TADC-SBM: a Time-varying, Attributed, Degree-Corrected Stochastic Block Model

Nelson A. R. A. Passos
University of Pisa
National Research Council
Pisa, Italy
0000-0003-1869-2976

Emanuele Carlini
National Research Council
Pisa, Italy
0000-0003-3643-5404

Salvatore Trani
National Research Council
Pisa, Italy
0000-0001-6541-9409

***Abstract*—We present a synthetic dataset generator that produces temporal graphs with varying community structures, attribute features, and temporal dynamics, allowing for the evaluation of node clustering methods in a systematic manner. Temporal graphs offer a robust framework for modeling dynamic systems, with far-reaching applications in various domains where the analysis of evolving relationships between entities over time is required, such as transportation networks and recommendation systems. However, detecting communities in such graphs poses significant challenges, as the underlying community structure is subject to change over time and the presence of additional node or edge attributes introduces further complexity. Recent advances in graph neural networks have shown promise for "neural" community detection, but their expressiveness and generalization capabilities in attributed temporal graphs remain unclear, largely due to the scarcity of suitable real-world datasets for evaluation. In an experimental evaluation using TADC-SBM, we observe that novel approaches for node clustering can display good performance in scenarios with low community stability, but do not consistently outperform most baselines, highlighting potential research opportunities and underscoring the need for more generalizable models and robust benchmarks and datasets.**

***Index Terms*—Temporal Graphs, Community Detection, Stochastic Block Modeling, Graph Representation Learning.**

NOMENCLATURE

$G, \mathcal{G}$: Graph (static/temporal).
$V, \mathcal{V}$: Nodes set.
$E, \mathcal{E}$: Edges set.
$X_V, \mathcal{X}_\mathcal{V}$: Node features matrix.
$X_E, \mathcal{X}_\mathcal{E}$: Edge features matrix.
$\mathbf{A}$: Adjacency matrix.
$\mathbf{B}$: Block matrix.
$\mathbf{H}$: Embedding matrix.
$\boldsymbol{\tau}$: Transition matrix.
$\mathbf{z}$: Node membership vector.
δ: Kronecker delta.
$\oplus$: Perm.-invariant operator.
ϕ: Readout function.
ψ: Aggregation function.
θ: Node degree distribution.
α: Power law exponent.
β: Edge sampling probability.
η: Community stability rate.
γ: Fixed transition probabilities.
σ: Inter-cluster feature distance.
σ_c: Intra-cluster feature distance.
Θ: Feature covariance matrix.

I. INTRODUCTION

Communities in networks are often defined as mesoscale structures of comparatively similar entities, according to some ad hoc criterion. Detecting them is a fundamental problem in Network Science, with multiple techniques proposed to tackle it and far-reaching applications for varied tasks, such as analyzing social patterns, discovering functional biological modules, and forecasting traffic [5]. In temporal and attributed graphs, the problem of community detection entails additional challenges, along with research opportunities — yet it has accrued considerably less attention in the literature so far [19].

More recently, advances in the field of Artificial Intelligence have led to the proposal of new machine learning models for non-Euclidean data, such as manifolds and graphs [27]. In the latter case, nodes, edges, or (sub)graphs are mapped into a real-dimensional space — elements are represented as vectors (embeddings), while their relative proximity reflects some notion of similarity among them — thus enabling downstream tasks such as node clustering, link prediction, and graph classification. Their strength mainly lies in jointly leveraging a graph's topology, attributes, and temporal dynamics altogether, aspects often present in tandem in real-world networks [19].

Despite their state-of-the-art performance in various applications, the effectiveness of these models for community detection in temporal graphs remains largely unexplored [13]. In previous work, machine learning models for graphs were rather shown to underperform when compared to more established algorithms [1]. Moreover, real-world datasets used in their evaluation are often static, unattributed, or lack verifiable ground truths, therefore hindering a thorough assessment of their performance under controlled experimental settings [22].

In this work, we focus on the problem of benchmarking node clustering models for graphs where node features are present, edges are associated with a timestamp, and their structure evolves over time. Our contribution is threefold:

- We present a principled approach for generating attributed temporal graphs with community ground truths, building on previous work on stochastic block modeling [6], [25];
- We employ the TADC-SBM model here presented to generate synthetic graphs and evaluate how several existing approaches for community detection perform on them;
- We discuss our experimental results, offer possible insights, and release the code[1] used in our experiments to foster reproducibility and further research on the topic.

The remainder of this paper is organized as follows. Section II reviews related work on relevant topics. Section III formalizes the research problem and presents our methodology. Section IV discusses experimental results. Lastly, Section V concludes the paper and outlines future research directions.

[1]Available at the repository: https://github.com/nelsonaloysio/tadc-sbm.

