# T3former: Temporal Graph Classification with Topological Machine Learning

**Md Joshem Uddin**[1], **Soham Changani**[1], **Baris Coskunuzer**[1]

[1]Department of Mathematical Sciences, The University of Texas at Dallas, Richardson, TX, USA
{mdjoshem.uddin, soham.changani, coskunuz}@utdallas.edu

## Abstract

Temporal graph classification plays a critical role in applications such as cybersecurity, brain connectivity analysis, social dynamics, and traffic monitoring. Despite its significance, this problem remains underexplored compared to temporal link prediction or node forecasting. Existing methods often rely on snapshot-based or recurrent architectures that either lose fine-grained temporal information or struggle with long-range dependencies. Moreover, local message-passing approaches suffer from oversmoothing and oversquashing, limiting their ability to capture complex temporal structures.

We introduce T3FORMER, a novel Topological Temporal Transformer that leverages sliding-window topological and spectral descriptors as first-class tokens, integrated via a specialized Descriptor-Attention mechanism. This design preserves temporal fidelity, enhances robustness, and enables principled cross-modal fusion without rigid discretization. T3former achieves state-of-the-art performance across multiple benchmarks, including dynamic social networks, brain functional connectivity datasets, and traffic networks. It also offers theoretical guarantees of stability under temporal and structural perturbations. Our results highlight the power of combining topological and spectral insights for advancing the frontier of temporal graph learning.

**Code** — https://github.com/joshem163/T3Former

**Extended version** — https://arxiv.org/pdf/2510.13789

## 1 Introduction

Temporal graph classification, the task of assigning a label to a single temporal graph whose nodes and edges carry timestamp information indicating their active intervals, plays a crucial role in domains such as cybersecurity intrusion detection, dynamic functional-connectivity mapping in neuroscience, social network analysis, and traffic pattern recognition. Current approaches predominantly rely on discrete snapshot methods or recurrent updates (e.g., EVOLVEGCN (Trivedi et al. 2019), DYSAT (Zhao et al. 2020)), inherently trading off between temporal granularity and the global structural context. Local message-passing networks further suffer from critical limitations like *oversmoothing* and *oversquashing*, restricting their capacity to identify long-range and higher-order temporal patterns (Alon and Yahav 2021). Moreover, the learned embeddings of these methods are frequently unstable under small perturbations, a notable disadvantage in inherently noisy and sparse settings such as neuroscience (Hajij et al. 2021).

In contrast, static graph classification techniques have recently made significant progress by leveraging global topological and spectral descriptors, such as persistent homology (Immonen, Souza, and Garg 2024; Hiraoka et al. 2024) and Laplacian density-of-states (DOS) curves (Dong, Benson, and Bindel 2019), providing robust and expressive graph-level summaries. However, extending these methods directly to temporal settings typically demands rigid snapshotting, compromising temporal resolution.

To bridge this gap, we introduce T3FORMER, a novel *Topological Temporal Transformer* specifically designed for temporal graph classification. T3FORMER effectively integrates topological signatures and Laplacian DOS vectors through sliding-windows as first-class descriptor tokens alongside global structural tokens generated by lightweight per-window graph neural networks. These tokens, enriched with relative-time embeddings, are effectively fused through a unified transformer backbone with dedicated *Descriptor-Attention* modules, explicitly capturing intricate interactions among structural, topological, and spectral modalities.

The *Descriptor-Attention* framework offers significant advantages: (1) it maintains fine-grained temporal resolution without ad-hoc snapshot discretization; (2) it enhances robustness through topological invariants; (3) it leverages mesoscopic structural insights provided by spectral descriptors; and (4) it ensures principled cross-modal integration via transformer-based attention rather than simple concatenation. Empirically, we demonstrate the superior performance of T3FORMER across standard benchmarks spanning social networks, dynamic brain connectivity graphs, and traffic networks, establishing new state-of-the-art results. Our extensive experiments further confirm the method's robustness, interpretability, and resilience to perturbations.

In summary, our contributions include:

- T3FORMER, a novel temporal graph classification model that integrates sliding-window topological and spectral descriptors via dedicated cross-attention modules.
- *Descriptor-Attention*, a transformer-based fusion mechanism that unifies structural, topological, and spectral information.

- Theoretical stability guarantees showing bounded sensitivity of topological and spectral descriptors to minor graph perturbations.
- We introduce new temporal graph classification datasets adapted from real-world traffic networks.
- Extensive experiments on social, brain connectivity, and traffic network benchmarks, demonstrating state-of-the-art performance, robustness, and broad applicability.

# 2 Background

## 2.1 Related Work

**Temporal Graph Neural Networks.** Temporal graph neural networks (TGNNs) and graph transformers have emerged as a popular framework for modeling dynamic graphs, where nodes, edges, and features evolve over time. A common strategy in TGNNs is to partition the temporal graph into discrete snapshots, apply graph neural networks (GNNs) to obtain node or graph representations for each snapshot, and then model the temporal evolution of these representations using sequence models such as recurrent neural networks (RNNs) or transformers. Representative examples include EVOLVEGCN (Pareja et al. 2020), which adapts GCN parameters through recurrent updates, and DYGFORMER (Yu et al. 2023), which models dynamic events through temporal point processes. More recent methods such as TGN (Rossi et al. 2020) leverage memory modules to capture fine-grained temporal dependencies between node interactions. While effective for dynamic prediction tasks such as link prediction or event forecasting, these methods often rely heavily on local neighborhood aggregation and sequential modeling of embeddings, making them sensitive to temporal noise and structural perturbations.

**Topological Methods for Temporal Graph Learning.** An alternative to sequence modeling over learned embeddings is to focus on improving the snapshot representations themselves using topological methods. Topological models utilizing persistent homology have recently become a strong alternative to GNNs, especially in graph-level learning (Loiseaux et al. 2024; Chen, Frias, and Gel 2024; Verma, Souza, and Garg 2024) as they have proven highly effective for capturing higher-order structural patterns in static graphs. In the context of dynamic or temporal graphs, recent works such as TAMP-S2GCNETS (Chen et al. 2022), GRAPHPULSE (Shamsi et al. 2024), and Dynamic Dowker Filtrations (Ye et al. 2023) demonstrate that topological summaries can serve as powerful snapshot encoders, significantly boosting performance and often outperforming traditional TGNN architectures. Topological methods capture long-range dependencies and temporal dynamics by modeling global and higher-order patterns, without relying solely on sequential updates.

