Visual Detection of Structural Changes in Time-Varying Graphs Using Persistent Homology

Mustafa Hajij*  
University of South Florida  
Bei Wang†  
University of Utah  
Carlos Scheidegger‡  
University of Arizona  
Paul Rosen§  
University of South Florida

Abstract
Topological data analysis is an emerging area in exploratory data analysis and data mining. Its main tool, persistent homology, has become a popular technique to study the structure of complex, high-dimensional data. In this paper, we propose a novel method using persistent homology to quantify structural changes in time-varying graphs. Specifically, we transform each instance of the time-varying graph into a metric space, extract topological features using persistent homology, and compare those features over time. We provide a visualization that assists in time-varying graph exploration and helps to identify patterns of behavior within the data. To validate our approach, we conduct several case studies on real-world datasets and show how our method can find cyclic patterns, deviations from those patterns, and one-time events in time-varying graphs. We also examine whether a persistence-based similarity measure satisfies a set of well-established, desirable properties for graph metrics.

Keywords: Topological data analysis, time-varying graph, persistent homology, graph visualization

1 Introduction
Time-varying graphs are ubiquitous across many disciplines, yet difficult to analyze, making them a natural target for visualization – a good visual representation of a time-varying graph will present its structure and structural changes quickly and clearly, to enable further analysis and exploration.

A major development in graph drawing has been the observation that using derived information can retain structure in static graph visualizations. For example, the dot layout uses node ranks to perform hierarchical drawings [31]; the neato algorithm employs graph distances within statistical multi-dimensional scaling [30]; and Noack’s energy model utilizes approximated clustering [46].

In this paper, we take the first steps toward using topological features – captured by persistent homology – with the design goal of detecting potentially important structural changes in time-varying graph data. By topological features, we do not mean the configuration of nodes and edges alone, but instead the 0- and 1-dimensional homology groups of a metric space that describe its connected components and tunnels, respectively.

This definition allows us to quantify structural elements within time-varying graphs to identify behavior patterns in the data. Persistent homology quantifies individual topological features (events) in the graph according to their significance (or persistence). The set of all features, encoded by the persistence diagram, can be seen as a fingerprint for the graph. Using this fingerprint, the most topologically important structures of two graphs can be compared in a manner that is robust to small perturbations in the data.

Well-understood techniques in topological data analysis typically focus on the qualitative study of point cloud data under the metric space setting. In order to study graph data, our approach is to embed the graph in a metric space, where topological techniques can be applied. In other words, the notion of metric space acts as an organizational principle [9] in interpreting the graph data.

Our approach, as seen Figure 2, can be summarized as follows. The input of our pipeline is a time-varying graph, which is an ordered sequence of graph instances. First, each instance is embedded into
a metric space. Second, topological features of each instance are extracted, using persistent homology, and encoded within persistence diagrams. Third, instances are compared by calculating the distance between persistence diagrams and projecting them using classical multi-dimensional scaling (MDS) [6].

The data is then visualized using an interactive timeline and node-link diagrams, as shown in Figure 1. The horizontal axis is used to represent time, while the vertical location is the first component of MDS; in other words, it captures the dissimilarities among instances. Graph instances from selected timeframes are drawn using a force-directed layout to demonstrate how the approach highlights different structures in the graph. The contributions of our paper are:

- A novel pipeline for detecting structural changes in time-varying graphs that uses persistent homology to summarize important structures, as opposed to directly comparing nodes and edges.
- An interface that uses conventional visualization approaches adapted to the design goal of highlighting structural changes.
- Two case studies of time-varying graphs showing how our approach can find cyclic patterns, deviations from those patterns, and unique one-time events in the graphs.
- A study of the suitability of using persistence-based similarity measure for detecting structural changes in time-varying graphs.

2 RELATED WORK

Static Graph Analysis and Visualization. We provide a brief overview here, see [53] for a survey.

The first automated technique for node-link diagrams is Tutte’s barycentric coordinate embedding [49], followed by linear programming [31], force-directed/mass-spring embeddings [28, 36], embeddings of the graph metric [30], and techniques using linear-algebraic properties of the connectivity structures (especially, the graph Laplacian and its associated eigenspace) [39, 40].