II. Related Work

This section briefly outlines related work in community detection, learning and clustering temporal graphs, synthetic graph generation, and benchmarking graph learning models.

Community detection: Groups of similar nodes in a network — considering their density of connections, shared attributes, and/or other criteria — are often referred to as communities, clusters or modules [19]. Commonly used methods for their detection include spectral techniques [31]; optimization algorithms that maximize an objective function, e.g., modularity [29]; statistical inference, which estimates the likelihood of the data through probabilistic generative processes [24]; label and belief propagation [21]; and graph representation learning models [9]. In temporal graphs, where nodes, edges, and their associated attributes may change, the task becomes significantly more complex (see Figure 1), though models adapted for this context have shown improved performance and detectability thresholds [6].

Learning and clustering temporal graphs: The goal of graph representation learning models is to find a function that efficiently maps complex, high-dimensional graph elements into dense, d-dimensional vectors, i.e., $f : \mathcal{G} \to \mathbf{H} \in \mathbb{R}^d$ [9]. Since the introduction of SkipGram-based models [16], which encode node similarity based on random walk sampling, numerous graph representation learning models have been proposed, evolving from "shallow" to "deep" architectures able to capture more complex patterns. Among the latter, Graph Neural Networks (GNNs) are a class of state-of-the-art deep learning models that can obtain node representations by recursively aggregating their and their neighbors' features, usually employing a message-passing scheme [7], i.e., $h_i = \phi\left(x_i, \bigoplus_{j \in \mathcal{N}_i} \psi(x_i, x_j, x_e)\right)$, where ψ and ϕ are differentiable functions, $\oplus$ is a permutation invariant aggregation, and x_i, x_j, and x_e are a node's, its neighbors', and their edges' features, respectively. For temporal graph learning, GNNs are often used to learn representations from a sequence of graph snapshots or edge-level events, relying on recurrent architectures [26], attention mechanisms [34], temporal decay [12],and others. However, despite the recent advances in the field and its potential relevance in many domains, the task of neural community detection in temporal graphs remains a relatively less-studied area of research [13].

Synthetic graph generation: Among generative graph approaches, Stochastic Block Models (SBMs) are widely used to produce synthetic graphs with community ground truths. The simplest model receives two parameters as input: a block matrix $\mathbf{B} = B_{rs}$, that describes the edge probability between nodes in communities r and s, and a vector $\mathbf{z}$, where each element is the node's community assignment, drawn from a prior distribution q. The likelihood of the data is given by $P(\mathbf{A}|\mathbf{z}) = \prod_{ij} P(A_{ij}|z_i, z_j)$ and adjacencies by a binomial distribution, i.e., $A_{ij} \sim \text{Bernoulli}(B_{z_i z_j})$. By

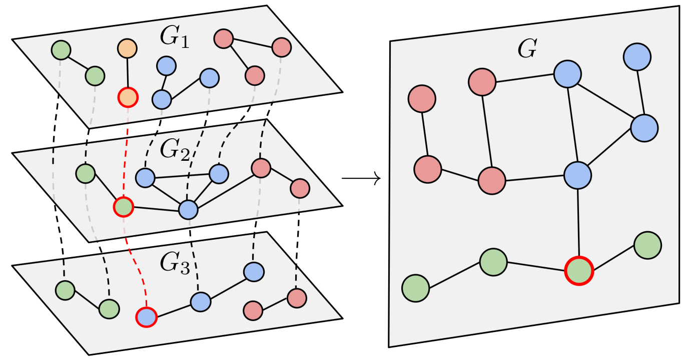

Fig. 1. Temporal graph snapshots (left) combined (right) to form a static graph, with communities obtained by modularity optimization [29]. Node colors represent their memberships and dashed lines indicate time-adjacent node copies. Notice how the orange and green communities are merged in G, while the highlighted node (in red) transitions communities on each snapshot.

employing optimization strategies such as Markov Chain Monte Carlo sampling, Expectation-Maximization, simulated annealing, or other variational inference methods, SBMs may also be employed for community detection, ultimately allowing for the faithful reconstruction of a graph by fitting the model that maximizes its likelihood and most accurately describes the observed data. Recent advances in the field have led to the introduction of more complex models that extend the original SBM framework, by parameterizing the node degree distribution [10]; accounting for nested (hierarchical) community structures [24] and contextual (node/edge) attributes [3]; or temporal dynamics [35].