**Prediction-Focused Temporal Learning.** Despite these advances, most temporal learning methods remain focused on prediction tasks such as dynamic link prediction, node classification, or next-event forecasting. Models like JODIE (Kumar, Zhang, and Leskovec 2019), DYREP (Trivedi et al. 2019), TGN (Rossi et al. 2020), and APAN (Wang and Zhang 2021) primarily target node- or edge-level predictions over time. Recent benchmarking efforts such as the Temporal Graph Benchmark (TGB) (Huang et al. 2023) also emphasize predictive tasks across diverse domains, highlighting the relative scarcity of methods designed specifically for graph-level classification in temporal settings. Furthermore, sequence-based models often exhibit sensitivity to timestamp irregularities and noise, which can degrade the learned representations when applied to tasks requiring robust global graph understanding.

**Temporal Graph Classification.** While much of the research on temporal graph learning has centered around *temporal graph property prediction* tasks, the problem of *temporal graph classification* remains relatively underexplored (Kim et al. 2022). Only a few recent efforts have extended TGNNs to temporal graph classification (Yang, Zhao, and Shen 2025; Cai et al. 2021), and many existing approaches struggle to effectively leverage fine-grained temporal information, particularly when the dynamic evolution cannot be easily discretized into snapshots (Ekle and Eberle 2024). Recognizing this gap, (Tieu et al. 2024) recently introduced a suite of benchmark datasets and baseline methods to promote further study of temporal graph classification. Nevertheless, most current models continue to rely heavily on rigid discretization schemes or sequence-based assumptions, limiting their ability to capture continuous structural evolution. Our work addresses these limitations by proposing a hybrid framework that integrates global structural modeling with timestamp-aware topological and spectral summaries, enabling effective classification of evolving graphs without requiring fixed snapshot sequences.

## 2.2 Temporal Graph Learning

Many real-world systems are inherently dynamic, where the graph structure evolves over time through the addition or removal of nodes and edges. To capture such temporal behavior, graphs can be extended to the temporal domain using either continuous-time or discrete-time representations.

Let a static undirected graph be defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of edges. Each node $v \in \mathcal{V}$ is associated with a feature vector $\mathbf{x}_v \in \mathbb{R}^d$, and each edge $(u, v) \in \mathcal{E}$ has a corresponding feature $\mathbf{e}_{uv}$. The neighborhood of a node $v$ is denoted by $\mathcal{N}(v)$.

A *temporal graph* models evolving graphs as a sequence of timestamped interactions:
$\mathcal{G} = \{(u, v, \mathbf{x}_u(t), \mathbf{x}_v(t), \mathbf{e}_{uv}(t), t)\}_{t=t_0}$, where each event indicates that nodes $u$ and $v$ interacted at time $t$, and $\mathbf{x}_u(t)$, $\mathbf{x}_v(t)$, and $\mathbf{e}_{uv}(t)$ are the corresponding node and edge features at that time.

## 2.3 Persistent Homology

Persistent Homology (PH) is a central tool in Topological Data Analysis (TDA) that captures multiscale topological features such as connected components, loops, and voids that persist across different scales (Dey and Wang 2022). Originally developed for point cloud data, PH has since been successfully extended to graphs, images, and other data modalities (Coskunuzer and Akçora 2024).

PH typically consists of three main steps: constructing a filtration, obtaining persistence diagrams, and performing vectorization. The process begins with *filtration*, where a nested sequence of simplicial complexes is built to track the evolution of topological features. Given an unweighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a filtration function $f : \mathcal{V} \to \mathbb{R}$ with thresholds $\mathcal{I} = \{\alpha_i\}$, we construct subgraphs $\mathcal{G}^i = (\mathcal{V}_i, \mathcal{E}_i)$, where $\mathcal{V}_i = \{v \in \mathcal{V} \mid f(v) \leq \alpha_i\}$ (See Figure 1 for toy example). The filtration function can be based on graph structural properties (e.g., degree, betweenness centrality) or derived from domain-specific information (e.g., atomic number in molecular graphs). For weighted graphs, edge weights can also serve as a natural filtration (Coskunuzer and Akçora 2024). The primary goal is to establish a hierarchy among nodes or edges that yields a meaningful sequence of subgraphs aligned with the downstream task. In temporal graphs, a natural choice would be timestamps on edges as a valuable filtration function.

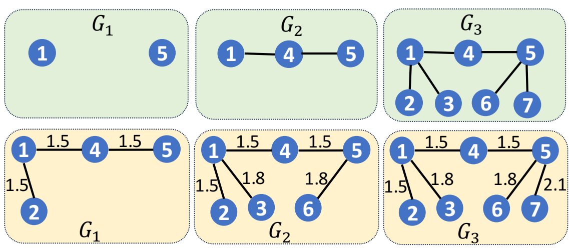


Figure 1: Graph Filtration: For $\mathcal{G} = \mathcal{G}_3$ in both examples, the top figure illustrates a *superlevel filtration using the node degree function* with thresholds $3 > 2 > 1$, where nodes of degree 3 are activated first, followed by those of lower degrees. Similarly, the bottom figure illustrates a *sublevel filtration based on edge weights* with thresholds $1.5 < 1.8 < 2.1$.

Once the subgraphs are defined, each $\mathcal{G}^i$ is extended to its clique complex $\widehat{\mathcal{G}}^i$, producing a sequence $\widehat{\mathcal{G}}^1 \subseteq \widehat{\mathcal{G}}^2 \subseteq \cdots \subseteq \widehat{\mathcal{G}}^N$. As the filtration evolves, topological features such as connected components (0-holes), loops (1-holes), and cavities (2-holes) emerge and disappear. Each $k$-dimensional feature $\sigma$ is associated with a birth index $b_\sigma$ and death index $d_\sigma$ ($1 \leq b_\sigma < d_\sigma \leq N$), recorded as a point $(b_\sigma, d_\sigma)$ in the $k$-th persistence diagram $\mathrm{PD}_k(\mathcal{G}) = \{(b_\sigma, d_\sigma) \mid \sigma \in H_k(\widehat{\mathcal{G}}^i)\}$. Since persistence diagrams are sets of points in $\mathbb{R}^2$, a *vectorization* step is necessary for compatibility with machine learning models. Techniques such as persistence images, landscapes, silhouettes, and Betti curves (Ali et al. 2023) are commonly used to transform diagrams into fixed-size feature representations.

