# Exploring Evolution of Dynamic Networks via Diachronic Node Embeddings

Jin Xu, Yubo Tao, Yuyu Yan, and Hai Lin

**Abstract**—Dynamic networks evolve with their structures changing over time. It is still a challenging problem to efficiently explore the evolution of dynamic networks in terms of both their structural and temporal properties. In this paper, we propose a visual analytics methodology to interactively explore the temporal evolution of dynamic networks in the context of their structure. A novel diachronic node embedding method is first proposed to learn latent representations of the structural and temporal features of nodes in a vector space. Diachronic node embeddings are then used to discover communities with similar structural proximity and temporal evolution patterns. A visual analytics system is designed to enable users to visually explore the evolutions of nodes, communities, and the network as a whole in terms of their structural and temporal properties. We evaluate the effectiveness of our method using artificial and real-world dynamic networks and comparisons with previous methods.

**Index Terms**—Graph/Network Data, Data Transformation and Representation, Sequence of Data, Dimensionality Reduction

## 1 Introduction

Networks describe the relationships between objects, e.g., social relationships within a group of friends, conversations and interactions within a population, and collaborations of researchers in publication records. Because network structures evolve over time, uncovering the structural properties in a single timestep, such as communities and centrality, as well as temporal properties over time, such as temporal patterns and shifts, can provide insights into the dynamic network evolution. For instance, the structure of a collaboration network may evolve over time, as some new research fields emerge and/or others disappear. Meanwhile, a researcher's collaboration relationships also evolve over time as he/she leaves and joins research fields. Understanding the temporal patterns of researchers' collaboration relationships in the context of the evolutionary structure can reveal their career development patterns. However, one of the biggest challenges in dynamic network analysis and visualization is finding the optimal trade-off between revealing the structural properties in any timestep and the temporal properties over time [1], [2].

Previous works can be summarized into three categories [3]: (a) animation-based methods that flip through snapshots (a network in one timestep) and highlight changes between adjacent timesteps, (b) timeline-based methods that juxtapose snapshots into the same view according to the timeline, and (c) projection-based methods that characterize network structures with representations in a vector space and then project these representations for automatic detection and summarization. Animation-based methods are adequate to show the structure of each snapshot but inadequate to track the temporal patterns over a long time period. With timeline-based methods, it is difficult to determine the number of snapshots, as few snapshots may lack temporal details, but many snapshots may result in a loss of structural details. Furthermore, these two types of methods rely on human cognition ability. Projection-based methods have been proven successful in the temporal evolution characterization of networks, but they fail to depict the structural properties simultaneously. It is essential to interpret how the network evolves while maintaining the overview of network structures [4]. Currently, there is a lack of projection-based methods for characterizing and presenting both the structural and temporal properties for network evolution analysis. Moreover, communities with similar temporal patterns are not fully considered in previous works, but are vital for summarizing the temporal properties of the evolution of dynamic networks and reduce the cognitive burden of visual clutter in dynamic network analysis and visualization.

To this end, we aim to summarize the temporal evolution in the context of network structures and support the exploration from the node, community, and network levels to facilitate comprehensive dynamic network analysis. For the first requirement, the node representations should preserve not only the structural proximity of nodes in each timestep but also the temporal continuity of nodes across time. Moreover, it is desirable to generate robust representations in the presence of noise. For instance, a node may have similar structural characteristics across time with different but similar neighbors, such as neighbors that belong to the same community. To meet these needs, we introduce node embeddings based on neural networks into dynamic network analysis and visualization for the first time. Owing to their effectiveness in capturing the structural characteristics of nodes, i.e., neighborhood similarity and community membership, node embeddings make the robust detection in static networks viable [5], [6]. However, node embeddings have not been fully explored to preserve temporal continuity for the visual analysis of dynamic networks. We rely on neural networks and embedding alignment to generate

• *Jin Xu, Yubo Tao, Yuyu Yan, and Hai Lin are with State Key Laboratory of CAD&CG, Zhejiang University. E-mail: jinxu.zju@gmail.com, taoyubo@cad.zju.edu.cn, yanyuyu001@gmail.com, lin@cad.zju.edu.cn. Yubo Tao is the corresponding author.*

*Manuscript received March 18, 2018; revised December 9, 2018.*

node embeddings, representing nodes as dense vectors in a high-dimensional vector space with both the structural and temporal features preserved. These generated node embeddings are called **diachronic node embeddings** in this paper. For the other requirement, because the generated diachronic node embeddings can characterize node evolutions, we can cluster diachronic node embeddings to extract communities from the structural and temporal properties. From these two requirements, our method is novel and provides new insights for dynamic network analysis.

In this paper, we propose a novel visual analytics methodology based on diachronic node embeddings for identifying the structural and temporal properties of nodes, communities, and networks. A diachronic node embedding generation method is first proposed to represent each node as a sequence of dense vectors in a high-dimensional vector space. To exploit diachronic node embeddings, a node classification method is introduced to identify dynamic and stable nodes, and two clustering methods are provided to cluster nodes with structural properties to uncover the structure of the network as well as temporal properties to summarize the temporal evolution of the network. Based on the proposed methodology, we design an interactive visual analytics system to enable users to effectively explore the temporal evolution of dynamic networks in the context of their structural properties. Two new visual designs are proposed: a stream-based design for showing an overview of the temporal evolution in the context of the structure, and a mapping time-to-color glyph for exploring the dynamic network in a specific time period.

In summary, this paper offers three main contributions:

- A diachronic node embedding generation method is proposed to represent the structural characteristics of each node in each timestep as a point in a high-dimensional vector space while preserving the structural proximity in the network and the temporal consistency of stable nodes over time.
- Two diachronic node-embedding-based clustering methods, which cluster nodes with similar structural and temporal properties, are provided for the exploration of dynamic networks from the community level.
- An interactive visual analytics system based on our methodology enables users to intuitively explore dynamic network temporal evolution in the context of the structure.

## 2 Related Work

As our work is a node-embedding-based visual analytics methodology for dynamic networks, we review related works from two perspectives: node embedding and dynamic network visualization.

### 2.1 Node Embedding

Node embedding represents the nodes of a network as vectors in a vector space and is of broad use in network analysis. It converts network analysis into the analysis of low-dimensional vectors and makes machine learning models applicable to tasks such as node classification, clustering, and link prediction [7]. These representations can be viewed as encoding or projecting nodes into a latent space, where geometric relations in the latent space correspond to connections in the original network [8].

Early methods of node embedding typically exploited the spectral properties of various matrices representing the connections of nodes in a network, such as a node adjacency matrix, Laplacian matrix, node transition probability matrix, and Katz similarity matrix. They would construct a similarity graph for a set of high-dimensional points based on the neighborhood, and nodes in the graph are embedded in a low-dimensional vector space where connected nodes are closer to each other. Laplacian eigenmaps [9] and Locally Linear Embedding (LLE) [10] are examples of algorithms based on this rationale. From the linear algebra perspective, these methods can be considered as dimensionality reduction techniques, and hence several linear and nonlinear dimensionality reduction methods are capable of node embedding, such as PCA [11] and IsoMap [12]. However, these methods have limited scalability and are time-consuming [7]. Moreover, they are not robust to the diverse patterns observed in networks [6].

Recent advances in distributed representation learning for natural language processing (NLP) have inspired the development of feature learning of discrete objects such as words. Particularly, Mikolov et al. [13] proposed a neural language model, called the Skip-Gram model, to learn distributed feature representations for words. A surge of research based on these neural language models has been proposed to generate node embeddings. Aiming to represent a network as a "document" composed of an ordered sequence of words, these techniques generally sample walks of nodes from the underlying network and turn a network into an ordered sequences of nodes. Perozzi et al. [5] first developed a random-walk-based method, DeepWalk, to learn representations for nodes by generalizing neural language models to preserve the higher-order proximity between nodes. Walklets [14], focusing on multiscale representation learning, modified the random walk strategy in DeepWalk by skipping over some nodes in the network. Node2vec [6] introduced biased random walks to smoothly interpolate between walks that are more akin to breadth-first or depth-first searching to preserve the community structure and structural equivalence between nodes.

These methods focused on node embedding for static network analysis. For dynamic networks, we propose a novel method to generate diachronic node embeddings based on node embeddings and alignment, which is introduced into dynamic network analysis and visualization for the first time.

### 2.2 Dynamic Network Visualization

Dynamic network visualization has been extensively studied for a long time. There are three major visual methods: animation, timeline, and projection-based methods.

Animation-based methods flip through snapshots [15]. Each snapshot is usually visualized based on node-link diagrams [16] to show its structural properties. The key idea of these methods is to allow users to better preserve a mental map in each snapshot [17], [18]. Animation techniques are desirable to reduce the complexity of dynamic

networks to facilitate visualizing the structure of networks, but are inadequate to support detailed network analysis and interpretation of temporal properties [17].

Timeline-based methods display snapshots in any timestep in a static image. This allows these methods to provide a better temporal overview of dynamic networks. Some timeline techniques leverage small multiples that generate snapshots in every timestep, and matrices [19], node-link diagrams [20], and node-link based diagrams [21] have been used for snapshot structure presentation. For small multiples, it is difficult to determine the number of multiples in the visualization, as larger images show more structural properties of the network in each timestep, but more temporal properties can be revealed when using smaller images. Other approaches based on node-link diagrams incorporate the timeline into the diagram [1]. These methods make the changes in edges over time stand out. For example, Burch et al. [4] leveraged a bipartite graph layout to place nodes with fixed vertical positions and computed a scalar density field to aggregate overlapping edges and provide an overview of very long graph sequences. These methods are capable of temporal property visualization, but temporal pattern discovery is limited by human cognition.

For automatic summarization, recent efforts try to convey graph or node representations for visualization. Van den Elzen et al. [22] reduced snapshots to points in a high-dimensional vector space by deriving representations of snapshots from adjacency matrices and projected snapshots to the 2D space for network state discovery. Dal Col et al. [3] represented nodes by the coefficients of graph wavelets for evolutionary local change discovery. These methods can present a summary of the temporal evolution of networks, but it is difficult to understand the temporal evolution as they fail to interpret temporal patterns in the context of the structure. Our method stems from the observations of the above projection-based methods. Most works do not adequately consider the simultaneous summary of the structural and temporal properties of dynamic networks, and the summary is able to bridge the gap between exploring the structural and temporal properties. Cui et al. [2] focused on the same challenge as this paper and proposed a new toolkit for capturing temporal patterns of a group of nodes while maintaining network structures. However, this method needs the node category information. We exploit diachronic node embeddings for better structural and temporal property preservation and design a system for the interactive evolutionary exploration of dynamic networks.

Furthermore, in order to facilitate detection of the structure in any timestep and temporal patterns over time, clustering methods are incorporated into our methodology. Various methods have been proposed to find community structures in any timestep [23], such as random-walk-based methods [24], [25]. We employ a k-means-based method to characterize the community structures in stable states. In contrast, there are few works concerning communities with similar temporal patterns. Most previous methods extract community structures and focus on the emergence of communities over time [26], [27], [28]. However, community detection with similar temporal patterns is also vital. A new method based on diachronic node embeddings is proposed in this paper to detect temporal communities.