Benchmarking models: Synthetic graphs generators are instrumental in evaluating and comparing models for community detection [11], e.g., based on stochastic blockmodeling [25]. Its capacity to produce datasets with varied community structures, both assortative and disassortative, and of varying degree distributions, presents a suitable choice to benchmark models for node clustering in static and temporal graphs within controlled experimental settings, in both transductive and inductive learning settings. In contrast, real-world temporal graph datasets for node classification are usually (i) narrowly themed, based on data from citation, political, or social communication networks; (ii) have static node memberships, unchanged over time; and/or (iii) ground truths that refer to predefined, domain-specific, or handcrafted categories, which do not necessarily correspond to observed activity patterns, nor nodes' attribute features [22]. These aspects severely limit their usefulness for model benchmarking purposes, highlighting the need for more suitable datasets and synthetic graph generators.

III. Methodology

In this work, we deal with the following research questions: (i) *how to evaluate graph learning models in attributed temporal graphs*; and (ii) *how well they perform in the task of community detection compared to other established methods.*

A temporal graph [13] is usually represented either as a sequence of snapshots, $\mathcal{G}_S := \{G_1, \ldots, G_t \,|\, t \in \mathbb{N}\}$, where

$G_t \coloneqq \{V, E, X_V, X_E\}$ is a static graph at time t, and V, E, X_V, and X_E are nodes, edges, and their features; or edge-level events, $\mathcal{G}_E \coloneqq \{\forall e \in \mathcal{E} : \{u, v, t, \delta, x_u, x_v, x_e\} \,|\, t \in \mathbb{R}^+\}$, where $\mathcal{E}$ are edges (interactions), u and v are nodes, $\delta \in \{0, 1\}$ represents an edge addition, removal, or interaction length, and x_u, x_v, and x_e are (optional) node- and edge-level features[2].

In our context, a model (algorithm) is used to learn (employ) a function $f : \mathcal{G} \to \mathcal{C}$, which maps nodes in a temporal graph $\mathcal{G}$ to a set of k clusters $\mathcal{C} = \{C_1, \ldots, C_k\}$. In the graph representation learning paradigm, two functions may be employed: first to encode nodes into a real $d-$dimensional space, i.e., $g : \mathcal{G} \to \mathbf{H} \in \mathbb{R}^d$, and then obtain their membership assignments from their latent representations, i.e., $h : \mathbf{H} \to \mathcal{C}$.

Lastly, the model's performance is evaluated based on the predicted clusters and the ground truths. Here, we focus on attributed temporal graphs with a fixed number of clusters and static attribute features, and do not consider overlapping (mixed-membership) communities, while the framework presented is flexible enough to support future extensions.

A. *Model description*

To enable the joint generation of temporal graphs with community ground truths and node- and edge-level features, we base ourselves on previously introduced generative models for degree-corrected temporal and attributed graphs [6], [25].

Graph generation: Initial node memberships $\mathbf{z} = \{z_0, z_1, ..., z_n\}$ are drawn from a prior distribution q, for n nodes and k clusters. Edges are generated from a power law distribution with exponent $\alpha > 0$ that controls the expected node degree distribution, with minimum and maximum degrees bounded by arbitrarily defined values d_{min} and d_{max}.

A square matrix $\mathbf{B}$ is used to generate a graph G_1, in which each element B_{rs} defines the expected number of edges between clusters r and s. The adjacency matrix is sampled from a binomial distribution, with the probability of an edge between nodes u and v given by $B_{z_u z_v}$. Therefore, both the expected degree d_v of each node v and their expected number of connections to other clusters $d_v^\star \leq d_v$ are considered during this process. At the end of this stage, the spectral detectability of each community c in the graph is given by $d_c - d_c^\star$ [18], and the power law's extremity by $d_{\text{max}} - d_{\text{min}}$.

Temporal dynamics: This process is repeated for a predefined number of snapshots $T \geq 1$. A $k \times k$ transition matrix $\boldsymbol{\tau}$ defines the probability τ_{rs} of a node changing communities from r to s at each snapshot $t > 1$, as depicted in Figure 2.

Two additional hyperparameters allow further control of this process. The first, $\gamma \in \{0, 1\}$, determines if node transition probabilities are either fixed or dynamic, i.e., based on their initial or current memberships, respectively — nodes have a fixed chance η of remaining in or returning to their original community if $\gamma = 1$, ensuring consistent transition probabilities over time; while $\gamma = 0$ may result in a harder task if

[2]Snapshot-based and event-based temporal graphs are also referred to in the literature as discrete-time and continuous-time temporal graphs [13].

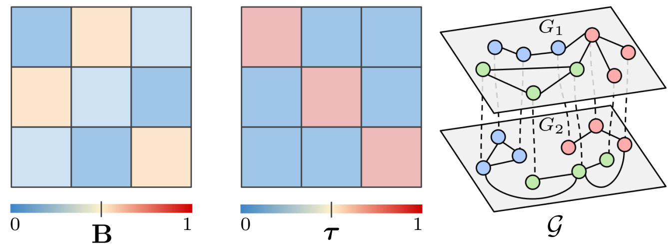

Fig. 2. Generation of a temporal graph $\mathcal{G}$ with $t = 2$ snapshots. The block matrix $\mathbf{B}$ is used to generate the adjacency matrix of each snapshot $G_t \in \mathcal{G}$, while the transition matrix $\boldsymbol{\tau}$ controls nodes changing communities over time.