While PH effectively captures complex structural patterns, its integration into temporal graph learning remains challenging. Most prior methods are limited to static graphs and fail to model evolving topological features. Additionally, combining PH summaries with temporal dynamics and node attributes is nontrivial. To address these issues, we propose a hybrid framework that extracts topological descriptors from temporal subgraphs, aligning them with sliding windows to track structural changes over time. This enables scalable, interpretable temporal graph classification without relying on rigid snapshots or heavy discretization.

### 2.4 Density of States for Graphs

The *density of states* (DoS) is a classical concept from spectral graph theory that captures the distribution of eigenvalues of a graph Laplacian (Chung 1997). It provides a coarse-grained summary of the graph's structural complexity, bottlenecks, and connectivity properties.

Given an undirected graph $G = (\mathcal{V}, \mathcal{E})$, let $L$ denote its normalized Laplacian matrix. The eigenvalues of $L$, denoted by $\{\lambda_i\}_{i=1}^{|\mathcal{V}|}$, lie within the interval $[0, 2]$. The DoS is then defined as the empirical distribution of these eigenvalues:

$$\mathrm{DoS}(\lambda) = \frac{1}{|\mathcal{V}|} \sum_{i=1}^{|\mathcal{V}|} \delta(\lambda - \lambda_i)$$

where $\delta(\cdot)$ denotes the Dirac delta function.

In practice, the DoS is approximated by constructing a histogram over a fixed number of bins partitioning $[0, 2]$ (Dong, Benson, and Bindel 2019). This yields a compact and interpretable vector representation that captures key spectral characteristics of the graph. The resulting descriptor is permutation-invariant, robust to small perturbations in the graph structure, and encodes information about fundamental graph properties such as connectivity, expansion, and community structure (Spielman 2011). When applied to temporal graphs, the DoS computed over sliding subgraphs provides a dynamic signature of the evolving spectral complexity (Huang et al. 2024), complementing topological features.

## 3 Methodology

We propose a hybrid framework that integrates global structural modeling with timestamp-aware topological and spectral descriptors for temporal graph classification. Our approach is composed of four key components: temporal filtration via sliding windows, topological and spectral descriptor extraction, static graph construction with temporal node features, and a unified classification architecture that combines multiple embeddings through self-attention.

### 3.1 Temporal Filtration via Sliding Window

A fundamental obstacle in temporal graph classification is that interactions arrive as continuously timestamped events rather than as a predetermined sequence of discrete snapshots. To address this, we introduce a *temporal filtration* framework inspired by persistent homology, in which the timestamp annotation itself induces a filtration over the underlying static graph.

Formally, let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \tau)$ denote a temporal graph with node set $\mathcal{V}$, edge set $\mathcal{E}$, and the relation $\tau : \mathcal{E} \to \mathbb{R}^+$ assigning each edge its timestamp (or multiset of timestamps for repeated interactions). Given a window length $\delta > 0$, we define a family of subgraphs $G_{[t,\, t+\delta]} = (\mathcal{V}_t, \mathcal{E}_t)$, where $\mathcal{E}_t = \{e \in \mathcal{E} \mid \tau(e) \in [t, t+\delta]\}$ and $\mathcal{V}_t = \{u \in \mathcal{V} \mid \exists\, e \in \mathcal{E}_t \text{ with } u \in e\}$ (See Figure 4 for toy example). To control the granularity of the sliding windows, we also use a stride parameter $\sigma \in (0, \delta)$ that defines the step size of the windowing process, resulting in the filtered sequence $\mathcal{G}_i = \mathcal{G}_{[i\sigma,\, i\sigma+\delta]}$ (See Appendix B.5).

This sliding-window construction yields a pseudo-filtration in time, where each edge may naturally appear in multiple windows if it carries multiple timestamps. Crucially, our approach avoids collapsing an edge to a single timestamp (e.g. min, max, or average), which can obscure temporal patterns. By avoiding coarse snapshotting, temporal filtration preserves high-resolution dynamics and yields a structured subgraph sequence that captures both local transitions and global evolution.

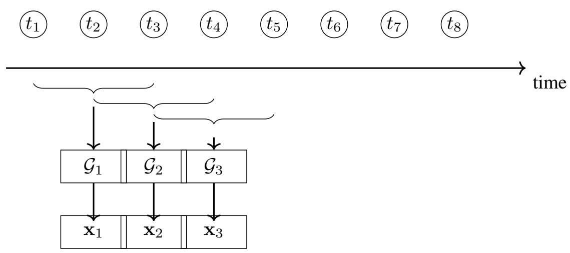


Figure 2: Sliding window construction: DoS and Betti vectors $\{\mathcal{X}_i\}$) are extracted for each induced subgraph $\mathcal{G}_t = \mathcal{G}_{[t,t+\delta]}$.

## 3.2 T3Former

We present T3Former, a *Topological Temporal Transformer* framework that effectively incorporates topological and spectral signatures of the temporal graph into a unified graph transformer architecture.

**Topological Descriptor Vectors.** For each subgraph $\mathcal{G}_{[t,t+\delta]}$, we compute a *topological descriptor vector*: $\phi_t = \left[|\mathcal{V}_t|, |\mathcal{E}_t|, \beta_0(\widehat{G}_{[t,t+\delta]}), \beta_1(\widehat{G}_{[t,t+\delta]})\right]$, where $\widehat{G}_{[t,t+\delta]}$ denotes the clique complex of $\mathcal{G}_{[t,t+\delta]}$, and $\beta_0$, $\beta_1$ denote the zeroth and first Betti numbers, respectively. These features capture key aspects of connectivity and cyclic structure over time, providing interpretable and noise-robust summaries of the evolving graph topology.

**Spectral Descriptor Vectors.** In parallel, we capture spectral properties of each subgraph using its *density of states* (DOS) vector. Given the normalized Laplacian $L_t$ of $\mathcal{G}_{[t,t+\delta]}$, the DOS is approximated by a histogram over the eigenvalues: $\psi_t = \text{Histogram}(\text{Eigenvalues}(L_t))$.

The DOS provides a complementary, geometry-sensitive representation of the graph, encoding notions such as graph complexity, bottlenecks, and expansion properties.

**Global Structural Modeling via Graph Neural Network.** While the topological and spectral descriptors extracted through the sliding window model fine-grained temporal dynamics, we also capture global structural patterns by applying a Graph neural network GSAGE (Hamilton et al. 2017) to the full static graph $\mathcal{G}$ (ignoring timestamps). To incorporate temporal activity, we assign each node a binary feature vector of length $|\tau|$, *temporal degree* (see Appendix B.1), where each entry in a node's feature vector is set to 1 if the node is involved in at least one edge at that time step, and 0 otherwise. This encoding enables the static graph to reflect temporal behavior while preserving its overall structure. Given initial node features (e.g., temporal degree, clustering coefficient, or learned embeddings), GSAGE generates node embeddings, aggregated via global pooling into a holistic graph-level representation.

