Time-Variant Graph Classification

Haishuai Wang, Jia Wu , *Member, IEEE*, Xingquan Zhu, *Senior Member, IEEE*,
Yixin Chen, and Chengqi Zhang , *Senior Member, IEEE*

Abstract—Graphs are commonly used to represent objects, such as images and text, for pattern classification. In a dynamic world, an object may continuously evolve over time, and so does the graph extracted from the underlying object. These changes in graph structure with respect to the temporal order present a new representation of the graph, in which an object corresponds to a set of time-variant graphs. In this paper, we formulate a novel time-variant graph classification task and propose a new graph feature, called a *graph-shapelet pattern*, for learning and classifying time-variant graphs. Graph-shapelet patterns are compact and discriminative *graph transformation subsequences*. A graph-shapelet pattern can be regarded as a graphical extension of a *shapelet*—a class of discriminative features designed for vector-based temporal data classification. To discover graph-shapelet patterns, we propose to convert a time-variant graph sequence into time-series data and use the discovered shapelets to find *graph transformation subsequences* as graph-shapelet patterns. By converting each graph-shapelet pattern into a unique tokenized graph transformation sequence, we can measure the similarity between two graph-shapelet patterns and therefore classify time-variant graphs. Experiments on both synthetic and real-world data demonstrate the superior performance of the proposed algorithms.

Index Terms—Classification, graph, graph-shapelet pattern, time-variant subgraph.
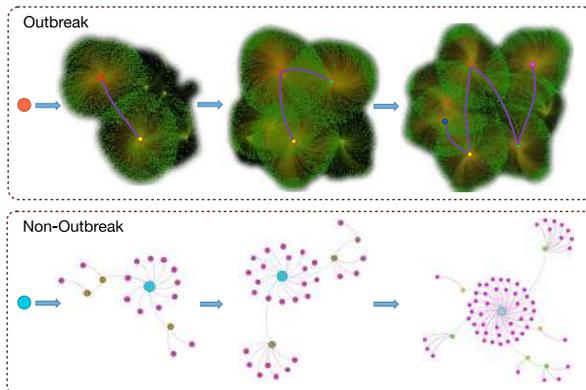

Fig. 1. An example of a time-variant graph representation for large-scale information propagation. The information propagation can be regarded as a series of individual graphs. At different time periods, both the vertex volume and the graph structure are diverse, i.e., the information propagation is a process that takes place on the graph. The information propagation outbreak prediction aims to build a time-variant graph classification solution to accurately differentiate an outbreak time-variant graph (the entire row above) from a nonoutbreak time-variant graph (the entire row below).

I. INTRODUCTION

GRAPH classification is an important branch of data mining research, given that much structured

and semi-structured data can be represented as graphs. Images [1], [2], text [3]–[5], fault diagnosis [6], decision making [7], and biological data [8], [9] present just a few examples. The main challenge in graph classification is that graphs do not have vectorial features directly available for classification, therefore traditional vector-based classifiers such as support vector machines are not applicable. Research efforts in this area have focused on extracting discriminative substructures from graph data, so that graphs can be represented as vectorial features for learning and classification. One of the most widely used methods extracts frequent subgraph patterns as discriminative features, and many studies have shown that this approach achieves good performance [10]–[12].

Despite positive results for many applications [13], [14], using discriminative subgraph patterns as features is based on a strict assumption that graph data are static and that frequent subgraph patterns are obtainable. However, time-variant graph data generated from dynamic graphs has recently entered the domain, for example, for the information propagation of graphs (i.e., a series of graphs in time series). As shown in Fig. 1, a sequence of graphs shows the information propagated over time. If we analyze the information propagation at different time window, then the target is to predict the class label that describes the outbreak or nonoutbreak of the information propagation. A sudden increase in occurrences of information

propagated over social media in a particular time and place is known as information propagation outbreak [15]. In other words, outbreak occurs when information is propagated to a large audience in a very short time (e.g., a common case of outbreak is rumor or malicious information diffusion [16]). At a specific point in time, the status of the information diffusion is a graph, but over time the graph is diverse. Therefore, an information diffusion at different stages forms a time-variant graph. As both the vertex volumes and structure keep changing in each time-variant graph, information propagation outbreak is characterized by a number of special structures that can be modeled as graph-shapelet patterns.

In this paper, we refer to a sequence of graphs with a strict temporal order as a *time-variant graph*, e.g., an entire row in Fig. 1. A time-variant graph can be used to characterize the changing nature of a structured object. For example, a chemical compound [8] can be represented as a graph. By varying the temperature or other environmental conditions, the structure of the chemical compound with respect to each changed condition can be recorded, and a set of graphs (i.e., a time-variant graph) will capture its evolution. This representation is much more comprehensive and more accurate than simply representing a chemical compound as a single static graph.

In time-variant graph classification, frequent subgraph patterns are very difficult to find because it is necessary to consider both the graph structure space and the temporal correlations to find subgraph candidates for validation. The search space becomes infinite and hence unlikely to yield stable structures. Second, the distance between subgraph patterns may change irregularly over time, therefore even though a relatively large threshold parameter could be set to mine frequent subgraph patterns, the resulting set may be empty.

Although there are many works on graph stream classification [17], [18], these works typically assume that graphs are independent of one another and flow in a stream fashion. As a result, subgraph features are still applicable because the structure of each graph is static and the static graph distribution assumption and satisfactory empirical results have been reported [19]. However, in time-variant graph classification, the entire graph data changes over time and therefore obtaining stable subgraph features is unlikely. New types of graph patterns are needed as features for time-variant graph classification.

