A Temporal Graphlet Kernel for Classifying Dissemination in Evolving Networks

Lutz Oettershagen[1], Nils M. Kriege[2,3], Claude Jordan[1], and Petra Mutzel[1]

[1] Institute of Computer Science, University of Bonn, Bonn, Germany
`{lutz.oettershagen, petra.mutzel}@cs.uni-bonn.de`, `claudejordan@gmail.com`
[2] Faculty of Computer Science, University of Vienna, Vienna, Austria
`nils.kriege@univie.ac.at`
[3] Research Network Data Science, University of Vienna, Vienna, Austria

Abstract. We introduce the *temporal graphlet kernel* for classifying dissemination processes in labeled temporal graphs. Such dissemination processes can be spreading (fake) news, infectious diseases, or computer viruses in dynamic networks. The networks are modeled as labeled temporal graphs, in which the edges exist at specific points in time, and node labels change over time. The classification problem asks to discriminate dissemination processes of different origins or parameters, e.g., infectious diseases with different infection probabilities. Our new kernel represents labeled temporal graphs in the feature space of temporal graphlets, i.e., small subgraphs distinguished by their structure, time-dependent node labels, and chronological order of edges. We introduce variants of our kernel based on classes of graphlets that are efficiently countable. For the case of temporal wedges, we propose a highly efficient approximative kernel with low error in expectation. We show that our kernels are faster to compute and provide better accuracy than state-of-the-art methods.

Keywords: Temporal Graphs, Dissemination, Kernel, Classification.
Kind of paper: Novel research paper

1 Introduction

Dissemination processes such as spreading information or disease can be challenging to analyze and track. Recent works [17,18,25] discuss the problem of classifying dissemination processes in social and human contact networks, e.g., discriminating the spread of real news from fake news, different infectious diseases, or malicious from benign network communications. Identifying such spreading processes correctly in real-world and online social networks can have immense social impacts. For example, detecting a new viral pathogen spreading in communities can help to react early and prevent severe outbreaks [4,19]. Similarly, identifying and limiting the spread of fake news on social networks like *Facebook* or *WeChat* can help to reduce resulting social unrest [16,24,28]. These kinds of spreading processes can naturally be modeled using labeled temporal graphs. A labeled temporal graph consists of a fixed set of nodes, a label function that assigns a discrete label to each node at each point in time, and a set of timestamped
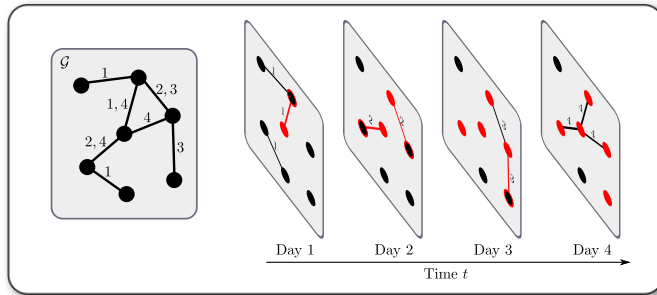
Fig. 1: Example for temporal graph $\mathcal{G}$ in which nodes represent persons and edges contacts over four days. The edge labels denote on which days the edges are available. An infection spreads in the network: On the first day, one person is initially infected (highlighted in red). The infection spreads along the edges highlighted in red. The next infected persons are outlined in red.

edges. Figure 1 shows an example of a temporal graph and a dissemination process over time (highlighted in red). Because temporal graphs are suitable models for a wide range of real-life scenarios with dynamic relations [7], research in temporal graphs has recently gained increasing attention [2,15,17,20]. Methods for analyzing and classifying static graphs, on the other hand, have been studied for decades and are an established area of research—dominating learning methods are based on graph kernels [10] and graph neural networks [30]. A well-known and popular static graph kernel is the *graphlet kernel* proposed in [23]. We introduce a temporal version, named *temporal graphlet kernel*, to account for the specific properties of temporal graphs. Our kernel is inspired by recent works on temporal motifs [12,13,14,20,21]. Temporal motifs are generalizations of small subgraph patterns, i.e., graphlets, that incorporate temporal properties like the chronological ordering of the edges. In contrast to previous works [17,18], our temporal graphlet kernel directly operates on the labeled temporal graphs without prior transformations into static graphs.

The authors of [18] proposed three classification tasks of dissemination on temporal networks. The first one concerns the discrimination of observations of a dissemination process and random observations. The second task aims to discriminate temporal graphs subject to two dissemination processes differing in the infection probability. Finally, the third classification task is similar to the second one, but under incomplete information, i.e., parts of the network's observations are unavailable. Our kernel achieves a high state-of-the-art classification accuracy and efficiency on all three tasks.

Contributions:

- We introduce the temporal graphlet kernel, defined as the inner product of normalized graph feature vectors counting the occurrences of labeled temporal graphlets.

– We present a highly efficient approximation variant of our temporal graphlet kernel that approximates the number of temporal wedges.
– We show that our temporal graphlet kernels reach high accuracies in classifying epidemic spreading in real-world data sets. In most cases, our temporal graphlet kernels beat the state-of-the-art kernels and neural networks and are on par otherwise. The gains in accuracy are often significant. Moreover, our kernels are up to several orders of magnitude faster than the baselines.

We give an overview of the related work in Appendix A.1.

2 Preliminaries

A *labeled, temporal graph* $\mathcal{G} = (V, \mathcal{E}, l)$ consists of a finite set V of nodes, a finite set $\mathcal{E}$ of (directed) *temporal edges* $e = (u, v, t)$ with u and v in V, $u \neq v$, *availability time* (or *time stamp*) $t \in \mathbb{N}$, and a labeling function $l: V \times \mathbb{N} \to \Sigma$. The labeling function l assigns a label to each node at each time step $t \in \mathbb{N}$. Let $d(v)$ be the degree of vertex v, i.e., the total number of incoming and outgoing temporal edges. We denote with $S(\mathcal{G}) = (V, E)$ the underlying static graph of the temporal graph $\mathcal{G} = (V, \mathcal{E})$ with $E = \{(u, v) \mid (u, v, t) \in \mathcal{E}\}$.

Kernels for Graphs: A *kernel* on a non-empty set $\mathcal{X}$ is a symmetric, positive semidefinite function $k: \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. Equivalently, a function k is a kernel if there is a *feature map* $\phi: \mathcal{X} \to \mathcal{H}$ to a Hilbert space $\mathcal{H}$ with inner product $\langle \cdot, \cdot \rangle$, such that $k(x, y) = \langle \phi(x), \phi(y) \rangle$ for all x and y in $\mathcal{X}$. Let $\mathbb{G}$ be the set of all (temporal) graphs, then a kernel $k: \mathbb{G} \times \mathbb{G} \to \mathbb{R}$ is a *(temporal) graph kernel*.

Static Graphlet Kernel: Shervashidze et al. [23] introduced the graphlet kernel for static (unlabeled) graphs. It counts the occurrences of subgraph patterns of a fixed size which are called *graphlets*. For a graph G and $k \in \{3, 4, 5\}$, the static graphlet kernel counts the isomorphism types of all induced subgraphs with k nodes. The subgraphs can be disconnected. Let $\phi(G)_{\sigma_i}$ be the number of occurrences of the isomorphism type σ_i for $1 \leq i \leq N$ with N the number of different types. The feature map of the kernel is then $\phi_{SG}(G) = (\phi(G)_{\sigma_1}, \ldots, \phi(G)_{\sigma_N})$, and the graphlet kernel is $k_{SG}(G, H) = \langle \phi_{SG}(G), \phi_{SG}(H) \rangle$ for all graphs G and H in $\mathbb{G}$. For small graphs representing molecules, labeled graphlets have also been considered [29].

Temporal Motifs: Our work is based on extending the commonly used definition of temporal motifs first introduced in [20].

Definition 1. *A k-node, ℓ-edge, δ-temporal graphlet is a sequence of ℓ temporal edges, $g = ((u_1, v_1, t_1), (u_2, v_2, t_2), \ldots, (u_\ell, v_\ell, t_\ell))$ that is (i) chronologically ordered, i.e., $t_1 < t_2 < \cdots < t_\ell$, (ii) within a δ time interval, i.e., $t_\ell - t_1 \leq \delta$, and (iii) the induced static graph is connected and has k nodes.*

3 Temporal Graphlet Kernel

We extend Definition 1 for labeled temporal graphs. Our approach is general and can be adapted for other definitions.