$\eta > 1/k$, especially as the number of snapshots T increase. The second, $\beta \in [0, 1]$, determines the probability of observing the edges of each snapshot, with $\beta = 1$ preserving all sampled edges — when $\beta < 1$, nodes with lower degrees have a higher chance to vanish and resurge over time, as the graph becomes sparser and isolates are filtered out — therefore introducing additional stochasticity to the temporal dynamics of the graph.

During this process, edges are generated independently for each graph $G_t \in \mathcal{G}$. At this point, the detectability of dynamic communities depends on the rate and strength of their change [6]: if the probability of nodes remaining in their community is given by η and probability of them transitioning to another random community by $1 - \eta$, the threshold then interpolates from the static value when $\eta = 1$ to zero when $\eta = 0$. In the event of the latter, the predictive accuracy of a model is expected to approximate that of a random guess, i.e., $1/k$, where k is the number of clusters.

Feature attribution: For a number of clusters $\hat{k}$, zero-mean centroids are drawn from an s_i-multivariate normal distribution with covariance matrix $\mathbf{\Theta}_i = \sigma_i^2 \cdot \mathbf{I}$, where σ_i^2 is the variance of the i-th cluster and $\mathbf{I}$ is the identity matrix [25]. Node features $X_V \in \mathcal{G}$ are drawn from an s-multivariate normal distribution with mean $\overline{x}_v$ and covariance $\mathbf{\Theta} = \sigma^2 \cdot \mathbf{I}$, with the ratio σ_i^2/σ^2 controlling the expected intra-community and inter-community centroid distances, i.e., the within and between sum of squares of the feature clusters. Features may be generated with hierarchical (nested) group structures [24], and the number of clusters in the feature space may differ from the number of communities in the graph, i.e., $\hat{k} \neq k$.

Edge features $X_E \in \mathcal{G}$ may be optionally generated considering node communities. Within-community edge features are sampled from a zero-mean, unit-covariance, s_e-multivariate normal distribution, while between-community edge features are sampled with unit covariance and mean vector $\overline{X}_E = \{\forall e \in \mathcal{E} : \overline{x}_e\}$. Larger values of $\overline{x}_e$ increase the intra-community and inter-community edge features distances, and decrease the difficulty of recovering node ground truths [25].

Note that, in contrast with the dynamic community structure, node-level features are generated once at the beginning of the process and remain unchanged over time. This approach was found suitable for node clustering benchmarks, allowing models to exploit them to retrieve the ground truths, although it may be extended in future work to allow for evolving features.

TABLE I
Evaluated models, divided into three groups: general algorithms, shallow graph representation learning models, and GNN models for node clustering. Input: X_V features, G, $\mathcal{G}_S$, $\mathcal{G}_E$ graphs (static, snapshot- or event-based).

Model	Input	Topology	Features	Temporal
K-Means	X_V		✓	
Spectral Clustering	G	✓		
Leiden	G	✓		
Node2Vec	G	✓		
Attri2Vec	G	✓	✓	
DynNode2Vec	$\mathcal{G}_S$	✓		✓
tNodeEmbed	$\mathcal{G}_S$	✓		✓
DAEGC	G	✓	✓	
DMoN	G	✓	✓	
TGC	$\mathcal{G}_E$	✓	✓	✓

B. Evaluated models

We compared several approaches that take a graph's topology, temporal dynamics, and/or attribute features as input, as summarized in Table I. Our selection includes both established algorithms for community detection and more recent graph learning models for node clustering, and prioritizes solutions implemented in open source frameworks [2], [4], [23].

K-Means [14] is an algorithm that partitions elements into clusters based on their positions in the feature space. **Spectral Clustering** [20] is a clustering algorithm based on eigendecomposition of the graph Laplacian. **Leiden** [29] is an optimization algorithm based on the graph's topology, here employing modularity as a quality function. **Node2Vec** [8] is a graph representation learning model that maps node similarities based on their co-occurrences in random walks. **Attri2Vec** [36] is an extension of Node2Vec that includes node attributes to encode their similarity. **DynNode2Vec** [15] extends Node2Vec to temporal graphs by sampling a sequence of graph snapshots, with initial weights defined by the previously learned embeddings. **tNodeEmbed** [28] employs matrix approximations on consecutive snapshots to align node embeddings. **DAEGC** [33] is an attentional graph autoencoder that minimizes a joint reconstruction and clustering loss to learn node embeddings based on their neighborhoods. **DMoN** [30] is an end-to-end GNN that optimizes modularity with a collapse regularization term to avoid trivial solutions. **TGC** [12] is a graph autoencoder that employs a temporal decay-based (Hawkes) function to learn embeddings — which, at the moment of writing, is the only GNN-based model designed for temporal node clustering proposed since a recent survey [13].

