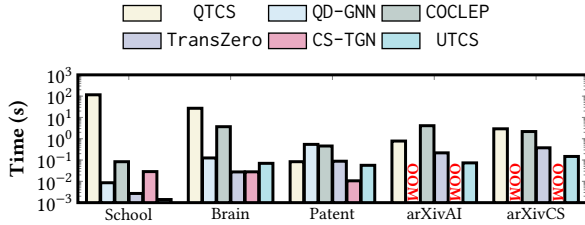
Figure 2: Efficiency results of CS methods.

methods in the search phase, with OOM denoting out-of-memory. Our model outperforms the baselines in terms of efficiency on most datasets, achieving over 100× speedup on the School dataset compared to QTCS due to our local search design. While CS-TGN is slightly more efficient than ours on the Brain and Patent datasets, it faces effectiveness issues and struggles to scale on large graphs.

• **Ablation Study.** In this experiment, we evaluate the effect of three key modules by F1-score: Temporal Dynamics Modeling (TM), Node-Leiden Subgraph Alignment (NA), and Batch-Level Embedding Refinement (BR). We develop three variants by removing these modules from UTCS, denoted as UTCS-NA, UTCS-TM, and UTCS-BR-TM. Table 3 shows the results: (1) The Temporal Dynamics Modeling module is crucial, as it captures temporal interaction information, which is vital for community search in temporal graphs. (2) The Node-Leiden Subgraph Alignment module is also important for effectively grouping similar nodes into the same community.

• **Case Study.** We demonstrate the effectiveness of our method

Table 3: Ablation study.

Models	School	Brain	Patent	arXivAI	arXivCS	+/-
UTCS-BR-TM	0.9347	0.4497	0.2754	0.1851	0.2133	-13.66%
UTCS-TM	0.9289	0.4530	0.2812	0.1789	0.2203	-13.01%
UTCS-NA	0.9282	0.4892	0.2971	0.3266	0.2156	-1.93%
UTCS	**0.9476**	**0.4894**	**0.3108**	**0.3285**	**0.2213**	-

(a) COCLEP	(b) QTCS	(c) UTCS	(d) Ground-truth
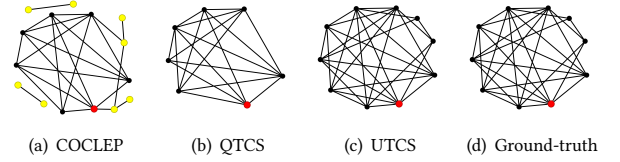
Figure 3: A case study on arXivAI dataset.

on the arXivAI dataset, where QD-GNN and CS-TGN encountered OOM issues. In Figure 3, the query node (red) is visualized alongside ground-truth community nodes (black) and non-community nodes (yellow). COCLEP misses several ground-truth nodes and includes many from other communities because it overlooks the temporal characteristics of the dataset. QTCS captures the temporal nature of the dataset but imposes strict constraints on temporal dimensions, missing nodes with weaker temporal associations. In contrast, our UTCS precisely identifies the ground-truth community.

5 Conclusion and Future Work

In this paper, we propose UTCS, the first unsupervised learning-based approach for the temporal CS problem. UTCS consists of two phases. In the pre-training phase, temporal information is captured via the Hawkes process, while structural relationships are learned through the node-Leiden subgraph alignment and batch-level refinement modules. In the online phase, some subgraphs are extracted to narrow the search scope, and communities are predicted based on a new scoring mechanism. Experiments on five public datasets validate the method's effectiveness.

Future work includes integrating traditional and learning-based methods to balance efficiency and accuracy, developing new metrics for unlabeled temporal graphs, and designing distributed algorithms to scale to massive temporal graphs.

References

[1] Esra Akbas and Peixiang Zhao. 2017. Truss-based community search: a truss-equivalence based indexing approach. *VLDB* 10, 11 (2017), 1298–1309.

[2] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations *(ICML '20)*. JMLR.org, Article 149, 11 pages.