In this paper, we outline a new class of graph patterns, called *graph-shapelet patterns*, that can be used as features for time-variant graph classification. Graph-shapelet patterns are inspired by the *shapelets* in [20], which were proposed as discriminative features for time-variant data stream classification. Shapelets are compact and discriminative subsequences that significantly reduce the time and space required for time-variant data classification [21], [22]. However, all the existing time series issues fall into either univariate or multivariate problems. They ignore the structural information behind time sequences that can be represented as graphs.

The design of graph-shapelet patterns requires a graph sequential pattern mining to be grafted onto time series shapelet pattern mining. The main technical challenges include the following.

1) *Challenge #1:* From the time series shapelet pattern mining viewpoint, existing shapelet pattern mining is only based on univariant/multivariant time series data. Extending existing shapelet pattern mining algorithms to time-variant graphs is the first challenge, and requires downscaling a graph sequence into a univariant/multivariant time series by designing a graph statistics, which can minimize information loss during the conversion of a graph into a univariant/multivariant time series.

2) *Challenge #2:* From the graph sequence mining viewpoint, existing graph sequence mining aims to find frequent transformation subsequences *directly* from graph sequences by using *graph edit similarity*, which measures the similarity between two neighboring graphs. However, such methods are unscalable to the large sets of graph sequences which commonly occur in dynamic graphs. Moreover, these graph sequence mining algorithms cannot always guarantee that the frequent graph transformation subsequences are *discriminative transformation subsequences* for graph classification. Therefore, how to design a scalable algorithm that can extend existing graph sequence mining algorithms to discover *discriminative transformation subsequences* is the second challenge.

3) *Challenge #3:* From the application viewpoint, time-variant graphs have the potential to represent large and dynamic graph data. How to evaluate the performance of the proposed method by collecting persuasive data sets and designing proper benchmark methods is the third challenge.

In this paper, we propose a new class of graph patterns, *graph-shapelet patterns*, as features for time-variant graph classification. Graph-shapelet patterns are compact and discriminative graph transformation subsequences that can be used to classify graph sequences. Technically, to solve these challenges, we first convert a graph sequence into a time series using fast graph statistics. Then, we extract shapelets from the converted time series by using a traditional shapelets mining algorithms. Next, we locate the graph subsequences that match the shapelets from the original graph sequences. Each *graph subsequence* corresponds to a unique *graph transformation subsequence* based on our new graph transformation rules. At the last step, we extract the most discriminative *graph transformation subsequences* as graph-shapelets based on their graph edit similarity. Experiments on both synthetic and real-world data demonstrate the performance of the proposed algorithms.

The remainder of this paper is organized as follows. Related works are reviewed in Section II. Section III presents the preliminaries with key definitions. Section IV introduces the proposed graph-shapelet pattern-based classifier. Section V reports the experimental results, with discussion being conducted in Section VI. We conclude this paper in Section VII.

II. RELATED WORK

The increasing availability of structured data holds great potential for knowledge discovery from graph data. Generally,

real graphs tend to be time-variant graphs with time changes. However, one of the difficulties when classifying time-variant graphs is converting the graph data into a format that allows the extraction of effective features for graph classification.

Exploring new graph features is a persistent research focus in the graph data mining community, and many feature extraction methods have been proposed [13], [14]. Existing graph classification approaches can be roughly categorized into two groups: 1) distance-based methods that include a pairwise similarity measure between two graphs, such as graph kernel [23], graph embedding [24]–[27], graph matching [28], [29], and transformation [30]–[32]; and 2) subgraph feature-based methods that identify significant subgraphs as signatures for one particular class. For example, the work in [33] proposed to the extraction of subgraph structural information for classification. The work in [34] formulated the subgraph selection problem as a combinatorial optimization problem and used heuristic rules and a combination of frequent subgraph mining algorithms such as gSpan [35] to find the subgraph features. Some boosting methods, such as [36] and [37] use individual subgraph features as weak classifiers to build an ensemble for graph classification, yet none of these methods consider the dynamic property of graphs.

Several graph stream mining methods have recently been proposed that address the streaming feature challenge. The work in [38] proposed a grafting algorithm based on a stage-wise gradient descent approach for streaming feature selection. Zhou *et al.* [39] studied streaming feature selection and proposed two novel algorithms based on stream-wise regression, information investing, and alpha investing. The work in [40] developed a new, adaptive complexity penalty, the information investing criterion, to the model to dynamically adjust the threshold of the entropy reduction required for adding a new feature. In [18], a graph ensemble boosting approach was proposed to handle the imbalance in noisy graph stream classification. Again, these graph streaming methods ignore the dynamic nature of graph sequences.

Discriminative features for temporal data classification have been studied extensively [41]–[43]. Examples include bursts [44], periods [45], anomalies [46], motifs [47], shapelets [20], and discords [48]. Time series shapelets have recently attracted increased interest in data mining [49]–[51], because shapelets are usually much shorter than the original time series, which allows us to use a single shapelet for classification instead of the entire data set. Shapelets were first proposed by [20] as time-series segments that maximally predict the target variable, however, the runtime of brute-force shapelet discovery is not feasible due to the large number of candidates. A series of speed-boosting techniques, such as early abandonment of distance computations and entropy pruning of the information gain metric, have therefore been proposed [20]. The work in [21] relies on the reuse of computations and pruning of the search space to speed up the discovery of shapelets. Grabocka *et al.* [22] proposed a novel method which learns near-to-optimal shapelets directly, without the need to search exhaustively among a pool of candidates

extracted from time-series segments. Nevertheless, all the time series issues addressed fall into either the univariate or multivariate problem category, and ignore the structural information behind a time series which could be represented by graphs.