C. Evaluation metrics

We assess the performance of the models based on three key metrics commonly employed in the literature [32]:

- **Accuracy**: we use the Kuhn-Munkres algorithm to solve label assignments and compute correctly predicted labels;
- **AMI**: Adjusted Mutual Information, based on the mutual information between true and predicted clusters.
- **ARI**: Adjusted Rand Index, based on the number of pairs of elements assigned to the same or distinct groups.

IV. Experimental Results

We employ the TADC-SBM model to generate a temporal undirected graph with $T = 8$ snapshots, $n = 1024$ nodes and $k = 8$ clusters, with similar parameterization to [6], [25].

Initial node memberships are drawn uniformly-at-random, i.e., $q = 1/k$. The degree vector $\mathbf{z}$ is obtained by setting average degree $d = 20$, average inter-community degree $d^\star = 2$, and power law parameters $\alpha = 2$, $d_{\text{min}} = 2$, $d_{\text{max}} = 20$. The expected number of edges per snapshot is $|E| = d \times n/2$ and the expected average degree is $\langle d \rangle = (d + (k-1)\, d^\star)/k$. The block matrix $\mathbf{B}$ is defined as $B_{rs} = d/n$ if $r = s$ and $d^\star/n$ otherwise, i.e., $p/q = (d - d^\star)/d^\star$. Node features $X_V = \{x_1, x_2, \ldots, x_n\}$ are generated from a multivariate normal distribution with $s = 32$ dimensions, $\hat{k} = k$ clusters, $\sigma^2 = 1$ intra-cluster variance, and $\sigma_c^2 = 6$ cluster variance.

We define the transition matrix by fixing the probability of a node remaining in its community to η, and changing its community uniformly-at-random with probability $1 - \eta$, i.e.,

$$\boldsymbol{\tau} := \eta\, \mathbf{I} + (1 - \eta)\, \frac{\mathbf{J} - \mathbf{I}}{k - 1},$$

where $\mathbf{I}$ is the identity matrix and $\mathbf{J}$ is a matrix of ones[3]. We vary the probability of nodes remaining in their community $\eta \in \{0, 0.25, 0.5, 0.75, 1\}$ and keep node transition and edge sampling hyperparameters both fixed at $\gamma = 0$ and $\beta = 1$, meaning that node transition probabilities are based on their current memberships and all sampled edges are observed. The temporal graph $\mathcal{G}_{\eta=1}$ therefore maintains the same community structure in all snapshots, while $\mathcal{G}_{\eta=0}$ displays completely random mesoscale temporal dynamics, with only node-level features providing information on the ground truths. Lastly, snapshots were reversed, $\mathcal{G} = \{G_t, \ldots, G_1\}$, to ensure that the last snapshot contained the node ground truths for prediction.

A. Discussion

This section summarizes our experimental results for the selected models in a transductive learning setting, where the whole graph was used for training and testing. The Leiden algorithm optimized modularity Q, with initial random node label assignments $C = \{\forall v \in \mathcal{V} : c(v) \leq k\}$. For SkipGram-based models, return and in-out parameters were set to 1, walk length to 80, and number of walks per node and window size to 10. Node embedding dimensionality was set to 128, and we performed hyperparameter tuning on the dropout rate and collapse regularization term for DMoN. Remaining hyperparameters were set to the values described in the papers, and we include the data and code in our repository for reproducibility.

Table II shows the performance of the evaluated models across all datasets. We report the mean and standard deviation of the metrics on the best epochs over 5 runs. K-Means is used as a baseline for the separability of communities considering node-level features, while Spectral Clustering serves as a baseline for their detectability regarding the graph's topology.

[3] Note that adding $(1-\eta)/k$ to the value of η yields the same probabilities as in [6], where uniform-at-random transition probabilities include the node's current community, i.e., $\tau_{rs} = \eta + (\eta/k)$ if $r = s$ and $(1-\eta)/k$ otherwise.