## 3 Overview

The proposed visual analytics methodology enables an integrated exploration of evolution from the node, community, and network levels to help analysts understand the structural and temporal properties of dynamic networks. As our method is based on node embeddings, it involves two main issues: 1) how to generate diachronic node embeddings that are able to preserve both high-order proximity and temporal consistency and 2) how to use diachronic node embeddings to analyze the evolution of nodes, communities, and networks. Fig. 1 shows the high-level workflow of our methodology.

At the dynamic network construction stage, we first produce a temporal sequence of snapshots from the dynamic network data (A). Next, temporal node embeddings are generated by a neural language model, and the structural characteristics of each node in each snapshot are then represented by a point in a high-dimensional vector space with geometric relationships reflecting the structure of the original snapshot (B). Because the generated temporal node embeddings of each snapshot are not in a common vector space owing to the randomness of representation learning, we apply an embedding alignment method to align the temporal node embeddings, and then diachronic node embeddings are generated to represent the evolution of each node. These diachronic node embeddings efficiently convert the evolutions of nodes into high-dimensional trajectories chronologically through their corresponding points (C).

In order to exploit the diachronic node embeddings to facilitate the exploration of the structural and temporal properties from the node, community, and network levels, we provide three methods: node classification (D), structural node clustering (E), and temporal trajectory clustering (F). Because the comparison of the diachronic node embeddings is reduced to simple vector operations between two corresponding points in a vector space, the node classification method classifies a node as a dynamic or stable node by identifying whether it has significant changes indicated by the proximity between its diachronic node embeddings in adjacent timesteps. Structural node clustering groups stable nodes into communities where they have close relationships measured by the proximity between their diachronic node embeddings in a time period. Temporal trajectory clustering discovers communities of all the nodes with similar temporal patterns according to the proximity between their evolutionary trajectories.

Finally, we design a visual analytics system to explore the evolutions of nodes, communities, and networks via dimensionality reduction and time mapping methods (G).

## 4 Diachronic Node Embeddings

This section introduces the generation of diachronic node embeddings, including dynamic network construction, embedding generation, and alignment.

### 4.1 Dynamic Network Construction

Dynamic networks generally contain timestamped activities, e.g., email communication files. We model a dynamic network as a sequence of $T$ snapshots, where each snapshot

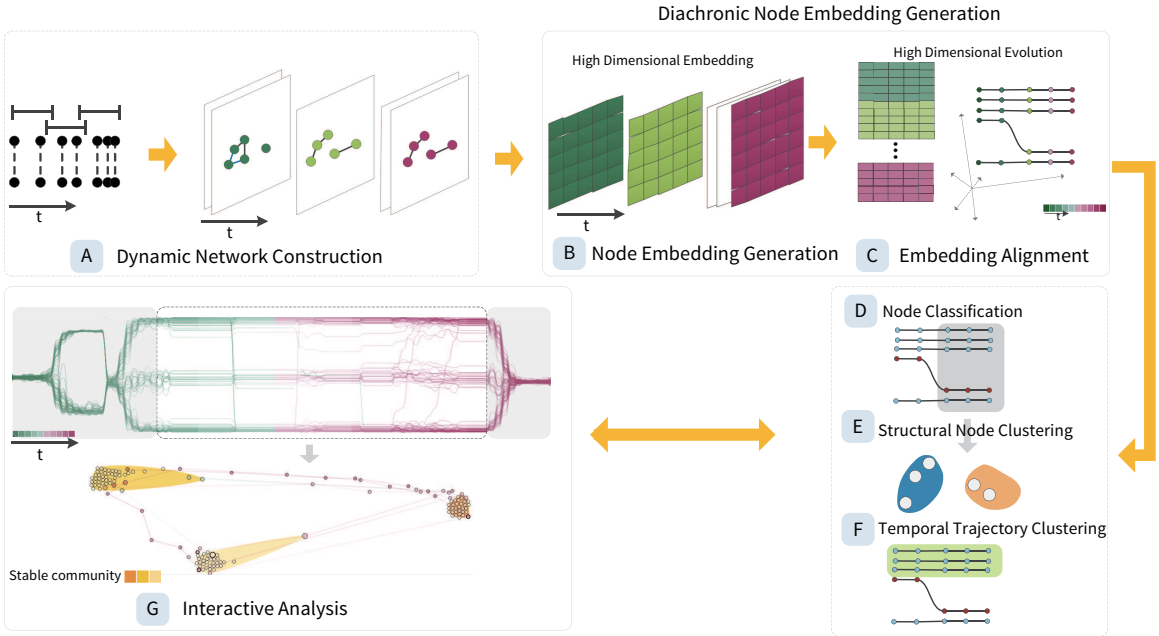


Fig. 1: Workflow of our methodology. We first extract a sequence of snapshots from the dynamic network data (A). Next, diachronic node embeddings are produced based on node embeddings (B) and embedding alignment (C). Then, we provide three methods: node classification (D), structural node clustering (E), and temporal trajectory clustering (F), to facilitate the analysis of these evolutions of nodes and communities. Finally, the structural and temporal properties can be interactively analyzed via dimensionality reduction and time mapping methods (G).

is a directed or undirected graph $G_t = (V_t, E_t, t)$ and $T$ is the number of timesteps. The snapshot, $G_t$, has an associated time-window $[t_i - \omega/2, t_i + \omega/2)$ with the width $\omega$. We assume that $t_{i+1} - t_i = \Delta t$ is equidistant for all $i$ and $\Delta t$ can be set to any time period, usually representing weeks, days, hours, etc. The graph edge set, $E_t$, consists of all the edges during the time window. We simply set the number of occurrences of an edge in the edge set as the weight of the edge in snapshot $G_t$. The node set, $V_t$, is the union of the nodes that occur in the edge set.

Because the effectiveness of our method would decrease when missing patterns of the dynamic network owing to the hard boundaries, we allow time-windows to overlap with neighboring time-windows. For example, we create a snapshot every 7 days ($\Delta t$), and the time-window width $\omega$ can be chosen larger than 7 days, such as 14 days.

## 4.2 Embedding Generation

We generate node embeddings to represent the structural characteristics of nodes in each snapshot. Learning embeddings for nodes has been widely studied in static network analysis as discussed in Section 2.1, while they have not been fully explored for the visual analysis of dynamic networks. We propose a novel method to generate temporal node embeddings based on a neural language model.

A neural language model embeds words in a high-dimensional vector space. Since the Skip-Gram model [13] has been widely used in node embedding generation, we adopt this model to learn node embeddings. We first provide an overview of the Skip-Gram model, and then illustrate the adaptation of the Skip-Gram model for temporal node embeddings generation.

### 4.2.1 *Skip-Gram Model*

The Skip-Gram model is trained by a large amount of text to produce high-dimensional representations of words incorporating both syntactic and semantic information. The Skip-Gram model uses the given word to predict the surrounding words.

In particular, a training corpus is a sequence of sentences composed of training words that belong to a vocabulary $\mathcal{V}$ whose size is $|\mathcal{V}|$. The context of a word consists of both the $n$ words before and after the target word $w_i$ in a sentence and the context-window size is $2n+1$. We associate word $w_i \in \mathcal{V}$ with a vector $z_{w_i} \in \mathbb{R}^d$ and context $w_j$ with a vector $\hat{z}_{w_j} \in \mathbb{R}^d$, where $d << |\mathcal{V}|$. The Skip-Gram model tries not only to maximize the occurrence probability of word pairs $(w_i, w_j)$ that occur in the training corpus, but also to minimize the occurrence probability of any other word pair $(w_i, w_m)$ that does not occur in the training corpus. To reduce the time complexity, "negative sampling" is applied to randomly draw $w_m$ from some distribution $P$, which is typically viewed as the unigram distribution over $\mathcal{V}$ (normalized word counts) [29]. Therefore, the objective function is defined as follows:

$$E = \frac{1}{|\mathcal{V}|}\sum_{i=1}^{|\mathcal{V}|}\Big(\sum_{-n\leqslant j\leqslant n, \neq 0}(log(\sigma(z_{w_i}^T \hat{z}_{w_{i+j}})) + N(z_{w_i}))\Big), \tag{1}$$

$$N(z) = \sum_{m=1}^{M}(\mathbb{E} \sim P(1 - log(\sigma(z^T \hat{z}_w)))), \tag{2}$$

where $M$ is the number of negative samples for each word (usually between 5 and 20) and $\sigma$ is the sigmoid function $\sigma(x) = \frac{1}{1+exp(-x)}$. The first term encourages word-contexts that co-occur to have a high likelihood, and are

consequently close in the embedding, whereas the second term $N(z)$ encourages words to be sufficiently distinct from randomly drawn contexts, and consequently far apart in the embedding.

#### 4.2.2 Temporal Node Embedding Generation

**Algorithm 1** Generation$(G_t, r, l, n, d)$

**Input:**
snapshot $G_t(V_t, E_t, t)$, walks per node $r$, walk length $l$
context-window size $2n+1$, node embedding size $d$
**Output:**
matrix of node embeddings $U_t \in \mathbb{R}^{|V_t| \times d}$

1: $walks = Empty$
2: **for** $i = 0$ to $r$ **do**
3: $\mathcal{O} = Shuffle(V_t)$
4: **for each** $v_i \in \mathcal{O}$ **do**
5: $walk$ = WeightedRandomWalk $(G_t, v_i, l)$
6: Append $walk$ to $walks$
7: **end for**
8: **end for**
9: **if** $t == 0$ **then**
10: Initialize $U_t$ randomly
11: **else**
12: $U_t = U_{t-1}$
13: **end if**
14: SkipGram$(U_t, walks, n)$

The basic idea of neural-model-based methods for static networks is to take nodes as words and random walks as sentences in a document, and then apply the Skip-Gram model to process the set of random walks to generate node embeddings. Thus, there are two components, the random walk generation and the Skip-Gram algorithm. Owing to the different objectives, various walk-strategy-based methods have been proposed, such as DeepWalk [5] based on unbiased random walks and node2vec [6] based on biased random walks. As our goal is to preserve the higher-order proximity between nodes and temporal continuity of nodes, we extend DeepWalk to dynamic networks.

The pseudocode is given in Algorithm 1. For each node in every snapshot, we generate $r$ random walks with a fixed length $l$. For weighted networks, we replace the random walk strategy in DeepWalk with a weighted-random-walk strategy to control the likelihood of visiting a neighbor node according to the weight of the edge (lines 1-8). This would better preserve the community structure in weighted networks. These walks are treated as sentences.

For the walks of each snapshot, we apply the Skip-Gram model to learn node embeddings with the same objective function as in Equations 1 and 2. The node embeddings are generally initialized randomly in static networks. However, to preserve the temporal continuity of the node embeddings for dynamic network analysis, we initialize the node embeddings $U_t$ in the snapshot $G_t$ ($t > 0$) with the generated node embeddings $U_{t-1}$ of $G_{t-1}$ (lines 9-14).

Based on the experimentation in DeepWalk, the number of walks per node $r$ is 10, the walk length $l$ is 40, the context-window size is 5 ($n = 2$), and the node embedding size $d$ is 64 in our experiments. We compare the random initialization method and the proposed initialization method using the email communication dataset (discussed in Section 6.3). The temporal continuity is measured by the mean Euclidean distance between pairs of node embeddings of each node in adjacent timesteps. The results are 0.98±0.01 and 0.37±0.01 for the random initialization method and our method, respectively. Thus, we find that a good initialization better preserves the temporal continuity.