The work in [52] proposes a method to mine frequent subsequences from graph sequence data. However, this paper is different. First, [52] focuses on frequent graph transformation sequence pattern mining rather than a classification problem. Therefore, it does not consider the label of each graph sequence. Second, it initially transfers the whole graph sequences to transformation sequences and then mines the frequent patterns. However, this method is only efficient for small time-variant graph data sets since the search space tends to be too large for large-scale data sets (such as information propagation outbreak). Instead of generating the transformation sequences from the whole data set directly, our method first finds discriminative graph subsequences (graph-shapelets) to reduce the transformation space. Finally, we use real-world information propagation to evaluate our proposed approach in the experimental section. We first model the information propagation as time-variant graph and then classify the outbreak diffusion from a nonoutbreak diffusion. However, [52] is not applicable to this problem.

In summary, this paper advances the state-of-the-art by combining existing research in graph sequence mining and time series shapelet mining to extract graph-shapelets for time-variant graph classification.

III. PROBLEM FORMULATION AND PRELIMINARIES

A. Problem Definition

Definition 1 (Graph): For a time-variant graph g_i, each graph $g_i^{(j)}$ is represented as $g_i^{(j)} = (\mathcal{V}, E)$, where $\mathcal{V} = \{v_1, \ldots, v_z\}$ is a set of vertices, $E = \{(v, v')|(v, v') \in \mathcal{V} \times \mathcal{V}\}$ is a set of edges. $\mathcal{V}(g_i^{(j)}), E(g_i^{(j)})$ are sets of vertices and edges of $g_i^{(j)}$.

Definition 2 (Time-Variant Graph): A time-variant graph (i.e., graph sequence) is represented by $g_i = \langle g_i^{(1)} \cdots g_i^{(j)} \cdots g_i^{(m)} \rangle$, where the superscript represents the temporal order of the graph (e.g., $g_i^{(1)}$) in the sequence, and each time-variant g_i is assigned a label L_i.

Definition 3 (Subtime-Variant Graph): For a time-variant graph (i.e., graph sequence) $g_i = \langle g_i^{(1)} \cdots g_i^{(m)} \rangle$, a subtime-variant graph $g_i' = \langle g_i^{(k)} \cdots g_i^{(m')} \rangle$ is a fragment of a time-variant graph, where $1 \leq k \leq m' \leq m$.

Given a set of time-variant graphs $\mathcal{G} = \{g_1, \ldots, g_n\}$, time-variant graph classification *aims* to learn classification models from $\mathcal{G}$ to accurately predict previously unseen time-variant graphs with maximum accuracy. For simplicity, we only consider binary time-variant graph classification tasks. Since an information propagation graph sequence is a special case of a time-variant graph, we use information propagation as an example of a time-variant graph. Thus, the specific classification task is to identify an information propagation outbreak from nonoutbreak.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

4   IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS
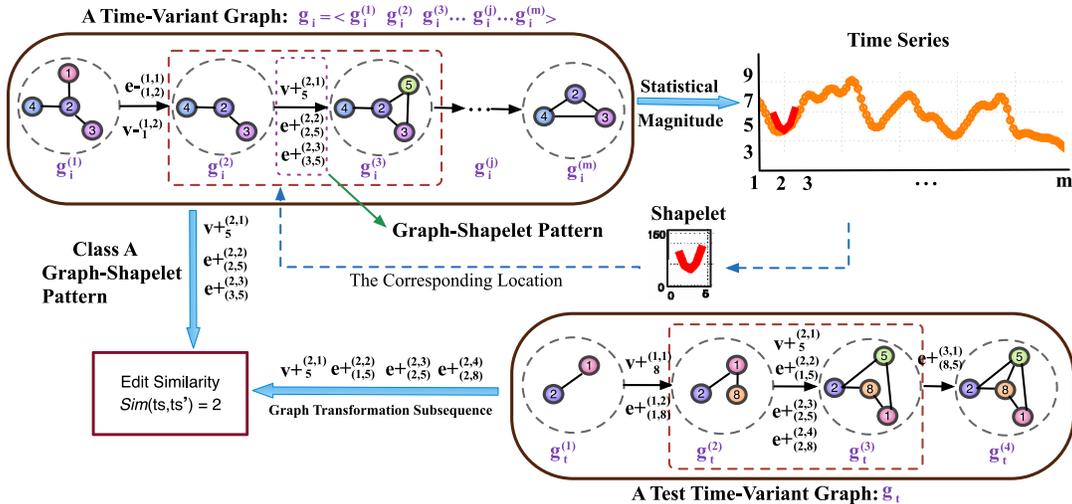
Fig. 2.   Toy example of graph-shapelet pattern mining to explain the definitions of related operations. We first explore univariate time series from a set of time-variant graphs (detailed in Section IV-B), where shapelets are discovered using shapelet pattern mining algorithms. The graph-shapelet patterns (discriminative graph transformation sequences) are then relocated by calculating the graph edit similarity (as in Definition 5) between shapelet patterns which match a set of graph subsequences. To make the above illustration clear, we use this example through Examples 1~4.

