Change Point Detection in Evolving Graph using Martingale

Shen-Shyang Ho
Rowan University
Glassboro, New Jersey, USA
hos@rowan.edu

Tarun Teja Kairamkonda
Rowan University
Glassboro, New Jersey, USA
kairam42@rowan.edu

ABSTRACT

Many real world applications are modeled as groups of interacting entities using dynamic graphs due to the nature of the application domains such as sensor network, social network, computer network, urban traffic network, and power grid. A change in the evolving graph or a change in the behavior of the interacting entities can be viewed as a change in the graph distribution or graph generating process.

We describe our proposed change-point detection approach for an evolving graph by monitoring the martingale values derived from the extracted graph features. We demonstrate empirically that the feature representation that encodes a graph property (or characteristics) is critical in the performance of the martingale test in detecting the change-point using two synthetic (random topology, scale-free) graph generators and a real-world dataset. On the other hand, we demonstrate that the theoretical false positive bound for a martingale change-detection is preserved even when different feature representations are used. We further discuss the use of multiple martingale tests on different graph features allowing one to identify which graph property changes and provide explanations to the change-point detected.

CCS CONCEPTS

• **Mathematics of computing** → **Probabilistic algorithms**; **Nonparametric statistics**; • **Computing methodologies** → **Machine learning algorithms**;

KEYWORDS

Change Point Detection, Graph Embeddings, Dynamic Graphs, Martingale

ACM Reference Format:
Shen-Shyang Ho and Tarun Teja Kairamkonda . 2024. Change Point Detection in Evolving Graph using Martingale. In *The 39th ACM/SIGAPP Symposium on Applied Computing (SAC '24), April 8–12, 2024, Avila, Spain.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3605098.3635979

1 INTRODUCTION

Many real world applications exist in dynamic environments which require the monitoring of real-time processes. Many such processes are modeled as groups of interacting entities using dynamic graphs due to the nature of the application domains such as sensor network, social network, computer network, urban traffic network, and power grid. One particular task of interest is anomaly detection (global or local) in an evolving graph [6, 22, 29, 30]. A more generic task is the change-point detection problem such that a change may not be an anomaly. Instead, it is a change in the process or a change in the behavior of the interacting entities [18, 20]. Such a change-point in time is a consequence of a change in the graph distribution.

In this paper, we focus on this particular change-point detection problem. The change point detection problem involves the identification of a time instance when there is a deviation from the current data generation model. We formally state the problem statement similar to [4]:

Given a sequence of observed random variables $(x_k)_{1\le k\le n}$ *with conditional density* $p_\theta(x_k|x_{k-1},\ldots,x_1)$. *Before change occurs, the conditional density parameter* $\theta=\theta_0$. *After the change,* $\theta=\theta_1$. *The online problem is to "detect the occurrence of the change as soon as possible, with a fixed rate of false alarms" [4].*

For the change point detection problem for an evolving graph, we observe a sequence of graph snapshots $(\mathcal{G}_k)_{1\le k\le n}$, one snapshot at a time. One assumes that at current time instance $t=n$. At some time instance $t=k$, the conditional density $p_\theta(\mathcal{G}_k|\mathcal{G}_{k-1},\ldots,\mathcal{G}_1)$ describes some properties or characteristics (or features) that can be extracted from snapshots of the evolving graph. Hence, to model a graph distribution or graph generation process, one needs to consider the question "which property or characteristic is used to model the evolving graph?" This, in turn, affects the performance of a change-point detection solution implementation.

In this paper, we describe our proposed change-point detection approach for an evolving graph by monitoring the martingale values derived using the features extracted from the snapshots of the evolving graph. Moreover, we demonstrate empirically that the feature representation that encodes a graph property (or characteristics) is critical in the performance of the martingale test in detecting the change-point. This shows that additional factors need to be taken into consideration when using a martingale test for change-detection problem for an evolving graph compared to its previous proposed uses in conventional data types and models [12, 13]. On the other hand, we demonstrate that the desirable theoretical false positive bound for a martingale change-detection is preserved even when different feature representations are used. We further discuss the use of multiple martingale tests on different graph features allowing one to identify which graph property changes and provide explanations to the change-point detected.

The paper is organized as follows. We provide a brief review of existing change detection methods for evolving graph in Section 2. In Section 3, we describe the modeling graph property distribution using graph features and embeddings. In Section 4, we describe our proposed evolving graph change point detection method using

SAC '24, April 8–12, 2024, Avila, Spain

ACM ISBN 979-8-4007-0243-3/24/04…$15.00
https://doi.org/10.1145/3605098.3635979

martingale in detail. We describe how the martingale methods can run in parallel to provide explanation to change-points detected. In Section 5, we present and discuss our results on synthetic data and a demonstration using a real-world dataset.

2 RELATED WORK

One of the earliest change-point detection approaches, called the sequential probability ratio test (SPRT) [26], was proposed to monitor the manufacturing process of military and naval equipment during World War II. Many early techniques were proposed by mathematicians and statisticians and they can be found in [4]. Some of these conventional statistical process control and monitoring techniques such as cumulative sum (CUSUM) control chart [19] and exponentially weighted moving average [21] can be applied to modern applications related to social networks [18].

Many recently proposed change-point detection approaches' for evolving graphs have the objective of identifying anomalies (local) in a graph time-series in node attributes or global (snapshots) graph anomalies. Aggarwal et al. [1] proposed a thresholding method for dynamic changes in node labels for real-time detection of changes in a node classification task setting. Koutra et al. [16] solves the problem of change detection in graph as the lack of similarity based on node/edge characteristics between two graphs. Wang et al. [27] proposed a method considering a graph sequence given a dissimilarity scoring function, and a threshold to detect events or changes when the dissimilarity of two consecutive snapshots is above the threshold in the latent space for the evolving graph.