Most graph visualization systems, including Gephi [3], NodeXL [34], and Graphviz [25], use variations on node-link visualizations to display graphs. For dense graphs, edge bundling can reduce visual clutter by routing graph edges to the same portion of the screen [35]. In terms of quality, divided edge bundling [48] produces high-quality results, while hierarchical edge bundling [29] scales to millions of edges with slightly lower quality. Because these quality and runtime trade-offs are so characteristic of node-link diagram visualizations, whether or not this class of diagrams can effectively unlock the insights hidden inside the structure of large networks remains an open research question.

Other visual metaphors have been proposed to reduce clutter, ranging from relatively conservative proposals [20, 21] to variants of matrix diagrams [18] and abstract displays of graph statistics [38].

Time-Varying Graph Analysis. The problem we address is closely related to the problem of measuring similarity or dissimilarity between graphs without knowing node correspondences. Comparing between graphs up to isomorphism is hard [1]. For this reason, many notions of graph similarities have been proposed [4, 47]. These methods rely on mapping the graphs into a feature space and then defining distances on that space. Other approaches use kernel functions to build a similarity measures on graphs [43, 51]. While large portions of the literature on graph similarity focus on graph comparison with known node correspondences, there are attempts to tackle the problem where node correspondence is unknown [51, 52]. Distance functions on the space of graphs have also been studied [12].

Time-Varying Graph Visualization. Beck et al. [5] provide a detailed survey of dynamic graph visualization. They divide the techniques into two major categories, animation and timelines. Our approach falls into the latter category. Animation approaches, such as the work of Misue et al. [44], vary the graph representation over time, while making the graph as legible as possible at any given instance. Timeline approaches, such as the work of Greilich et al. [33], use a non-animated, often spatially oriented, visual channel to show the changes in the graph over time. Timeline approaches seem to provide a better overview of the data as it tries to capture the entire graph sequence in a single image. These approaches include multiple techniques such as node-link-based methods [37], matrix-based approaches [8] and feature vector-based method [50]. For more references see also [53].

Topological Data Analysis of Networks. Persistent homology is becoming an emerging tool in studying complex networks [19, 22] including collaboration [2, 10] and brain networks [11, 15]. To the best of our knowledge, our approach is the first to connect topological techniques with the visualization design of (time-varying) graphs.

3 APPROACH

Our approach uses persistent homology to identify and compare features in a time-varying graph. Our visual design goal is to identify high-level structural changes in the graph. To do this, consider a time-varying graph \( \mathcal{G} = \{G_0, \ldots, G_n\} \) that contains an ordered sequence of static graph instances \( G_i = (V_i, E_i) \).

We are interested in quantifying and visualizing structural changes of \( \mathcal{G} \). Our analysis pipeline (see Figure 2) is described...
We see the commute-time distance produces a smoother gradient homology, see [23] and the references within. For more background on persistence our process, we first briefly review persistent homology. We then persistent homology to its metric space representation. To describe in Figure 3: The (a) shortest-path and (b) commute-time distance than the shortest-path distance. These distance metrics are illustrated in Figure 3, which shows the distance from a point source to all other locations on the surface. The first few nonzero eigenvectors, since the higher eigenvectors do not contribute significantly. These distance metrics are illustrated in Figure 3, which shows the distance from a point source to all other locations on the surface. We see the commute-time distance produces a smoother gradient than the shortest-path distance.

3.1 Graphs and Metric Space Representations

Suppose an instance $G_i$ is represented as a weighted, undirected graph with a vertex set $V$ and an edge set $E$ equipped with a positive edge weight $w$. We associate each graph instance $G_i$ with a metric space representation, which yields a symmetric distance matrix $d_i$. Consider the positive edge weight as the length of an edge. Then a natural metric $d_{sp}$ is obtained on $G_i$, where for every pair of vertices $x$ and $y$ in $G_i$, the distance $d_{sp}(x,y)$ is the length of the shortest path between them. This is the classic shortest-path distance, which is typically computed with Dijkstra’s algorithm [17] and its variations.

Alternatively, other distance metrics based on the graph Laplacian [14], such as commute-time distance, discrete biharmonic distance, and diffusion distance, can be considered. For instance, the commute-time distance is defined as [27]

$$d_{ct}(x,y) = \frac{|V|-1}{2} \sum_{\lambda \in \lambda_0} \frac{1}{\lambda} (\phi(x) - \phi(y))^2.$$  

(1)

Here $\{\lambda_i\}_{i=0}^{|V|-1}$ and $\{\phi_i\}_{i=0}^{|V|-1}$ are the generalized eigenvalues and eigenvectors of the graph Laplacian of $G_i$, respectively [13]. In practice, we approximate the summations of Equation (1) by considering the first few nonzero eigenvectors, since the higher eigenvectors do not contribute significantly.