[3] Yankai Chen, Yixiang Fang, Reynold Cheng, Yun Li, Xiaojun Chen, and Jie Zhang. 2018. Exploring communities in large profiled graphs. *IEEE Transactions on Knowledge and Data Engineering* 31, 8 (2018), 1624–1629.

[4] Yankai Chen, Jie Zhang, Yixiang Fang, Xin Cao, and Irwin King. 2021. Efficient community search over large directed graphs: An augmented index-based approach. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*. 3544–3550.

[5] Lingyang Chu, Yanyan Zhang, Yu Yang, Lanjun Wang, and Jian Pei. 2019. Online density bursting subgraph detection from temporal graphs. *Proceedings of the VLDB Endowment* 12, 13 (2019), 2353–2365.

[6] 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.

[7] Wanyun Cui, Yanghua Xiao, Haixun Wang, and Wei Wang. 2014. Local search of communities in large graphs. In *SIGMOD*. 991–1002.

[8] Leon Danon, Albert Diaz-Guilera, Jordi Duch, and Alex Arenas. 2005. Comparing community structure identification. *Journal of statistical mechanics: Theory and experiment* 2005, 09 (2005), P09008.

[9] Yixiang Fang, Reynold Cheng, Yankai Chen, Siqiang Luo, and Jiafeng Hu. 2017. Effective and efficient attributed community search. *The VLDB Journal* 26 (2017), 803–828.

[10] Yixiang Fang, Xin Huang, Lu Qin, Ying Zhang, Wenjie Zhang, Reynold Cheng, and Xuemin Lin. 2019. A survey of community search over big graphs. *The VLDB Journal* 29, 1 (July 2019), 353–392. https://doi.org/10.1007/s00778-019-00556-x

[11] 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 (Feb. 2021), 1006–1018. https://doi.org/10.14778/3447689.3447704

[12] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *SIGKDD*. 855–864.

[13] Bronwyn H Hall, Adam B Jaffe, and Manuel Trajtenberg. 2001. The NBER patent citation data file: Lessons, insights and methodological tools.

[14] Farnoosh Hashemi, Ali Behrouz, and Milad Rezaei Hajidehi. 2023. Cs-tgn: Community search via temporal graph neural networks. In *WebConf*. 1196–1203.

[15] Alan G Hawkes. 1971. Spectra of some self-exciting and mutually exciting point processes. *Biometrika* 58, 1 (1971), 83–90.

[16] Xin Huang, Laks VS Lakshmanan, Jeffrey Xu Yu, and Hong Cheng. 2015. Approximate closest community search in networks. *arXiv preprint arXiv:1505.05956* (2015).

[17] 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. arXiv:2104.03583 [cs]

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

[19] Ling Li, Siqiang Luo, Yuhai Zhao, Caihua Shan, Zhengkui Wang, and Lu Qin. 2023. Contrastive Learning-based Semi-Supervised Community Search. In *ICDE*.

[20] Longlong Lin, Pingpeng Yuan, Rong-Hua Li, Chunxue Zhu, Hongchao Qin, Hai Jin, and Tao Jia. 2024. QTCS: Efficient Query-Centered Temporal Community Search. *Proceedings of the VLDB Endowment* 17, 6 (2024), 1187–1199.

[21] Meng Liu, Yue Liu, Ke Liang, Wenxuan Tu, Siwei Wang, Sihang Zhou, and Xinwang Liu. 2024. DEEP TEMPORAL GRAPH CLUSTERING. (2024).

[22] Wensheng Luo, Yixiang Fang, Chunxu Lin, and Yingli Zhou. 2024. Efficient Parallel D-Core Decomposition at Scale. *Proceedings of the VLDB Endowment* 17, 10 (2024), 2654–2667.

[23] Wensheng Luo, Qiaoyuan Yang, Yixiang Fang, and Xu Zhou. 2023. Efficient core maintenance in large bipartite graphs. *Proceedings of the ACM on Management of Data* 1, 3 (2023), 1–26.