There are research focusing on the learning of graph similarity function to better compare graphs to identify changes. Sulem et al. [23] proposed the use of a Siamese graph neural network to learn a graph similarity function to compare the current graph and its recent history. The main issues with the approach is the need to train and learn the similarity function and it is not clear how the threshold is chosen for the change-point detection. Recently, Flossdorf et. al [8] utilized multiple sets of metrics for online monitoring of evolving graphs. They described guidelines on how to choose a suitable metric set together with the choice of a meaningful parametric or non-parametric control chart. However, they showed that the performance of their approach is application dependent.

There are research focusing on the learning of graph embeddings to better represent graphs to identify changes. Grattarola et al. [9] proposed two change detection tests for evolving graphs by using graph embeddings learned using an autoencoder to represent graph instances over time on the (non-Euclidean) constant-curvature Riemannian manifolds (CCMs) which enable better computation of the metric geodesic distances between graphs. Ferrari and Richard [7] proposed a non-parametric approach to detect a change point located on an unknown cluster in an evolving graph by monitoring the nodes using graph-filtered signals that take into consideration the graph topology. Huang et al. [14] proposed the use of the spectrum of the Laplacian matrix of the graph to capture the temporal relationship and compare graph snapshots across time.

Recently, Xie et al. [28] proposed the multi-view feature interpretable change point detection method (MICPD) based on a vector autoregressive (VAR) model to reduce high-dimensional network data into a low-dimensional representation for tracking the change points for multiple targets. Note that the multi-view approach is to find multiple change points in the graph structure using a a single multidimensional vector encoding multiple feature time series.

3 MODELING AND MONITORING FEATURE DISTRIBUTION FOR EVOLVING GRAPHS

In this paper, an evolving graph is represented by a sequence of static graph snapshots $(\mathcal{G}_k)_{1\leq k\leq n}$. Each graph snapshot is observed one after another, and only once. As a snapshot is observed, graph properties and features are extracted from it.

We define the conditional density of the graph feature of interest by

$$p_\theta(f(\mathcal{G}_k)|f(\mathcal{G}_{k-1}),...,f(\mathcal{G}_1))$$

such that $f : G \to \mathcal{F}$ is a transformation for an input graph G and returning an output vector $\mathcal{F}$ (or a matrix, in general). For simplicity, we assume

$$p_\theta(f(\mathcal{G}_k)|f(\mathcal{G}_{k-1}),...,f(\mathcal{G}_1)) = p_\theta(f(\mathcal{G}))$$

is the density function for some feature vector random variables.

The transformation f represents a great variety of feature extraction or embedding methods. It could be as simple as degree centrality calculation, SVD (Singular value decomposition), and spectral embedding. It can also be more computationally expensive static graph embeddings such as Node2Vec [10] and GraphSAGE [11] or dynamic graph embeddings [3].

We briefly describe the five graph features / embeddings and their graph properties used in our empirical evaluation:

(1) **Degree Centrality**: It is a vector with the number of elements equals to the number of nodes. Each element is the fraction of nodes a particular node is connected to. The values are normalized by dividing by the maximum possible degree in the graph.
(2) **Singular value decomposition (SVD) of the adjacency matrix**: The embedding ensures that nodes that are closed to each other based on the adjacency matrix will be closed in a d-dimensional space such that $d < |V|$, the number of nodes in the graph. This embedding is sensitive to the node order in the adjacency matrix.
(3) **Spectral Embedding:** Eigendecomposition is performed on the graph Laplacian. Similar to embedding using SVD on adjacency matrix, nodes that are closed to each other in a graph are closed in a d-dimensional space such that $d < |V|$, the number of nodes in the graph. This embedding is independent of the node order in the Laplacian and hence, hidden patterns in the original space are preserved in the embedding.
(4) **Singular Values of Spectral Embedding (LSVD) [14]:** It is a vector containing the singular values of the spectral embedding of a graph snapshot. The singular values (or eigenvalues) of the Laplacian matrix encode graph structural properties.
(5) **Node2Vec**: It is a random walk-based node embedding. For each node, the random walks are used to derive the representation of the neighborhood of that node. The objective is to have node embedding as similar as possible to embeddings of nodes in its neighborhood.

Note that we do not explicitly model the graph feature distribution function when running the online martingale test. A martingale

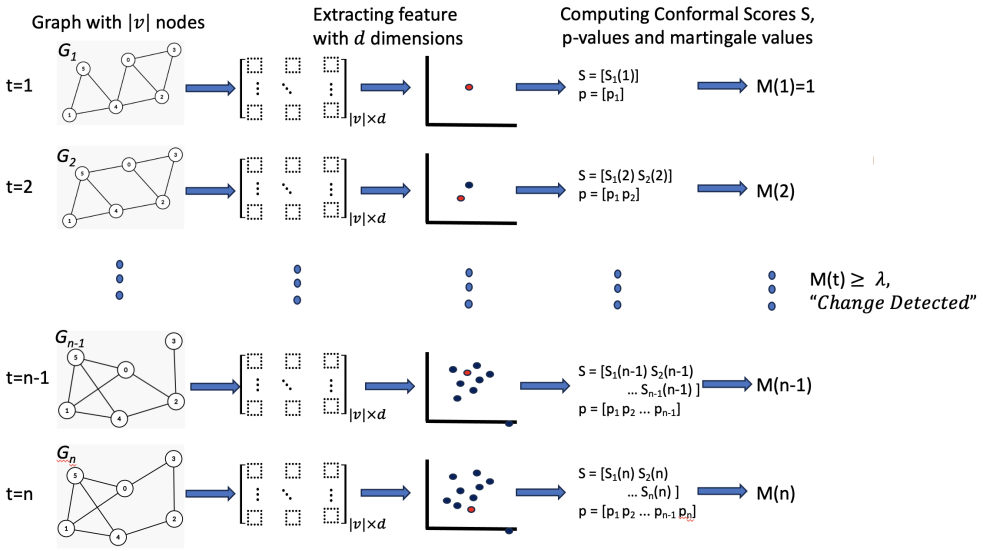

Figure 1: Overview of evolving graph change-point detection using martingale over time.

value is computed as a graph snapshot is observed and change-point decision is based on the martingale value. We discuss the methodology in detail in the next section.