### 4.3 Embedding Alignment

Although we initialize node embeddings with the learnt result of the previous snapshot to keep them temporally consistent, they may still not be comparable with one another. The reason for this is that the learning process is inherently stochastic and the resulting embedding sets are invariant under rotation [30]. Therefore, diachronic node embeddings are achieved by further aligning these temporal node embeddings to one vector space.

Different methods have been employed for embedding alignment. Kulkarni et al. [31] leveraged a local linear regression by fitting a sample of vectors from the neighborhood of a focal word and minimizing the mean squared error. This method poses a potential drawback: it must be applied separately for each focal word. Hamilton et al. [32] applied orthogonal Procrustes, which is similar to linear regression, aiming to learn a transformation of one vector space onto another and minimizing the distance between pairs of vectors, to the full space by using a different mathematical method. As this can achieve the best rotational alignment, we adapt orthogonal Procrustes to further align the node embeddings of dynamic networks.

When aligning the vector space of $G_i$ to the base vector space of $G_b$, the intersection of vertex set $V_{in}$ is first extracted. Two matrices, $M_i \in \mathbb{R}^{|V_{in}| \times d}$ and $M_b \in \mathbb{R}^{|V_{in}| \times d}$, are established by stacking node embeddings in $G_i$ and $G_b$, respectively. We then leverage orthogonal Procrustes to find an orthogonal matrix $\Omega$ that most closely maps $M_i$ to $M_b$. Specifically, we aim to minimize $||M_i\Omega - M_b||_F$, where $||*||_F$ is Frobenius norm.

The assumption of embedding alignment is that two snapshots should have small changes of structure. We have different choices for the base vector space. One is to select one vector space, such as in the timestep with the most nodes, and all the other vector spaces are aligned to the selected vector space. This choice would achieve better performance in dynamic networks with small changes. Another choice is to select an adjacent vector space as the base vector space and multiply the corresponding orthogonal matrices to align all the vector spaces to the vector space in the first timestep. As adjacent timesteps may have small changes, this choice would achieve the least error when aligning two adjacent vector spaces. However, as the cumulative error increases, this method is inadequate when the dynamic network has too many timesteps. The last choice is to select a base vector space every several timesteps, such as every three or five timesteps. In contrast to the second choice, this method can reduce the cumulative error.

We compare the accuracy of the three choices for the base vector space using the email communication dataset: the vector space of the snapshot with the most nodes, the adjacent vector space, and the vector space every two

timesteps. The alignment errors of the three choices are 0.27±0.01, 0.23±0.01, and 0.30±0.01, respectively. We find that the adjacent vector space achieves the best embedding alignment for this dynamic network.

Generally, we can select the adjacent vector space as the base vector space for embedding alignment. In practice, the base vector space can be chosen based on the domain knowledge and features of dynamic networks.

## 5 Node Classification and Clustering

This section describes the utilization of diachronic node embeddings to facilitate the dynamic network exploration from different levels.

### 5.1 Node Classification

Diachronic node embeddings have been generated and we can use them to classify stable and dynamic nodes in the dynamic network. Intuitively, stable nodes are more likely located at the same position in a vector space or have small changes over several timesteps. In contrast, dynamic nodes have significant changes, and their node embeddings form evolutionary trajectories in a vector space. This classification allows users to focus on dynamic nodes to reduce the analysis burden when exploring large dynamic networks.

The distance between two node embeddings can be quantified by many methods, such as the cosine similarity and the Euclidean distance. Since the Euclidean distance is widely used for NLP tasks, we use it to measure the distance between two node embeddings. Given two node embeddings, $u_x$ and $u_y$, the Euclidean distance between them is defined as:

$$d(u_x, u_y) = ||u_x - u_y||_2. \quad (3)$$

The distance between two node embeddings can be used in node classification, structural node clustering, and temporal trajectory clustering.

The evolution of a node $v_i$ in a selected time period $[t_j, t_{j+m}]$ is represented by a temporal sequence of diachronic node embeddings $(u_i^j, u_i^{j+1}, ..., u_i^{j+m})$. The evolution value of the node between adjacent timesteps can be evaluated by $d(u_i^{p-1}, u_i^p)$, where $p \in [j+1, j+m]$. If any evolution value between adjacent timesteps is above the threshold $\theta$, then this node is classified as a dynamic node. Otherwise, this node is a stable node in this time period. The dynamic node set $D$ is defined as:

$$D = \{v_i | d(u_i^{p-1}, u_i^p) > \theta, \exists p \in [j+1, j+m]\}. \quad (4)$$

Different dynamic networks may require different thresholds to classify nodes. Thus, the threshold is interactively specified by the user during dynamic network analysis.

### 5.2 Structural Node Clustering

Many dynamic networks evolve over time but still have stable states. **A stable state of a dynamic network** is regarded as a state with a few nodes evolving and most nodes remaining unchanged during a time period $[t_j, t_{j+m}]$. It is desirable to reveal the network structure (communities of stable nodes) and evolutions of dynamic nodes (evolutionary trajectories among communities) in stable states.

For the network structure, there are many methods on static community detection [33]. As node embedding based on k-means clustering has consistently better accuracy for static networks [34], we compute a mean diachronic node embedding of each stable node in a time period and perform k-means clustering on these mean vectors of stable nodes to extract the stable communities as the network structure.

In a stable state of the network, dynamic nodes are still evolving in the time period, and we highlight their evolutionary trajectories among the stable communities. Dynamic nodes also have stable states. **A stable state of a dynamic node** is described as a state, belonging to a stable community with small changes. To characterize their stable states, we identify whether a dynamic node $v_i$ belongs to a community in each timestep by the distance $d(u_i^p, u_c)$, where $u_c$ is the centroid of a stable community. $d_c$ is the maximum distance between its member nodes and $u_c$. If $d(u_i^p, u_c) < d_c$, $v_i$ belongs to the community and stays in a stable state in the timestep $t_p \in [t_j, t_{j+m}]$. Otherwise, the dynamic node is in a transition pattern, changing from one community to another or is occurring anomalously.

### 5.3 Temporal Trajectory Clustering

As there are various temporal behaviors of nodes, discovering communities of nodes with similar evolutionary trajectories enables the summarization of the temporal patterns of dynamic networks. An evolution of a node is represented by a temporal sequence of diachronic node embeddings. This temporal sequence can be regarded as a trajectory through these points in a high-dimensional vector space. Therefore, we aim at clustering similar trajectories.

K-Nearest Neighbor Graph (K-NNG) is used to construct partitions with similar trajectories for the following reasons. First, K-NNG does not require the number of clusters as an input. Second, because some trajectories may be abnormal and these trajectories should not be allocated to communities, K-NNG is adequate for this problem. Finally, considering the time efficiency and interactions with users, K-NNG is able to provide different scales of clustering results by changing the parameter $K$.

First, we need to define the distance function between two trajectories. Different distance functions have been proposed for trajectory clustering according to different analysis goals, such as similar routes and similar directions [35]. Because we focus on identifying similar trajectories, we compute the Euclidean distance of the pair of points in the same timestep and take the average distance as the distance between two trajectories.

The basic idea of K-NNG is to find the $K$ most similar neighbors for each trajectory and then add its neighbors to the partition to which the trajectory belongs. In order not to allocate abnormal trajectories to communities, we set a distance threshold $\theta_{K-NNG}$ (0.3 in our experiment) to determine whether a trajectory's neighbors can be added to a partition. Obviously, the higher the value of $K$ is, the smaller is the number of temporal communities. Different values of $K$ can uncover temporal patterns of dynamic networks on different scales. The evolution of a temporal community can be represented by the average trajectory of its member nodes. Therefore, a temporal community summarizes similar temporal patterns of its member nodes.

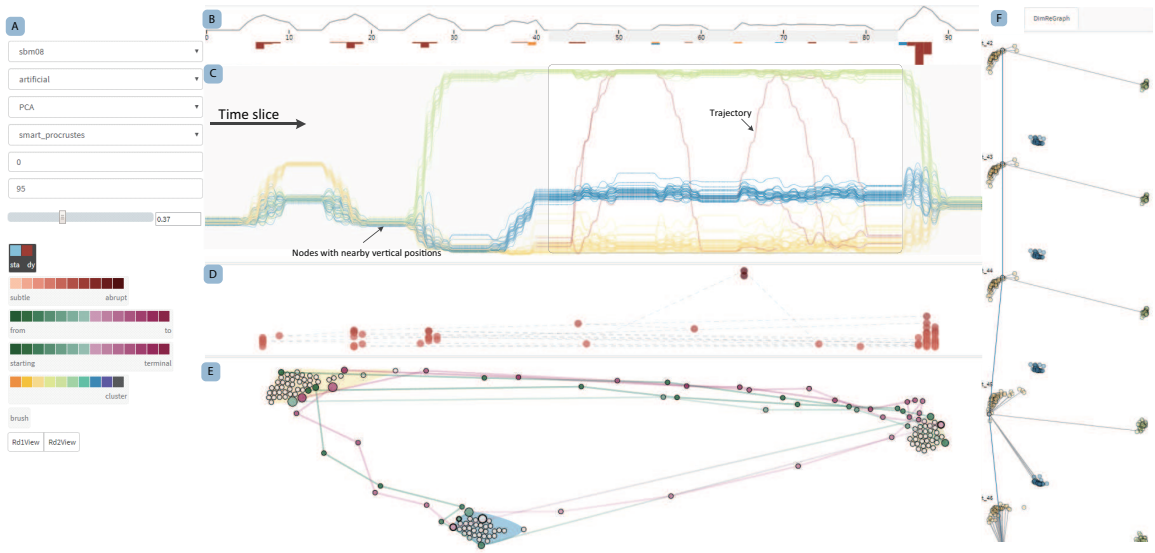


Fig. 2: Our visual analytics system for dynamic network analysis: (A) the control panel allows users to adjust different parameters, (B) the statistical view provides a high-level evolution overview of the network, (C) the trend view depicts an overview of the structural and temporal properties of nodes and communities, (D) the node view shows dynamic nodes in each timestep, (E) the structure view presents the structure of the network in a time period, and (F) the snapshot view displays the original network structures.

## 6 User Interface

To enable visual exploration of dynamic networks, we propose a visual analytics system for identifying temporal patterns in the context of the structure from the node, community, and network levels based on diachronic node embeddings. In particular, we propose five goals (G) for our system, which are inspired by the dynamic network tasks taxonomy by Bach et al. [18] and Dal Col et al. [3]:

**G1: The evolution analysis of the network as a whole.** An evolution summary of the whole network over time could guide the user to focus on the important time periods. Because each node in every timestep is represented as a point in a high-dimensional vector space, we can sum the evolution change of each node to generate the evolution change of the network.

**G2: The evolution analysis of dynamic nodes.** Uncovering dynamic nodes in each timestep could facilitate identifying which nodes are changing in each timestep. Furthermore, showing the evolutions of nodes could reveal how a node evolves over time, including stable states and transition patterns among stable communities.