These distance metrics are illustrated in Figure 3, which shows the distance from a point source to all other locations on the surface. We see the commute-time distance produces a smoother gradient than the shortest-path distance.

3.2 Extracting Topological Features

To extract topological features from each graph instance, we apply persistent homology to its metric space representation. To describe our process, we first briefly review persistent homology. We then describe persistence diagrams, which encode topological features of a given graph instance. For more background on persistence homology, see [23] and the references within.

**Topological features.** Homology deals with topological features of a space. Given a topological space $\mathbb{X}$, the 0-, 1- and 2-dimensional homology groups, denoted respectively as $H_0(\mathbb{X})$, $H_1(\mathbb{X})$ and $H_2(\mathbb{X})$, correspond intuitively to (connected) components, tunnels and voids of $\mathbb{X}$.

In our context, we care about the 0- and 1-dimensional topological features of a graph instance $G_i$ that, roughly speaking, correspond to (connected) components and tunnels formed by points in its metric space representation.

**Persistent homology.** In practice, there might not exist a unique scale that captures topological structures of the data. Instead, we adapt a multi-scale notion of homology, called persistent homology, a main tool in topological data analysis, to describe the topological features of a space at different spatial resolutions.

Persistent homology typically starts with a finite set of points in a metric space. In our setting, each graph instance $G_i$ is associated with a metric space, where vertices in $G_i$ form a finite set of points $S$, and $d_i$ encodes the pairwise distance among points in $S$. We then apply a geometric construction, such as a Rips complex, on the point set $S$, that describes the combinatorial structure among the points. For a real number $r > 0$, a Rips complex, denoted as $R(r)$, is obtained by considering a set of balls of diameter $r$ centered at points in $S$. A 1-simplex (an edge) is formed between two points in $S$ if and only if their balls intersect (see Figure 4 left). A 2-simplex (a triangular face) is formed among three points if the balls intersect between every pair of points (see Figure 4 right).

**Persistence Diagrams.** Topological features of a graph instance and their persistence are recorded by pairing their birth and death event. For example, at $r = 0.25$, there are six components alive, one per vertex, all of which are born at $r = 0$. At $r = 0.5$, the two components in the upper right combine into one. This causes the death of one component, represented by a barcode of length 0.5 on the right.

**Topological features appear and disappear as the diameter increases:** when a topological feature appears, that is, a component (i.e. a cluster) or a tunnel forms, this is called a birth event; when a topological feature disappears, that is, two components merge into one, this is called a death event. Each topological feature is represented by a single bar, with the position of left and right sides representing the birth and death times, respectively. The persistence of a topological feature is the time difference between the death and the birth event. For example, at $r = 0.25$, there are six components alive, one per vertex, all of which are born at $r = 0$. At $r = 0.5$, the two components in the upper right combine into one. This causes the death of one component, represented by a barcode of length 0.5 on the right.

**Persistence Diagrams.** Topological features of a graph instance and their persistence are recorded by pairing their birth and the death events as a multi-set of points in the plane, called the persistence diagram (see [24]).

Each topological feature is represented as a point $(u,v)$, where $u$ is the birth time, and $v$ is the death time of the feature. Certain
features may “live” forever; in that case, they are assigned a death
time of ∞. Therefore, a persistence diagram contains a multi-set of
points in the extended plane (i.e., (R ∪ {±∞})²). For technical reasons,
we add the points on the diagonal to the diagram, each with infi-
nite multiplicity. The persistence of the pair (u, v) is simply |v − u|.
Features with higher persistence carry more significant topological
information. Features with low persistence are typically considered
noise. A persistence diagram can be visualized as persistence bar-
codes [32] (see Figures 2 and 5), where each bar starts at time u and
ends at time v. We are interested in 0- and 1-dimensional topological
features, so we consider the 0- and 1- persistence diagrams, denoted
as PD₀(F) and PD₁(F), respectively.

3.3 Comparing Sets of Topological Features

A persistence diagram can be thought of as a summary of topological
features of a graph instance Gᵢ. To quantify the structural difference
between two instances Gᵢ and Gⱼ, we compute the bottleneck and
Wasserstein distances between their persistence diagrams.