4 METHODOLOGY

In Section 4.1, we provide an overview of our change point detection approach. In Section 4.2, we describe in detail the martingale change-point detection method for evolving graphs. In Section 4.3, we discuss how the multiple extracted properties and features allow us to monitor and explain the possible change-points with respect to the feature distributions considered by multiple martingale tests.

4.1 Approach Overview

Figure 1 shows an overview of the process of using a martingale-based change-point detection approach for an evolving graph represented by a sequence of graph snapshots $(\mathcal{G}_k)_{1\leq k\leq n}$ over time. Feature matrix $f(\mathcal{G}_k)\in R^{|V|\times d}$ are extracted from a graph at time instance k such that V is the set of nodes for an evolving graph and d is the feature dimension for each node. Over time, the extracted features are used to compute the set S of conformal scores (different values at different time instance k for a particular graph snapshot) for all the graph snapshots observed so far. Set p contain the p-values p_k computed so far. $M(t)$ is the martingale values computed at time instance t. Decision to declare a change-point detected is based on the threshold value λ. When a change is declared, the process is reset (i.e., martingale value is reset to one at the next time instance).

4.2 Conformal Evolving Graph Change-Point Detection

The martingale change detection approach was first proposed in [12] for labeled data stream. It was further extended to handle the change detection problem for regression model and unlabeled data stream [13].

In this section, we describe and discuss in detail the proposed evolving graph change-point detection using martingale. First, a particular graph feature vector or matrix is extracted from an observed graph instance $\mathcal{G}_k$ as described in Section 3. In an online setting, graph feature vectors $f(\mathcal{G}_1)$,...,$f(\mathcal{G}_{n-1})$,$f(\mathcal{G}_n)$ from time instance $t=1$ to current time instance $t=n$ are used to compute the martingale value M_n at $t=n$.

Definition 4.1. Given a sequence of random variables $\{M_i : 0\leq i<\infty\}$. It is a **martingale** M with respect to the sequence $\{Z_1 : 0\leq i<\infty\}$ (in particular, M_0 is a constant value), if, for all $i\geq 0$ the following conditions hold:

- M_i is a measurable function of $Z_0,Z_1,\cdots,Z_i$,
- $E(|M_i|)<\infty$,
- $E(M_{n+1}|Z_0,\cdots,Z_i)=M_n$.

To compute the martingale value at current time instance $t=n$, one used p-values p_t computed from $t=1$ to n. A function $h:X^n\rightarrow[0,1]$ is a *p-value function* with respect to any probability distribution P over X if for all $n\in N$ and $r\in[0,1]$,

$$P_n\{x\in X^n : h(x)\leq r\}\leq r \tag{1}$$

In statistical significance testing, the p-value provides a measure on how well the data support or discredit the statistical null hypothesis.

A family of martingales, called the *power martingale* [25], indexed by $\epsilon\in[0,1]$, is defined as

$$M_n^{(\epsilon)}=\prod_{i=1}^{n}\left(\epsilon p_i^{\epsilon-1}\right) \tag{2}$$

where the p_is are the output p-values from the function h, and the initial martingale $M_0^{(\epsilon)}=1$.

Vovk et al. [25] introduced the idea of testing exchangeability of streaming data using (2). Ho [12] utilized the idea that a lack of data exchangeability in the observed streaming data implies a change occurs in the data distribution. In this paper, the data points are

Algorithm 1 Martingale value computation at time step t

Input: Graph $\mathcal{G}_t$

1: $v_t = f(\mathcal{G}_t)$.
2: Compute conformal scores for $(\mathcal{G}_k)_{1\le k\le t-1}$ and $\mathcal{G}_t$ using $v_i, i=1\le j\le t$ based on (3)
3: Compute the p-value p_t using conformal scores based on (4);
4: Compute $M(t)$ using (2) with all previously computed $p_i, i=1\le j\le t-1$ and p_t;
5: **if** $M(t)\ge\lambda$ **then**
6: **Change Point Detected;**
7: Alert user of Change Point;
8: Break (from the test process);
9: **else**
10: **Normal (continue the test process)**
11: **end if**

snapshots of an evolving graph and the distribution change is with respect to some properties or characteristics represented by some feature vectors/ embeddings for the snapshots.

The fundamental building block of the martingale (2) is called the *conformal score* which quantifies how much a data instance (an observation or a prediction) is different from other data instance [13]. For our change-point detection problem, a snapshot of the evolving graph is represented by a feature matrix or vector. To compute the conformal scores for the snapshots of the graph observed so far represented by a sequence of features $(f(\mathcal{G}_k))_{1\le k\le n}$, we utilize $\mathcal{K}-$mean clustering such that $\mathcal{K}=1$. Let $C(f(\mathcal{G}_k))_{1\le k\le n}$ be the cluster center. The conformal score for graph $\mathcal{G}_k$

$$S(\mathcal{G}_k) = ||f(\mathcal{G}_k) - C(f(\mathcal{G}_k))_{1\le k\le n}|| \tag{3}$$

such that $||c||$ is some suitable distance measure (for vector or matrix).

A p-value p_n at time instance n in (2) is computed using the p-value function,

$$p_n(\{(\mathcal{G}_k)_{1\le k\le n}\},\theta_n) = \frac{\#\{j : cs_j > cs_n\} + \theta_n \#\{j : cs_j = cs_n\}}{n} \tag{4}$$

where cs_j is the *conformal score* for $\mathcal{G}_j, j=1,2,\ldots,n$ and θ_n is randomly chosen from $[0,1]$ at time instance n [25]. The computed p-values are independent and uniformly distributed on $[0,1]$ if the graph snapshots are exchangeable [24].