**G3: The temporal analysis of temporal communities.** Characterizing temporal communities facilitates the summarization of the temporal patterns of the network. Nodes are clustered with similar temporal patterns as temporal communities based on the temporal trajectory clustering, and the temporal patterns should be analyzed from the community level.

**G4: The structural analysis of stable communities.** In a stable state of the dynamic network, stable communities—groups of nodes that often correspond to functional modules—are crucial to understanding the network structure. Nodes are clustered with close structural properties as stable communities based on the structural node clustering in a stable state, and the structural properties should be visually explored from the community level.

**G5: Relate the analysis to the original networks.** It is vital for users to be able to retrieve the original dynamic networks at any stage of the analysis process. Moreover, this can also verify the analysis results of our method.

These goals are reflected in the proposed visual analytics system with six interactive views in Fig. 2: control panel (A), statistical view (B), trend view (C), node view (D), structure view (E), and snapshot view (F). They are connected by brushing and linking to allow for flexibly exploring how the nodes, communities, and networks evolve over time. The statistical view provides a high-level overview of the evolution of the network as a whole (G1). The node view facilitates dynamic node identification (G2). The trend view is the fundamental view, providing an overview of the structural and temporal properties simultaneously from the node and community levels (G3). The structure view enables users to focus on the structure of the network and the evolutionary trajectories of dynamic nodes (G4). The snapshot view provides the original network (G5).

### 6.1 Statistical View

The statistical view provides a high-level overview of the evolution of the network as a whole and the statistical information of nodes to support quick identification of the important time periods, as shown in Fig. 2(B).

The evolution value of the network in each timestep is defined as the sum of the evolution value of each node between adjacent timesteps. When the evolution value is low, the network is in a stable state. When it is high, the network is undergoing a large change.

We employ the widely used line chart to present the time-evolving trend of the evolution of the whole network. The horizontal and vertical axes are encoded with the time and the evolution value, respectively. Dynamic nodes,

newly joined nodes, and disappearing nodes are important for network evolution analysis. Their numbers are shown at the bottom of the statistical view via stacked bars, colored in red, green, and purple, respectively. The time period of interest, such as a stable state of the network, can be selected via brushing in the line chart, and the following trend view, node view, structure view, and snapshot view are updated.

### 6.2 Node View

The node view shows the dynamic nodes with large evolution values in each timestep to reveal what is changing significantly in a time period, as presented in Fig. 2(D).

The horizontal axis is encoded with the time, which is aligned with the time in the statistical view. Because dynamic nodes are the primary focus in this view, the vertical axis from the bottom to the top is scaled to the range from the threshold $\theta$ in the node classification to the largest evolution value of nodes between adjacent timesteps. Therefore, stable nodes are not displayed in this view. A circle represents a node that has a large change compared with the previous timestep, and its color from yellow to red is used to encode the change from small to large. If a node has large changes in successive or intermittent timesteps, the circles representing the same node are connected by a solid or dashed line, respectively. This view is updated when the threshold $\theta$ changes. It is used to reveal which nodes have a dramatic evolutionary process over time.

### 6.3 Trend View

The trend view provides the temporal evolution in the context of the structure from the node and community levels, as shown in Fig. 2(C). The evolutions of nodes are high-dimensional trajectories, which are difficult to comprehend and visualize. Thus, we leverage a dimensionality reduction method to project these trajectories to a lower subspace.

Many dimensionality reduction techniques have been proposed, including principal component analysis (PCA) [36], multidimensional scaling (MDS) [37], and t-distributed stochastic neighbour embedding (t-SNE) [38]. Because a comparative review mentions that the linear dimensionality reduction technique PCA was desirable for real-world dataset analysis [39], we utilize PCA in our methodology and project all the node embeddings in all timesteps to a 1D space simultaneously.

Stream-based designs are often used to show evolutions while encoding time to the vertical or horizontal axis. We propose a stream-based trend view by mapping the projected points to the vertical axis and then positioning them along the horizontal time axis, as shown in Fig. 2(C). This allows the vertical space to present the structure of each snapshot; if two nodes have two nearby vertical positions, this indicates that they have close structural properties in the timestep. The evolution of each node is represented by a line connecting its corresponding points chronologically in a 2D space. As the color can be used to reveal patterns in the data [1], different colorings are applied in the trend view. We propose to color the lines based on the absolute time, lifetime of each node, node classification, evolution values between adjacent timesteps, stable communities, and temporal communities, and these colorings can be switched by clicking different color bars in the control panel. The parameter $K$ in the temporal trajectory clustering can be interactively adjusted to present the temporal communities in the trend view.

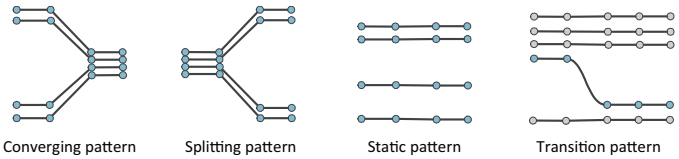


Fig. 3: Converging and splitting patterns indicate that nodes/communities gather together or fall apart, respectively. Static and transition patterns demonstrate that nodes/communities remain stable or evolve from one community to another community, respectively.

The trend view can reveal the temporal patterns of evolutions of nodes in the context of the structure, and the pattern classes are defined in Fig. 3. A converging pattern indicates that nodes/communities approach each other, such as if they tend to have frequent contacts and close relationships. A splitting pattern corresponds to the inverse converging pattern and demonstrates that nodes/communities tend to have few contacts and remote relationships. A static pattern describes a stable state of nodes/communities. A transition pattern shows that one node/community moves from one community to another community.

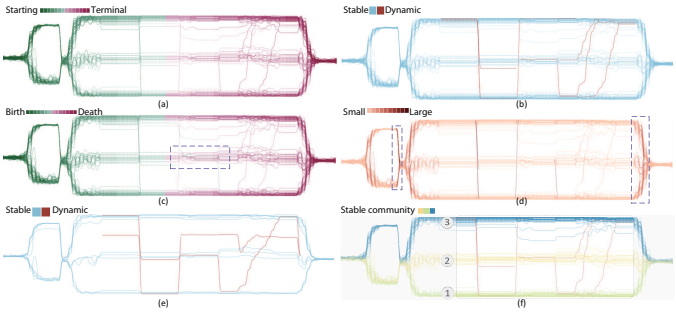


Fig. 4: Different color encodings in the trend view. The color from green to red is encoded to the absolute time and the lifetime of each node in (a) and (c), respectively. The blue and red lines represent trajectories of stable and dynamic nodes in (b) or temporal communities in (e). The color from light red to dark red represents the evolution values from small to large in (d). Stable nodes are colored by their associated stable communities in (f).

We use an artificial dynamic network to demonstrate the capabilities of the trend view. The network contains 107 nodes and 85 timesteps. First, the network consists of 100 nodes belonging to one community. Next, the community starts to split and two communities emerge. Then, the two communities start to converge into one community. After that, the community falls apart, and three communities emerge. In this stable state, there are seven newly joined nodes with different temporal patterns. Finally, seven nodes are removed, and the three communities converge into one community. Fig. 4 shows the evolutions of the artificial dynamic network with two splitting and two converging patterns. There are transition patterns during the stable state of the network, which is consistent with the construction of

the artificial dynamic network. We encode the color from green to red to the absolute time and the lifetime of each node in (a) and (c), respectively. The green lines reveal the newly joined nodes, marked in the purple rectangle in (c). The blue and red lines represent trajectories of stable and dynamic nodes in (b) or temporal communities in (e). The color from light red to dark red corresponds to the evolution values from small to large, and the two time periods with large evolution values are marked in two purple rectangles in (d). Stable nodes are colored by their associated stable communities in (f).

### 6.4 Structure View

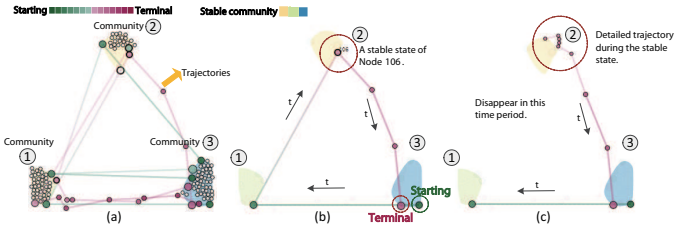


Fig. 5: (a) The structure view shows three stable communities and evolutionary trajectories of dynamic nodes among these stable communities in a selected time period, (b) displays a trajectory of a selected dynamic node, and by clicking on a combined circle, its detailed trajectory in the stable state is presented in (c).

In contrast to the trend view for temporal properties, the structure view supports further exploration of a specific time period and visualizes the network structure based on structural node clustering, including the stable communities of stable nodes and evolutions of dynamic nodes in a stable state. In order to depict the relationships of more than two communities, we project them to a 2D space in the structure view using PCA, as shown in Fig. 2(E).

Because a stable node stays in a nearby region in the stable state of the network, it is represented by a gray circle in the center position of its corresponding projected points. Circles that are close to each other indicate that the nodes have close relationships. To indicate its stability, we map the size of each circle to the variance of the node embeddings in the time period. A large circle means that the node has a relatively large evolution. A convex hull of the circles in the same stable community represents a stable community, and different colors correspond to different stable communities.

A dynamic node is represented by a trajectory in the context of stable communities. Traditionally, there are two common methods to visualize trajectories: time-to-space and time-to-color mappings. The trend view maps time-to-space to emphasize the temporal patterns, whereas in the structure view, we use a time-to-color mapping from green to pink to describe the evolutionary trajectories. Thus, an evolutionary trajectory of a dynamic node is a line chronologically through its corresponding projected points. Moreover, we combine the points in a stable state to one circle sized according to the duration of the stable state to represent a stable state of a dynamic node and reduce visual clutter.

When a node first appears or reappears, it is highlighted with a bold border. A detailed evolutionary trajectory of a stable/dynamic node can be displayed via interactions.

Fig. 5 shows the structure view in the selected time period in Fig. 4(f). Three stable communities are highlighted with different colors, and evolutionary trajectories and circles in each timestep of dynamic nodes are displayed with colors from green to pink encoded to time. Fig. 5(b) displays a trajectory of a selected dynamic node, which starts with Community 3, moves to Community 1 and Community 2, and finally returns to Community 3. Then we select a combined circle with a bold border. We find its detailed evolutionary trajectory in Fig. 5(c) and it disappears for a while before joining Community 3. This design can effectively present relationships between a dynamic node and stable communities and reveal more detailed information about the temporal properties.

### 6.5 Snapshot View

The snapshot view shows the structure of each snapshot. We propose a new method to layout the nodes in each snapshot, which better preserves the mental map compared with previous methods. Because the nodes in each snapshot are represented by diachronic node embeddings, projecting these embeddings to a 2D space using PCA can preserve the graph information and keep stable nodes fixed. As shown in Fig. 2(G), each node is represented as a circle in each snapshot and the snapshots of all the time periods are placed along the vertical time axis. The color of each circle from light red to dark red and its size are encoded to the evolution values from the previous adjacent timestep from small to large. When hovering the mouse over a node, its occurrences in all snapshots are linked by blue lines and its neighborhoods are highlighted in each snapshot view.