[24] Rossana Mastrandrea, Julie Fournet, and Alain Barrat. 2015. Contact patterns in a high school: a comparison between data collected using wearable sensors, contact diaries and friendship surveys. *PloS one* 10, 9 (2015), e0136497.

[25] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *NeurIPS* 26 (2013).

[26] Maria Giulia Preti, Thomas AW Bolton, and Dimitri Van De Ville. 2017. The dynamic functional connectome: State-of-the-art and perspectives. *Neuroimage* 160 (2017), 41–54.

[27] Hongchao Qin, Rong-Hua Li, Guoren Wang, Lu Qin, Yurong Cheng, and Ye Yuan. 2019. Mining periodic cliques in temporal networks. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 1130–1141.

[28] Mauro Sozio and Aristides Gionis. 2010. The community-search problem and how to plan a successful cocktail party. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. 939–948.

[29] Vincent A Traag, Ludo Waltman, and Nees Jan Van Eck. 2019. From Louvain to Leiden: guaranteeing well-connected communities. *Scientific reports* (2019).

[30] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, 11 (2008).

[31] Jianwei Wang, Kai Wang, Xuemin Lin, Wenjie Zhang, and Ying Zhang. 2024. Efficient Unsupervised Community Search with Pre-trained Graph Transformer. arXiv:2403.18869 [cs]

[32] Jianwei Wang, Kai Wang, Xuemin Lin, Wenjie Zhang, and Ying Zhang. 2024. Neural Attributed Community Search at Billion Scale. arXiv:2403.18874 [cs]

[33] Kuansan Wang, Zhihong Shen, Chiyuan Huang, Chieh-Han Wu, Yuxiao Dong, and Anshul Kanakia. 2020. Microsoft academic graph: When experts are not enough. *Quantitative Science Studies* 1, 1 (2020), 396–413.

[34] Long Yuan, Lu Qin, Wenjie Zhang, Lijun Chang, and Jianye Yang. 2017. Index-based densest clique percolation community search in networks. *TKDE* 30, 5 (2017), 922–935.

[35] Yuanyuan Zeng, Yixiang Fang, Chenhao Ma, Xu Zhou, and Kenli Li. 2024. Efficient distributed hop-constrained path enumeration on large-scale graphs. *Proceedings of the ACM on Management of Data* 2, 1 (2024), 1–25.

[36] Yuanyuan Zeng, Chenhao Ma, and Yixiang Fang. 2024. Distributed Shortest Distance Labeling on Large-Scale Graphs. *Proceedings of the VLDB Endowment* 17, 10 (2024), 2641–2653.

[37] Yao Zhang, Yun Xiong, Yun Ye, Tengfei Liu, Weiqiang Wang, Yangyong Zhu, and Philip S Yu. 2020. SEAL: Learning heuristics for community detection with generative adversarial networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1103–1113.

[38] Yifei Zhang, Hao Zhu, Zixing Song, Yankai Chen, Xinyu Fu, Ziqiao Meng, Piotr Koniusz, and Irwin King. 2024. Geometric view of soft decorrelation in self-supervised learning. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4338–4349.

[39] Yingli Zhou, Yixiang Fang, Wensheng Luo, and Yunming Ye. 2023. Influential community search over large heterogeneous information networks. *VLDB* 16, 8 (2023), 2047–2060.

[40] Yingli Zhou, Yixiang Fang, Chenhao Ma, Tianci Hou, and Xin Huang. 2024. Efficient Maximal Motif-Clique Enumeration over Large Heterogeneous Information Networks. *Proceedings of the VLDB Endowment* 17, 11 (2024), 2946–2959.

[41] Yingli Zhou, Qingshuo Guo, Yixiang Fang, and Chenhao Ma. 2024. A Counting-based Approach for Efficient k-Clique Densest Subgraph Discovery. *Proceedings of the ACM on Management of Data* 2, 3 (2024), 1–27.

[42] Yuan Zuo, Guannan Liu, Hao Lin, Jia Guo, Xiaoqian Hu, and Junjie Wu. 2018. Embedding Temporal Network via Neighborhood Formation. In *SIGKDD*. 2857–2866.