Theorem 4.2. *(**Doob's Maximal Inequality**) Suppose that $\{M_k : 0\le k<\infty\}$ is a non-negative martingale. Then for any $\lambda>0$ and $n\in\mathbb{N}$,*

$$\lambda P\left(\max_{0\le k\le n} M_k \ge \lambda\right) \le E(M_n). \tag{5}$$

If $E(M_n) = E(M_0) = 1$, then one has

$$P\left(\max_{k\le n} M_k \ge \lambda\right) \le \frac{1}{\lambda} \tag{6}$$

This inequality means that there is low probability that M_k is higher than some high λ value. However, there is some probability for a false detection. Inequality (6) is an upper-bound for the false positive rate for change-point detected when there is none. The amount of risk one is willing to take for a detection to be a false alarm will determine the λ value.

Algorithm 1 shows step-by-step the computation of martingale value for a snapshot of the evolving graph $\mathcal{G}_t$ at time t. Line 1 extracts the feature of interest using f on $\mathcal{G}_t$. Line 2 computes the conformal scores for all the graph snapshots observed so far using (3). Line 3 computes the p-value p_t using (4) with the conformal scores computed in Line 2 . Line 4 computes the martingale value at time t using all previously computed p-values. If the martingale value is greater than a pre-defined threshold λ, the test signals a change-point detection, alert the user, and stop (or reset) the change detection process. If not, the process will continue to monitor the evolving graph.

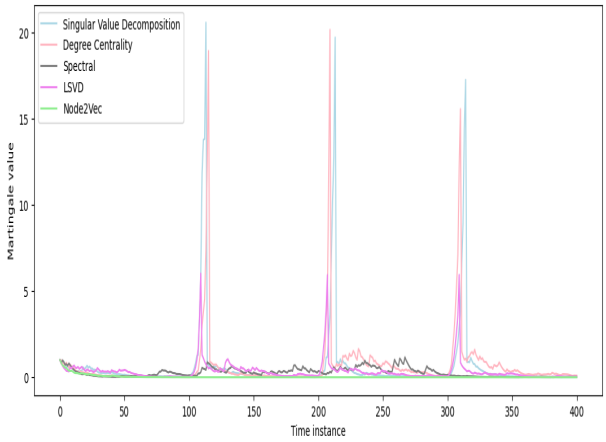

Figure 2: Monitoring evolving graph feature distributions change for multiple features over time.

4.3 Feature-based Explanation for Detected Change in Evolving Graph

Figure 2 shows an example of the martingale values of an evolving graph with structural changes occurring at t=100, 200, and 300 for an evolving graph generated using the Barabási–Albert model (see Section 5.1 for dataset generation description). The martingale value is reset when it is greater than $\lambda=20$. One observes from the figure that martingale-based change-point detection using the degree centrality vector, SVD embedding, and LSVD representations are able to detect all 3 change-points. The martingale method is unable to detect the structural change in the evolving graph using spectral embedding and Node2Vec features.

We hypothesize that graph features exhibit different time-varying distribution characteristics in the evolving graph. Moreover, there exist some features whose distributions do not show a change or shift even when a structural change occurs. These features do not contain useful information about the particular structural change. Hence, by monitoring the martingale values for different features, one can provide explanation on what type of feature distribution change occurs as a structural change occurs. It provides feature-based explanation on why the change occurs in the evolving graph.

5 EXPERIMENTAL RESULTS

In Section 5.1, we describe the dataset and two graph generators used in our experiment. In Section 5.2, we describe the evaluation measures used in our experimental results. In Section 5.3, we describe our experimental design. In Section 5.4, we present and discuss our empirical results.

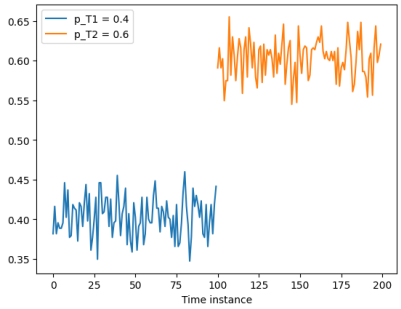

Figure 3: The mean values of the elements in the degree centrality feature vector for $|V|$ nodes for an evolving graph generated using the Erdős-Rényi model with change-point at t = 101.

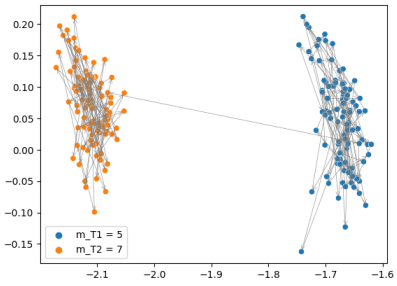

Figure 4: Locations and movement of the evolving graph center in the 2D SVD graph feature space for an evolving graph with $|V|$ nodes generated using the Barabási–Albert model. The change-point is at $t = 101$.

5.1 Dataset Descriptions

5.1.1 Sequences of Synthetic Evolving Graph Dataset: We generate sequences of evolving graphs using two random graph models: Erdős-Rényi model [5] and Barabási–Albert model [2], for our empirical study of our proposed change-point detection approach. The random graphs are generated using the graph generators in NetworkX package [1].