TABLE I
OPERATION DEFINITIONS

Vertex Insertion $v+_u^{(j,k)}$	Insert a vertex and a unique ID u into $g^{(j)}$ to transform to $g^{(j+1)}$ $\neg \exists v \in \mathcal{V}(g_i^{(j)})\ s.t.\ id(v) = u$ and $\exists v' \in \mathcal{V}(g_i^{(j+1)})\ s.t.\ id(v') = u\}$
Vertex Deletion $v-_u^{(j,k)}$	Delete an isolated vertex having a unique ID u in $g^{(j)}$ to transform to $g^{(j+1)}$ $\exists v \in \mathcal{V}(g_i^k)\ s.t.\ id(v) = u,$ $\neg \exists (v_1, v_2) \in E(g_i^{(k)})\ s.t.\ (id(v_1) = u) \vee (id(v_2) = u)$ and $\neg \exists v' \in \mathcal{V}(g_i^{k+1})\ s.t.\ id(v') = u\}$
Edge Insertion $e+_{(u_1,u_2)}^{(j,k)}$	Insert an edge between two vertices having unique IDs u_1 and u_2 into $g^{(j)}$ to transform to $g^{(j+1)}$ $\neg \exists (v_1, v_2) \in E(g_i^{(j)}), \exists v_1 \in \mathcal{V}(g_i^{(j)}),$ $\exists v_2 \in \mathcal{V}(g_i^{(j)})\ s.t.(id(v_1) = u_1) \wedge (id(v_2) = u_2)$ and $\exists (v_1', v_2') \in E(g_i^{(j+1)})s.t.(id(v_1' = u_1) \wedge (id(v_2' = u_2)$
Edge Deletion $e-_{(u_1,u_2)}^{(j,k)}$	Delete an edge between two vertices having unique IDs u_1 and u_2 in $g^{(j)}$ to transform to $g^{(j+1)}$ $\neg \exists (v_1, v_2) \in E(g_i^{(j)}), s.t.(id(v_1) = u_1) \wedge (id(v_2) = u_2)$ and $\neg \exists (v_1', v_2') \in E(g_i^{(j+1)}), \exists v_1' \in \mathcal{V}(g_i^{(j+1)}),$ $\exists v_2' \in \mathcal{V}(g_i^{(j+1)})\ s.t.(id(v_1' = u_1) \wedge (id(v_2' = u_2)$

∗: k is the index of operation in a transformation sequence.

B. Preliminaries

In this section, we formally define notations and introduce time series data and graph transformation subsequences. We state the distance between the time-variant graphs and the two approaches to convert the time-variant graphs into graph transformation sequences. Fig. 2 shows a toy example of the given definitions and the proposed approaches. The main operations used in the graph dynamic changes are defined in Table I.

Since the proposed method is related to time series shapelets, and the shapelet is from time series segments, we first give some prior knowledge on time series segments and shapelets to better understand the proposed approach.

Time Series Segments: Consider a sliding window of length t: a set of segments can be obtained when the window slides along a time series. For time series $x_j \in X$, we can generate a total of $q - t + 1$ segments by sliding the window from $x_j^{(1)}$ to $x_j^{(q-t+1)}$. Thus, for the entire time series X, there is a total of $(q - t + 1) \times n$ segments, i.e., $\boldsymbol{\Omega} = [\varphi_1, \ldots, \varphi_{(q-t+1)\times n}]$ where each $\varphi_j \in \boldsymbol{\Omega}$ denotes a segment. Each element $s_{(j,k)}$ is the distance between time series x_j and segment φ_k. It can be defined as the differential minimum distance that approximately denotes the minimum distance between the time series and the segment. Note that the segment length $t \ll q$, and the number of segments $(q - t + 1) \times n$ is very large.

Shapelets: Shapelets are defined as the most discriminative subsequences (segments) of time series that can best predict the target variable. Shapelets can capture inherent structures of time series, contributing to a high prediction of accuracy as explainable features [20]. Therefore, time series segments are shapelet candidates, and we can use $\boldsymbol{\Omega}$ as the feature space for shapelet selection. To represent each time series $x_j \in X$ in the space $\boldsymbol{\Omega}$, we use a column vector $s_i = [s_{i,1}, \ldots, s_{(i,(q-t+1)\times n)}]$ to record $x_j's$ feature values, where each element $s_{i,j}$ depends on a distance function between x_j and segment $\varphi_j \in \boldsymbol{\Omega}$, i.e., $s_{i,j} = d_{(x_i, \varphi_j)}$. This way, the time series data set X can be represented by a data matrix $S = [s_1, s_2, \ldots, s_n] \in \mathbb{R}^{(q-t+1)\times n, n}$, where each column vector s_j represents a time series x_j in space $\boldsymbol{\Omega}$.

We propose the graph-shapelet by borrowing the idea of shapelets in the time series. The similarity between time series and time-variant graphs is that they are time-related sequential data, whereas the difference at each time point is a numerical value in a time series and a graph in a time-variant graph. Therefore, we can calculate the Euclidian distance after learning about shapelets from the time series. In contrast, we can calculate the edit similarity after learning about graph-shapelets from time-variant graphs.

In a time-variant graph g_i, we assume that vertices v and v' represent different objects (e.g., users) in $g_i^{(j)}$. We define a set of unique IDs $\mathrm{ID}(\mathcal{V})$ and pairs of unique IDs $\mathrm{ID}(E)$ as:

$$\mathrm{ID}(\mathcal{V}) = \left\{ id(v) | v \in \cup_{g_i^{(j)} \in g_i} \mathcal{V}\left(g_i^{(j)}\right) \right\}$$
$$\mathrm{ID}(E) = \left\{ (id(v), id(v')) | (v, v') \in \cup_{g_i^{(j)} \in g_i} E\left(g_i^{(j)}\right) \right\}.$$

A transformation from $g_i^{(k)}$ to $g_i^{(k+1)}$ is represented by $op_{g_i}^k$, and $op_{g_i}^k = \langle op_{g_i}^{(k,1)} \cdots op_{g_i}^{(k,l_k)} \rangle$ with l_k denoting the length of $op_{g_i}^k$, is a series of transformations such as insertions and deletions of vertices and edges between neighboring graphs, e.g., $v+, e+, e-, v-$.