## 7 Use Cases

We apply our visual analytics system to artificial and real-world datasets to demonstrate the effectiveness and usefulness of our methodology based on diachronic node embeddings. The datasets cover a range of network domains, network sizes, and evolution types. By analyzing and exploring these datasets, our method enables users to better understand the structural and temporal properties of the evolution from the node, community, and network levels.

### 7.1 Artificial Dynamic Networks

We first evaluate our method on an artificial dynamic network. The Stochastic Block Model (SBM) has been widely used in network analysis and we use it to generate experimental dynamic networks. We first define the number of timesteps, number of communities in different timesteps, and stable nodes of each community. Next, we add new nodes and create evolutionary changes for these dynamic nodes. Specifically, for a dynamic node, we create its stable states based on when and how long it belongs to a community and its transition patterns by interpolating between the stable states over a predefined number of timesteps. In this case, we create a dynamic network with 95 timesteps. First, the network consists of 100 nodes belonging to one

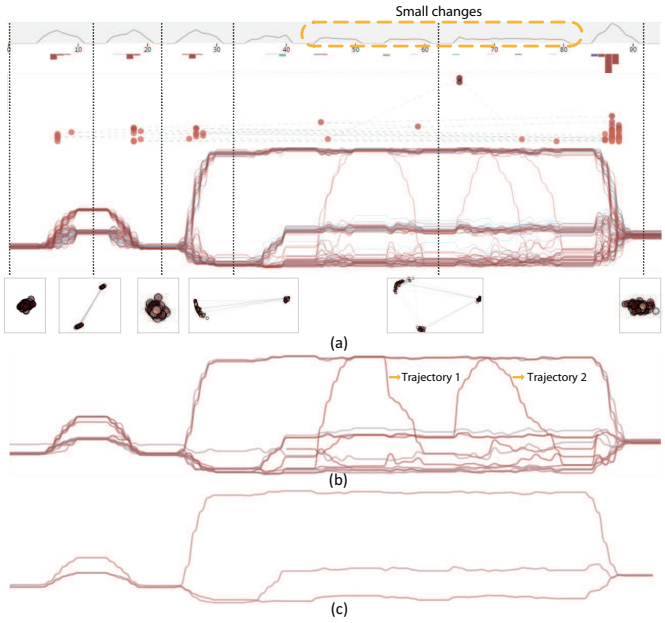


Fig. 6: (a) The trend view provides an overview of the evolution of the artificial network and reveals splitting, converging, and transition patterns. (b) and (c) summarize the temporal patterns and present the evolution from the community scale when $K$ is set to 1 and 2 respectively in the temporal trajectory clustering.

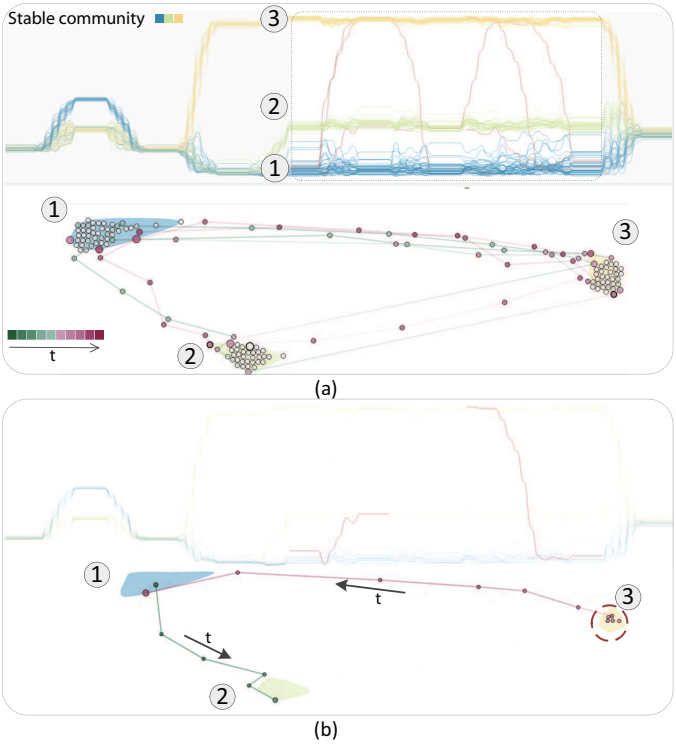


Fig. 7: (a) Three stable communities are shown in the structure view, and their corresponding temporal trajectories are presented in the trend view. (b) The detailed evolutionary trajectory and its neighborhood of a selected dynamic node are highlighted in the structure view.

community. Next, the community starts to split and two communities emerge. Then, the two communities start to converge into one community. After that, the community falls apart again, and two communities emerge with different nodes. The larger community is then divided into two communities. In this stable state with three communities, there are 10 newly joined nodes with different temporal patterns. Finally, the 10 nodes are removed, and the three communities converge into one community.

We first demonstrate how our method can effectively characterize the evolutions of nodes, communities, and networks from their temporal properties. As shown in Fig. 6(a), the line chart shows five high peaks and a time period with relatively small evolution values (G1). The trend view displays the evolutionary trajectories of each node in the context of the network structure over time. The five high peaks demonstrate the time of the splitting and converging patterns, and the relatively small evolution values reveal the time of the stable states of the network with some nodes in the transition pattern (G2). According to the temporal patterns, we infer that the network contains one community at the start, and then two communities, one community, two communities, three communities, and finally one community. The analysis results are consistent with the snapshot view and network construction (G5). When $K$ is 1 in the temporal trajectory clustering, the nodes with the most similar trajectories over time are clustered, and the central line of each temporal community is shown in Fig. 6(b). The temporal communities at this scale greatly reduce the visual clutter and highlight the trajectories with large evolution values, such as Trajectory 1 and 2. When $K$ is 2, the three main temporal communities reveal the evolution summary of the network, and the splitting and converging patterns are clearly visible in Fig. 6(c) (G3).

We further explore the network structure and dynamic node evolutions in the stable state of the network in the selected time period marked by a rectangle in Fig. 6 (G4). As shown in Fig. 7(a), three stable communities are extracted and displayed in the structure view and their temporal patterns over the whole time are highlighted in the trend view with the same color encoding. We find that the nodes in the same stable community have similar temporal patterns, as communities of stable nodes are consistent with the temporal communities in Fig. 6. As we would like to explore the evolutions of dynamic nodes, we select Node 100 and its detailed path is displayed in Fig. 7(b). It starts from Community 1, moves to Community 2, disappears for a while, reappears in Community 3, and finally moves to Community 1. By hovering the mouse over a pink circle in Fig. 7(b), its neighbors in this timestep are highlighted and we see that most of them belong to Community 3 when Node 100 belongs to Community 3.

Through this case study, we demonstrate the usefulness of our method for the exploration of the artificial dynamic network. At the network level, the evolution overview of the network as a whole is provided, which quickly guides users to the time period of interest. At the node level, the evolutionary trajectories with low-dimensional projections facilitate the evolution analysis of nodes. At the community level, the structural and temporal communities present a summary of the structural and temporal properties for a

better understanding of the evolution of the network.

### 7.2 School Contact Network

In this case study, we show how our system explores the evolution of dynamic networks with significant changes. The school contact network dataset is a collection of face-to-face contacts between students and teachers from a school during two school days [40]. This school has five grades with each grade divided into two classes, and each class has an assigned teacher. The original data provides the class information of 232 students and 10 teachers. There is a lunch break and two breaks around 10:30 and 15:30 on a school day. As the results of the two days are similar in nature, we restrict our attention to the first day. We create a snapshot every 6 minutes ($\Delta t$) and choose a window width of 60 minutes ($\omega$) with an overlap. Thus, we finally create 89 undirected snapshots where nodes and edges exist. We first explore temporal evolution of networks, then compare our method with related methods, and finally introduce more unique findings of our method.

We first explore the temporal evolution of the network using the trend view in Fig. 8(a) (G2, G3, and G5). We see that some temporal communities evolve over time. After the temporal trajectory clustering with $K = 2$, ten temporal communities are extracted and displayed in Fig. 8(b), and they may correspond to ten classes. To verify this hypothesis, we color the trajectory of each node with two color encodings, mapping color to the temporal community and the class information in this dataset, which are shown in Fig. 8(c) and (d), respectively. We find that the temporal communities are consistent with the classes as expected. In addition, the teachers (black lines in (d)) are also grouped with the students of their assigned classes. This indicates that the individuals in the same class have similar temporal patterns owing to the class schedules.

Furthermore, because the trajectories of the temporal communities corresponding to the classes in the same grades are close, as shown in Fig. 8(b), this indicates that the classes in the same grades have frequent contacts and tend to have similar class schedules, including Classes 5A and 5B, Classes 3A and 3B, and Classes 2A and 2B. However, the trajectories of the temporal communities corresponding to Classes 4A and 4B, as well as 1A and 1B are placed relatively far away.

Thus, we further explore how the ten temporal communities evolve over time and analyze the differences between them. As shown in Fig. 8(d), after 10:00, the temporal communities of Classes 5A, 5B, and 4A are separated, but the temporal communities of the other classes are closer together. This may be due to the two breaks. As only some of the classes have breaks at the same time due to the limited playground space, the breaks of the classes except for Classes 5A, 5B, and 4A, may be arranged around 10:30. Correspondingly, the temporal communities of the classes except for Classes 2A, 2B, and 1B get closer starting from 15:30, and this indicates that these classes have breaks around 15:30. Thus, we can infer that the reason for the differences between trajectories of Classes 4A and 4B, as well as Classes 1A and 1B is the different schedule of breaks. Between 12:00 and 14:00, there are no clear temporal

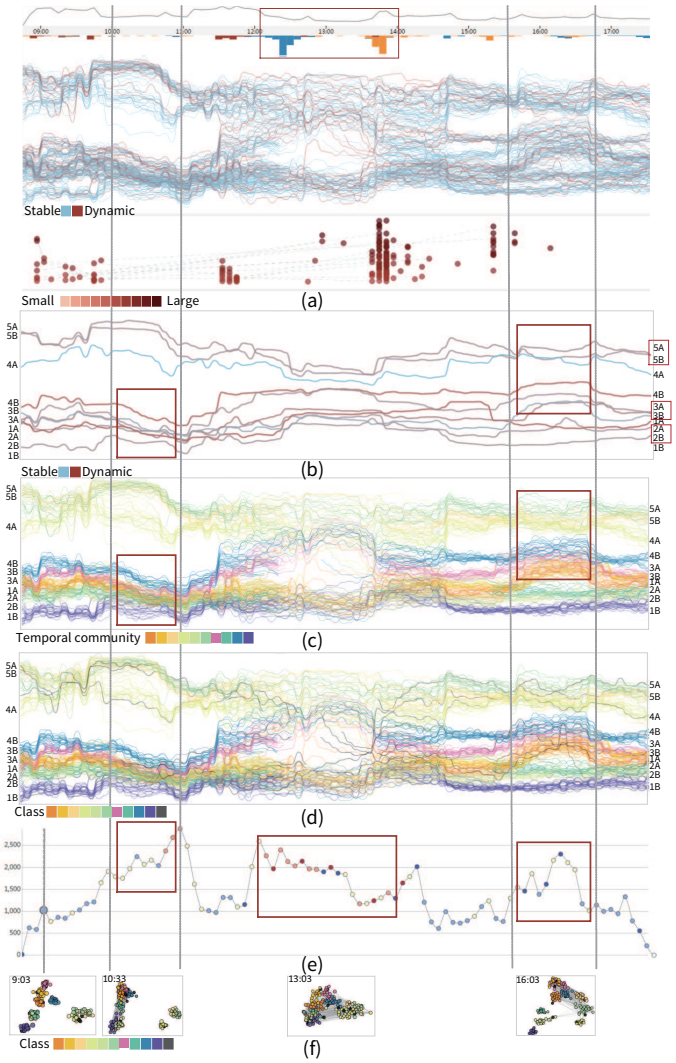


Fig. 8: (a) The temporal evolution of the school contact network. (b) Ten temporal communities corresponding to ten classes after the temporal trajectory clustering with $K = 2$. (c) Each trajectory colored by its temporal community, and (d) each trajectory colored by its class. (e) The network analyzer view from WaveletVis. (f) The original networks in four different timesteps are shown in four snapshot views.

communities with many nodes removed after 12:00 and added before 14:00, as shown in the bar chart (G1). This is explained by the fact that many students leave the school during the lunch break. These findings about the network evolution are reflected in the original networks in the snapshot view in Fig. 8(f). Previous research [40] describes the spatiotemporal trajectories of some classes, which are also generally consistent with our findings. This demonstrates the usefulness of the trend view for characterizing the temporal evolution of temporal communities.