For each snapshot of an evolving graph with $|V|$ nodes generated based on the Erdős-Rényi model, an edge is created between two nodes based on probability p. For the first T_1 time instances, the probability is fixed at p_{T_1}. For the next T_1 time instances, the probability is p_{T_2}. The change-point occurs at T_1+1. In our experiment, different $\delta p = |p_{T_1} - p_{T_2}|$ values are used to quantify the amount of differences before and after change occurs in an evolving graph. Smaller δp signifies more difficult change-point detection problem. Figure 3 shows the mean values of the elements in the degree centrality feature vector for an evolving graph with $|V|$ nodes generated using the Erdős-Rényi model with $T_1 = T_2 = 100$, $p_{T_1} = 0.4$ and $p_{T_2} = 0.6$. Hence, $\delta p = 0.2$. The change-point is at $t = 101$.

A random graph generated based on the Barabási–Albert model is a graph whose degree distribution follows the power law. To generate such a random graph, an initial graph (a star graph with $m+1 << |V|$ nodes) is grown by attaching new nodes each with edges that are preferentially attached to existing nodes with high degree until we reach $|V|$ nodes. m is defined as the number of edges to attach from a new node to existing nodes. For the first T_1 time instances, m is fixed at m_{T_1}. For the next T_1 time instances, $m = m_{T_2}$. The change-point occurs at T_1+1. In our experiment, different $\delta m = |m_{T_1} - m_{T_2}|$ values are used to generate changes in the scaling factor in an evolving graph. Smaller δm signifies more difficult change-point detection problem. Figure 4 shows the locations and movement of the evolving graph center in the 2D SVD graph feature space for an evolving graph with $|V|$ nodes generated using the Barabási–Albert model with $T_1 = T_2 = 100$, $m_{T_1} = 7$ and $m_{T_2} = 5$. Hence, $\delta m = 2$. The change-point is at $t = 101$. The longest distance between the centers at t=100 and 101.

5.1.2 MIT Reality Dataset [17]: It is a dataset that record the social network evolution in a student dormitory based on students' proximity. It utilizes bluetooth signals sent and received between mobile phones over time. The signals indicate the senders' mobile phones were within 10 meters of receivers' mobile phones at the time of the record. Proximity probability greater than 0.3 is considered as an edge between two users. All the edges that have formed in a single day irrespective of time are considered as edges of a single graph. 289 days of graphs are in the dataset with some missing weeks.

[1] https://networkx.org/documentation/stable/reference/generators.html

5.2 Evaluation Measures

We evaluate the detection performance using mainly (i) recall, (ii) precision, and (iii) F1 measure:

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive+False Positive}}$$

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive+False Negative}}$$

$$F_1 = \frac{2\times\text{Recall}\times\text{Precision (Prec)}}{\text{Recall+Precision}}$$

Precision is the probability that a detected change point is a true change-point and detected within t' time instances. Recall is the probability that an algorithm detected the change within t' time instances.

We also present the mean delay time (MDT) for change detection. We define the *delay time* for a detected change-point as the difference in time instances from the start of the change to the time instance when it is detected and within t' time instances. If the change cannot be detected within t' time instances, it is a *missed detection* or *false negative*. A *false positive* (FP)(i.e., alarm) is a change-point detection call by the algorithm when there is no change.

5.3 Experimental Design

For each experimental trial, the sequence of graph snapshots have the following settings: $|V| = 30$, $T_1 = T_2 = 100$ with change-point at t = 101 (see Figure 3). We perform 20 trials on each experimental setting: δp (or δm), feature representation, and the threshold λ. We varying δp (or δm), feature representation, and the threshold λ to study their impact on the performance of our proposed method.

We used the four graph feature representations/embeddings described in Section 3. For SVD embedding, spectral embedding, and node2vec, we use $d = 2$. In other words, each node is represented by a 2D vector. The degree centrality vector has 30 elements since $|V| = 30$. For threshold λ, we use 5, 10, 15 and 20, corresponding to a false positive upper bound from 20% reducing to 5%. For Erdős-Rényi model parameter δp, we vary from 0.05 to 0.2. Since δp quantifies the amount of differences before and after change occurs in an evolving graph. and $\delta p = 0.05$ does not registered any detection, our results start with $\delta p = 0.075$. For δm, we vary from 1 to 4.

We implement a sequential probability ratio test (SPRT) [26] similar to [15] for structural time series change detection as a baseline to compare our proposed martingale approach. A sequences of feature embedding for the graph snapshots can be considered as time series. Table 1 shows its performance on the change-point detection task for an evolving graph generated using the Barabási–Albert model. We use SVD embedding as the feature representation here as it performs best for our approach (see Section 5.4). We vary the Type I error (false positive rate) α from 0.05 to 0.2 to control the detection threshold for SPRT. One observe that while we can control well the false positive close to our approach, there are many missed detections which affecting the detection performance.

δm	α	Prec	FPR	Recall	F1	MDT
1	0.05	0.14	0.01	0.31	0.19	17.6
	0.1	0.13	0.02	0.31	0.18	18.44
	0.15	0.12	0.02	0.31	0.17	18.40
	0.2	0.12	0.02	0.34	0.18	23.83
2	0.05	0.09	0.05	0.5	0.15	12.6
	0.1	0.09	0.05	0.49	0.15	12.97
	0.15	0.08	0.05	0.5	0.14	12.83
	2	0.07	0.06	0.49	0.13	12.77
3	0.05	0.04	0.12	0.5	0.07	3
	0.1	0.03	0.13	0.5	0.06	3
	0.15	0.03	0.13	0.5	0.06	3
	2	0.03	0.13	0.5	0.06	3
4	0.05	0.04	0.12	0.5	0.07	1
	0.1	0.04	0.12	0.5	0.07	1
	0.15	0.04	0.12	0.5	0.06	1
	0.2	0.03	0.12	0.5	0.06	1

Table 1: SPRT performance on evolving graphs generated using the Barabasi-Albert model with the graph snapshots represented using 2-D SVD embedding.