Given two persistence diagrams X and Y, let η be a bijection between
points in the diagram. The bottleneck distance [24] is defined as

\[ W_∞(X, Y) = \inf_{\eta : X \to Y} \sup_{x \in X} \| x - \eta(x) \|_∞. \]  

(2)

The Wasserstein distance is

\[ W_q(X, Y) = \left( \inf_{\eta : X \to Y} \int_X \| x - \eta(x) \|^q \right)^{1/q}, \]  

(3)

for any positive real number q; in our setting, q = 2.

The set of points in the persistence diagram can be considered as a
feature vector, where the feature space consists of all persistence
diagrams for the time-varying graph G. Given all pairwise distances
between persistence diagrams, classical MDS is then used to reduce
the dimensionality of the feature vectors for visualization, and to
identify the instances where topologically interesting events occur.

3.4 Visualization

The design goal of our interactive visualization tool is to provide
insights about variation in the structural properties of time-varying
graphs. In this way, we hope to identify time periods of uniform
behavior (low variation) and outlier behavior (instances of high
variation). Our visualization tool provides a number of capabilities
to support this form of investigation.

Timeline. The timeline view uses the horizontal axis to rep-
tresent time and the vertical axis to represent the first dimension
returned by applying classical MDS to the space of persistence dia-
grams. This in essence highlights the dissimilarity between graph
instances. Each point on the timeline represents a single instance
of the time-varying graph. The points are colored using cyclic
colormaps, such as the time-of-day colormap of Figure 1 or the
day-of-the-week colormap of Figure 11.

Cyclic Patterns. Two techniques are available for showing
repetitive patterns in the data, both being variations of the timeline.
The first technique simply splits the data based upon a user-specified
period length. Each period is colored uniquely. Figure 7 shows
an example of this. For the second technique, the time periods are
clustered based upon their L²-norm using k-means clustering with a
user-specified k, see Figure 12 for an example where the points are
colored by day of the week.

Graph Visualization. For investigating the behavior of specific
graph instances, the instances are displayed by two visualization
mechanisms. The first is a node-link diagram created using a force-
directed layout. If categorical information is available (such as

in Figure 1), the nodes are colored by those categories. For 1-
dimensional topological features, nodes can be parameterized around
the tunnel using a 1-dimensional cyclic parameterization [16, 54].
An example of this is seen in Figure 9. In other cases, nodes receive
a fixed color. The second mechanism visualizes the persistence
diagram for a given graph instance using its barcodes (see fourth row
of Figure 6). The barcode is a variation on a bar chart that represents
the birth and death of all topological features in the graph.

3.5 Example
We provide an illustrative example of our pipeline in Figure 6.
In step 1 (1st row), a time-varying graph $G$ is given as a sequence
of graph instances, where each instance is a connected, weighted
graph. In step 2 (2nd row), each graph instance is embedded in
a metric space by calculating a distance matrix using the shortest-
path metric. In step 3 (3rd row), each distance matrix is used to
compute a series of filtrations. In reality, additional filtrations are
created, but we show only those that produce 0-dimensional features.
In step 4 (4th row), the 0-dimensional persistence diagrams of the
filtrations are extracted and shown as barcodes. The final step (5th
row) consists of computing the distances between these diagrams
using bottleneck and Wasserstein distances.

The bottleneck or Wasserstein distance as a persistence-based
similarity measure helps quantify topologically similarity between a
pair of instances. For example, under both distances, $G_0$ and $G_1$ are
much closer to one another than are the pairs $(G_0, G_2)$ and $(G_1, G_2)$.

4 Case Studies
To validate our approach, we look at case studies of two publicly
available datasets. Both are communication networks, one involves
interpersonal communication of high school students; and the other
contains e-mail communications between researchers. These case
studies help demonstrate how our approach can identify cyclic pat-
terns in data, deviations from patterns, and one-time events in time-
varying graphs.

Our pipeline requires a number of tools for processing. Graph
processing and metric space embedding are coded using Python.
Persistent homology calculations and the bottleneck and Wasserstein
distances are computed using Dionysus\(^1\). Finally, visualizations are
implemented using Processing\(^2\).

4.1 High School Communications
The High School Communications dataset [26] is a time-varying
graph that tracks the contact between high school students. The data
was collected for 180 students in five classes over seven school days
in November 2012 in Marseilles, France. The graph tracks Monday
through Friday of the first week and Monday and Tuesday of the
following week.