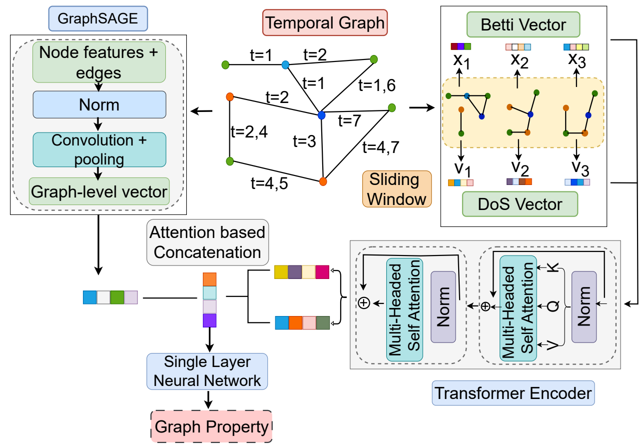


Figure 3: T3Former Flowchart: Given a temporal graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \tau)$, GraphSAGE generates time-aware structural embeddings. In parallel, a sliding-window module extracts topological (Betti) and spectral (density of states) vectors. These sequential features are processed through a Transformer to capture temporal dependencies. An attention-based fusion module merges the multi-view representations, which are then passed through a final neural layer for graph property prediction.

**Integration and Classification.** Topological ($\phi_t$) and spectral ($\psi_t$) descriptors are processed through a Transformer encoder, yielding two streams of temporal embeddings that capture localized structural dynamics. These are combined with global embeddings from GSAGE. A self-attention mechanism adaptively fuses the three views, and the resulting embedding is passed to a linear layer for final graph classification.

The T3Former architecture (Figure 3) uses GraphSAGE to encode time-agnostic global structure, applies a sliding-window filtration to extract timestamp-aware topological and spectral features, and fuses these views via attention and a transformer encoder, capturing both high-level global structures and fine-grained temporal dynamics in a scalable framework for temporal graph classification.

## 3.3 Stability of Topological and Spectral Descriptors

For temporal graph classification, robustness to small perturbations in the graph structure is critical, particularly in real-world settings where data may be noisy, incomplete, or prone to timestamp inaccuracies. In T3Former, we utilize topological and spectral descriptors that are inherently stable: small changes to the graph, such as edge insertions or deletions for given time periods induce only bounded changes in the extracted features. This stability ensures that the learned representations are resilient to noise and minor fluctuations over time, which is essential for reliable temporal graph classification.

We formalize these stability properties below. First, we prove the stability of our topological descriptors. Notice that our theorem is more general, and it applies to any sliding window $\mathcal{G}_{[t,t+\delta]}$.

**Theorem 3.1** (Stability of Topological Descriptors)**.** *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph and let $\tau_1, \tau_2 : \mathcal{E} \to \mathbb{R}$ be two timestamp functions on $\mathcal{G}$. Then, for $k \geq 0$, we have*

$$\|\vec{\beta}_k(\mathcal{G}, \tau_1) - \vec{\beta}_k(\mathcal{G}, \tau_2)\|_1 \leq C_k \cdot \|\tau_1 - \tau_2\|_1$$

*where $\vec{\beta}_k(\mathcal{G}, \tau_i)$ represents the Betti vector corresponding to $\mathrm{PD_k}(\mathcal{G}, \tau_i)$ obtained by sublevel filtration with respect to $\tau_i$.*

Next, we show the stability of our spectral descriptors.

**Theorem 3.2** (Stability of Spectral Descriptors)**.** *Let $\mathcal{G}$ and $\mathcal{G}'$ be two temporal graphs differing by at most $k$ edge modifications (insertions or deletions) within any window $[t, t+\delta]$. Then, the Wasserstein distance between the spectral descriptors $\psi_t$ and $\psi'_t$ is bounded by $Ck/n$, where $n = |\mathcal{V}|$ and constant $C$ depends on the eigenvalue distribution.*

The proofs of the theorems are given in Appendix A.

# 4 Experiments

## 4.1 Experimental Setup

**Datasets.** For the temporal graph classification experiments, we employed benchmarks across three domains: social, brain, and traffic networks. The social-network datasets consist of five temporal graphs from the TU-Dataset collection, `Infectious`, `DBLP`, `Tumblr`, `MIT`, and `Highschool` (Morris et al. 2020). The brain-network benchmarks are drawn from the NeuroGraph collection and include `DynHCP-Task`, `DynHCP-Gender`, and `DynHCP-Age` (Said et al. 2023). The traffic benchmarks are derived from the PEMS collection, `PEMS04`, `PEMS08`, and `PEMSBAY`, (Guo et al. 2019). These traffic datasets were originally designed for temporal regression tasks; to enable classification, we reformulated them as binary and three-class temporal graph classification problems. Our adaptation extends their utility to new learning settings. Key dataset characteristics are summarized in Table 1, with additional details on temporal structure and task definitions provided in Appendix B.1.

**Task.** We focus on temporal graph classification, where each graph $\mathcal{G}$ has timestamped edges capturing structural evolution over time. The label $y$ is fixed and time-invariant. The goal is to predict $y$ by leveraging temporal connectivity patterns and relevant domain-specific signals.

**Model Configuration.** We select the window length $\delta = 6$ and stride $\sigma = 4$ based on validation (Appendix B.5), resulting in $N = \lceil \frac{t_{\max} - t_{\min} - \delta}{\sigma} \rceil + 1$ windows per temporal graph. From each window, we extract the following information:

- *Topological tokens:* Betti-0 and Betti-1 computed from the clique complex of each window.

| Dataset | Graphs | Class | Avg.Node | Avg.Edge* | timesteps |
|---|---|---|---|---|---|
| Infectious | 200 | 2 | 50.00 | 459.72 | 48 |
| DBLP | 755 | 2 | 52.87 | 99.78 | 46 |
| Tumblr | 373 | 2 | 53.11 | 71.63 | 89 |
| MIT | 97 | 2 | 20 | 1469.15 | 5576 |
| Highschool | 180 | 2 | 52.32 | 544.81 | 203 |
| DynHCP-Task | 7443 | 7 | 100 | 843.04 | 34 |
| DynHCP-Gender | 1080 | 2 | 100 | 874.88 | 34 |
| DynHCP-Age | 1067 | 3 | 100 | 875.42 | 34 |
| PEMS04 | 708 | 2/3 | 307 | 680 | 24 |
| PEMS08 | 744 | 2/3 | 170 | 548 | 24 |
| PEMSBAY | 2172 | 2/3 | 325 | 2694 | 24 |

**Avg. Edges counts edge occurrences per snapshot (with repeats)*

Table 1: Temporal Graph Classification datasets.