Definition 4 (Graph Transformation Sequence and Graph Transformation Subsequence): Given a time-variant graph $g_i = \langle g_i^{(1)} \cdots g_i^{(m)} \rangle$, the corresponding graph transformation sequence can be denoted as $\mathrm{ts}(g_i) = \langle op_{g_i}^1 \cdots op_{g_i}^{(m-1)} \rangle$, with $\mathrm{ts}(g_i') = \langle op_{g_i'}^1 \cdots op_{g_i'}^{(m'-1)} \rangle$ denoting a subsequence of $\mathrm{ts}(g_i)$. $\mathrm{ts}(g_i') \subseteq \mathrm{ts}(g_i), m' \leq m$, iff $\forall op_{g_i'}^{k'} \in \mathrm{ts}(g_i'), 1 \leq k' \leq m' - 1, \exists op_{g_i}^k \in \mathrm{ts}(g_i), 1 \leq k \leq m - 1; op_{g_i'}^{k'} = op_{g_i}^k$ holds.

We can use time series subsequences to locate subtime-variant graphs which are potential graph-shapelets. To predict a new testing time-variant graph, we can also convert it to a transformation sequence to compare the distances of time-variant graphs by calculating the distance of their corresponding transformation sequences.

In Fig. 2, for a time-variant graph g_i, when its individual graph $g_i^{(1)}$ changes to $g_i^{(2)}$, the corresponding transformation sequence is $op_{g_i}^1 = \langle e-_{(1,2)}^{(1,1)} v-_1^{(1,2)} \rangle$, where superscripts $(1, 1)$ and $(1, 2)$ represent the first and second graph operations changing from graph $g_i^{(1)}$ to $g_i^{(2)}$. $e-$ and $v-$ represent the removal of an edge and a vertex. The order of the operations subscripts $(1, 2)$ and 1 represent that we remove an edge from the vertex of ID 1 to ID 2, and remove a vertex of ID 1. Similarly, $op_{g_i}^2 = \langle v+_5^{(2,1)} e+_{(2,5)}^{(2,2)} e+_{(3,5)}^{(2,3)} \rangle$.

Example 1: A subtime-variant graph $\langle g_i^{(1)} g_i^{(2)} g_i^{(3)} \rangle$ generates a graph transformation sequence $\langle e-_{(1,2)}^{(1,1)} v-_1^{(1,2)} v+_5^{(2,1)} e+_{(2,5)}^{(2,2)} e+_{(3,5)}^{(2,3)} \rangle$ as shown in Fig. 2, so we can also generate its transformation subsequence $\langle v-_1^{(1,2)} e+_{(2,5)}^{(2,2)} e+_{(3,5)}^{(2,3)} \rangle$ with $e+_{(3,5)}^{(2,3)}$ denoting the edge adding operation between vertices 3 and 5.

Definition 5 (Edit Similarity): For two graph transformation sequences $\mathrm{ts}(g_i)$ and $\mathrm{ts}(g_j)$, we use $\mathrm{Sim}(\mathrm{ts}(g_i), \mathrm{ts}(g_j))$ to denote the Edit Similarity for measuring their similarity. $\mathrm{ts}(g_i) = \langle op_{g_i}^1 \cdots op_{g_i}^{(m-1)} \rangle, 1 \leq k \leq m - 1$, with $op_{g_i}^k = \langle op_{g_i}^{(k,1)} \cdots op_{g_i}^{(k,l_k)} \rangle$ with l_k denoting the length of $op_{g_i}^k$. $\mathrm{Sim}(\mathrm{ts}(g_i), \mathrm{ts}(g_j)) = \sum_{k,v} \sum_{p=1}^{l_k} \sum_{q=1}^{l_v} \hbar_{(i,j)}^{(k,p),(v,q)}$, where $\hbar_{(i,j)}^{(k,p),(v,q)} = I(op_{g_i}^{(k,p)} = op_{g_j}^{(v,q)})$, $I(\cdot)$ is 1 if the condition inside is true; otherwise, 0.

Example 2: In Fig. 2, the graph transformation sequence for the graph-shapelets of class A is $\langle v+_5^{(2,1)} e+_{(2,5)}^{(2,2)} e+_{(3,5)}^{(2,3)} \rangle$, denoted by $\mathrm{ts}(g_i)$. The graph transformation sequence of the corresponding subtime-variant graph in the testing time-variant graph is $\langle v+_5^{(2,1)} e+_{(1,5)}^{(2,2)} e+_{(2,5)}^{(2,3)} e+_{(2,8)}^{(2,4)} \rangle$, denoted by $\mathrm{ts}(g_t)$,
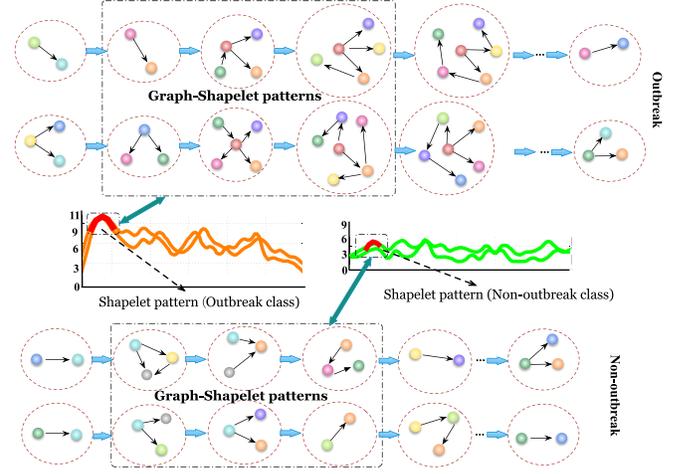

Fig. 3. Illustration of graph-shapelet patterns. Graph-shapelet patterns are compact and discriminative graph transformation subsequences that describe the graph transformation patterns shared in the same class of time-variant graphs, e.g., two time-variant graphs (the two entire rows above) with the outbreak labels. When we explore a univariate time series from a time-variant graph, we can see that the location of a *graph-shapelet pattern* is consistent with that of a *shapelet* in time series (detailed in Section IV-B). In this case, graph-shapelet patterns can be used for time-variant graph prediction such as dynamic graph outbreak prediction.