We compute both shortest-path and commute-time distances and
both 0- and 1-dimensional persistence diagrams. Then, both the
bottleneck and Wasserstein distances are used to compare persistence
diagrams. We present a small set of configurations and draw a
few conclusions from them. Many similar conclusions have been
identified in other configurations that are not shown.

4.1.1 An Average Day
First, to examine an average day of communication, we look at
the 0-dimensional features of the first Monday of the dataset in
Figure 1. Commute-time is used to generate persistence diagrams
and bottleneck distance is used to compare diagrams. In this figure,
a number of phases can be seen. In the early and late hours, no
interactions occur (e.g., time A). As the school day begins at time B,
light, loosely connected communications begin. By mid-morning
(time C), class MP*1, PC, PC*, and PSI* are all interacting heavily
internally and externally. Midday (times D & E), shows classes
heavily interacting once again. Early afternoon (time F) shows
mostly internal communications for classes PC, PC*, and PSI*
and internal and external communications for MP*1 and MP*2.
Finally, the end of the day, time G, shows much sparser group
communications.

4.1.2 Comparison with Other Days
While observing patterns within a single day is interesting, compar-
ing Monday with other days can help to better identify regular and

\(^1\)http://www.mrzv.org/software/dionysus/
\(^2\)https://processing.org/
irregular daily behavior. Figure 7 shows just such a comparison; it uses commute-time to generate 0-dimensional persistence diagrams, and Wasserstein distance to compare diagrams.

The top chart of Figure 7 compares the first Monday and the first Tuesday. Ignoring outlier graph instances, two main differences can be observed. First, the early morning of Tuesday shows different levels of activity than Monday. This can be confirmed by looking at examples from those days. Figure 7 (top left) shows example graphs from Monday and Tuesday morning. Second, at the beginning and the end of midday, Tuesday shows higher activity than Monday.

The middle chart of Figure 7 compares Wednesday, Thursday, and Friday. Wednesday and Friday show more early morning activity than Monday, but Thursday shows activity levels similar to Tuesday. Individual graph instances of the time-varying graph from this timeframe can be seen in Figure 7 (middle left). Late morning shows that Wednesday is extremely active, while Thursday and Friday are mostly inactive. Midday (midrange active) and the afternoons (inactive) across all three days remains similar. Sample graphs for this timeframe are shown in Figure 7 (middle right).

The bottom of Figure 7 shows the second Monday and Tuesday. These days show almost no morning activity (also see Figure 7 (bottom left)) and normal midday activity. Early afternoon shows midrange and high activity for Monday and Tuesday, respectively. Graphs associated with these activity levels can be seen in Figure 7 (bottom right).

As a means to compare results to a more traditional analytic, Figure 8 bottom is a timeline that captures the number of interaction events for a given graph instance in the time-varying graph (i.e., the sum of the weights). Comparing this chart to our approach in Figure 8 top, it is clear that our approach captures a different type of behavior than edge counting alone.

4.1.3 1-Dimensional Topological Features

The High School Communications dataset ultimately contains very few 1-dimensional topological features, the majority of which have low persistence. The one-time exception, which appears on the first Monday, can be seen in Figure 9. Between 11:48 am and 12:48 pm, a high-persistence 1-dimensional pattern appears in the graph. The nodes of the graph are parameterized using that feature and visualized using a cyclic rainbow colormap. The graph shows a large tunnel (loop) toward the upper left.

Figure 8: Top: Persistent homology timeline for the first Monday and Tuesday of the High School Communications dataset. Bottom: Timeline counting the number of events (sum of all weights) in each graph instance. The timeline shows how different features can be identified in our approach as compared to edge counts alone.

Figure 9: Timeline of the High School Communications dataset for 1-dimensional features. The timeline was generated by comparing the commute-time features using bottleneck distance. The single outlier is a graph with a high persistence cycle. To highlight that feature, the graph is parameterized and visualized with a cyclic rainbow colormap [54].

4.2 EU Research Institution E-Mail

The EU Research Institution E-Mail [42] dataset is an anonymized time-varying graph tracking e-mails between members of a large European research institution. We have used the smaller of the available networks containing 986 nodes and 332,334 temporal edges. The graph tracks the activity for 803 days. A period of about 200 days is missing toward the end of the dataset, so we have analyzed the first 500 days. A single graph instance is created per day and shared 45% overlap with neighboring days. Once again, edge weight is chosen by counting the number of communications between vertices in a graph instance.