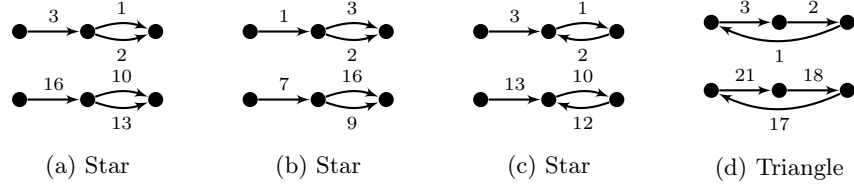(a) Star   (b) Star   (c) Star   (d) Triangle

Fig. 2: Examples of 3-node, 3-edge, 10-temporal graphlets without labels. The two graphlets in each of (a)–(d) are equivalent. All other pairs of graphlets are non-equivalent, e.g., the graphlets in (a) are non-equivalent to those in (b)–(d).

Definition 2. *In a labeled temporal graph, the label of a k-node, ℓ-edge, δ-temporal graphlet $g = ((u_1, v_1, t_1), (u_2, v_2, t_2), \ldots, (u_\ell, v_\ell, t_\ell))$ is*

$$l(g) = (l(u_1, t_1), l(v_1, t_1 + 1), l(u_2, t_2), l(v_2, t_2 + 1), \ldots, l(u_\ell, t_\ell), l(v_\ell, t_\ell + 1)).$$

We are interested in the classification of dissemination that spreads along temporal edges. The labels can be used to encode different dissemination patterns. Next, we define an equivalence relation on graphlets.

Definition 3. *Two δ-temporal graphlets $g = ((u_1, v_1, t_1), \ldots, (u_\ell, v_\ell, t_\ell))$ and $g' = ((u'_1, v'_1, t'_1), \ldots, (u'_{\ell'}, v'_{\ell'}, t'_{\ell'}))$ are equivalent, written $g \sim g'$, if (i) $\ell = \ell'$, (ii) there exists a bijection ψ from the nodes of g to the nodes of g' with $\psi(u_i) = u'_i$ and $\psi(v_i) = v'_i$ for all $i \in \{1, \ldots, \ell\}$, and, in case of labels, (iii) $l(g) = l(g')$.*

A temporal graphlet is edge-induced, in contrast to the static graphlet kernel, which counts the number of node-induced, possibly disconnected, subgraphs. Moreover, our equivalence relation considers the chronological order of the edges but not the actual time stamps, which would be too restrictive. Figure 2 shows an example for the equivalence relation. Note that non-equivalence can also arise from different labels in the case of labeled temporal graphs. Next, we use the equivalence relation to define graph feature vectors based on temporal graphlets.

Definition 4. *For a parameter δ in $\mathbb{N}$, let $\mathcal{T}$ be the equivalence classes of $\sim$. Given a temporal graph $\mathcal{G}$, we define $\hat{\phi}^{\mathcal{T}}_{TG}(\mathcal{G})$ as the vector with a component for each $\tau \in \mathcal{T}$ counting the occurrences of temporal graphlets equivalent to τ in $\mathcal{G}$. We denote the normalized feature vector by $\phi^{\mathcal{T}}_{TG}(\mathcal{G}) = \frac{\hat{\phi}^{\mathcal{T}}_{TG}(\mathcal{G})}{\|\hat{\phi}^{\mathcal{T}}_{TG}(\mathcal{G})\|_1}$.*

We derive the temporal graph kernel from these feature vector.

Definition 5 (Temporal Graphlet Kernel). *Given two temporal graphs $\mathcal{G}_1$ and $\mathcal{G}_2$, the temporal graphlet kernel is $k^{\mathcal{T}}_{TG}(\mathcal{G}_1, \mathcal{G}_2) = \langle \phi^{\mathcal{T}}_{TG}(\mathcal{G}_1), \phi^{\mathcal{T}}_{TG}(\mathcal{G}_2) \rangle$.*

Fig. 3: The four non-equivalent temporal wedges with time stamps $t_1 < t_2$.

4 Counting Temporal Graphlets

Counting temporal graphlets is a hard problem. In general, deciding if a δ, k-star graphlet exists in a given temporal graph is `NP`-complete [12]. The known counting and enumeration algorithms for general temporal graphlets have exponential worst-case running times, e.g., the backtracking enumeration algorithm of Mackey et al. [14] or the general counting algorithm presented in [20]. Besides these negative results, several special cases of temporal graphlets can be computed efficiently. In the following, we discuss the counting of general temporal graphlets and the efficient counting of temporal graphlets with two or three nodes and edges, respectively, which are elementary classes of motifs for the characterization of temporal networks [26,20], as well as the underlying dissemination process.

4.1 Labeled Temporal Graphlets for Dissemination

Our goal is to classify disease or information spreading in temporal graphs. Therefore, we use a binary label alphabet Σ that encodes if a node is *infected* or *susceptible* or if a node has obtained some information or not. It is possible to use our approach for larger alphabet sizes that include, e.g., *exposed* or *recovered* labels to model complex epidemiological behaviors [3]. In the following, we discuss the counting of small graphlets with two or three nodes and edges, respectively. The motivation is that the dissemination of, e.g., viruses or (fake) news on social networks is usually a mainly local process [9,28]. Therefore, we expect small and connected graphlets to capture these processes well—we verify this hypothesis in Section 5. The authors of [20] identified 36 non-equivalent temporal unlabeled graphlets for $k \in \{2, 3\}$ nodes and $\ell = 3$ edges. More specifically, eight non-equivalent triangles and 24 stars with three edges. In the case of three vertices and two edges, the temporal graphlet is called a *temporal wedge*. Figure 3 shows the four non-equivalent (unlabeled) temporal wedges. If we take node labelings into account, the number of non-equivalent graphlets increases.

Lemma 1. *Let $\mathcal{G}$ be a labeled temporal graph and $L = |\Sigma|$ the size of its label alphabet. The number of distinct labels of ℓ-edge temporal graphlets is $L^{2\ell}$.*

It follows from Lemma 1 that for $\Sigma = \{$infectected, susceptible$\}$ there are a total number of 2304 pair-wise non-equivalent labeled graphlets in $\mathcal{G}$ with two or three nodes and three edges. Similarly, the number of non-equivalent labeled temporal wedges is 64.

4.2   Counting General Temporal Graphlets

In order to count the occurrences of general labeled graphlets, we adapt the counting algorithm introduced in [20]. Counting the occurrences of a (non-labeled) temporal graphlet g is done in three steps:

1. Find all instances of the subgraph $S(g)$ in $S(\mathcal{G})$, where $S(g)$ denotes the underlying static graph of the graphlet g, i.e., all temporal edges are replaced by static edges, and multi-edges are merged.
2. For each $S(g)$, collect all temporal edges incident only to nodes in $S(g)$ in a chronologically ordered sequence σ.
3. Count the occurrences of g in σ using dynamic programming.

The first step is done using an algorithm for enumerating subgraphs in the static graph $S(\mathcal{G})$. The second step is just collecting the corresponding temporal edges. The third step uses a dynamic programming approach in which a temporal graphlet is considered as a chronologically ordered sequence of temporal edges. Therefore, the count of a graphlet of length ℓ edges can be computed using the count of the prefix of length $\ell - 1$. To this end, for each possible edge sequence of lengths smaller or equal to ℓ, a counter keeps track of its occurrences. Let m be the number of static edges in $S(g)$, then all occurrences of g in the sequence σ can be counted with a single pass over the edge sequence σ in $\mathcal{O}(m^\ell \cdot |\sigma|)$ running time because each edge leads to maximal $\mathcal{O}(m^\ell)$ counter updates when it is processed and when it leaves the time window δ. The sliding time window is used to ensure that only graphlets are counted that respect the maximal temporal distance of δ. Furthermore, after σ is processed, the counters contain all occurrences of graphlets with maximal ℓ edges in $S(g)$.

   In order to apply the counting framework for labeled temporal graphlets, we map the time-dependent node labels to edge labels.

Lemma 2. *Given a temporal graphlet* $g = (e_1, \ldots, e_\ell)$, *its label* $l(g)$ *can be mapped one-to-one to an edge label sequence* $l_e(g) = (l_e(e_1), \ldots, l_e(e_\ell))$.