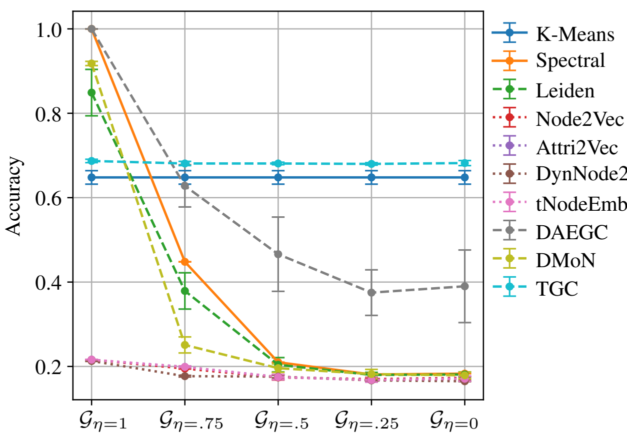

Fig. 3. Model accuracy averaged over 5 runs on synthetic graphs.

In general, model performance was highly dependent on the community stability level of the graphs, rapidly decreasing with the probability η of nodes remaining in their communities — as the task increased in difficulty due to the number of snapshots ($T = 8$) and dynamic node transition probabilities ($\gamma = 0$). In contrast, employing spectral decomposition sufficed to correctly retrieve the ground truths only in the graph generated with static communities, as evidenced in Fig. 3.

Spectral clustering and DAEGC were the best-performing models on $\mathcal{G}_{\eta=1}$, followed by DMoN and Leiden optimizing modularity ($Q = 0.43$). Performance employing spectral clustering degraded much faster than DAEGC as η approached zero, which can be attributed to the former obtaining clusters from the eigenvectors of the graph Laplacian, while the latter leverages both the graph's topology and node-level features.

DMoN, which optimizes spectral modularity and also considers node-level features, was the second best-performing model on average on $G_{\eta=1}$. As most other models, it did not maintain performance on graphs generated with $\eta = 0.75$, figuring on par with other methods in graphs with $\eta \leq 0.5$.

TGC, the only GNN-based model designed for node-level clustering in temporal graphs, was the only model that did not show performance degradation on $\eta < 1$ graphs. This result may be attributed to the model's design and the fact that the last snapshot contained the community ground truths, benefiting the temporal decay-based loss it optimizes. However, it was outperformed by Spectral, Leiden, and GNN-based methods on $\mathcal{G}_{\eta=1}$, and closely followed by K-Means in the other graphs. Further experimentation, including a suitable hyperparameter tuning strategy and examining its loss landscape, is warranted to properly assess its expressiveness.

Conversely, all evaluated random walk-based models, such as Node2Vec, performed poorly across datasets, in spite of leveraging the graph's topology in conjunction with node-level features (Attri2Vec) or temporal dynamics (DynNode2Vec, tNodeEmbed). As communities were not well-separated over time, it may be that the underlying community structure was not captured during sampling, although a different parameterization or extending the models to account for both the graph's temporal dynamics and node features could yield better results.

TABLE II

Model performance in a transductive setting. We mark the best results for each dataset in bold and italic. Generated graphs have varying community transition probabilities, given by $1 - \eta$, for every node in each snapshot.