4.2.1 Bottleneck vs. Wasserstein Distance

The bottleneck and Wasserstein distances both capture important but distinct differences among sets of topological features. Intuitively, the bottleneck distance (\(p = \infty\)) captures the most perturbed topological feature (or the extreme behavior); while the Wasserstein distance (\(p = 2\)) captures the perturbation across all features (or the average behavior). Figure 10 shows how this difference impacts the analysis of the EU E-Mail dataset. For 0-dimensional (Figure 10(a)) and 1-dimensional (Figure 10(b)) bottleneck distances, the result is noisy; as the value captured has the most variation. For 0-dimensional (Figure 10(c)) and 1-dimensional (Figure 10(d)) Wasserstein distances,

\[\text{http://snap.stanford.edu/data/email-Eu-core.html}\]

Figure 10: Comparing shortest-path bottleneck ((a) and (b)) and Wasserstein ((c) and (d)) distance on 0-dimensional ((a) and (c)) and 1-dimensional ((b) and (d)) features in the EU E-Mail dataset. Since bottleneck distance captures the most perturbed feature, the result may be noisy. Wasserstein distance captures variation across all features in the graph, resulting in a smoother pattern.
the result is smoother, since it encodes the perturbations across all features. For our analysis of the EU E-Mail data, this property is more desirable.

4.2.2 Revealing Cyclic Patterns
Upon investigating the data, cyclic patterns were immediately apparent with all configurations of the Wasserstein distance (0- & 1-dimensional features and shortest-path) and commute-time. Figure 11 A & B show the 1-dimensional shortest-path version, where the cyclic patterns are most prominent (also see supplemental material for the complete 1-dimensional feature timeline).

It is notable that this pattern is related to the natural cycle of the week. To identify the pattern of the “standard” week, we divided the data into seven-day segments and used k-means clustering to group similar weeks. Figure 12 shows the result with five clusters. Each cluster shows a version of the typical week for this institution.

Figure 12: Clustering of the weekly behavior in the EU E-Mail dataset using the Wasserstein distance on 1-dimensional persistence diagrams based on shortest-path metric. A & B show graphs from a timeframe of normal weekly cyclic activity. C & F show timeframes of limited activity from December of 2004. E shows an unexpected boost in activity on June 13, 2004 that is correlated with the release of results for the EU Parliamentary Election. E shows a 3- to 4-week period of low activity in November and December of 2004. We could not identify any externally correlated event to explain this occurrence.

4.2.3 One-time Events
When looking at the entire timeline (see supplemental material), a number of one-time events are easily discovered. Figure 11 C & F are two such events. During these time periods, very little activity is present in the graphs. These times happen to be the last week of December and first few days of January, during the Christmas and New Year’s holidays. Figure 11 D is a one-time event that shows an extreme increase in activity for a 1- to 2-day period. After entering the date, June 13, 2004, into Google, we discovered that this day corresponds to the release day of the results for the EU Parliamentary Election. Finally, Figure 11 E shows a 3- to 4-week period of significantly decreased activity. Despite our best efforts, we could not identify a major external event that would have caused such a reduction, and since the data is anonymized, we could not identify the institution to investigate a local or internal cause.

5 DISCUSSION
In the previous section, we construct a similarity measure between two graph instances of a time-varying graph by utilizing the bottleneck or Wasserstein distance between their persistence diagrams, which encode the topological features associated with each instance. However, one might ask: why persistent homology? We argue that using topological data analysis and in particular, persistent homology, to study graphs, has complementary benefits and offers new insights. In this section, we conduct several experiments to justify our approach. In addition, we describe some intuition behind the information encoded by the persistence diagram of a graph, and the distance metric defined on them.

5.1 Persistent Diagram As a Graph Fingerprint
Conventional graph-theoretical approaches typically utilize the statistical properties of the vertices and edges, for instance, degrees, connectivity and path lengths, to describe the short range and pairwise interactions in the system. On the other hand, topological summaries, such as the persistence diagrams, are compressed feature representation of the underlying data, that can capture long-range and higher order interactions.

We test our persistence-based similarity measure against a set of desirable properties for a similarity measure on a graph (the first four conditions are introduced in [41]):

1. **Edge importance**: An edge whose insertion or deletion changes the number of connected components is more important than an edge that does not.
2. **Weight awareness**: In weighted graphs, the bigger the weight of the removed edge is, the greater the impact on the similarity measure should be.
3. **Edge submodularity**: Changing an edge in a dense graph is less important than changing an edge in an equally sized, sparse graph.
4. **Focus awareness**: Random changes in graphs are less important than targeted changes of the same extent.
5. **Node awareness**: We add an extra condition in this paper, i.e., deleting a large number of nodes in a graph has a larger impact than deleting a small number of nodes from the same graph.