To count the labeled graphlets, we now can apply Lemma 2 to the counting framework, where we use, for each possible *labeled edge sequence* σ_l of lengths smaller or equal to ℓ, a counter that keeps track of the occurrences of σ_l. The running time for counting labeled temporal graphlets is then in $\mathcal{O}((m \cdot |\Sigma|^2)^\ell \cdot |\sigma_l|)$.

4.3   Counting Two and Three Node Graphlets

The running time of the general counting algorithm described in the previous section can be improved for graphlets with two or three nodes and three edges. Efficient variants for graphlets with two or three nodes and three edges exist, namely for *stars* and *triangles* [20]. Counting the number of all labeled star graphlets over three nodes and edges is possible in $\mathcal{O}(|\mathcal{E}| \cdot |\Sigma|^2)$. We can count temporal wedges similarly. And, counting the numbers of all labeled temporal triangles has a running time in $\mathcal{O}(\mathrm{T}_\Delta + c_\Delta |\mathcal{E}| \cdot |\Sigma|^2)$, where T_Δ is the running time for the enumeration of static triangles in $S(\mathcal{G})$ and c_Δ the number of static triangles in $S(\mathcal{G})$. We refer the reader to [20] for further details of the algorithms.

4.4   Temporal Wedge Kernel Approximation

As we will see in Section 5, the wedge-based temporal graphlet kernel shows an excellent trade-off between running time and classification accuracy. The reason is that the temporal wedges can capture dissemination patterns well, and they are easy to count. We further improve this trade-off with a highly efficient approximation algorithm. First, note that we can count the labeled wedges by iterating over all nodes and considering all incoming and outgoing edge combinations. For each pair of edges, we increase the counter of the correspondent labeled wedge. The running time for counting all labeled wedges is then in $\mathcal{O}(\sum_{v \in V} d(v)^2)$.

We introduce an approximation for the temporal wedge kernel based on this counting variant. The idea is to sample temporal wedges by extending techniques for wedge sampling in static graphs [22]. First, let $\mathcal{G} = (V, \mathcal{E}, l)$ be a labeled temporal graph that has no vertex with two incident edges with the same time stamp (we discuss the general case later). Then $w = \sum_{v \in V} \binom{d(v)}{2}$ is the total number of temporal wedges in $\mathcal{G}$. We first sample a vertex with a probability of $p_v = \binom{d(v)}{2}/w$ and then a pair of to v incident edges uniformly at random, i.e., with probability $1/\binom{d(v)}{2}$. The probability of the sampled temporal wedge is then $1/w$, and therefore, it is a uniform sample. Algorithm 1 shows the approximation for the feature vector $\widetilde{\phi}_{\mathrm{TG}}(\mathcal{G})$ of normalized temporal wedge counts.

Theorem 1. *The running time of Algorithm 1 is in* $\mathcal{O}(|V| + |\Sigma|^2 + s)$.

Algorithm 1

Input: A temporal graph $\mathcal{G}$, sample size $s \in \mathbb{N}$
Output: A feature vector $\widetilde{\phi}_{\mathrm{TG}}(\mathcal{G})$ of normalized temporal wedge counts
 1: Initialize feature vector $\widetilde{\phi}_{\mathrm{TG}}(\mathcal{G})$ to a vector of zeros
 2: Compute probabilities p_v for all $v \in V$
 3: **for** $1, \ldots, s$ **do**
 4:     Sample vertex v with probability p_v
 5:     Sample two edges $\{e, f\}$ incident to v
 6:     Let τ be the temporal wedge defined by e and f
 7:     $\widetilde{\phi}_{\mathrm{TG}}(\mathcal{G})_\tau \leftarrow \widetilde{\phi}_{\mathrm{TG}}(\mathcal{G})_\tau + \frac{1}{s}$
 8: **end for**
 9: **return** $\widetilde{\phi}_{\mathrm{TG}}(\mathcal{G})$

Theorem 2. *Let $\mathbb{G}$ be a set of temporal graphs with label alphabet Σ. Moreover, let W be number of non-equivalent labeled temporal wedges, $\lambda \in \mathbb{R}_{>0}$, and $\delta \in (0, 1)$. For $s = \left\lceil \frac{\log(2 \cdot |\mathbb{G}| \cdot W/\delta)}{2(\lambda/W)^2} \right\rceil$, Algorithm 1 approximates the normalized temporal graphlet wedge kernel k_{TG} with probability $(1 - \delta)$, such that*

$$\sup_{\mathcal{G}_1, \mathcal{G}_2 \in \mathbb{H}} \left| k_{TG}(\mathcal{G}_1, \mathcal{G}_2) - \langle \widetilde{\phi}_{TG}(\mathcal{G}_1), \widetilde{\phi}_{TG}(\mathcal{G}_2) \rangle \right| \leq 3\lambda.$$

To support temporal graphs that have equal time stamps of edges at the same vertex, and to obtain a random uniform temporal wedge respecting a given

time interval δ, we sample a vertex v with probability p_v and one of the edges incident to v uniformly at random. Then we choose the second edge, f, only from the incident edges such that the difference of the availability times of e and f is in $[1, \delta]$. To correct the introduced bias, we apply rejection sampling. Let p_f be the probability of the second edge f. The probability of a temporal wedge τ respecting δ and with center v is then $P_\tau = \frac{p_v \cdot p_f}{d(v)}$. A lower bound for the probability of any temporal wedge is $P_{min} = 1/w$. To obtain a uniformly sampled temporal wedge respecting the interval δ, we accept a wedge with probability $\frac{P_{min}}{P_\tau}$, and we achieve a uniform probability of $P_\tau \cdot \frac{P_{min}}{P_\tau} = 1/w$.

   If the input graphs have similar distributions of the edge times and degrees, forgoing the rejection step leads to similar biases during the wedge sampling. In this case, we may waive the rejection step and still achieve high accuracy.

5   Experiments

We evaluate our temporal graphlet kernel and compare the effectiveness and efficiency to the baselines provided in [17,18,25]. Our research questions are:

- **Q1.** How do our new kernels compare to the baselines in terms of accuracy?
- **Q2.** What are the running times of our temporal graphlet kernels?
- **Q3.** What is the solution quality and running time of our approximation?
- **Q4.** How is the classification accuracy affected by incomplete knowledge of the dissemination process?

Data Sets: We use the temporal dissemination classification data sets introduced in [18]. The data sets contain the three different classification tasks described in the introduction based on real-world temporal graphs that model physical human interactions and social networks. Please refer to Appendix A.3 for a detailed description.

Kernel Instances: We compare the following variants of our temporal graphlet kernel for $k = 3$ nodes and $\ell \in \{2, 3\}$ edges: 1) *TGK-∧* counts labeled temporal wedges, 2) *TGK-⋆* counts labeled temporal star graphlets, 3) *TGK-all* counts all labeled temporal graphlets, and 4) *Approx-s* is our approximation of *TGK-∧* with sample size $s \in \{50, 100, 200\}$. Note that setting $k = \{2, 3\}$, i.e., the kernels also count graphlets on two nodes, did not improve the results.

   Furthermore, we use state-of-the-art approaches based on kernels and neural networks as baselines. The authors of [18] introduce three different transformations of temporal graphs to static graphs, which differ in the size of the resulting graphs and the amount of loss of temporal information. After the transformations, they apply static kernels to the static graphs for classifying the dissemination process. They use the k-step random walk and the Weisfeiler-Leman subtree kernel. This approach results in the following kernel instances: (1) *RD-RW* and *RD-WL*, which use the *reduced graph representation*, (2) *DL-RW* and *DL-WL*, which use the *directed line graph expansion*, and (3) *SE-RW* and *SE-WL*, which use the *static expansion*. In [17], the authors of [18] introduced graph neural networks (GNNs) based on the three graph transformations. However,

Table 1: Classification accuracy in percent and standard deviation for the first and second classification tasks. For each data set, we highlight the highest accuracy in bold. OOT—Computation did not finish within the time limit.

(a) Classification accuracy for the first classification task.