Several methods of presenting a summary of the temporal evolution of networks have been proposed recently, and we compare our method with three state-of-the-art methods: WaveletVis [3], RSP [22], and TimeArcs [41], as illustrated in Fig. 8 and Fig. 9. The dataset of this case was chosen because of three factors: it was used in the released system although not used in the paper of WaveletVis, it is a collection of

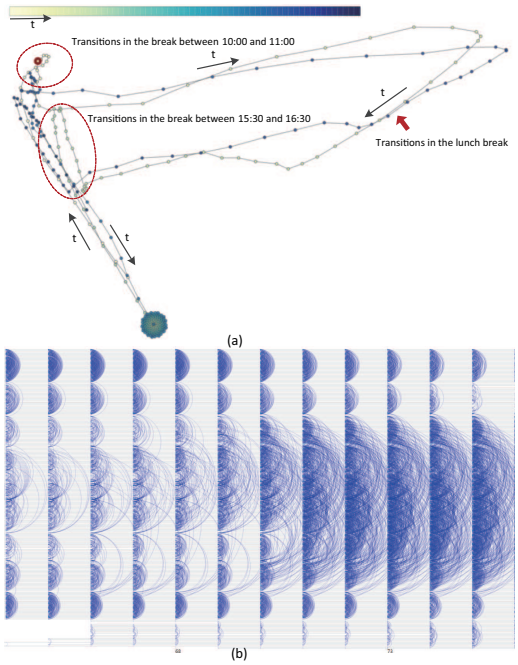


Fig. 9: (a) RSP and (b) TimeArcs methods are applied to the school contact network.

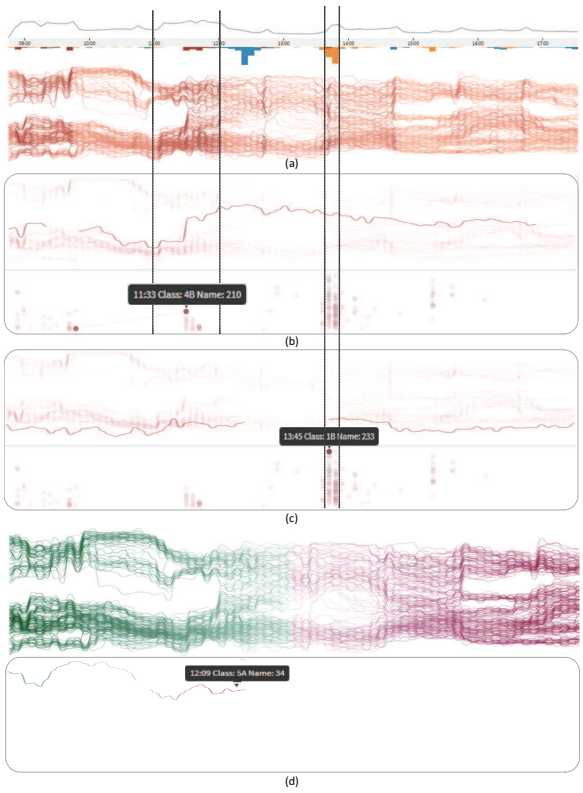


Fig. 10: The trend view in (a) shows that evolutionary trajectories are colored from light red to dark red corresponding to the evolution values of nodes between adjacent timesteps from small to large. The trend views in (b) and (c) highlight the evolutionary trajectories of Students 210 and 233 selected in the node view, respectively. The top trend view in (d) displays that evolutionary trajectories are colored from green to pink corresponding to the lifetime of each node, and the bottom view highlights the evolutionary trajectory of Student 34.

school contacts similar to the dataset used in RSP with the same parameters in the dynamic network construction, and it has not been used in other works on the evolution visualization of dynamic networks.

WaveletVis applies graph wavelets to analyze the network. In Fig. 8 (e), each circle represents a snapshot, and a circle with a high position or color close to red represents an event or the occurrence of significant changes to the network.

We see that WaveletVis is able to reveal the important timesteps falling in the two breaks, but it is inadequate to uncover the structural differences between different events. In addition, there are many red circles between 12:00 and 14:00, which stands out and indicates there are significant changes occurring in this time period. As discussed above, this is due to the lunch break, but the structural characteristics of students at school generally remain stable in this time period.

RSP reduces snapshots to points for dynamic network exploration, with the distance between two points corresponding to the structural similarity of two snapshots. In Fig. 9(a), each circle represents a snapshot, and the trajectory composed of circles close to yellow describes the evolution of the network on the first day. We find the transitions of the network during three time periods, between 10:00 and 11:00 and between 15:30 and 16:30, corresponding to the times of the two breaks, and between 12:00 and 14:00, corresponding to the lunch break. However, some stable states are not easily identified, such as during class time between 14:00 and 15:00, and the transitions between states are difficult to interpret without the context of the network structure.

TimeArcs utilizes constraints on a force-directed layout algorithm to show the evolutions of nodes and highlight communities of nodes. In Fig. 9(b), the horizontal axis is encoded to the time, and blue arcs represent edges. Indeed, we see the evolution of the communities between 14:00 and 16:00, around the breaks. However, it is difficult to trace the evolution of a node, and it is limited by the length of time with long arcs interfering with that of other time slices.

Compared with these three methods, our method provides an overview of the temporal evolution in the context of the network structure in the presence of noise. This facilitates understanding and interpreting the temporal patterns. Moreover, our method extracts similar temporal evolutions, which reduces the burden of a high cognitive load and enables users to explore the network from different scales.

Our method can further explore temporal patterns of nodes (G2). In the trend view in Fig. 10(a), the color of the evolutionary trajectories of nodes from light to dark red is encoded to the evolution values from small to large. The time periods when most nodes have large evolution values are revealed, such as the time period between 11:00 and 12:00. Together with the node view in Fig. 10(b), we quickly find that Student 210 has the largest change between 11:00 and 12:00 and its evolutionary trajectory is highlighted in the trend view in Fig. 10(b). In addition, the node view also reveals that there are many students with large evolution values between 13:45 and 13:51. When selecting Student 233

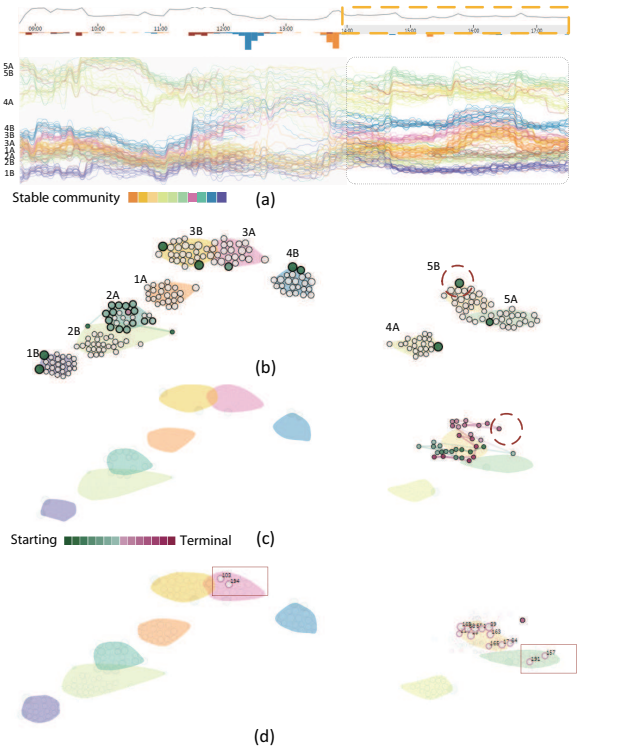


Fig. 11: Between 14:00 and 17:30, the lines are colored according to their stable communities in the trend view (a), and ten stable communities are shown in the structure view (b), which are consistent with the temporal communities. The detailed evolutionary trajectory of a student is shown in (c) and his neighborhoods at 17:00 are shown in (d).

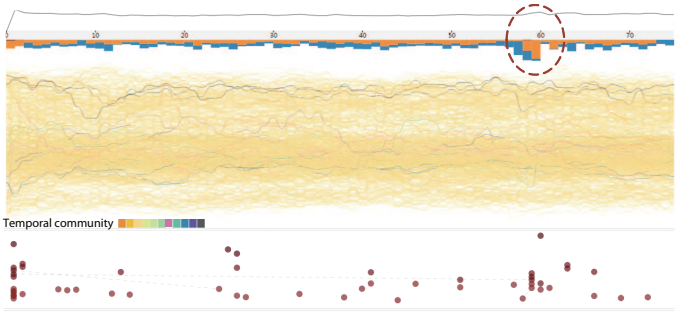


Fig. 12: Temporal evolution of the email communication network with the color representing the temporal trajectory community.

with the largest evolution value at 13:45, its evolutionary trajectory is highlighted in the trend view in Fig. 10(c) and it disappears for a while in the afternoon. This indicates that many students have large evolution values in this time period due to the lunch break. In the top trend view in Fig. 10(d), the color of evolutionary trajectories from green to pink is encoded to the lifetime of each node. The students leaving the school early are uncovered, such as Student 34 in Class 5A. Its evolution is highlighted in the bottom trend view.

We finally explore the network from the structural aspect (G4). As shown in Fig. 11, the time period of the afternoon (14:00-17:30) is selected. The network structure with ten stable communities is displayed in the structure view in Fig. 11(b). By coloring the trajectories according to the stable communities in the trend view in Fig. 11(a), we see that the ten stable communities are consistent with ten classes and temporal communities. This indicates that the students in the same class have close face-to-face contacts and similar temporal patterns over time. Moreover, in the structure view, most nodes, colored gray, remain stable and have close relationships with their classmates, whereas there are still some dynamic students with changes over time in each class, represented by evolutionary trajectories or combined circles from green to pink. We notice that Class 2A has the most dynamic students (green circles). We further select a student in Class 5B and his detailed evolutionary trajectory is shown in Fig. 11(c). When hovering the mouse over the pink circles, we find that the student has frequent contacts with Students 103 and 194 in Class 3A and Students 191 and 197 in Class 5A after school (16:30). We infer that they may be friends.