$\mathrm{Sim}(\mathrm{ts}(g_i), \mathrm{ts}(g_t)) = 2$ because of the same sequential operations. In our method, we calculate the graph edit distance [53] by using graph edit similarity.

When comparing two operations, we ignore their superscripts. For example, $e+_{(2,5)}^{(2,3)} = e+_{(2,5)}^{(2,2)}$. We use sliding windows to extract time series subsequences.

Definition 6 (Graph-Shapelet Pattern): From a given time-variant graph data set $\mathcal{G}$, $\mathcal{G}$ can be categorized into two classes: 1) outbreak and 2) nonoutbreak. The graph-shapelet pattern is a graph transformation subsequence (Definition 4) which can maximally identify an outbreak time-variant graph from a nonoutbreak time-variant graph by maximizing the edit similarity Definition 5.

Graph-shapelet patterns can be taken as a graphical extension of the traditional shapelets used in time series classification. From the time series classification viewpoint, this paper extends the univariate and multivariant time series data classification into time-variant graph classification where at each time stamp we obtain a graph instead of a single variable or a vector. As shown in Fig. 3, graph-shapelet patterns are compact and discriminative graph transformation subsequences extracted from a sequence of graph data that describes graph transformation patterns. The graph-shapelet patterns, although they looks similar to the frequent graph subsequences proposed in graph sequence mining [52], are discriminative graph subsequences that can differentiate two classes of graph sequences with high accuracy. In contrast, frequent graph subsequences do not contain the class label information and thus are not helpful for time-variant graph classification.

When converting time-variant graphs to transformation sequences, we use the following strategies to represent the time-variant graphs and make the transformation sequences between two successive graphs unique.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

6

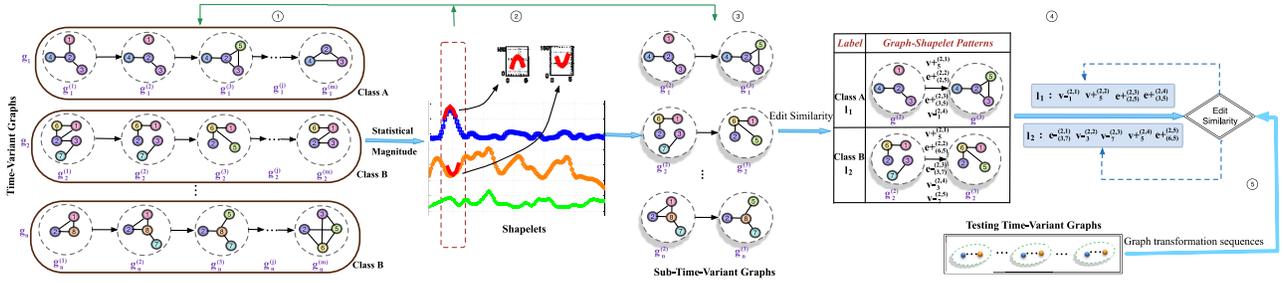IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS

Fig. 4. A conceptual view of the proposed time-variant graph classification framework. We first explore univariate time series from time-variant graphs via a sample kernel method in each graph (step ①, Section IV-B). Then, we find shapelet patterns from the time series by using shapelet pattern mining (step ②, Section IV-C). Next, we locate the subtime-variant graphs that match the shapelet patterns from the original time-variant graphs (step ③, Section IV-D). Note that each subtime-variant graph corresponds to a unique graph transformation subsequence by the proposed time-variant graph representation approach. At the last step, we calculate the graph edit similarity between graph transformation subsequences and find the most discriminative transformation subsequences as graph-shapelet patterns (step ④, Section IV-E). Step ⑤ shows the process of time-variant graph prediction.

Strategy 1 (The Admissibility): When transforming two successive graphs to a transformation sequence, the transformation order is fixed to make it unique, such that an insertion has precedence over a deletion, vertex insertion has precedence over edge insertion, while edge deletion has precedence over vertex deletion. Consequently, the transformation strategy between two successive graphs $g^{(j)}$, $g^{(i)}$ is as follows:

$$v+_{u_1}^{(j,k)} \prec e+_{(u_1,u_2)}^{(j,k')}, e+_{(u_1,u_2)}^{(j,k)} \prec e-_{(u_1,u_2)}^{(j,k')}, e-_{(u_1,u_2)}^{(j,k)} \prec v-_{u_1}^{(j,k')}$$

where $k < k'$. The definitions of $v+_u^{(j,k)}, v-_u^{(j,k)}, e+_{(u_1,u_2)}^{(j,k)}, e-_{(u_1,u_2)}^{(j,k)}$ are illustrated in Table I.

In terms of the same operation with different IDs, we use a lexicographic order to generate a unique sequence, e.g., $v+_1^{(1,1)} \prec v+_2^{(1,2)}$.

Strategy 2 (The Representation): Since all operations between two successive graphs can be enumerated as edge insertion, edge deletion, vertex insertion, and vertex deletion, results in any time-variant graph can be represented by the four operations in Table I with a given initial $g_i^{(1)}$ [52].

Table II summarizes major notations used in this paper.