	Kernel	Data set					
		MIT	HIGHSCHOOL	INFECTIOUS	TUMBLR	DBLP	FACEBOOK
	TGK-$\wedge$	**93.45** ± 1.8	98.00 ± 0.9	**98.30** ± 0.4	93.19 ± 0.7	98.47 ± 0.1	95.21 ± 0.2
	TGK-$\star$	87.02 ± 1.6	96.50 ± 0.9	98.05 ± 0.4	93.19 ± 0.8	98.05 ± 0.2	95.32 ± 0.2
	TGK-all	87.46 ± 0.9	97.17 ± 0.8	97.90 ± 0.4	93.57 ± 0.8	98.12 ± 0.1	94.69 ± 0.3
	$Approx$-50	88.96 ± 2.1	94.50 ± 0.8	91.75 ± 1.0	91.63 ± 0.4	97.12 ± 0.2	94.05 ± 0.2
	$Approx$-100	89.19 ± 1.6	95.88 ± 1.3	95.15 ± 0.7	91.84 ± 0.6	97.66 ± 0.3	94.10 ± 0.1
	$Approx$-200	90.78 ± 1.5	97.66 ± 0.6	96.80 ± 1.0	92.86 ± 0.5	98.06 ± 0.5	95.02 ± 0.1
Baselines	RG-RW	61.31 ± 2.7	90.16 ± 1.0	89.30 ± 1.0	74.99 ± 1.9	90.60 ± 1.0	82.86 ± 0.6
	RG-WL	81.88 ± 1.1	89.88 ± 0.9	91.75 ± 1.0	70.50 ± 1.0	90.45 ± 0.5	81.15 ± 0.8
	DL-RW	92.91 ± 0.9	98.33 ± 0.7	97.05 ± 0.8	**94.64** ± 0.5	98.16 ± 0.1	96.46 ± 0.1
	DL-WL	90.67 ± 1.6	**98.88** ± 0.4	97.35 ± 1.5	94.05 ± 0.9	98.56 ± 0.3	**96.59** ± 0.4
	SE-RW	88.56 ± 1.0	96.89 ± 1.2	97.60 ± 0.6	93.97 ± 0.9	**98.65** ± 0.3	95.46 ± 0.2
	SE-WL	87.31 ± 1.9	96.72 ± 0.7	94.45 ± 1.1	93.51 ± 0.6	97.38 ± 0.2	95.39 ± 0.4

(b) Classification accuracy for the second classification task.

	Kernel	Data set					
		MIT	HIGHSCHOOL	INFECTIOUS	TUMBLR	DBLP	FACEBOOK
	TGK-$\wedge$	68.52 ± 3.5	93.83 ± 0.8	89.65 ± 0.8	**79.06** ± 0.7	83.76 ± 0.5	76.63 ± 0.3
	TGK-$\star$	77.03 ± 3.7	**95.33** ± 0.8	**90.55** ± 1.6	78.13 ± 1.1	85.42 ± 0.8	**82.12** ± 0.3
	TGK-all	**78.43** ± 4.2	94.22 ± 1.3	90.40 ± 0.9	76.57 ± 1.0	**85.84** ± 0.6	81.13 ± 0.6
	$Approx$-50	55.78 ± 3.5	84.77 ± 1.5	84.50 ± 1.1	75.92 ± 0.5	78.00 $\pm 0,3$	74.12 ± 0.4
	$Approx$-100	60.11 ± 4.5	89.61 ± 1.9	85.10 ± 1.0	76.16 ± 0.7	79.32 ± 0.5	74.41 ± 0.3
	$Approx$-200	62.97 ± 2.8	91.94 ± 1.2	85.45 ± 1.4	78.46 ± 0.8	79.61 ± 0.5	76.32 ± 0.3
Baselines	RG-RW	58.03 ± 3.7	77.33 ± 2.4	72.05 ± 2.2	68.48 ± 1.5	63.24 ± 1.2	66.68 ± 0.9
	RG-WL	66.81 ± 2.0	82.78 ± 1.3	77.40 ± 1.2	68.25 ± 1.2	66.16 ± 0.5	66.96 ± 0.7
	DL-RW	OOT	91.44 ± 1.1	87.35 ± 1.3	76.51 ± 0.5	81.79 ± 0.9	79.97 ± 0.5
	DL-WL	40.87 ± 3.6	87.11 ± 1.7	77.55 ± 2.0	78.69 ± 0.8	74.47 ± 1.1	79.44 ± 0.5
	SE-RW	51.03 ± 5.1	90.77 ± 1.1	83.60 ± 1.1	77.09 ± 1.0	83.31 ± 1.0	79.56 ± 0.6
	SE-WL	46.52 ± 3.9	91.55 ± 0.9	79.60 ± 1.5	78.64 ± 1.4	81.24 ± 0.6	74.68 ± 0.7

the evaluation in [17] showed that the kernel-based approaches outperform the GNNs in almost all instances while requiring less computation time. Tortorella and Micheli [25] introduced a dynamic echo state network called $DynGESN$ for classification of dissemination in temporal networks. Further details and a comparison of our kernels with these neural baselines can be found in Appendix A.6.

We implemented our kernels in C++ using GNU CC Compiler 9.3.0. The source code and data sets are available online.[4] The C++ implementation of the baseline kernels from [18] were provided by the authors and compiled with the same settings as our kernels. The experimental protocol is in Appendix A.4.

---

[4] `https://gitlab.com/tgpublic/tgraphlet`

5.1   Results

Q1. Table 1a and Table 1b show the accuracies for the first and second classification tasks, respectively. Our temporal graph kernels perform better than the baselines for eight of the twelve data sets and are on par for the remaining four. For the first classification task, our kernels have the highest accuracies for Infectious, and TGK-$\wedge$ also has the highest accuracy for Mit. For the other data sets, the accuracies are similar to the best performing baselines. In the case of the second classification task, TGK-$\wedge$, TGK-$\star$, and TGK-all achieve higher accuracies the baselines for all data sets. Similarly, Appendix A.6 shows that our new kernels usually reach higher accuracies compared to the GNN baselines. In general, the accuracies of our kernels are very close to each other for most data sets, and they can all capture the discriminating information well.

Q2. Table 2a shows the running times for the kernel computations for the second classification task. We observed similar results for the first task (see Appendix A.5). The impact of the time window parameter δ on the running time of the temporal graphlet kernels is limited. TGK-$\wedge$ is the fastest kernel for all data sets but the Mit data set. Here the WL-kernel based on the reduced graph representation (RG-WL) is faster. Its running time also comes close to TGK-$\wedge$ for the other data sets, but it has much worse accuracy (see **Q1**). The reason is that there are no multiple edges between pairs of nodes in graphs in reduced graph representation, which leads to loss of temporal information and affects the accuracy. Our TGK-$\star$ is faster than the random walk kernels and the WL-kernels for Highschool and Mit. Notice that the running time of DL-RW exceeded the time limit of one hour, where our TGK-$\star$ kernel only needed 102 ms.

Q3. Table 1a and Table 1b show the accuracies of our approximation algorithm for sample sizes $s \in \{50, 100, 200\}$. With increasing sample size, the approximation error is reduced as expected. The approximation error is generally low, while the speed-up compared to TGK-$\wedge$ for low sample sizes is high. However, for $s = 200$, the difference in running time of the approximation and TGK-$\wedge$ is small for Tumblr and Dblp. The reason is that the exact algorithm is already very efficient due to the low average maximal degree in these data sets (see Table 4). Due to Theorem 2, our approximation is suited for large data sets. To further evaluate the performance on a larger data set, we generated a synthetic data set consisting of 100 graphs generated using the Barabási–Albert preferential attachment method [1] and randomly chosen availability times $t(e) \in [0, 1000]$ for each edge e. Each of the graphs contains 5000 nodes and 49 900 edges with an average maximal degree of 390.3. The dissemination process is simulated analogously to the second classification task. Table 3 shows the running times and accuracies. The results suggest very good scalability of our approximation. The running times are only a fraction of the exact methods, and the accuracy is high.

Q4. In order to evaluate how the accuracy is affected by incomplete knowledge, we applied our kernel to the third classification task, in which classification tasks one and two are modified by partly reset node labels. We compare our kernels with the best performing baselines DL-RW, DL-WL, SE-RW, and SE-WL.

Table 2: Running times in ms for the second classification task.

(a) Random walk length $k = 3$ ($k = 2$ for *DL-RW*), number of iterations of WL $h = 3$ ($h = 2$ for *DL-WL*). OOT—Computation did not finish within the time limit.