This case study demonstrates that when network structures radically change over time, our method can effectively explore the evolutionary trajectories of nodes in the context of structure and detect similar temporal trajectory communities as well as stable structural communities. Compared with state-of-the-art methods, our method presents the evolution of the network from the node, community, and network levels, simultaneously. This bridges the gap between the exploration of different levels and uncovers events with noise removed and cognitive load reduced.

### 7.3 Email Communication Dynamic Network

In this case study, we demonstrate how our system can effectively gain insights into the evolution of dynamic networks with small changes. We use a collection of emails between members of different departments at a large European research institution [42]. The dataset contains 986 nodes and 332,334 temporal edges over 803 days. Owing to privacy protection, people's department information is not provided. We create a snapshot every 7 days ($\Delta t$), choose a window width of 14 days ($\omega$) with an overlap, and use the contact frequency as the weight of edges. We finally create 76 directed and weighted snapshots.

Fig. 12 shows the temporal evolution of nodes. In the statistical view, the line chart shows that the network evolves smoothly across time and only has large evolution values during the 59th week with many newly joined nodes presented in the bar chart (G1). In the trend view, the trajectories are colored according to the temporal communities, and one can see that most nodes (yellow lines) have similar temporal behaviors; that is, the relationships of most nodes remain stable without significant changes over time. Other nodes with very distinctive temporal patterns are extracted and can be taken as outliers (G3). The node view displays dynamic nodes with large evolution values in each timestep. These views quickly guide users to these nodes (G2).

Because the network remains stable over time, we select the whole time period and explore the network from the structural aspect (G4). In Fig. 13, the trend view highlights the evolutionary trajectories of dynamic nodes in red while preserving the color of other lines according to their stable communities, and the structure view depicts

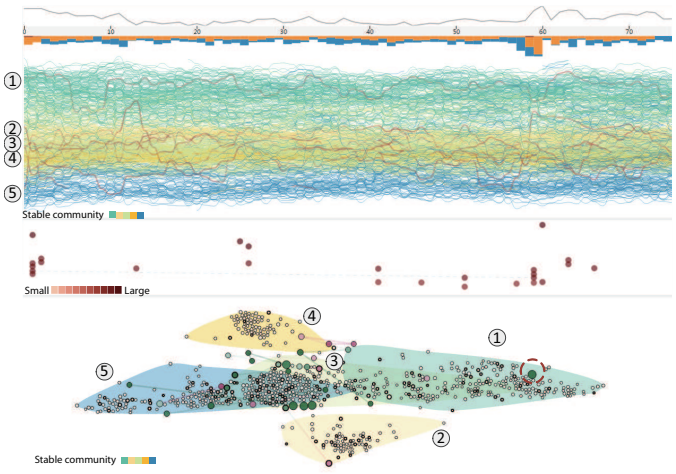


Fig. 13: In the whole time period, five stable communities are shown in the structure view, and their corresponding trajectories are displayed in the trend view.

that the network consists of five stable communities with Community 3 located in the middle. Community 3 may have frequent contacts with the other four communities. Furthermore, the transition patterns of the dynamic nodes mostly occur between Community 3 and other communities, and most nodes move to other communities and finally return to Community 3. This reveals that there are the most email communications between Community 3 and the other communities. Considering different departments in this dataset, Communities 1, 2, 4, and 5, may correspond to four departments and Community 3 may be composed of the nodes from each department responsible for the contact with other departments. Community 2 has few nodes leaving or joining, which indicates that Community 2 may be an independent department and its research may be very different from the other departments.

As there is only one trajectory of a dynamic node in Community 1 in the trend view, we click the corresponding combined green circle in the structure view in Fig. 13 and its detailed evolutionary trajectory among the stable communities is shown in Fig. 14(a). We see that the node is evolving in Community 1, then moves to Community 3, and returns to Community 1. Before moving to Community 3, it has contacts with Nodes 487 and 692 that belong to Community 3 in Fig. 14(b). When it belongs to Community 3, it has many contacts with nodes in Community 3, including Node 692, and some contacts with the other communities in Fig. 14(c). After returning to Community 1, we find that it has few contacts with the other communities, but still has contacts with Node 692. This reveals that the node has close relations with Node 692 and their research may have intersections.

This case study describes that when network structures remain stable, our method is able to detect abnormal temporal patterns based on the temporal trajectory clustering and discover relationships of stable communities based on the structural node clustering and the structure view.

## 8 Discussions and Limitations

Our experiments show that our method is adequate to explore the evolution of dynamic networks in these cases

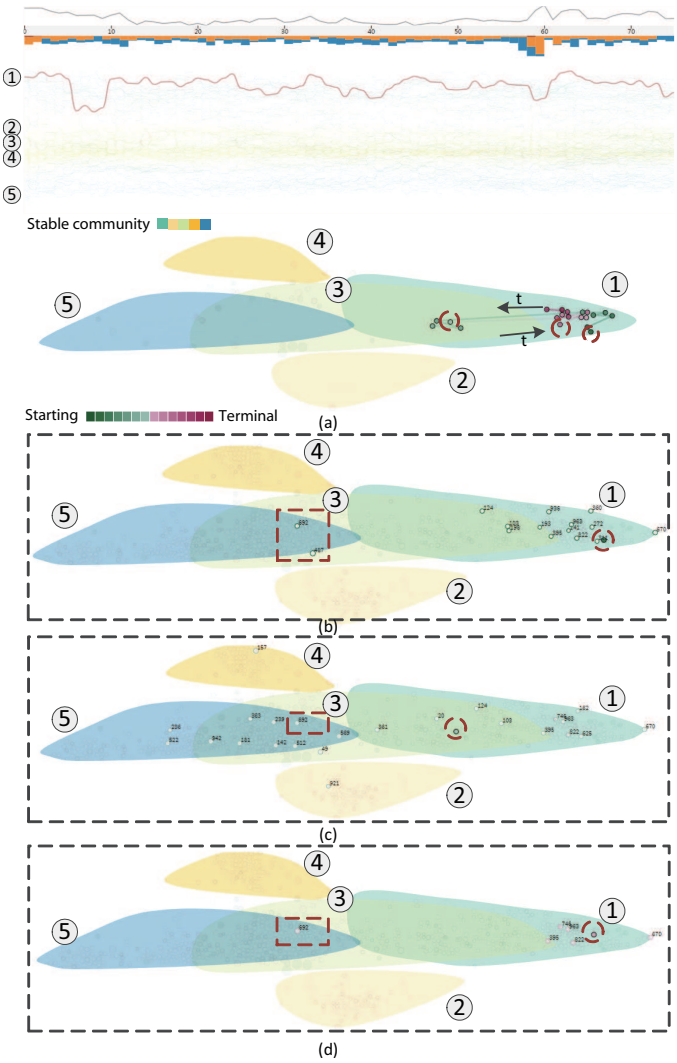


Fig. 14: When selecting a node in the structure view, (a) displays its detailed evolutionary trajectory, starting from Community 1, moving to Community 3 and returning to Community 1, in the structure view and trend view. (b), (c), and (d) show its neighborhoods in different timesteps.

covering a broad range of network domains, including face-to-face contacts and email communications.

**Parameter Study.** Our visual analytics system supports interactive parameter specification for the threshold $\theta$ to identify stable or dynamic nodes in the node classification and the parameter $K$ to uncover temporal patterns at different scales in the temporal trajectory clustering. The trend view changes with these parameters and users can adjust their values according to the trend view to find dynamic nodes and temporal communities of interest. The larger $\theta$ is, the smaller the number of dynamic nodes is, and the higher $K$ is, the smaller the number of temporal communities is. Moreover, the number of stable communities in the structural node clustering can be determined iteratively according to the overview in the trend view and the structure view. For example, in Section 6.2, ten stable communities are depicted clearly in the first structure view in Fig. 8(f). These parameters provide flexibility to our method for a broad range of datasets.

**Scalability of the Visualization.** We design the trend view to present temporal patterns in the context of structures of dynamic networks. There are two types of visual scalability concerns in this view: the number of nodes and number of timesteps. If the number of nodes is large, the temporal trajectory clustering can be applied to combine similar evolutionary trajectories, which enables users to explore the network from different scales and reduce visual clutter. Moreover, filtering can be used to only display the evolution of important dynamic nodes. If the number of timesteps is large, we propose an interaction via brushing to enable users to focus on the temporal properties in the selected time period of interest. Furthermore, clustering timesteps with similar network structures can be used to reduce the timesteps and present more information in the limited screen space.

| | K=1 | K=2 | K=3 |
|---|---|---|---|
| Artificial Network | $10 \pm 1$ ms | $10 \pm 1$ ms | $10 \pm 1$ ms |
| School Contact Network | $45 \pm 1$ ms | $46 \pm 1$ ms | $49 \pm 1$ ms |
| Email Communication Network | 870 $\pm$20 ms | $872 \pm 20$ ms | $875 \pm 10$ ms |

TABLE 1: Computation times of temporal trajectory clustering with different $K$.

| | t=20 | t=30 | t=40 |
|---|---|---|---|
| Artificial Network | $15 \pm 1$ ms | $18 \pm 1$ ms | $20 \pm 1$ ms |
| School Contact Network | $44 \pm 1$ ms | $50 \pm 1$ ms | $57 \pm 1$ ms |
| Email Communication Network | $190 \pm 20$ ms | $230 \pm 20$ ms | $330 \pm 20$ ms |

TABLE 2: Computation times of structural node clustering with different time periods.

**Scalability of Node Embedding and Analysis.** Our experiments are evaluated on a computer with a 3.20 GHz Intel Core i5 CPU and 32 GB of memory. Diachronic node embeddings are trained in the preprocessing and requires a relatively long time depending on the network size. For the example of the email communication use case, this process costs approximately 40 minutes. Temporal trajectory clustering and structural node clustering are performed during the exploration of dynamic networks, and hence, they should meet the needs of real-time interactions. We compute node distances in the preprocessing and the time complexities of the two clustering methods are $O(log(|V|)+K^2log(K))$ and $O(k|V|)$, respectively. For the three datasets in our use cases, they contain 105 nodes and 95 timesteps, 242 individuals and 89 timesteps, as well as 986 people and 76 timesteps, respectively. Their computation times for temporal trajectory clustering and structural node clustering are shown in Table 1 and Table 2, respectively. Our method can achieve interactive clustering on the current networks, but more efficient clustering methods would be required for larger dynamic networks.