IV. PROPOSED METHOD

A. Overall Framework

Fig. 4 shows the framework of our method which first explore a univariate time series from each time-variant graph via a simple kernel method (Section IV-B). We then calculate shapelets from the time series using shapelets mining (Section IV-C). After that, subtime-variant graphs that match the shapelets from the original time-variant graphs are located. Each subtime-variant graph corresponds to a unique graph transformation subsequence by applying the admissibility strategy and the representation strategy. In the last step, we calculate the graph edit similarity between graph transformation subsequences and find the most discriminative transformation subsequences as graph-shapelets (Section IV-D).

B. Graph-Shapelet Pattern Location

Although many methods exist to find shapelets for time series classification, it is very challenging to restore subtime-variant graphs from the time series shapelets. To address this

TABLE II
SYMBOLS AND NOTATIONS

Symbols	Descriptions
T	time series
ts	transformation sequence
op	transformation operation
l_k	the length of $op_{g_i}^k$
m, m'	the length of time-variant graph g_i and sub-time-variant graph g_i'
A, B	class label
$\mathcal{V}, E$	vertex and edge sets
$\mathcal{V}_{v+}, \mathcal{V}_{v-}$	a set of vertex insertions and deletions
E_{e+}, E_{e-}	a set of edge insertions and deletions
vertex insertion $v+_u^{(j,k)}$	insert a vertex of ID u into $g_i^{(j)}$ which leads to $g_i^{(j+1)}$
vertex deletion $v-_u^{(j,k)}$	delete an isolated vertex of ID u in $g_i^{(j)}$ which leads to $g_i^{(j+1)}$
edge insertion $e+_{(u_1,u_2)}^{(j,k)}$	insert an edge between two vertices u_1 and u_2 into $g_i^{(j)}$ and obtain $g_i^{(j+1)}$
edge deletion $e-_u^{(j,k)}$	delete an edge between two vertices u_1 and u_2 in $g_i^{(j)}$ to obtain $g_i^{(j+1)}$

∗: k is the index of operation in a transformation sequence.

challenge, we record the locations of time series segments that have the minimum Euclidean distance, while introducing a transformation of time-variant graphs to the time series.

We first design a kernel method to explore time-series information $t_i = \{t_i^{(1)}, \ldots, t_i^{(m)}\}$ from a time-variant graph g_i, $K_{t_i}^{(j)} = N_n(g_i^{(j)}) + N_e(g_i^{(j)})$, where $N_n(g_i^{(j)})$ and $N_e(g_i^{(j)})$ denotes the statistical information of the vertices and edges in an individual graph $g_i^{(j)}$ at time j with $1 \leq j \leq m$. After that, efficient shapelet pattern mining algorithms can be applied to quickly locate graph changes in time-variant graphs. The reason why we use the graph statistic method is that we can find the location of graph-shapelets and return to graphs after discovering graph-shapelets. By using this method, we can speed up the search time and save the graph information since we only use a segment of a time-variant graph.

Example 3: In Fig. 2, the statistics of vertices and edges in graphs $g_i^{(1)}, g_i^{(2)}, \ldots, g_i^{(m)}$ are $7, 5, \ldots, 6$, respectively. Thus, a corresponding time series $\{7, 5, \ldots, 6\}$ can be explored. Note that, we use the sum of the statistical information of the vertices and edges (i.e., $K_{t_i}^{(j)}$) when transforming a graph in a

---

Algorithm 1 SubTVG: Subtime-Variant Graph Candidates

Input:

$\mathcal{G}$: A set of time-variant graphs;

l: Length of graph-shapelet patterns;

Output:

$\mathcal{G}'$: Graph subsequence candidates;

1: $D \leftarrow$ Apply $\mathcal{G}$ to obtain corresponding time series D;

2: $cSet \leftarrow \emptyset$; // Time series candidates with length l

//Generate time series candidate with length l

3: **for** t **in** D **do**

4:    $\mathcal{S}_t^l \leftarrow$ Extract time series subsequence with length l from t

5:    $cSet \leftarrow cSet \bigcup \mathcal{S}_t^l$;

6: **end for**

7: Shapelets $\leftarrow$ Apply $cSet$ to D based on shapelet mining[20].

8: Record the Shapelet locations in each time series

9: **for each** g_i **in** $\mathcal{G}$ **do**

10:    $g_i' \leftarrow$ Apply recoded Shaplelet locations to the related sub-time-variant graphs in g_i;

11:    $\mathcal{G}' = \mathcal{G}' \bigcup g_i'$

12: **end for**

13: **return** $\mathcal{G}'$

---

time-variant graph to a numerical value in the corresponding time-series.

C. Graph-Shapelet Pattern Candidates

All subtime-variant graphs are potential graph-shapelet candidates. The maximum similarity of a candidate to all training sequences can be used for graph prediction by ranking the similarity of a candidate *with respect to* a testing time-variant graph. To find graph-shapelet patterns, the algorithm needs to generate potential subtime-variant graphs as candidates and to locate the one, which has maximum similarity *with respect to* other subsequences from the candidate subtime-variant graphs.