Kernel		Data set					
		MIT	HIGHSCHOOL	INFECTIOUS	TUMBLR	DBLP	FACEBOOK
TGK-∧	$\delta = 10$	290	**36**	**39**	**69**	293	**459**
TGK-⋆	$\delta = 10$	102	192	202	467	1 605	2 819
TGK-all	$\delta = 10$	311	755	1 442	815	5 154	3 935
TGK-∧	$\delta = 10^3$	482	56	41	70	**287**	463
TGK-⋆	$\delta = 10^3$	103	200	211	478	1 726	2 796
TGK-all	$\delta = 10^3$	283	742	1 608	799	4 809	4 024
RG-RW		803	9 868	10 504	5 653	17 516	10 444
RG-WL		**14**	62	76	109	304	580
DL-RW		OOT	17 879	5 747	1 555	3 523	2 344
DL-WL		56 877	1 331	577	312	906	812
SE-RW		3 901	1 887	2 464	984	3 359	2 858
SE-WL		339	268	193	218	673	820

(The leftmost rows RG-RW through SE-WL are labeled "Baselines".)

(b) Running times in ms of *Approx-s*.

Kernel	Data set					
	MIT	HIGHSCHOOL	INFECTIOUS	TUMBLR	DBLP	FACEBOOK
Approx-50	5	16	19	66	264	443
Approx-100	6	17	20	67	267	458
Approx-200	7	19	22	71	268	466

Table 3: Running times and accuracy in percent for the synthetic large data set.

|	*NTWL*	*SEWL*	*TGK-⋆*	*TGK-∧*	*Approx-100*	*Approx-1000*	*Approx-10000*
Run. time	10.21 s	36.34 s	19.02 s	3.6 s	13 ms	42 ms	240 ms
Acc.	$36.32_{\pm 2.7}$	$90.31_{\pm 1.3}$	$91.61_{\pm 0.6}$	$91.73_{\pm 0.6}$	$87.91_{\pm 1.4}$	$91.63_{\pm 0.5}$	$91.73_{\pm 0.4}$

For the first modified task and 50% to 80% missing data, *TGK-∧* has the highest mean accuracy. Our other kernels are on par with the best performing baseline *SE-RW* and are significantly better than the other baselines. In the case of the second modified task, our kernels beat *SE-RW*, *SE-WL*, and *DL-WL* for all percentages, often with significant gaps in the accuracies, e.g., for the WL-kernels. Our kernels have the highest mean accuracy for 20% and from 40% to 80%. The accuracies are on par with the baselines for the remaining percentages. In conclusion, our new temporal graphlet kernels often achieve higher mean accuracy than the kernels introduced in [18], especially in cases where large percentages of information are not available. Hence, our kernels are competitive and often better in the task of classification under missing information.

6 Conclusion

We introduced a temporal graphlet kernel for classifying dissemination processes on temporal graphs. Our experimental evaluation showed that our approach
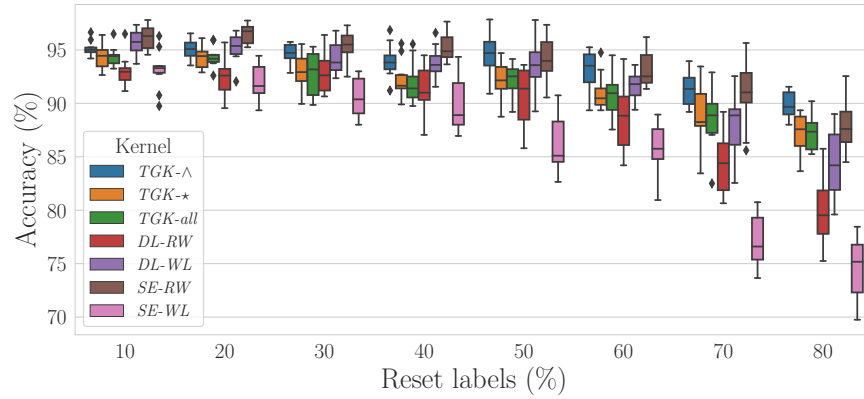
(a) Modified first classification task.
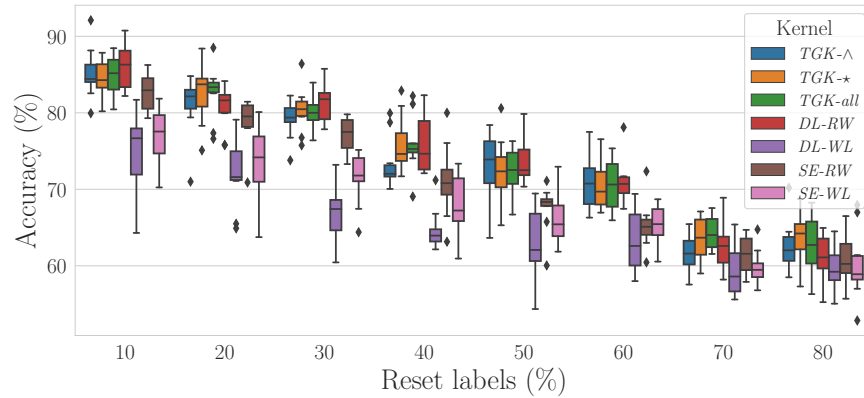
(b) Modified second classification task.

Fig. 4: Results for the INFECTIOUS data set under incomplete data.

beats the state-of-art baselines in most data sets and is on par for the remaining data sets. Our approximative kernel has high accuracy while running significantly faster on large data sets. The wedge-based kernel has an excellent trade-off between running time and accuracy. Finally, our kernels have high accuracy even when information about the dissemination is missing, especially when a majority of the information is unknown.

References

1. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. science **286**(5439), 509–512 (1999)
2. Braha, D., Bar-Yam, Y.: Time-Dependent Complex Networks: Dynamic Centrality, Dynamic Motifs, and Cycles of Social Interactions, pp. 39–50. Springer Berlin Heidelberg, Berlin, Heidelberg (2009)
3. Brauer, F.: Compartmental models in epidemiology. In: Mathematical epidemiology, pp. 19–79. Springer (2008)
4. Brouwer, A.F., Eisenberg, J.N., Pomeroy, C.D., Shulman, L.M., Hindiyeh, M., Manor, Y., Grotto, I., Koopman, J.S., Eisenberg, M.C.: Epidemiology of the silent polio outbreak in rahat, israel, based on modeling of environmental surveillance data. Proc. of the National Academy of Sciences **115**(45), E10625–E10633 (2018)
5. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology **2**, 27:1–27:27 (2011)
6. Eagle, N., Pentland, A.S.: Reality mining: Sensing complex social systems. Personal and Ubiquitous Computing **10**(4), 255–268 (2006)
7. Holme, P.: Modern temporal network theory: A colloquium. The European Physical Journal B **88**(9), 234 (2015)
8. Isella, L., Stehlé, J., Barrat, A., Cattuto, C., Pinton, J.F., Van den Broeck, W.: What's in a crowd? Analysis of face-to-face behavioral networks. Journal of Theoretical Biology **271**(1), 166–180 (2011)
9. Kaslow, R.A., Stanberry, L.R., Le Duc, J.W.: Viral infections of humans: epidemiology and control. Springer (2014)
10. Kriege, N.M., Johansson, F.D., Morris, C.: A survey on graph kernels. Applied Network Science **5**(1), 1–42 (2020)
11. Leskovec, J., Backstrom, L., Kleinberg, J.: Meme-tracking and the dynamics of the news cycle. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 497–506 (2009)
12. Liu, P., Benson, A.R., Charikar, M.: Sampling methods for counting temporal motifs. In: Proceedings of the twelfth ACM international conference on web search and data mining. pp. 294–302 (2019)
13. Liu, P., Guarrasi, V., Sariyuce, A.E.: Temporal network motifs: Models, limitations, evaluation. IEEE Transactions on Knowledge and Data Engineering (2021)
14. Mackey, P., Porterfield, K., Fitzhenry, E., Choudhury, S., Chin, G.: A chronological edge-driven approach to temporal subgraph isomorphism. In: 2018 IEEE international conference on big data (big data). pp. 3972–3979. IEEE (2018)
15. Masuda, N., Holme, P.: Detecting sequences of system states in temporal networks. Scientific Reports **9**(1), 1–11 (2019)
16. Murayama, T., Wakamiya, S., Aramaki, E., Kobayashi, R.: Modeling the spread of fake news on twitter. Plos one **16**(4), e0250419 (2021)
17. Oettershagen, L., Kriege, N.M., Morris, C., Mutzel, P.: Classifying dissemination processes in temporal graphs. Big Data **8**(5), 363–378 (2020)
18. Oettershagen, L., Kriege, N.M., Morris, C., Mutzel, P.: Temporal graph kernels for classifying dissemination processes. In: Proceedings of the 2020 SIAM International Conference on Data Mining. pp. 496–504. SIAM (2020)
19. Palladino, R., Bollon, J., Ragazzoni, L., Barone-Adesi, F.: Excess deaths and hospital admissions for covid-19 due to a late implementation of the lockdown in italy. Intl. Journal of Environmental Research and Public Health **17**(16), 5644 (2020)