We conduct several experiments on synthetic and real-world datasets to test the above conditions.

For the node awareness (property 5), we consider the graphs $BR$ shown in Figure 13 (c) top left. Each graph $n_iBR$ is obtained from the original graph $BR$ by deleting $i$ number of nodes (in blue). The bottleneck and Wasserstein distance matrices of $PD_0$ between these graphs are shown in the top of Figure 13 (a)-(b). The $PD_1$
Figure 13: Given synthetic, small exemplar graphs in (c), we study the node awareness (property 5, a-b, top) and the edge importance (property 1, a-b, bottom) on these graphs by computing the bottleneck (a) and Wasserstein distances (b) matrix between PDₜ of the corresponding graphs. All edge weights are assumed to be 1.

As shown in Figure 13 (a)-(b), top, we observe that the persistence-based similarity measure is more aware of the level of (dis)connectedness between non-adjacent nodes. Similarly, to test edge importance (property 1) against our similarity measure, we delete a set of edges from a graph LP, shown in Figure 13 (c) top right. The graph eₐLP is obtained from LP by deleting i edges (in blue). The bottleneck and Wasserstein distance matrices of PDₜ among these graphs are shown in Figure 13 (a-b) bottom. We observe that our persistence-based similarity measure is sensitive to edge deletions that change the connectivity of the graph, that is, it satisfies edge importance. Notice how the Wasserstein distance is more aware of the level of (dis)connectedness between the graphs than the bottleneck distance.

To test weight awareness (property 2), we run our test on three randomly generated, weighted graphs A₁ = (V₁, E₁, W₁), A₂ = (V₂, E₂, W₂) and A₃ = (V₃, E₃, W₃), where |V₁| = 50, |V₂| = 60, |V₃| = 70, |E₁| = 200, |E₂| = 250 and |E₃| = 300 respectively. Each is generated from the Gₘₙ random graph model, where a graph is chosen uniformly at random from the set of all graphs with n nodes and m edges (by setting n = |V| and m = |E| for 1 ≤ i ≤ 3).

The weights on the edges are drawn uniformly from (0,1,1). For each graph Aᵢ = (Vᵢ, Eᵢ, Wᵢ), we obtain a set of |Eᵢ| modified graphs Bᵢ' = (Vᵢ, Eᵢ, uᵢ) by modifying only the weight of an edge e (for all edges) in Aᵢ such that uᵢ(e) = wᵢ(e) + δᵢ, where δᵢ is drawn uniformly randomly from (4, 5); similarly, we obtain a set of modified graphs Cᵢ' = (Vᵢ, Eᵢ, vᵢ) from Aᵢ by modifying only the weight of edge e such that vᵢ(e) = wᵢ(e) + δᵢ', where δᵢ' is drawn uniformly randomly from (2, 3). Let graph eᵢ denote the graph obtained from Aᵢ by deleting an edge e. Property 2 holds when W(eᵢAᵢ, Bᵢ') = W(eᵢAᵢ, Cᵢ') ≥ 0 for all e in Aᵢ.

In Figure 14, we represent the difference W(eᵢAᵢ, Bᵢ') = W(eᵢAᵢ, Cᵢ') by plotting the points (W(eᵢAᵢ, Bᵢ'), W(eᵢAᵢ, Cᵢ')) against the weight of edge A that is deleted. For example, property 2 holds for points (0, 0) and above the diagonal (i.e., y ≥ x). Note that our similarity measure satisfies weight awareness for dimension 0 but violates the condition for dimension 1. This is because Wₚₜ(eᵢAᵢ, Bᵢ') = Wₚₜ(eᵢAᵢ, Cᵢ') for some e captures the creation or the destruction of a cycle.

To test edge submodularity (property 3), we consider a set of four graphs A, B, C and D. These graphs share the same number of nodes. Graph A is denser than graph C; while graph B and D are obtained from A and C, respectively, by deleting an edge. We test property 3 against four sets of small synthetic graphs in Figure 13 (c) bottom; the results are shown in Table 1. We see that both Wasserstein and bottleneck on PDₜ better capture the changes for dimension 2 and bottleneck on PDₜ better capture the changes for dimension 3.