5.4 Results and Discussions

5.4.1 Effect of Varying Detection Threshold on False Positive Rate. Figure 5 shows the false positive rate when we vary the detection threshold λ from 5 to 20 on the two types of evolving graphs (Erdos-Renyi model with $\delta p = 0.2$ and Barabasi-Albert model with $\delta m = 2$) using SVD embedding and degree centrality. We observe the false positive decreasing trend follows Theorem 4.2 but with a much smaller false positive rate.

5.4.2 Effect of Graph Representations and Problem Difficulty on Detection Performance. The problem difficulty for evolving graphs generated using the Erdős-Rényi model is quantified by δp. The smaller the δp, the more difficult is the change-point detection problem. Similarly, the problem difficulty for evolving graph generated using the Barabasi-Albert model is quantified by δp. The smaller the δm, the more difficult is the change-point detection problem.

Node2vec and spectral embedding did not perform well in our empirical study (with majority of experiments showing high missed detection rate). Hence, the detection performance results are not presented here. Only results of our proposed martingale approach using degree centrality vector representation, Laplacian SVD Singular values (LSVD), and the SVD embeddings are shown in Table 2, 3, 4, 5, 6 and 7.

From the results in Table 2, 3, 4, 5, 6 and 7, we have the following general observations:

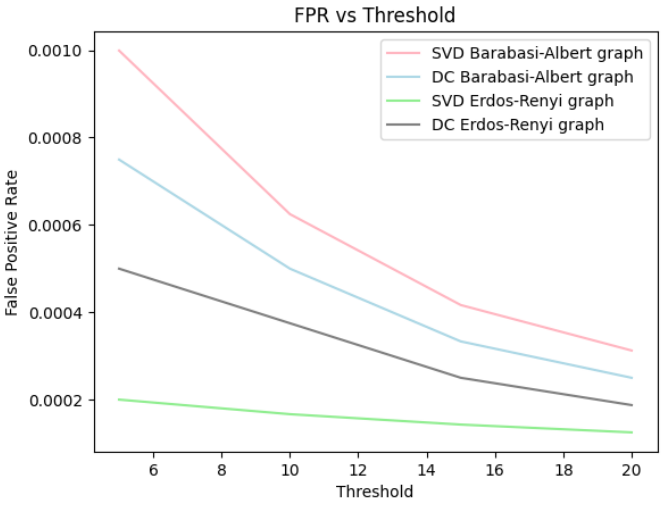

Figure 5: False positive rate (FPR) decreasing trend with increasing threshold λ

(1) For an evolving graph with a change defined by the difference (δp) in probability of an edge forming between any two nodes (Erdős-Rényi model) in the two time segments of an evolving graph (Table 3 and 5):
   (a) degree centrality and SVD embedding show great detection performance for the martingale method.
   (b) mean delay time (MDT) before correct detection increases with increase difficulty.
(2) For an evolving graph with a change defined by a shift (δm) in the degree distribution that follows a power law (Barabasi-Albert model) in the two time segments of an evolving graph (Table 2 and 4):
   (a) degree centrality and SVD embedding show great detection performance for the martingale method.
   (b) mean delay time (MDT) before correct detection increases with increase difficulty when SVD embedding is used. MDT when using degree centrality remains relatively stable. This should not be a surprising observation as the degree centrality vector distribution could approximate well degree distribution for the graph snapshots.
(3) LSVD feature has lower precision and recall, i.e., higher miss detection. But MDT is lower, i.e., faster detection.
(4) Threshold λ does not significantly affects MDT. The difficulty of the change detection scenario (δp,δm) has more impact on MDT.

5.4.3 Monitoring Multiple Graph Features for Changes in Evolving Graph. Figure 6 shows the martingale values of the five graph features on the social network evolution in a student dormitory over 289 days [17]. Here, the martingale peaks in the figure for degree centrality, SVD, and LSVD features show some correlation between the change-points detected and holidays. Hence, the three features may be good in representing and explaining changes in student interactions in student dormitory. More investigations are needed to study these features and their relationships to changes in object interactions in a graph.

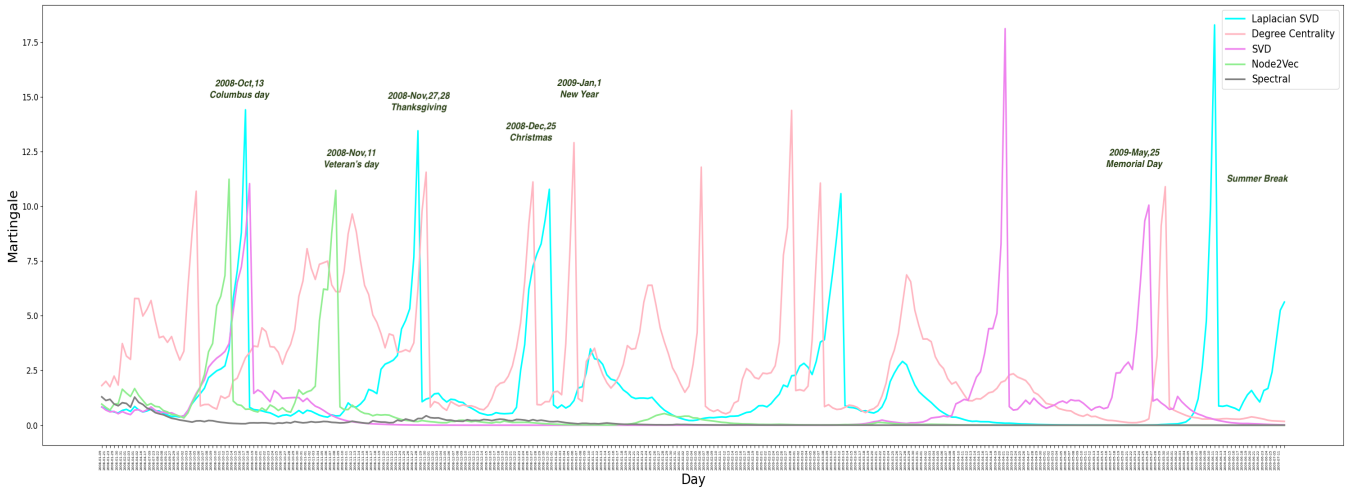

Figure 6: Monitoring the martingale values of the four graph features on the social network evolution in a student dormitory over 289 days, with λ threshold set to 10.