20. Paranjape, A., Benson, A.R., Leskovec, J.: Motifs in temporal networks. In: Proc. of the ACM Intl. Conf. on Web Search and Data Mining. pp. 601–610 (2017)
21. Sarpe, I., Vandin, F.: oden: Simultaneous approximation of multiple motif counts in large temporal networks. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management. pp. 1568–1577 (2021)
22. Seshadhri, C., Pinar, A., Kolda, T.G.: Wedge sampling for computing clustering coefficients and triangle counts on large graphs. Statistical Analysis and Data Mining: The ASA Data Science Journal **7**(4), 294–307 (2014)
23. Shervashidze, N., Vishwanathan, S., Petri, T., Mehlhorn, K., Borgwardt, K.: Efficient graphlet kernels for large graph comparison. In: Artificial intelligence and statistics. pp. 488–495. PMLR (2009)
24. Shu, K., Sliva, A., Wang, S., Tang, J., Liu, H.: Fake news detection on social media: A data mining perspective. ACM SIGKDD explorations newsl. **19**(1), 22–36 (2017)
25. Tortorella, D., Micheli, A.: Dynamic graph echo state networks. In: Proceedings of the 29th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN). pp. 99–104 (2021)
26. Užupytė, R., Wit, E.C.: Test for triadic closure and triadic protection in temporal relational event data. Social Network Analysis and Mining **10**(1), 1–12 (2020)
27. Viswanath, B., Mislove, A., Cha, M., Gummadi, K.P.: On the evolution of user interaction in facebook. In: Proc. ACM Works. on Onl. Soc. Net. pp. 37–42 (2009)
28. Vosoughi, S., Roy, D., Aral, S.: The spread of true and false news online. Science **359**(6380), 1146–1151 (2018)
29. Wale, N., Watson, I.A., Karypis, G.: Comparison of descriptor spaces for chemical compound retrieval and classification. Knowl. and Inf. Sys. **14**(3), 347–375 (2008)
30. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Yu, P.S.: A comprehensive survey on graph neural networks. IEEE Trans. Neural Net. Learn. Syst. **32**(1), 4–24 (2021)

A Appendix

A.1 Related Work

Graph kernels have been studied extensively in the past 15 years; see [10] for an overview. Important approaches include random walk and shortest path kernels [32,35,42], as well as the Weisfeiler-Leman subtree kernel [48,57]. Further recent works focus on assignment-based approaches [40,50], spectral techniques [39], and graph decomposition [49]. Most kernels are designed for static graphs, with few exceptions considering different aspects of temporal graphs. Li et al. [43] present a family of algorithms that efficiently recomputes the random walk kernel when graphs are modified. Paaßen et al. [53] use graph kernels for predicting the next graph in a dynamically changing series of graphs. Similarly, Anil et al. [31] propose spectral graph kernels to predict the evolution of social networks.

Closely related to our work are [17] and [18], which also discuss the classification of dissemination in temporal networks. To this end, the authors propose transforming the labeled temporal graphs into labeled static graphs. In [18], they compare three transformations that differ in loss of information and size of the resulting static graph. The authors combine their transformations with the random walk and the Weisfeiler-Leman subtree kernels. The kernel-based approach

with the most expressive transformation achieves state-of-the-art classification accuracy but does not scale to large graphs. In [17], the transformations have been used with graph neural networks, but the overall performance was worse compared to graph kernels. Another closely related work uses dynamic echo state networks [25]. While the approach does not outperform the accuracy of the transformation-based temporal graph kernels, an advantage is its ability to process graphs in an online setting requiring less space. Furthermore, various principled extensions of graph neural networks to the temporal domain exist; see this recent survey [38] and references therein.

Ribeiro et al. [55] give an overview of motifs in static networks. General overviews of temporal graphs can be found in [7,47]. Various attempts to extend the concept of motifs to evolving graphs have been proposed [37,13]. Paranjape et al. [20] define temporal motifs as induced subgraphs on sequences of temporal edges. They introduce an algorithm for counting general temporal graphlets and efficient variants for specific small graphlets. We generalize these algorithms for counting labeled temporal graphlets. Mackey et al. [14] introduced an enumeration algorithm for temporal graphlets. Several recent works discuss the sampling of temporal motifs [12,13,21].

A.2   Omitted Proofs

Proof (Proof of Lemma 1).   For each of the ℓ temporal edges (u, v, t) in the sequence of temporal edges, we consider the number of different labels for the first node u at time t and the second node v at time $t+1$, leading to L^2 possible combinations. Having ℓ edges leads to $L^{2\ell}$ possible sequences in total.   □

Proof (Proof of Lemma 2).   We use an edge labeling function ξ and for each edge $e = (u, v, t) \in \mathcal{E}$, we construct a new label $\xi(e)$ by the concatenation of the time-dependent node labels $l(u, t)\#l(v, t+1)$, where $\#$ is an additional symbol not in Σ. The edge label sequence is then $l_e(g) = (\xi(e_1), \ldots, \xi(e_\ell))$. Hence, there is a one-to-one mapping between $l(g)$ and $l_e(g)$ for all labeled temporal graphlets g.   □

Proof (Proof of Theorem 1).   If the temporal graph is given in incident list representation, i.e., each vertex has a list of incident temporal edges, and the degree $d(v)$ can be determined in constant time. Then, computing the number of wedges and the probabilities p_v is possible in $\mathcal{O}(|V|)$ time. Due to Lemma 1 the initialization takes $\mathcal{O}(|\Sigma|^2)$ time, and each operation in the for loop is possible in constant time.   □

Proof (Proof of Theorem 2).   First, by an application of the Hoeffding bound together with the union bound, it follows that by setting

$$s = \frac{\log(2 \cdot |\mathbb{G}| \cdot W/\delta)}{2\varepsilon^2},$$

it holds that

$$P\left(|\phi_{\mathrm{TG}}(\mathcal{G}_i)_j - \widetilde{\phi}_{\mathrm{TG}}(\mathcal{G}_i)_j| \leq \varepsilon\right) \geq 1 - \delta,$$

Table 4: Statistics and properties of the data sets.

Property	Data set					
	MIT	HIGHSCHOOL	INFECTIOUS	TUMBLR	DBLP	FACEBOOK
# Graphs	97	180	200	373	755	995
Avg. $|V|$	20	52.3	50	53.1	52.9	95.7
Min $|\mathcal{E}|$	63	151	109	48	105	88
Max $|\mathcal{E}|$	3 363	589	505	190	275	181
Avg. $|\mathcal{E}|$	734.6	272.4	229.9	99.9	160.0	134.5
Avg. max $d(v)$	712.1	96.8	46.1	25.1	27.1	21.2

for all $1 \leq j \leq W$, and all temporal graphs $\mathcal{G}_i$ in $\mathcal{G}$. Let $\mathcal{G}_1$ and $\mathcal{G}_2$ in $\mathcal{G}$, then

$$\widetilde{k}_{\text{TG}}(\mathcal{G}_1, \mathcal{G}_2) = \left\langle \widetilde{\phi}_{\text{TG}}(\mathcal{G}_1), \widetilde{\phi}_{\text{TG}}(\mathcal{G}_2) \right\rangle$$

$$\leq \sum_{i=1}^{W} (\phi_{\text{TG}}(\mathcal{G}_1)_i \cdot \phi_{\text{TG}}(\mathcal{G}_2)_i) + \varepsilon \cdot \sum_{i=1}^{W} (\phi_{\text{TG}}(\mathcal{G}_1)_i + \phi_{\text{TG}}(\mathcal{G}_2)_i) + \sum_{i=1}^{W} \varepsilon^2$$

$$\leq k_{\text{TG}}(\mathcal{G}_1, \mathcal{G}_2) + 2W \cdot \varepsilon + W \cdot \varepsilon.$$

The last inequality holds because the components of $\phi_{\text{TG}}(\cdot)$ are in $[0, 1]$. Finally, by setting $\varepsilon = \lambda/W$ the result follows. □

A.3 Data Sets

- INFECTIOUS and HIGHSCHOOL: Contain face-to-face contacts between visitors of the exhibition *Infectious: Stay Away* [8] and interactions between high school students.
- MIT: A temporal graph of contacts between students [6].
- FACEBOOK and TUMBLR: The first graph is a subset of the activity of a Facebook community over three months [27]. The Tumblr graph contains quoting between Tumblr users [11].
- DBLP: A subset of the Digital Bibliography & Library Project (DBLP) database representing temporal co-author graphs. Nodes represent authors, and the time stamp of an edge is the year of a joint publication.