Table 1: Testing edge submodularity (property 3) using the graphs from Figure 13 (c).

![Figure 14: Testing weight awareness (property 2). Points (W(eᵢAᵢ, Cᵢ'), W(eᵢAᵢ, Bᵢ')) on and above the diagonal correspond to instances where property 2 is satisfied. Three sets of graphs are represented by blue, orange, and green points respectively.](image)

![Figure 15: Testing focus awareness (property 4). Each colored curve represents a graph among three randomly generated graphs. The difference between the targeted corruption and the random corruption is plotted against the percentage of the deleted edges.](image)
The persistence diagram computation depends on the distance matrix \( G \) and the Wasserstein distance. Furthermore, the range of the y-axis for each plot has been chosen to directly comparable, the range of the y-axis for each plot has been chosen to be a particular similarity measure. Since these similarity measures are not uniformly at random; and such a perturbation is repeated 20 times to obtain (almost) unbiased results. We perform a total of 20 perturbation steps, that is, up to 20% of edges can be deleted from the baseline.

### Similarity Measures

We compare variations among various similarity measures. Recall \( G_0 \) is the baseline graph, and \( d_0 \) is the distance matrix of its metric space representation. Let \( G_i \) be an instance of a perturbed graph at the \( i \)-th perturbation step and \( d_i \) be its distance matrix of its metric space representation. The first set of similarity measures is based on bottleneck and Wasserstein distances. We examine the bottleneck distance \( W \) and the Wasserstein distance \( W_t \) between the 0- and 1-dimensional persistence diagrams associated with \( G_0 \) and \( G_i \), respectively. The second set of similarity measures is based upon matrix norms on the distance matrices. We measure the \textit{matrix max norm}, that is, \( \|d_i - d_0\|_{\text{max}} \), where \( \|A\|_{\text{max}} := \max_{i,j} |a_{ij}| \) for a matrix \( A \). We also measure the \textit{matrix Frobenius norm}, that is, \( \|d_i - d_0\|_F \), where \( \|A\|_F := \sqrt{\sum_{i,j} (a_{ij})^2} \).

### Experimental Results

Figure 16 shows our experimental results. Figure 16(a) uses the shortest-path distance metric in the computation of various similarity measures, and Figure 16(b) uses the commute-time distance metric.

Each sub-figure is a box-plot whose y-axis corresponds to a particular similarity measure. Since these similarity measures are not directly comparable, the range of the y-axis for each plot has been normalized to \([0,1]\) according to the maximum similarity measure across all experimental instances.

In Figure 16(a), under the shortest-path distance metric, there appears to be a linear relationship between perturbation and the bottleneck distance (and Wasserstein distance). Furthermore, the Wasserstein distance has a smaller variance than the bottleneck distance, making it suitable to study global perturbation in the data. On the other hand, similarity measures based on matrix norms are relatively unstable. Both max norm and Frobenius norm show large fluctuations and variance, making them less suitable for analysis. Moreover, these measures completely fail when the perturbed graph becomes disconnected, which is not an issue for our approach.

In Figure 16(b), under the commute-time distance, we observe that the persistence-based measure appears to be less noisy and more stable than the shortest-path distance metric.

### 6 Conclusion

Time-varying graphs are becoming increasingly important in data analysis and visualization. In this paper, we address the problem of capturing and visualizing structural changes in time-varying graphs using techniques originated from topological data analysis, in particular, persistent homology. We provide a simple and intuitive visual interface for investigating structural changes in the graph using persistence-based similarity measures.

There are many on-going and future research avenues based upon our approach. For example, in our work, we restrict topological feature extraction to Rips filtrations. Other types of filtrations, such as clique filtration \([55]\), can be used to analyze and understand time-varying graphs.

One interesting question that arises in our approach is how best to convert edge weights into distances. The conventional wisdom is that the stronger the communication between nodes (i.e., higher edge weight), the closer together they should be. However, we have some evidence that such a conversion may not always capture the underlying structural changes, and sometimes, an inverse weighting scheme may be more effective.

It would also be interesting to perform a systematic comparison of a wide range of similarity measures in the study of time-varying graphs \([45]\), in particular, to see how these different measures can complement one another in enriching our current visual analytic framework. A final note is that we hope the work described here could inspire more graph visualization research to move beyond graph-theoretical measures and venture into techniques from topological data analysis.

### ACKNOWLEDGEMENTS

This work was supported in part by NSF IIS-1513616.

### References