- *Spectral tokens:* Degree of Spectrality (DoS) histogram with 4 bins over the normalized Laplacian spectrum.
- *Global encoder:* A 2-layer GSAGE encoder (Hamilton et al. 2017) is applied to the static graph, using either *temporal degree* features or domain-specific node features.

Topological and spectral tokens are processed through separate Transformer encoders, while the static graph with node features is encoded using GSAGE. To better exploit temporal information in node features, we replace the standard one-hot degree vectors with *temporal degree vectors* as initial embeddings (Appendix B.2). Each of the three branches produces a 10-dimensional temporal graph representation. These are concatenated into a 30-dimensional representation, which is then passed through a self-attention layer to adaptively fuse the information. The final attended representation is fed into a linear layer for classification.

**Training Details.** We train our model using the Adam optimizer with a cross-entropy loss function and a weight decay of $1e{-}4$. Hyperparameter tuning is performed via grid search over the following ranges: learning rate $\{0.01, 0.005, 0.001\}$, dropout rate $\{0.0, 0.3, 0.5\}$, and hidden dimensions $\{16, 32, 64, 128\}$. For the social and traffic networks, we follow a 5-fold cross-validation protocol consistent with the experimental setup in (Tieu et al. 2024). For the brain network dataset, we adopt a 70/10/20 train/validation/test split using the same random seed as the baseline paper (Said et al. 2023). Our implementation is written in Python and experiments are conducted on a server equipped with an NVIDIA H100 NVL GPU and 768 GB of RAM.

**Computational Complexity and Runtime.** Let $n = |\mathcal{V}|$, $m = |\mathcal{E}|$, and $N = \left\lceil \frac{t_{\max} - t_{\min} - \delta}{\sigma} \right\rceil + 1$ be the number of sliding windows. Temporal filtration takes $O(m)$ per window (total $O(Nm)$), topological descriptors cost $C_{\text{topo}}(n, m)$ per window (worst-case $O(n^3)$, typically $O(m^\omega)$ for sparse graphs where $\omega < 2.4$ (Otter et al. 2017; Milosavljevic et al. 2011)), and spectral DOS extraction with $k$ Lanczos/Chebyshev steps requires $O(km)$ per window (Weisse et al. 2006). Running GSAGE with $L$ layers and hidden dimension $d$ on the static graph costs $O(Lmd)$, and fusing the

| | Method | Infectious | DBLP | Tumblr | MIT | Highschool |
|---|---|---|---|---|---|---|
| | Shortest Path (Borgwardt et al. 2005) | 67.00±7.50 | 56.00±4.90 | 58.00±14.30 | 50.80±2.90 | 56.00±8.00 |
| **Kernels** | Random Walk (Vishwanathan et al. 2010) | 67.00±7.30 | 53.00±5.80 | 58.00±11.20 | 56.80±14.20 | 53.74±5.40 |
| | WL-Subtree (Shervashidze et al. 2011) | 60.00±4.40 | 52.00±6.80 | 57.00±12.10 | 55.50±11.40 | 55.20±8.20 |
| | GSAGE (Hamilton et al. 2017) | 58.00±4.11 | 56.29±0.66 | 59.78±4.95 | 72.00±9.91 | 62.78±7.51 |
| **Static GNNs** | Graph2Vec (Narayanan et al. 2017) | 56.50±8.10 | 53.90±3.10 | 54.70±7.10 | 52.20±10.10 | 57.60±7.00 |
| | NetLSD (Tsitsulin et al. 2018) | 62.50±6.10 | 55.80±3.50 | 55.20±4.60 | 60.10±12.40 | 55.20±5.90 |
| | GL2Vec (Chen and Koga 2019) | 54.50±5.10 | 56.20±3.00 | 55.80±8.00 | 55.20±4.80 | 57.60±4.10 |
| | TGN (Rossi et al. 2020) | 52.00±1.90 | 58.00±0.30 | 51.70±2.50 | 60.23±5.37 | 63.33±6.83 |
| **Temporal GNNs** | EvolveGCN (Pareja et al. 2020) | 52.10±9.30 | 40.00±8.90 | 39.50±8.90 | 40.00±9.45 | 41.73±9.32 |
| | GraphMixer (Cong et al. 2023) | 50.00±0.00 | 56.30±1.10 | 50.90±5.00 | 56.28±3.33 | 57.61±7.88 |
| | Temp-G$^3$NTK (Tieu et al. 2024) | **74.00±5.80** | 60.00±6.30 | 63.00±6.80 | 58.40±11.50 | 58.10±4.20 |
| **Ours** | T3Former | 68.50±6.30 | **60.90±0.70** | **63.20±3.20** | **73.16±4.13** | **67.20±3.20** |

Table 2: Social Networks: Temporal graph classification accuracy results on social network datasets, following the Temporal Graph Benchmark (Tieu et al. 2024).

| Method | PEMS04 | | PEMS08 | | PEMSBAY | |
|---|---|---|---|---|---|---|
| | Binary | Multi | Binary | Multi | Binary | Multi |
| GSAGE | 85.31 ± 3.48 | 80.79 ± 4.37 | 93.41 ± 1.11 | 85.62 ± 1.99 | 84.61 ± 1.74 | 71.40 ± 6.10 |
| GAT | 87.42 ± 6.32 | 84.59 ± 5.20 | 92.61 ± 2.37 | 86.69 ± 3.78 | 93.39 ± 0.43 | 88.30 ± 3.93 |
| UniMP | 86.73 ± 5.05 | 80.93 ± 4.38 | 93.28 ± 1.95 | 85.49 ± 3.00 | 90.60 ± 8.04 | 73.51 ± 8.08 |
| Graphormer | 93.64 ± 1.76 | 90.26 ± 5.48 | 93.82 ± 3.09 | 87.71 ± 2.89 | 94.08 ± 0.49 | 89.93 ± 1.49 |
| TGN | 86.30 ± 4.11 | 80.09 ± 3.25 | 90.73 ± 2.66 | 83.87 ± 3.19 | 95.44 ± 1.15 | 88.85 ± 1.30 |
| GCN+LSTM | 91.18 ± 3.26 | 86.02 ± 1.61 | 90.05 ± 1.86 | 84.81 ± 2.77 | 94.86 ± 2.21 | 85.47 ± 2.52 |
| TGAT | 90.85 ± 7.18 | 86.02 ± 3.18 | 82.77 ± 9.49 | 87.23 ± 3.21 | 94.08 ± 2.64 | 88.75 ± 4.66 |
| T3Former | **96.76 ± 1.92** | **92.66 ± 1.93** | **95.16 ± 1.50** | **89.65 ± 2.02** | **96.68 ± 1.30** | **92.35 ± 1.52** |