The authors of [18] proposed three classification tasks and provided the labeled data sets.[5] Each classification data set consists of a set of graphs belonging to two classes. Table 4 shows the statistics of the data sets. The first classification task is the discrimination of observations of a dissemination process and random labeling. The second classification task aims to discriminate temporal graphs that were subject to two different dissemination processes differing in the infection probability. The third classification task is the classification under incomplete information. The reasons for missing information about the dissemination process are manifold, e.g., non-symptomatic infections, people hiding their infection for various reasons, or spreading fake news recorded at known spreaders only.

---

[5] http://graphlearning.io/

Based on the INFECTIOUS data set and the first two classification tasks, for each graph, the labels of $\{10\%, \dots, 80\%\}$ of randomly chosen infected nodes are set back to non-infected. This process is repeated ten times resulting in 80 data sets for each of the two classification tasks.

A.4 Experimental Protocol

We computed the normalized Gram matrices and report the classification accuracies obtained with the C-SVM implementation of LIBSVM [5], using 10-fold cross-validation. The C-parameter was selected from $\{10^{-3}, 10^{-2}, \dots, 10^2, 10^3\}$ by 10-fold cross-validation on the training folds. We repeated each 10-fold cross-validation ten times with different random folds and report average accuracies and standard deviations. The time window for the temporal graphlets ($\delta \in \{10^i \mid 1 \le i \le 4\}$) was selected by fold-wise 10-fold cross-validation. Likewise, the number of steps of the random walk kernel ($k \in \{1, \dots, 5\}$) and the number of iterations of the Weisfeiler-Leman subtree kernel ($h \in \{1, \dots, 5\}$) were selected by fold-wise 10-fold cross-validation. All experiments ran on a computer cluster. Each experiment had an exclusive node with an Intel(R) Xeon(R) Gold 6130 CPU @ 2.10GHz and 192 GB of RAM. The time limit was one hour. Analogous to [17] and to compare the running times, we set the walk length of DL-RW to $k = 2$ and the number of iterations of DL-WL to $h = 2$.

A.5 Running Times for the First Classification Task

Table 5 shows the running times for the first classification.

Table 5: Running times in ms for the first classification task. Random walk length $k = 3$ ($k = 2$ for DL-RW), number of iterations of WL $h = 3$ ($h = 2$ for DL-WL). OOT—Computation did not finish within the time limit.

Kernel		Data set					
		MIT	HIGHSCHOOL	INFECTIOUS	TUMBLR	DBLP	FACEBOOK
TGK-$\wedge$	$\delta = 10$	291	35	40	68	297	443
TGK-$\star$	$\delta = 10$	100	180	191	409	1365	2366
TGK-all	$\delta = 10$	202	793	1407	439	3692	1336
TGK-$\wedge$	$\delta = 10^3$	501	59	42	70	290	454
TGK-$\star$	$\delta = 10^3$	100	180	195	414	1363	2425
TGK-all	$\delta = 10^3$	193	619	1326	374	3540	1234
RG-RW		793	9272	9153	5040	14428	9700
RG-WL		18	61	76	109	294	564
DL-RW		OOT	17871	5683	1552	3532	2322
DL-WL		56762	1323	574	306	884	787
SE-RW		3899	1833	2461	973	3370	2810
SE-WL		338	264	189	209	631	787

(Baselines: RG-RW, RG-WL, DL-RW, DL-WL, SE-RW, SE-WL)

A.6 Comparison to Graph Neural Networks

We present the classification results using the neural approaches presented in [17] and [25]. We compare them to our temporal graphlet kernels. The GNN model

in [17] is based on the GIN architecture introduced in [59]. The final GNN layer is fed into a four-layer MLP followed by a softmax function. Alternatively, a Jumping Knowledge (JK) approach [60] to combine the outputs of all layers is used. The neural networks were trained for 200 epochs using the Adam optimizer with cross-entropy loss. For each of our three transformations introduced in [18], the following GNNs were trained on the transformed data sets:

1. *RG-GIN* and *RG-JK* for the reduced graph,
2. *DL-GIN* and *DL-JK* for the directed line graph expansion, and
3. *SE-GIN* and *SE-JK* for the static expansion.

The GNNs in [17] were implemented using the *PyTorch Geometric* library [33].
    DynGESN is the dynamic echo state network introduced in [25] as an adaption of static graph echo state networks (see [34]) for temporal graphs. The approach is implemented in Matlab[6].

Results Table 6a and Table 6b show the results for our kernels and the GNNs. Our temporal graphlet kernels beat the neural approaches for all but one data set. In the case of the second classification task, our kernel accuracies are often significantly higher compared to the GNN accuracies.

Appendix References

31. Anil, A., Sett, N., Singh, S.R.: Modeling evolution of a social network using temporalgraph kernels. In: Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval. pp. 1051–1054 (2014)
32. Borgwardt, K.M., Kriegel, H.P.: Shortest-path kernels on graphs. In: Fifth IEEE International Conference on Data Mining. pp. 8–pp. IEEE (2005)
33. Fey, M., Lenssen, J.E.: Fast graph representation learning with PyTorch Geometric. In: ICLR Workshop on Representation Learning on Graphs and Manifolds (2019)
34. Gallicchio, C., Micheli, A.: Graph echo state networks. In: The 2010 international joint conference on neural networks (IJCNN). pp. 1–8. IEEE (2010)
35. Gärtner, T., Flach, P., Wrobel, S.: On graph kernels: Hardness results and efficient alternatives. In: Learning Theory and Kernel Mach., pp. 129–143. Springer (2003)
36. Holme, P.: Modern temporal network theory: A colloquium. The European Physical Journal B **88**(9), 234 (2015)
37. Jazayeri, A., Yang, C.C.: Motif discovery algorithms in static and temporal networks: A survey. CoRR **abs/2005.09721** (2020)
38. Kazemi, S.M., Goel, R., Jain, K., Kobyzev, I., Sethi, A., Forsyth, P., Poupart, P.: Representation learning for dynamic graphs: A survey. J. Mach. Learn. Res. **21**, 70:1–70:73 (2020)
39. Kondor, R., Pan, H.: The multiscale Laplacian graph kernel. In: Advances in Neural Information Processing Systems. pp. 2990–2998 (2016)
40. Kriege, N.M., Giscard, P.L., Wilson, R.: On valid optimal assignment kernels and applications to graph classification. In: Advances in Neural Information Processing Systems. pp. 1623–1631 (2016)

---

[6] https://github.com/dtortorella/dyngraphesn

Table 6: Classification accuracy in percent and standard deviation for the first and second classification task. For each data set, we highlight the highest accuracy in bold. OOM—Out of memory.

(a) Classification accuracy for the first classification task.