Dataset	Model	Accuracy	AMI	ARI
$\mathcal{G}_{\eta=1}$	K-Means	.648 ± .016	.400 ± .015	.375 ± .018
	Spectral	**1.000 ± .000**	**1.000 ± .000**	**1.000 ± .000**
	Leiden	.849 ± .055	*.945 ± .022*	*.848 ± .048*
	Node2Vec	.216 ± .000	.066 ± .000	.041 ± .000
	Attri2Vec	.216 ± .000	.066 ± .000	.041 ± .000
	DynNode2Vec	.213 ± .001	.060 ± .002	.037 ± .001
	tNodeEmbed	.216 ± .000	.066 ± .000	.041 ± .000
	DAEGC	**1.000 ± .000**	**1.000 ± .000**	**1.000 ± .000**
	DMoN	*.918 ± .005*	.813 ± .011	.815 ± .011
	TGC	.687 ± .004	.438 ± .005	.421 ± .005
$\mathcal{G}_{\eta=.75}$	K-Means	*.648 ± .016*	*.400 ± .015*	*.375 ± .018*
	Spectral	.448 ± .000	.152 ± .000	.135 ± .000
	Leiden	.379 ± .043	.132 ± .016	.115 ± .017
	Node2Vec	.195 ± .001	.023 ± .001	.014 ± .000
	Attri2Vec	.199 ± .002	.026 ± .001	.017 ± .000
	DynNode2Vec	.177 ± .002	.012 ± .002	.006 ± .001
	tNodeEmbed	.199 ± .002	.026 ± .001	.017 ± .000
	DAEGC	.628 ± .050	.356 ± .040	.337 ± .055
	DMoN	.251 ± .019	.051 ± .007	.062 ± .007
	TGC	**.681 ± .005**	**.434 ± .006**	**.415 ± .007**
$\mathcal{G}_{\eta=.5}$	K-Means	*.648 ± .016*	*.400 ± .015*	*.375 ± .018*
	Spectral	.210 ± .000	.025 ± .000	.018 ± .000
	Leiden	.204 ± .017	.019 ± .008	.012 ± .005
	Node2Vec	.174 ± .006	.007 ± .004	.004 ± .002
	Attri2Vec	.175 ± .006	.005 ± .004	.003 ± .002
	DynNode2Vec	.176 ± .003	.005 ± .001	.003 ± .000
	tNodeEmbed	.175 ± .006	.005 ± .004	.003 ± .002
	DAEGC	.466 ± .088	.218 ± .050	.180 ± .058
	DMoN	.196 ± .010	.014 ± .004	.026 ± .003
	TGC	**.681 ± .003**	**.432 ± .005**	**.415 ± .005**
$\mathcal{G}_{\eta=.25}$	K-Means	*.648 ± .016*	*.400 ± .015*	*.375 ± .018*
	Spectral	.181 ± .000	.006 ± .000	.004 ± .000
	Leiden	.180 ± .006	.010 ± .002	.006 ± .002
	Node2Vec	.170 ± .004	.002 ± .001	.001 ± .001
	Attri2Vec	.168 ± .005	.002 ± .002	.001 ± .001
	DynNode2Vec	.167 ± .001	.006 ± .001	.003 ± .000
	tNodeEmbed	.168 ± .005	.002 ± .002	.001 ± .001
	DAEGC	.375 ± .054	.151 ± .017	.110 ± .025
	DMoN	.182 ± .011	.007 ± .004	.018 ± .004
	TGC	**.680 ± .003**	**.431 ± .004**	**.414 ± .004**
$\mathcal{G}_{\eta=0}$	K-Means	*.648 ± .016*	*.400 ± .015*	*.375 ± .018*
	Spectral	.183 ± .000	.017 ± .000	.010 ± .000
	Leiden	.181 ± .006	.008 ± .004	.005 ± .002
	Node2Vec	.173 ± .003	.004 ± .001	.002 ± .001
	Attri2Vec	.171 ± .006	.004 ± .004	.002 ± .002
	DynNode2Vec	.165 ± .001	.005 ± .000	.002 ± .000
	tNodeEmbed	.171 ± .006	.004 ± .004	.002 ± .002
	DAEGC	.390 ± .086	.173 ± .054	.128 ± .060
	DMoN	.179 ± .008	.004 ± .002	.016 ± .002
	TGC	**.682 ± .006**	**.433 ± .009**	**.416 ± .009**

In summary, as η approached zero, most models performed only slightly better than random guessing in the task of community detection, with ground truths set to the last snapshot. Future benchmarking efforts may investigate if model performance degrade as fast as observed for temporal graphs generated with different parameter values. Lastly, we highlight that in case of the Leiden algorithm, optimizing a temporal, e.g., multislice modularity [17] could result in better partitioning, as well as other techniques, such as asymptotically optimal spectral decomposition for dynamic community detection [6].

V. Conclusion

In this work, we have presented a framework for generating synthetic attributed temporal graphs with community ground truths, and compared distinct existing solutions for node-level clustering in this context. Most evaluated solutions displayed decreasing performance as community-level changes were introduced to the graph in a topological level, while node attribute features remained static. In particular, the temporal GNN-based model evaluated did not show performance degradation despite the mesoscale perturbations introduced by the temporal dynamics, and benchmarking it in other simulated scenarios, such as varying the number of snapshots, node features, and communities, may allow to further assess its expressiveness in different scenarios in a systematic way.

We note that our experiment considered only undirected graphs, with a relatively small number of nodes and clusters, exclusively in a transductive learning setting. As model performance may vary significantly depending on the dataset and the choice of hyperparameters, the same experimental setup may likely yield distinct results in an inductive learning setting, possibly allowing to evaluate how well neural community detection models generalize. Finally, the TADC-SBM model may be extended to generate directed, weighted, and multigraphs, as well as dynamic node features and overlapping (mixed-membership) communities, accounting for a wider range of simulated scenarios, which we leave for future work.

Acknowledgments

We thank the members of the Inverse Complexity Lab at the Interdisciplinary Transformation University Austria, Linz, and the Department of Network and Data Science at the Central European University, Vienna, especially Prof. Tiago Peixoto, for their guidance and support in the initial phase of this study, during a fruitful visiting research period in the spring of 2024.

References

[1] Nelson A. R. A. Passos, Emanuele Carlini, and Salvatore Trani. Deep community detection in attributed temporal graphs: Experimental evaluation of current approaches. In *Proceedings of the 3rd GNNet Workshop on Graph Neural Networking Workshop*, GNNet '24, pages 1–6, New York, NY, USA, 2024. Association for Computing Machinery.

[2] CSIRO's Data61. Stellargraph machine learning library, 2018.