Table 3: Traffic Networks: Temporal graph classification accuracy results on traffic network datasets, evaluated under both binary and 3-class label settings.

| Method | Task | Gender | Age |
|---|---|---|---|
| GSAGE (Hamilton et al. 2017) | **90.93** | 66.20 | 40.65 |
| GAT (Veličković et al. 2018) | 89.67 | 67.13 | 44.39 |
| k–GNN (Morris et al. 2019) | 73.03 | 68.45 | 44.25 |
| UniMP (Shi et al. 2020) | 89.66 | 72.30 | 44.41 |
| Gen-GNN (You et al. 2020) | 68.84 | 62.04 | 42.99 |
| T3former | 90.76 | **75.79** | **58.73** |

Table 4: Brain Networks: Temporal graph classification accuracy results on brain networks, following NeuroGraph benchmark (Said et al. 2023).

DoS and Betti sequences of length $N$ via self-attention incurs $O(N^2d + Nd^2)$. Hence, the overall complexity is

$$\mathcal{O}\big(N\,(m+C_{\text{topo}}(n,m)+km) + Lmd + (N^2d+Nd^2)\big),$$

which remains practical for moderate $n$, $m$, and $N$.

Our model demonstrates a clear computational advantage by efficiently integrating temporal information without the need for separate static GNN processing followed by sequential learning. This streamlined design results in significantly reduced runtime. On the large-scale `Task` dataset, a complete run of our model takes approximately 98 minutes. More notably, for a single fold, our model requires only 5.34 minutes, whereas the GCN+LSTM baseline takes 84.42 minutes under the same configuration, making our approach roughly 15 times faster. The detailed runtime comparison is presented in Appendix B.1, highlighting the efficiency and scalability of our method.

### 4.2 Results

**Baselines.** We evaluate T3FORMER across three domains: social, brain, and traffic networks. For **social networks**, we adopt the Temporal Graph Benchmark (Tieu et al. 2024). Comparison models include (i) *Graph kernel methods* such as WL-SUBTREE, SHORTEST-PATH, and RANDOM-WALK, which respectively capture neighborhood structures, path-length distributions, and walk-based similarities; (ii) *Static GNNs* including GSAGE (inductive message passing), GRAPH2VEC (skip-gram on rooted subgraphs), NETLSD (spectral heat trace descriptors), and GL2VEC (graphlet-based features); and (iii) *Temporal GNNs* such as GRAPHMIXER, TGN, EVOLVEGCN, and TEMP-G$^3$NTK, which incorporate temporal dynamics through attention, memory modules, or time-aware kernels. For **brain networks**, we follow the NeuroGraph benchmark (Said et al. 2023), com-

| | Social Networks | | | | | Brain Networks | | | Traffic Networks | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | **Infect.** | **DBLP** | **Tumblr** | **MIT** | **HSchool** | **Task** | **Gender** | **Age** | **P04-2** | **P04-3** | **P08-2** | **PS08-3** | **PBAY-2** | **PBAY-3** |
| GSage | 58.00 | 56.29 | 59.78 | 72.00 | 62.78 | **91.57** | 63.83 | 37.55 | 85.31 | 80.79 | 93.41 | 85.62 | 84.61 | 71.40 |
| Topo+Tr | 63.50 | 58.60 | 61.70 | 61.90 | 63.90 | 22.10 | 58.80 | 45.64 | 85.03 | 69.63 | 80.11 | 71.51 | 94.54 | 81.22 |
| DoS+Tr | 65.00 | 59.80 | 61.90 | 70.10 | 65.00 | 23.31 | 57.87 | 45.83 | 82.49 | 67.66 | 78.09 | 68.55 | 89.46 | 76.61 |
| Concat-Fuse | 63.50 | 58.94 | 63.00 | 72.16 | 65.56 | 90.58 | 73.58 | 57.78 | 95.90 | 92.52 | 95.03 | **89.92** | **96.82** | 90.47 |
| T3Former | **68.50** | **60.90** | **63.20** | **73.16** | **67.20** | 90.76 | **75.79** | **58.73** | **96.76** | **92.66** | **95.16** | 89.65 | 96.68 | **92.35** |

Table 5: Ablation Study: Comparison of the standalone performance of each component in our architecture. Concat-fuse represents concatenation of these features instead of attention-based integration.

paring against UNIMP, K-GNN, GAT, GSAGE, and GENGNN. Relevant references are provided in Tables 2 and 4. In **traffic networks**, we additionally evaluate against the transformer-based GRAPHORMER (Ying et al. 2021), as well as temporal baselines including TGN (Rossi et al. 2020), GCN+LSTM (Yu et al. 2018), and TGAT (Xu et al. 2020).

**Social Network Results.** Table 2 compares T3FORMER against classical graph kernels, static GNNs, and recent temporal GNNs on five temporal social graphs. Graph kernels and static GNNs achieve only moderate accuracy and exhibit wide variations across datasets, while temporal GNNs improve on some networks but remain uneven in others. In contrast, T3FORMER delivers consistently strong performance on every benchmark, outperforming most baselines on each dataset and narrowing the gap to the very best method where it does not lead outright. This uniform gain highlights the strength of our hybrid design: by fusing timestamp-aware topological signatures with spectral descriptors through self-attention, T3FORMER robustly captures both local dynamics and global structure in evolving graphs.

**Traffic Network Results.** The temporal graph classification accuracy across three benchmark datasets, `PEMS04`, `PEMS08`, and `PEMSBAY`, under both binary and 3-class label settings is presented in Table 3. Our proposed model, T3FORMER, consistently outperforms all baseline methods across all settings. In particular, T3FORMER achieves the highest accuracy on every task, with an average gain of 1.90% and 2.25% over the best-performing baseline for the binary and multi-class settings, respectively. Among the baselines, the transformer-based GRAPHORMER and inductive GNNs such as GSAGE and GAT perform competitively but fall short in capturing fine-grained temporal dynamics. For a fair comparison, static GNN baselines are evaluated on concatenated node features across all time steps, effectively flattening the temporal structure. While some temporal models, such as GCN+LSTM and TGN, show improvements over static counterparts, they still underperform compared to T3FORMER, particularly on more complex multi-class tasks. These results highlight the strength of our multi-view designed model.