	Kernel	Data set					
		MIT	HIGHSCHOOL	INFECTIOUS	TUMBLR	DBLP	FACEBOOK
	TGK-∧	**93.45** $\pm_{1.8}$	**98.00** $\pm_{0.9}$	**98.30** $\pm_{0.4}$	93.19 $\pm_{0.7}$	**98.47** $\pm_{0.1}$	95.21 $\pm_{0.2}$
	TGK-⋆	87.02 $\pm_{1.6}$	96.50 $\pm_{0.9}$	98.05 $\pm_{0.4}$	93.19 $\pm_{0.8}$	98.05 $\pm_{0.2}$	**95.32** $\pm_{0.2}$
	TGK-all	87.46 $\pm_{0.9}$	97.17 $\pm_{0.8}$	97.90 $\pm_{0.4}$	**93.57** $\pm_{0.8}$	98.12 $\pm_{0.1}$	94.69 $\pm_{0.3}$
Baselines	*RG-GIN*	50.65 $\pm_{4.2}$	51.11 $\pm_{2.5}$	58.20 $\pm_{4.0}$	72.63 $\pm_{1.8}$	86.36 $\pm_{0.9}$	89.54 $\pm_{0.8}$
	RG-JK	50.74 $\pm_{3.1}$	50.83 $\pm_{4.9}$	47.85 $\pm_{2.7}$	69.14 $\pm_{3.6}$	86.43 $\pm_{0.7}$	87.27 $\pm_{0.6}$
	DL-GIN	OOM	88.67 $\pm_{2.1}$	92.85 $\pm_{1.7}$	90.39 $\pm_{1.4}$	97.72 $\pm_{0.4}$	94.29 $\pm_{0.2}$
	DL-JK	OOM	86.22 $\pm_{2.6}$	91.55 $\pm_{2.3}$	89.30 $\pm_{1.5}$	97.57 $\pm_{0.3}$	93.05 $\pm_{0.6}$
	SE-GIN	75.98 $\pm_{3.7}$	92.28 $\pm_{1.2}$	93.10 $\pm_{1.9}$	92.78 $\pm_{1.1}$	97.87 $\pm_{0.3}$	94.72 $\pm_{0.5}$
	SE-JK	75.37 $\pm_{3.6}$	92.33 $\pm_{2.7}$	93.50 $\pm_{1.9}$	92.30 $\pm_{0.9}$	97.14 $\pm_{0.9}$	95.02 $\pm_{0.6}$
	DynGESN	69.70 $\pm_{9.2}$	94.40 $\pm_{5.3}$	94.70 $\pm_{5.3}$	93.30 $\pm_{3.9}$	97.70 $\pm_{1.8}$	93.00 $\pm_{2.5}$

(b) Classification accuracy for the second classification task.

	Kernel	Data set					
		MIT	HIGHSCHOOL	INFECTIOUS	TUMBLR	DBLP	FACEBOOK
	TGK-∧	68.52 $\pm_{3.5}$	93.83 $\pm_{0.8}$	89.65 $\pm_{0.8}$	**79.06** $\pm_{0.7}$	83.76 $\pm_{0.5}$	76.63 $\pm_{0.3}$
	TGK-⋆	77.03 $\pm_{3.7}$	**95.33** $\pm_{0.8}$	**90.55** $\pm_{1.6}$	78.13 $\pm_{1.1}$	85.42 $\pm_{0.8}$	82.12 $\pm_{0.3}$
	TGK-all	**78.43** $\pm_{4.2}$	94.22 $\pm_{1.3}$	90.40 $\pm_{0.9}$	76.57 $\pm_{1.0}$	**85.84** $\pm_{0.6}$	81.13 $\pm_{0.6}$
Baselines	*RG-GIN*	53.80 $\pm_{16.0}$	53.61 $\pm_{3.5}$	51.80 $\pm_{4.2}$	64.70 $\pm_{2.4}$	60.24 $\pm_{1.8}$	67.75 $\pm_{1.3}$
	RG-JK	51.80 $\pm_{9.7}$	54.61 $\pm_{2.9}$	52.60 $\pm_{3.2}$	65.50 $\pm_{2.6}$	61.00 $\pm_{1.1}$	67.75 $\pm_{1.0}$
	DL-GIN	OOM	89.11 $\pm_{2.0}$	80.60 $\pm_{2.2}$	75.45 $\pm_{2.4}$	80.05 $\pm_{1.1}$	**83.16** $\pm_{0.9}$
	DL-JK	OOM	85.00 $\pm_{2.8}$	75.70 $\pm_{3.5}$	73.10 $\pm_{1.8}$	79.98 $\pm_{1.3}$	82.26 $\pm_{1.1}$
	SE-GIN	51.40 $\pm_{11.1}$	85.88 $\pm_{2.1}$	75.05 $\pm_{3.4}$	73.23 $\pm_{1.7}$	80.72 $\pm_{1.1}$	82.21 $\pm_{0.9}$
	SE-JK	51.40 $\pm_{10.9}$	82.44 $\pm_{2.0}$	74.25 $\pm_{2.4}$	74.55 $\pm_{1.4}$	81.18 $\pm_{1.0}$	80.46 $\pm_{0.9}$
	DynGESN	63.30 $\pm_{11.0}$	92.80 $\pm_{5.2}$	80.60 $\pm_{9.1}$	76.80 $\pm_{6.2}$	74.30 $\pm_{4.7}$	76.10 $\pm_{3.9}$

41. Kriege, N.M., Johansson, F.D., Morris, C.: A survey on graph kernels. Applied Network Science **5**(1), 1–42 (2020)

42. Kriege, N.M., Neumann, M., Morris, C., Kersting, K., Mutzel, P.: A unifying view of explicit and implicit feature maps of graph kernels. Data Mining and Knowledge Discovery **33**(6), 1505–1547 (2019)

43. Li, L., Tong, H., Xiao, Y., Fan, W.: Cheetah: Fast graph kernel tracking on dynamic graphs. In: Proceedings of the 2015 SIAM International Conference on Data Mining. pp. 280–288. SIAM (2015)

44. Liu, P., Benson, A.R., Charikar, M.: Sampling methods for counting temporal motifs. In: Proceedings of the twelfth ACM international conference on web search and data mining. pp. 294–302 (2019)

45. Liu, P., Guarrasi, V., Sariyuce, A.E.: Temporal network motifs: Models, limitations, evaluation. IEEE Transactions on Knowledge and Data Engineering (2021)

46. Mackey, P., Porterfield, K., Fitzhenry, E., Choudhury, S., Chin, G.: A chronological edge-driven approach to temporal subgraph isomorphism. In: 2018 IEEE international conference on big data (big data). pp. 3972–3979. IEEE (2018)

47. Michail, O., Spirakis, P.G.: Elements of the theory of dynamic networks. Communications of the ACM **61**(2), 72–72 (2018)

48. Morris, C., Kersting, K., Mutzel, P.: Glocalized Weisfeiler-Lehman graph kernels: Global-local feature maps of graphs. In: 2017 IEEE International Conference on Data Mining. pp. 327–336. IEEE (2017)
49. Nikolentzos, G., Meladianos, P., Limnios, S., Vazirgiannis, M.: A degeneracy framework for graph similarity. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence. pp. 2595–2601 (2018)
50. Nikolentzos, G., Meladianos, P., Vazirgiannis, M.: Matching node embeddings for graph similarity. In: Thirty-First AAAI Conference on Artificial Intelligence (2017)
51. Oettershagen, L., Kriege, N.M., Morris, C., Mutzel, P.: Classifying dissemination processes in temporal graphs. Big Data **8**(5), 363–378 (2020)
52. Oettershagen, L., Kriege, N.M., Morris, C., Mutzel, P.: Temporal graph kernels for classifying dissemination processes. In: Proceedings of the 2020 SIAM International Conference on Data Mining. pp. 496–504. SIAM (2020)
53. Paaßen, B., Göpfert, C., Hammer, B.: Time series prediction for graphs in kernel and dissimilarity spaces. Neural Processing Letters **48**(2), 669–689 (2018)
54. Paranjape, A., Benson, A.R., Leskovec, J.: Motifs in temporal networks. In: Proc. of the ACM Intl. Conf. on Web Search and Data Mining. pp. 601–610 (2017)
55. Ribeiro, P., Paredes, P., Silva, M.E., Aparicio, D., Silva, F.: A survey on subgraph counting: Concepts, algorithms, and applications to network motifs and graphlets. ACM Computing Surveys (CSUR) **54**(2), 1–36 (2021)
56. Sarpe, I., Vandin, F.: oden: Simultaneous approximation of multiple motif counts in large temporal networks. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management. pp. 1568–1577 (2021)
57. Shervashidze, N., Schweitzer, P., Van Leeuwen, E.J., Mehlhorn, K., Borgwardt, K.M.: Weisfeiler-Lehman graph kernels. Journal of Machine Learning Research **12**(77), 2539–2561 (2011)
58. Tortorella, D., Micheli, A.: Dynamic graph echo state networks. In: Proceedings of the 29th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN). pp. 99–104 (2021)
59. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? arXiv preprint arXiv:1810.00826 (2018)
60. Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K., Jegelka, S.: Representation learning on graphs with jumping knowledge networks. In: 35th International Conference on Machine Learning. pp. 5449–5458 (2018)