δm	λ	Prec	Recall	F1	MDT
1	5	1	1	1	12.98
	10	1	1	1	13
	15	1	1	1	13.12
	20	1	1	1	13.35
2	5	0.95	1	0.98	7.9
	10	0.97	1	0.99	8.37
	15	0.98	1	0.99	8.94
	20	0.99	1	0.99	9.44
3	5	1	1	1	8.8
	10	1	1	1	9.7
	15	1	1	1	10.23
	20	0.99	1	0.99	10.75
4	5	1	1	1	10.25
	10	1	1	1	11.40
	15	1	1	1	11.90
	20	1	1	1	12.38

Table 2: Change point detection performance on evolving graphs with change points generated using the Barabasi-Albert Model and graph snapshots represented by degree centrality vector.

δp	λ	Prec	Recall	F1	MDT
0.075	5	1	1	1	11.7
	10	1	1	1	14.75
	15	1	1	1	15.8
	20	1	1	1	16.65
0.1	5	1	1	1	12
	10	1	1	1	13.5
	15	1	1	1	14.1
	20	1	1	1	14.71
0.15	5	1	1	1	11.85
	10	1	1	1	12.825
	15	1	1	1	13.584
	20	1	1	1	14.137
0.2	5	1	1	1	6.4
	10	1	1	1	7.13
	15	1	1	1	7.48
	20	1	1	1	8.0

Table 3: Change point detection performance on evolving graphs with change points generated using the Erdos-Renyi Model and graph snapshots represented by degree centrality vector.

δm	λ	Prec	Recall	F1	MDT
1	5	0.86	1	0.90	15.1
	10	0.91	1	0.95	17.1
	15	0.94	1	0.97	18.45
	20	0.95	1	0.98	19.58
2	5	92	1	0.96	17
	10	91	1	0.95	15.64
	15	0.91	1	0.95	14.8
	20	0.92	1	0.96	14.35
3	5	0.84	1	0.91	7.2
	10	0.91	1	0.95	7.7
	15	0.94	1	0.97	8.22
	20	0.95	1	0.98	8.68
4	5	1	1	1	5.55
	10	1	1	1	6.55
	15	1	1	1	7.02
	20	1	1	1	7.31

Table 4: Change point detection performance on evolving graphs with change points generated using the Barabasi-Albert Model and graph snapshots represented by SVD embedding.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we describe our proposed change-point detection approach for evolving graph by monitoring the martingale value derived from the graph features. We demonstrate empirically that the feature representation that encodes a graph property (or characteristics) is critical in the performance of the martingale test in detecting the change-point using two synthetic (random topology, scale-free) graph generators and a real-world dataset. Future work includes extensive comparisons using (1) graph generators creating different evolving graphs with change-points, (2) additional graph features, and (3) recently proposed new approaches.

REFERENCES

[1] Charu C Aggarwal, Yao Li, and Philip S Yu. 2020. On supervised change detection in graph streams. In *Proceedings of the 2020 SIAM International Conference on Data Mining*. SIAM, 289–297.

[2] Albert-László Barabási and Réka Albert. 1999. Emergence of scaling in random networks. *science* 286, 5439 (1999), 509–512.

δp	λ	Prec	Recall	F1	MDT
0.075	5	0.95	1	0.97	17.8
	10	0.91	1	0.95	20.63
	15	0.94	1	0.97	22.76
	20	0.95	1	0.97	24.33
0.1	5	0.90	1	0.95	5.3
	10	0.95	1	0.97	6
	15	0.96	1	0.98	6.36
	20	0.96	1	0.98	6.55
0.15	5	0.95	1	0.97	5.45
	10	0.98	1	0.99	6.5
	15	0.98	1	0.99	7
	20	0.99	1	0.99	7.45
0.2	5	0.91	1	0.95	4.65
	10	0.96	1	0.97	5.57
	15	0.97	1	0.98	6.01
	20	0.98	1	0.98	6.36

Table 5: Change point detection performance on evolving graphs with change points generated using the Erdos-Renyi Model and graph snapshots represented by SVD embedding.

δp	λ	Prec	Recall	F1	MDT
0.075	5	0.57	0.70	0.63	3.5
	10	0.67	0.82	0.74	2.65
	15	0.73	0.88	0.80	3.08
	20	0.78	0.91	0.83	3.63
0.1	5	0.54	0.68	0.61	2
	10	0.71	0.82	0.76	2.06
	15	0.79	0.88	0.83	2.22
	20	0.84	0.90	0.87	2.49
0.15	5	0.60	0.64	0.62	2.06
	10	0.69	0.77	0.73	2.24
	15	0.78	0.84	0.81	2.3
	20	0.82	0.88	0.85	2.56
0.2	5	0.70	0.80	0.74	1.5
	10	0.77	0.83	0.8	1.88
	15	0.80	0.88	0.84	2.23
	20	0.85	0.91	0.88	2.33

Table 6: Change point detection performance on evolving graphs with change points generated using the Erdos-Renyi Model and graph snapshots represented by Laplacian SVD Singular values.