**Brain Connectivity Results.** Table 4 presents our results on the three DynHCP tasks alongside the NeuroGraph benchmarks (Said et al. 2023). T3FORMER delivers competitive performance compared to the top-performing method on the task detection, illustrating its ability to capture global temporal patterns in brain connectivity. It outperforms all baselines on gender classification and achieves a substantial gain on age prediction, where modeling fine-grained, evolving connectivity is particularly challenging. These improvements, together with the strong performance observed on social and traffic benchmarks, underscore the adaptability and versatility of T3FORMER, which consistently delivers superior results across diverse temporal-graph domains.

## 4.3 Ablation Studies

We evaluate several ablated variants of T3FORMER (Table 5), including a static baseline (GSAGE), unimodal transformer models based on topological (TOPO-TR) or spectral (DOS-TR) descriptors, and a non-attentive fusion model (CONCAT-FUSE). Unimodal variants perform well on temporally driven datasets (e.g., `Infectious`, `Tumblr`) but degrade in feature-dominated settings (e.g., `Task`), while GSAGE shows the opposite behavior due to limited temporal expressiveness.

This complementary nature strongly motivates the adaptive cross-modal integration provided by T3FORMER's Descriptor-Attention mechanism. As shown in Table 7, attention allocation varies significantly across datasets, structural embeddings dominate social network tasks, whereas topological and spectral descriptors receive higher weights in certain traffic and brain network scenarios. Such adaptive weighting enables T3FORMER to flexibly harness the most relevant features, consistently delivering superior performance, robustness, and enhanced representation quality, as also validated through improved class separation in t-SNE visualizations (Appendix B.4).

# 5 Conclusion

We introduced **T3former**, a temporal graph classification framework that unifies global topological invariants, spectral descriptors, and local structural features via a transformer-based *Descriptor-Attention* mechanism. Extensive experiments across diverse temporal domains highlight both the adaptability of T3former and the complementary strengths of its descriptors, demonstrating superior performance. Promising future directions include extending T3former to streaming and continuous-time settings without relying on predefined sliding windows, exploring adaptive temporal resolutions, and incorporating additional topological invariants for richer feature extraction.

# References


Ali, D.; et al. 2023. A survey of vectorization methods in topological data analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Alon, U.; and Yahav, E. 2021. On the Bottleneck of Graph Neural Networks and its Practical Implications. In *ICLR*.

Borgwardt, K. M.; et al. 2005. Shortest-path kernels on graphs. In *Fifth IEEE international conference on data mining (ICDM'05)*, 8–pp. IEEE.

Cai, L.; Chen, Z.; Luo, C.; Gui, J.; Ni, J.; Li, D.; and Chen, H. 2021. Structural temporal graph neural networks for anomaly detection in dynamic graphs. In *Proceedings of the 30th ACM international conference on Information & Knowledge Management*, 3747–3756.

Chen, C.; Petty, K.; Skabardonis, A.; Varaiya, P.; and Jia, Z. 2001. Freeway performance measurement system: mining loop detector data. *Transportation research record*, 1748(1): 96–102.

Chen, H.; and Koga, H. 2019. Gl2vec: Graph embedding enriched by line graphs with edge features. In *Neural Information Processing: 26th International Conference, ICONIP 2019, Sydney, NSW, Australia, December 12–15, 2019, Proceedings, Part III 26*, 3–14. Springer.

Chen, Y.; Frias, J.; and Gel, Y. R. 2024. TopoGCL: Topological Graph Contrastive Learning. In *AAAI*, volume 38, 11453–11461.

Chen, Y.; Segovia-Dominguez, I.; Coskunuzer, B.; and Gel, Y. 2022. TAMP-S2GCNets: Coupling Time-Aware Multipersistence Knowledge Representation with Spatio-Supra Graph Convolutional Networks for Time-Series Forecasting. In *International Conference on Learning Representations (ICLR)*.

Chung, F. R. 1997. *Spectral Graph Theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. American Mathematical Society.

Cong, W.; Zhang, S.; Kang, J.; Yuan, B.; Wu, H.; Zhou, X.; Tong, H.; and Mahdavi, M. 2023. Do we really need complicated model architectures for temporal networks? *arXiv preprint arXiv:2302.11636*.

Coskunuzer, B.; and Akçora, C. G. 2024. Topological Methods in Machine Learning: A Tutorial for Practitioners. *arXiv preprint arXiv:2409.02901*.

Dey, T. K.; and Wang, Y. 2022. *Computational Topology for Data Analysis*. Cambridge University Press.

Dłotko, P.; and Gurnari, D. 2023. Euler characteristic curves and profiles: a stable shape invariant for big data problems. *GigaScience*, 12: giad094.

Dong, K.; Benson, A. R.; and Bindel, D. 2019. Network density of states. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1152–1161.

Ekle, O. A.; and Eberle, W. 2024. Anomaly detection in dynamic graphs: A comprehensive survey. *ACM Transactions on Knowledge Discovery from Data*, 18(8): 1–44.

Guo, S.; Lin, Y.; Feng, N.; Song, C.; and Wan, H. 2019. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *AAAI*, volume 33, 922–929.

Hajij, M.; Osses, A.; Glisse, M.; and Rosen, P. 2021. On the Expressive Power of Persistent Homology. In *ICLR Workshop on Geometric and Topological Representation Learning*.

Hamilton, W. L.; et al. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, volume 30.

Hiraoka, Y.; Imoto, Y.; Lacombe, T.; Meehan, K.; and Yachimura, T. 2024. Topological Node2vec: Enhanced Graph Embedding via Persistent Homology. *Journal of Machine Learning Research*, 25(134): 1–26.

Huang, S.; Coulombe, S.; Hitti, Y.; Rabbany, R.; and Rabusseau, G. 2024. Laplacian change point detection for single and multi-view dynamic graphs. *ACM Transactions on Knowledge Discovery from Data*, 18(3): 1–32.

Huang, S.; Poursafaei, F.; Danovitch, J.; Fey, M.; Hu, W.; Rossi, E.; Leskovec, J.; Bronstein, M.; Rabusseau, G.; and Rabbany, R. 2023. Temporal Graph Benchmark for Machine Learning on Temporal Graphs. *Advances in Neural Information Processing Systems*, 36: 2056–2073.