**Limitations.** Firstly, as our method is based on node embeddings and alignment, it is less effective for dynamic networks with few edges (hard to train node embeddings) and with random and significant changes between neighboring timesteps (difficult to align node embeddings). To reduce the impact of edges, our method can control the network topology of neighboring snapshots by adjusting the overlaps of the time-windows when constructing dynamic networks. Secondly, for large dynamic networks with tens of thousands of nodes, the computation efficiency of structural node clustering and temporal trajectory clustering is still a challenge. One possible solution is to perform these clusterings under common parameters in the preprocessing, such as K from 1 to 3. Thirdly, although the snapshot view can highlight the neighborhoods of nodes over time, it fails to uncover the changes in edges. More animation techniques can be adopted into the snapshot view in the future. Finally, our method is designed to explore the temporal evolution in the context of the structure. When the structure is not very clear, such as in the case of one chaotic community, we can still classify dynamic nodes and cluster nodes with similar temporal trajectories. However, it would be difficult to interpret and understand the evolution of dynamic nodes and temporal communities in the chaotic structure. Instead of showing all nodes, we can only display neighbor nodes of dynamic nodes in each timestep to reduce visual clutter and better reveal their semantic structural changes, which would be our future work.

## 9 Conclusion

In this paper, we have proposed a novel methodology based on diachronic node embeddings to interactively explore temporal patterns in the context of the structure of dynamic networks from the node, community, and network levels. Diachronic node embeddings are employed to analyze dynamic networks for the first time. These diachronic node embeddings are able to preserve both the structural proximity and temporal consistency of the evolutions of nodes. Two clustering methods are provided to extract nodes from the temporal and structural properties for the summarization of the temporal patterns across time and the network structure in a time period. A new stream-based temporal overview is designed to display the temporal patterns in the context of the network structure from different scales, and a new structural overview design maps time to color to present the structure of stable communities and the evolutionary trajectories of dynamic nodes. Our method is applied to artificial and real-world dynamic networks and compared with the state-of-the-art methods to demonstrate its effectiveness and usefulness.

In the future, we plan to improve the evolution extraction method to better capture the temporal features of nodes by jointly computing the node embeddings and alignment. Clustering timesteps with similar network structures can be adopted to reduce timesteps in the future.

## Acknowledgments

This work was supported by the National Key Research & Development Program of China (2017YFB0202203), National Natural Science Foundation of China (61472354, 61672452 and 61890954), and NSFC-Guangdong Joint Fund (U1611263).

## REFERENCES


[1] S. van den Elzen, D. Holten, J. Blaas, and J. J. van Wijk, "Dynamic network visualization withextended massive sequence views," *IEEE Transactions on Visualization & Computer Graphics*, vol. 20, no. 8, pp. 1087–1099, 2014.

[2] W. Cui, X. Wang, S. Liu, N. H. Riche, T. M. Madhyastha, K. L. Ma, and B. Guo, "Let it flow: a static method for exploring dynamic graphs," in *Visualization Symposium (PacificVis)*. IEEE, 2014, pp. 121–128.

[3] A. D. Col, P. Valdivia, F. Petronetto, F. Dias, C. T. Silva, and L. G. Nonato, "Wavelet-based visual analysis of dynamic networks," *IEEE Transactions on Visualization & Computer Graphics*, vol. 24, no. 8, pp. 2456–2469, Aug 2018.

[4] M. Burch, M. Hlawatsch, and D. Weiskopf, "Visualizing a sequence of a thousand graphs (or even more)," *Computer Graphics Forum*, vol. 36, no. 3, pp. 261–271, 2017.

[5] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 701–710.

[6] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 855–864.

[7] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *Knowledge-Based Systems*, vol. 151, pp. 78 – 94, 2018.

[8] P. D. Hoff, A. E. Raftery, and M. S. Handcock, "Latent space approaches to social network analysis," *Journal of the american Statistical association*, vol. 97, no. 460, pp. 1090–1098, 2002.

[9] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Advances in neural information processing systems*, 2002, pp. 585–591.

[10] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[11] I. T. Jolliffe, "Principal component analysis and factor analysis," in *Principal component analysis*. Springer, 1986, pp. 115–128.

[12] J. B. Tenenbaum, V. De Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[13] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[14] B. Perozzi, V. Kulkarni, H. Chen, and S. Skiena, "Don't walk, skip!: Online learning of multi-scale network embeddings," in *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, ser. ASONAM '17. ACM, 2017, pp. 258–265.

[15] F. Reitz, M. Pohl, and S. Diehl, "Focused animation of dynamic compound graphs," in *Information Visualisation, 2009 13th International Conference*. IEEE, 2009, pp. 679–684.

[16] F. Beck, M. Burch, S. Diehl, and D. Weiskopf, "A taxonomy and survey of dynamic graph visualization," *Computer Graphics Forum*, vol. 36, no. 1, pp. 133–159, 2017.

[17] D. Archambault, H. Purchase, and B. Pinaud, "Animation, small multiples, and the effect of mental map preservation in dynamic graphs," *IEEE Transactions on Visualization & Computer Graphics*, vol. 17, no. 4, pp. 539–552, 2011.

[18] B. Bach, E. Pietriga, and J.-D. Fekete, "Graphdiaries: animated transitions andtemporal navigation for dynamic networks," *IEEE Transactions on Visualization & Computer Graphics*, vol. 20, no. 5, pp. 740–754, 2014.

[19] A. Perer and J. Sun, "Matrixflow: temporal network visual analytics to track symptom evolution during disease progression," in *AMIA annual symposium proceedings*, vol. 2012. American Medical Informatics Association, 2012, p. 716.

[20] I. Boyandin, E. Bertini, and D. Lalanne, "A qualitative study on the exploration of temporal changes in flow maps with animation and small-multiples," *Computer Graphics Forum*, vol. 31, no. 3pt2, pp. 1005–1014, 2012.

[21] M. Burch and D. Weiskopf, "A flip-book of edge-splatted small multiples for visualizing dynamic graphs," in *Proceedings of the 7th International Symposium on Visual Information Communication and Interaction*. ACM, 2014, p. 29.

[22] S. van den Elzen, D. Holten, J. Blaas, and J. J. van Wijk, "Reducing snapshots to points: A visual analytics approach to dynamic network exploration," *IEEE Transactions on Visualization & Computer Graphics*, vol. 22, no. 1, pp. 1–10, 2016.

[23] Y. Wu, W. Wu, S. Yang, Y. Yan, and H. Qu, "Interactive visual summary of major communities in a large network," in *Visualization Symposium (PacificVis)*. IEEE, 2015, pp. 47–54.

[24] M. Rosvall, D. Axelsson, and C. T. Bergstrom, "The map equation," *The European Physical Journal Special Topics*, vol. 178, no. 1, pp. 13–23, Nov 2009.

[25] M. Rosvall and C. T. Bergstrom, "Multilevel compression of random walks on networks reveals hierarchical organization in large integrated systems," *PLOS ONE*, vol. 6, no. 4, p. e18209, 2011.

[26] ——, "Mapping change in large networks," *PLOS ONE*, vol. 5, no. 1, pp. 1–7, 01 2010.

[27] D. Greene, D. Doyle, and P. Cunningham, "Tracking the evolution of communities in dynamic social networks," in *Advances in social networks analysis and mining (ASONAM), 2010 international conference on*. IEEE, 2010, pp. 176–183.

[28] K. Reda, C. Tantipathananandh, A. Johnson, J. Leigh, and T. Berger-Wolf, "Visualizing the evolution of community structures in dynamic social networks," *Computer Graphics Forum*, vol. 30, no. 3, pp. 1061–1070, 2011.

[29] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[30] A. Kutuzov, L. Øvrelid, T. Szymanski, and E. Velldal, "Diachronic word embeddings and semantic shifts: a survey," in *Proceedings of the 27th International Conference on Computational Linguistics*. Association for Computational Linguistics, 2018, pp. 1384–1397.

[31] V. Kulkarni, R. Al-Rfou, B. Perozzi, and S. Skiena, "Statistically significant detection of linguistic change," in *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2015, pp. 625–635.

[32] W. L. Hamilton, J. Leskovec, and D. Jurafsky, "Diachronic word embeddings reveal statistical laws of semantic change," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016, pp. 1489–1501.

[33] A. Lancichinetti and S. Fortunato, "Community detection algorithms: A comparative analysis," *Phys. Rev. E*, vol. 80, p. 056117, Nov 2009.

[34] W. Ding, C. Lin, and P. Ishwar, "Node embedding via word embedding for network community discovery," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 3, no. 3, pp. 539–552, 2017.

[35] G. Andrienko and N. Andrienko, "Spatio-temporal aggregation for visual analysis of movements," in *Visual Analytics Science and Technology(VAST)*. IEEE, 2008, pp. 51–58.

[36] H. Hotelling, "Analysis of a complex of statistical variables into principal components." *Journal of educational psychology*, vol. 24, no. 6, p. 417, 1933.

[37] J. B. Kruskal, "Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis," *Psychometrika*, vol. 29, no. 1, pp. 1–27, 1964.

[38] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[39] L. Van Der Maaten, E. Postma, and J. Van den Herik, "Dimensionality reduction: a comparative," *J Mach Learn Res*, vol. 10, pp. 66–71, 2009.

[40] J. Stehlé, N. Voirin, A. Barrat, C. Cattuto, L. Isella, J.-F. Pinton, M. Quaggiotto, W. Van den Broeck, C. Régis, B. Lina *et al.*, "High-resolution measurements of face-to-face contact patterns in a primary school," *PLOS ONE*, vol. 6, no. 8, p. e23176, 2011.

[41] D. T. Nhon, N. Pendar, and A. G. Forbes, "Timearcs: Visualizing fluctuations in dynamic networks," *Computer Graphics Forum*, vol. 35, pp. 61–69, 2016.

[42] A. Paranjape, A. R. Benson, and J. Leskovec, "Motifs in temporal networks," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, ser. WSDM '17. ACM, 2017, pp. 601–610.

**Jin Xu** received the B.S. degree in Computer Software from Northeast Normal University, Changchun, China. She is currently pursuing the Ph.D. degree at the State Key Laboratory of CAD&CG in the College of Computer Science and Technology at Zhejiang University, Hangzhou, China. Her research interests include information visualization and visual analytics.

**Yubo Tao** is an associate professor at the State Key Laboratory of CAD&CG in the College of Computer Science and Technology at Zhejiang University. He received his B.S. and Ph.D. degrees in computer science and technology from Zhejiang University in China, in 2003 and 2009, respectively. From August 2010 to July 2012, he worked as a Research Fellow in the Centre for Computer Graphics & Visualization (CCGV) at University of Bedfordshire. His research interests include scientific visualization, visual analytics and computational electromagnetics.

**Yuyu Yan** received the B.S. degree in Computer Software from Xidian University, Xi'an, China. He is currently pursuing the Ph.D. degree at the State Key Laboratory of CAD&CG in the College of Computer Science and Technology at Zhejiang University, Hangzhou, China. His research interests include information visualization, visual analytics, and data mining.

**Hai Lin** is a professor at the State Key Laboratory of CAD&CG, Zhejiang University. He received the M.Eng and B.Eng degrees from Xidian University in 1990 and 1987, respectively. He joined the State Key Laboratory of CAD&CG in 1990. After he obtained a PhD in computer science from Zhejiang University, he had been a Research Fellow in Medical Visualization at De Montfort University, UK from 2000 to 2003. He was a visiting professor of the Department of Computing and Information Systems, University of Bedfordshire, UK. His research interests include computer graphics, scientific visualization, volume rendering, virtual reality and graphical electromagnetic computing.