δp	λ	Prec	Recall	F1	MDT
1	5	0.65	0.59	0.62	6.3
	10	0.77	0.68	0.72	6.8
	15	0.83	0.75	0.79	7.31
	20	0.87	0.80	0.83	7.46
2	5	0.44	0.64	0.53	7
	10	0.56	0.74	0.64	7.1
	15	0.61	0.80	0.69	7.5
	20	0.67	0.84	0.74	7.79
3	5	0.69	0.71	0.70	6.05
	10	0.78	0.83	0.81	6.1
	15	0.82	0.88	0.85	6.3
	20	0.85	0.90	0.88	6.51
4	5	0.50	0.61	0.55	9.60
	10	0.62	0.73	0.67	9.97
	15	0.70	0.79	0.74	10.15
	20	0.75	0.83	0.79	10.58

Table 7: Change point detection performance on evolving graphs with change points generated using the Barabasi-Albert Model and graph snapshots represented by Laplacian SVD Singular values.

[3] Claudio DT Barros, Matheus RF Mendonça, Alex B Vieira, and Artur Ziviani. 2021. A survey on embedding dynamic graphs. *ACM Computing Surveys (CSUR)* 55, 1 (2021), 1–37.

[4] Michele Basseville, Igor V Nikiforov, et al. 1993. *Detection of abrupt changes: theory and application*. Vol. 104. prentice Hall Englewood Cliffs.

[5] Paul Erdős, Alfréd Rényi, et al. 1960. On the evolution of random graphs. *Publ. math. inst. hung. acad. sci* 5, 1 (1960), 17–60.

[6] Dhivya Eswaran, Christos Faloutsos, Sudipto Guha, and Nina Mishra. 2018. Spotlight: Detecting anomalies in streaming graphs. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1378–1386.

[7] André Ferrari and Cédric Richard. 2020. Non-parametric community change-points detection in streaming graph signals. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 5545–5549.

[8] Jonathan Flossdorf, Roland Fried, and Carsten Jentsch. 2023. Online monitoring of dynamic networks using flexible multivariate control charts. *Social Network Analysis and Mining* 13, 1 (2023), 87.

[9] Daniele Grattarola, Daniele Zambon, Lorenzo Livi, and Cesare Alippi. 2019. Change detection in graph streams by learning graph embeddings on constant-curvature manifolds. *IEEE Transactions on neural networks and learning systems* 31, 6 (2019), 1856–1869.

[10] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. 855–864.

[11] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems* 30 (2017).

[12] Shen-Shyang Ho. 2005. A martingale framework for concept change detection in time-varying data streams. In *Proceedings of the 22nd international conference on Machine learning*. 321–327.

[13] Shen-Shyang Ho and Harry Wechsler. 2010. A martingale framework for detecting changes in data streams by testing exchangeability. *IEEE transactions on pattern analysis and machine intelligence* 32, 12 (2010), 2113–2127.

[14] Shenyang Huang, Yasmeen Hitti, Guillaume Rabusseau, and Reihaneh Rabbany. 2020. Laplacian change point detection for dynamic graphs. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 349–358.

[15] Hiromichi Kawano, Tetsuo Hattori, and Ken Nishimatsu. 2008. Structural change point detection method of time series using sequential probability ratio test. *IEEJ Transactions on Electronics, Information and Systems* 128, 4 (2008), 583–592.

[16] Danai Koutra, Neil Shah, Joshua T Vogelstein, Brian Gallagher, and Christos Faloutsos. 2016. Deltacon: Principled massive-graph similarity function with attribution. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 10, 3 (2016), 1–43.

[17] Anmol Madan, Manuel Cebrian, Sai Moturu, Katayoun Farrahi, et al. 2011. Sensing the" health state" of a community. *IEEE Pervasive Computing* 11, 4 (2011), 36–45.

[18] Ian McCulloh and Kathleen M Carley. 2011. Detecting change in longitudinal social networks. *Journal of social structure* 12, 3 (2011), 1–37.

[19] ES Page. 1961. Cumulative sum charts. *Technometrics* 3, 1 (1961), 1–9.

[20] Leto Peel and Aaron Clauset. 2015. Detecting change points in the large-scale structure of evolving networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 29.

[21] SW Roberts. 1959. Control Chart Tests Based on Geometric Moving Averages. *Technometrics* 1, 3 (1959), 239–250.

[22] Seyyid Emre Sofuoglu and Selin Aviyente. 2022. Gloss: Tensor-based anomaly detection in spatiotemporal urban traffic data. *Signal Processing* 192 (2022), 108370.

[23] Deborah Sulem, Henry Kenlay, Mihai Cucuringu, and Xiaowen Dong. 2022. Graph similarity learning for change-point detection in dynamic networks. *arXiv preprint arXiv:2203.15470* (2022).

[24] Vladimir Vovk, Alex Gammerman, and Glenn Shafer. 2005. *Algorithmic Learning in a Random World*. Springer-Verlag, Berlin, Heidelberg.

[25] Vladimir Vovk, Ilia Nouretdinov, and Alexander Gammerman. 2003. Testing exchangeability on-line. In *Proceedings of the 20th International Conference on Machine Learning*. 768–775.

[26] Abraham Wald. 1947. Sequential analysis. *john wiley & sons, New York, NY* (1947).

[27] Yu Wang, Aniket Chakrabarti, David Sivakoff, and Srinivasan Parthasarathy. 2017. Fast change point detection on dynamic social networks. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 2992–2998.

[28] Yingjie Xie, Wenjun Wang, Minglai Shao, Tianpeng Li, and Yandong Yu. 2023. Multi-view change point detection in dynamic networks. *Information Sciences* 629 (2023), 344–357.

[29] Minji Yoon, Bryan Hooi, Kijung Shin, and Christos Faloutsos. 2019. Fast and accurate anomaly detection in dynamic graphs with a two-pronged approach. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 647–657.

[30] Li Zheng, Zhenpeng Li, Jian Li, Zhao Li, and Jun Gao. 2019. AddGraph: Anomaly Detection in Dynamic Graph Using Attention-based Temporal GCN.. In *IJCAI*, Vol. 3. 7.