Immonen, J.; Souza, A.; and Garg, V. 2024. Going beyond persistent homology using persistent homology. *Advances in Neural Information Processing Systems*, 36.

Kim, H.; Lee, B. S.; Shin, W.-Y.; and Lim, S. 2022. Graph anomaly detection with graph neural networks: Current status and challenges. *IEEE Access*, 10: 111820–111829.

Kumar, S.; Zhang, X.; and Leskovec, J. 2019. Predicting Dynamic Embedding Trajectory in Temporal Interaction Networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, 1269–1278.

Loiseaux, D.; Scoccola, L.; Carrière, M.; Botnan, M. B.; and Oudot, S. 2024. Stable vectorization of multiparameter persistent homology using signed barcodes as measures. *Advances in Neural Information Processing Systems*, 36.

Milosavljevic, N.; et al. 2011. Zigzag Persistent Homology in Matrix Multiplication Time. In *Proceedings of the 27th Annual Symposium on Computational Geometry*, 378–387.

Morris, C.; Kriege, N. M.; Bause, F.; Kersting, K.; Mutzel, P.; and Neumann, M. 2020. TUDataset: A collection of benchmark datasets for learning with graphs. ArXiv preprint, arXiv:2007.08663.

Morris, C.; Ritzert, M.; Fey, M.; Hamilton, W. L.; Lenssen, J. E.; Rattan, G.; and Grohe, M. 2019. Weisfeiler and Leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 4602–4609.

Narayanan, A.; Chandramohan, M.; Venkatesan, R.; Chen, L.; Liu, Y.; and Jaiswal, S. 2017. graph2vec: Learning distributed representations of graphs. *arXiv preprint arXiv:1707.05005*.

Otter, N.; Porter, M. A.; Tillmann, U.; Grindrod, P.; and Harrington, H. A. 2017. A roadmap for the computation of persistent homology. *EPJ Data Science*, 6: 1–38.

Pareja, A.; Domeniconi, G.; Chen, J.; Ma, T.; Suzumura, T.; Kanezashi, H.; Kaler, T.; Schardl, T.; and Leiserson, C. 2020. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In *AAAI*, volume 34, 5363–5370.

Rossi, E.; Chambers, J.; Ying, R.; Bronstein, M.; and Monti, F. 2020. Temporal Graph Networks for Deep Learning on Dynamic Graphs. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

Said, A.; Bayrak, R.; Derr, T.; Shabbir, M.; Moyer, D.; Chang, C.-H.; and Koutsoukos, X. 2023. NeuroGraph: Benchmarks for Graph Machine Learning in Brain Connectomics. In *Advances in Neural Information Processing Systems*, volume 36, 6509–6531.

Shamsi, K.; Poursafaei, F.; Huang, S.; Ngo, B. T. G.; Coskunuzer, B.; and Akcora, C. G. 2024. GraphPulse: Topological Representations for Temporal Graph Property Prediction. In *The Twelfth International Conference on Learning Representations (ICLR)*.

Shervashidze, N.; Schweitzer, P.; Van Leeuwen, E. J.; Mehlhorn, K.; and Borgwardt, K. M. 2011. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(9).

Shi, Y.; Huang, Z.; Feng, S.; Zhong, H.; Wang, W.; and Sun, Y. 2020. Masked label prediction: Unified message passing model for semi-supervised classification. *arXiv preprint arXiv:2009.03509*.

Skraba, P.; and Turner, K. 2020. Wasserstein stability for persistence diagrams. *arXiv:2006.16824*.

Spielman, D. A. 2011. Spectral Graph Theory and Its Applications. http://cs-www.cs.yale.edu/homes/spielman/spectral/. Tutorial presented at IEEE FOCS 2011.

Tieu, K.; Fu, D.; Zhu, Y.; Hamann, H.; and He, J. 2024. Temporal Graph Neural Tangent Kernel with Graphon-Guaranteed. *Advances in Neural Information Processing Systems*, 37: 94568–94606.

Trivedi, R.; Farajtabar, M.; Biswal, P.; and Zha, H. 2019. DyRep: Learning Representations over Dynamic Graphs. In *ICLR*.

Tsitsulin, A.; Mottin, D.; Karras, P.; Bronstein, A.; and Müller, E. 2018. Netlsd: hearing the shape of a graph. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2347–2356.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2018. Graph attention networks. *ICLR*.

Verma, Y.; Souza, A. H.; and Garg, V. 2024. Topological neural networks go persistent, equivariant, and continuous. In *Proceedings of the 41st International Conference on Machine Learning*, 49388–49407.

Vishwanathan, S. V. N.; Schraudolph, N. N.; Kondor, R.; and Borgwardt, K. M. 2010. Graph kernels. *The Journal of Machine Learning Research*, 11: 1201–1242.

Von Luxburg, U. 2007. A tutorial on spectral clustering. *Statistics and computing*, 17: 395–416.

Wang, H.; and Zhang, J. 2021. Adaptive Path Aggregation for Temporal Graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, 20547–20558.

Weisse, A.; Wellein, G.; Alvermann, A.; and Fehske, H. 2006. The kernel polynomial method. *Reviews of Modern Physics*, 78(1): 275–306.

Xu, D.; Ruan, C.; Korpeoglu, B.; Kumar, S.; and Prakash, B. A. 2020. Inductive Representation Learning on Temporal Graphs. In *ICLR*.

Yang, X.; Zhao, X.; and Shen, Z. 2025. A Generalizable Anomaly Detection Method in Dynamic Graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 22001–22009.

Ye, D.; Jiang, H.; Jiang, Y.; and Li, H. 2023. Stable distance of persistent homology for dynamic graph comparison. *Knowledge-Based Systems*, 278: 110855.

Ying, C.; et al. 2021. Do transformers really perform badly for graph representation? *NeurIPS*, 34: 28877–28888.

You, J.; et al. 2020. Design space for graph neural networks. *Advances in Neural Information Processing Systems*, 33: 17009–17021.

Yu, B.; et al. 2018. Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*, 3634–3640.

Yu, L.; Sun, L.; Du, B.; and Lv, W. 2023. Towards better dynamic graph learning: New architecture and unified library. *Advances in Neural Information Processing Systems*, 36: 67686–67700.

Zhao, L.; Song, Y.; Zhang, C.; Cui, Z.; Wang, P.; and Sun, X. 2020. DySAT: Dynamic Self-Attention Network for Learning on Dynamic Graphs. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 3465–3475.