[3] Yash Deshpande, Andrea Montanari, Elchanan Mossel, and Subhabrata Sen. Contextual stochastic block models, 2018.

[4] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[5] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5):75–174, 2010.

[6] Amir Ghasemian, Pan Zhang, Aaron Clauset, Cristopher Moore, and Leto Peel. Detectability thresholds and optimal algorithms for community structure in dynamic networks. *Phys.Rev.X*, 6(3), 2016.

[7] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 1263–1272. JMLR.org, 2017.

[8] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *22nd ACM SIGKDD*. ACM, 2016.

[9] William L. Hamilton. *Graph Representation Learning*, volume 14. Morgan and Claypool, 2017.

[10] Brian Karrer and M. E. J. Newman. Stochastic blockmodels and community structure in networks. *Phys.Rev.E*, 83(1), 2011.

[11] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E*, 78(4), October 2008.

[12] Meng Liu, Yue Liu, Ke Liang, Wenxuan Tu, Siwei Wang, Sihang Zhou, and Xinwang Liu. Deep temporal graph clustering. In *The 12th International Conference on Learning Representations*, 2024.

[13] Antonio Longa, Veronica Lachi, Gabriele Santin, Monica Bianchini, Bruno Lepri, Pietro Lio, Franco Scarselli, and Andrea Passerini. Graph neural networks for temporal graphs: State of the art, open challenges, and opportunities. *Transactions on Machine Learning Research*, 2023.

[14] JB MacQueen. Some methods for classification and analysis of multivariate observations, proceedings of 5-th berkeley symposium on mathematical statistics and probability. *Berkeley, University of California Press*, 1:281–297, 1967.

[15] Sedigheh Mahdavi, Shima Khoshraftar, and Aijun An. dynnode2vec: Scalable dynamic network embedding. In *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, 2018.

[16] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013.

[17] Peter J. Mucha, Thomas Richardson, Kevin Macon, Mason A. Porter, and Jukka-Pekka Onnela. Community structure in time-dependent, multiscale, and multiplex networks. *Science*, 328(5980):876–878, 2010.

[18] Raj Rao Nadakuditi and M. E. J. Newman. Graph spectra and the detectability of community structure in networks. *Physical Review Letters*, 108(18), 2012.

[19] Mark Newman. *Networks*. Oxford University Press, 2 edition, 2018.

[20] Andrew Ng, Michael Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001.

[21] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufman Publishers, 1988.

[22] Leto Peel, Daniel B. Larremore, and Aaron Clauset. The ground truth about metadata and community detection in networks. *Science Advances*, 3(5), 2017.

[23] Tiago P. Peixoto. The graph-tool python library. *figshare*, 2014.

[24] Tiago P. Peixoto. Hierarchical block structures and high-resolution model selection in large networks. *Physical Review X*, 4(1), 2014.

[25] Anton Tsitsulin; Benedek A Rozemberczki; John Palowitch; Bryan Perozzi. Synthetic graph generation to benchmark graph learning. In *Workshop on Graph Learning Benchmarks*, 2021.

[26] Luana Ruiz, Fernando Gama, and Alejandro Ribeiro. Gated graph recurrent neural networks. *IEEE Transactions on Signal Processing*, 68:6303–6318, 2020.

[27] F. Scarselli, M. Gori, Ah Chung Tsoi, M. Hagenbuchner, and G. Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

[28] Uriel Singer, Ido Guy, and Kira Radinsky. Node embedding over temporal graphs. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI'19, pages 4605–4612. AAAI Press, 2019.

[29] V. A. Traag, L. Waltman, and N. J. van Eck. From louvain to leiden: guaranteeing well-connected community. *Scientific Reports*, 9(1):5233, 2019.

[30] Anton Tsitsulin, John Palowitch, Bryan Perozzi, and Emmanuel Müller. Graph clustering with graph neural networks. *J. Mach. Learn. Res.*, 24(1), 2024.

[31] Deepak Verma and Marina Meilă. A comparison of spectral clustering algorithms, 2003.

[32] Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09. ACM, 2009.

[33] Chun Wang, Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, and Chengqi Zhang. Attributed graph clustering: A deep attentional embedding approach. In *Twenty-Eighth International Joint Conference on Artificial Intelligence*, 2019.

[34] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs, 2020.

[35] Kevin S. Xu and Alfred O. Hero. Dynamic stochastic blockmodels for time-evolving social networks. *IEEE Journal of Selected Topics in Signal Processing*, 8(4):552–562, 2014.

[36] Daokun Zhang, Jie Yin, Xingquan Zhu, and Chengqi Zhang. Attributed network embedding via subspace discovery. *Data Mining and Knowledge Discovery*, 33(6):1953–1980, 2019.