Algorithm 1 shows the detailed procedures for finding subtime-variant graph candidates from time-variant graphs. Given a time-variant graph database $\mathcal{G}$ with a user-defined graph-shapelet length l. In Algorithm 1, line 1 converts time-variant graph set $\mathcal{G}$ to a time series D based on the number of vertices and edges in each graph. Specifically, it calculates $K_{t_i}^{(j)}$ based on $N_n(g_i^{(j)})$ and $N_e(g_i^{(j)})$ in $\mathcal{G}$. Thus, we can obtain the corresponding time series D. Then, the algorithm generates all candidates within a sliding window (lines 3–6). Line 4 divides an input time series into a sequence of discrete segments by using a sliding window. The sliding window algorithm works by anchoring the left point of a potential segment at the first data point of a time series, then attempts to approximate the data to the right with increasing longer segments of length l. Line 7 mines shapelets from time series D. The most straightforward way for finding the shapelet is using the brute-force method. Given the time series D and the user-defined maximum and minimum lengths of the shapelet, we first generate all of the segments with length l. Then the algorithm checks how well each segment can separate D in class A (outbreak)

and class B (nonoutbreak). For each shapelet candidate, the algorithm calculates the information gain [20] achieved if using that candidate to separate the data. The algorithm returns the candidates with the highest information gain as shapelets. The reason we use an exhaustion method for finding shapelets is that the transformed time series from the time-variant graph is short so that the search space is low. Thus, the simple yet efficient brute-force shapelet search algorithm is applicable in this case. Since the shapelet mining method can return the locations of shapelets, we find the same locations from the time-variant graph to discover the graph-shapelet (lines 9–12).

D. Finding Graph-Shapelet Patterns

Graph-shapelet patterns can be found from the generated subtime-variant graph candidates. To calculate graph-shapelet patterns from subtime-variant graphs, time-variant graphs are converted to transformation sequences so that edit similarity can be used as the measure. The transformation sequences between two graphs in a given time-variant graph can be represented by a series of operations, e.g., vertex and edge insertions and deletions based on Strategy 1 and Strategy 2. According to the time-variant graph representation, an algorithm to compile $\mathcal{G} = \{g_i | g_i = \langle g_i^{(1)} \cdots g_i^{(m)} \rangle\}$ to a set of transformation sequences $ts(\mathcal{G}) = \{ts(g_i) | g_i \in \mathcal{G}\}$ is summarized in Algorithm 2.

To obtain the graph transformation sequence ts in Algorithm 2, we first represent each two successive graphs in the time-variant graph $\mathcal{G}$ as transformation sequences. Strategy 2 indicates that there are four operations (vertex insertion, vertex deletion, edge insertion, and edge deletion) between two successive graphs. Strategy 1 limits the order of the operations to ensure a unique transformation sequence. For example, for the graph sequence $\langle g_i^{(2)} g_i^{(3)} \rangle$ shown in Fig. 2, the transformation sequence can be $\langle v+_5^{(2,1)}, e+_{(2,5)}^{(2,2)}, e+_{(3,5)}^{(3,3)} \rangle$ or $\langle e+_{(2,5)}^{(2,2)}, v+_5^{(2,1)}, e+_{(3,5)}^{(3,3)} \rangle$ if we do not follow Strategy 1. Therefore, the procedure basically follows Strategies 1 and 2 to convert graph sequences (line 4). Then, the vertex and edge insertions are appended to the transformation sequences, as shown on lines 6–8 and 9–11, respectively. After that, vertex and edge deletions are carried out on lines 12–14 and 15–17, respectively, with the changes appended to the transformation sequences.

Example 4: In the graph sequence $\langle g_i^{(2)} g_i^{(3)} \rangle$ given in Fig. 2, the insertion of vertex ID 5 and edge insertions (2, 5) and (3, 5) from $g_i^{(2)}$ to $g_i^{(3)}$, result in the transformation sequences, i.e., $v+_5^{(2,1)}, e+_{(2,5)}^{(2,2)}, e+_{(3,5)}^{(2,3)}$.

E. Classification With Graph-Shapelet Patterns

Fig. 4 gives an example of finding two graph-shapelet patterns for a binary classification problem, and the algorithm details are summarized in Algorithm 3. First, the algorithm calls SubTVG() to generate all graph-shapelet pattern candidates (line 2) and calls function GraphSequenceCompiler()

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

8   IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS

Algorithm 2 GraphSequenceCompiler

Input:
    $\mathcal{G}'$: time-variant graph candidates;

Output:
    ts: graph transformation sequences;

1: $ts \leftarrow \emptyset$;
2: **for each** $g_i \in \mathcal{G}'$ **do**
3:    **for each** $g_i^{(j)} \in g_i$ **do**
4:      Represent $g_i^{(j)}$ to $\mathcal{V}_{v+}, \mathcal{V}_{v-}, E_{e+}, E_{e-}$ follow *Strategies 1, 2*;
5:      $k \leftarrow 1$
6:      **for each** v in $\mathcal{V}_{v+}$ **do**
7:        $ts \leftarrow ts \bigcup v+_{id(v)}^{(j,k++)}$;   // Get the insertion vertices set
8:      **end for**
9:      **for each** (v, v') in E_{e+} **do**
10:       $ts \leftarrow ts \bigcup e+_{(id(v),id(v'))}^{(j,k++)}$; // Get the insertion edges set
11:     **end for**
12:     **for each** (v, v') in E_{e-} **do**
13:       $ts \leftarrow ts \bigcup e-_{(id(v),id(v'))}^{(j,k++)}$; // Get the deletion edges set
14:     **end for**
15:     **for each** v in $\mathcal{V}_{v-}$ **do**
16:       $ts \leftarrow ts \bigcup v-_{id(v)}^{(j,k++)}$; // Get the deletion vertices set
17:     **end for**
18:    **end for**
19: **end for**
20: **return** ts

---

to generate graph transformation sequence for each generated subtime-variant graph candidate (line 4). The similarity between each generated subtime-variant graph and all time-variant graphs in each class (line 6) is calculated, and the subtime-variant graphs with maximum similarity are selected as the graph-shapelet patterns, and the transformation sequences used as classification features (lines 7–10).

Whenever a test time-variant graph arrives, the algorithm converts it to transformation sequences and computes the distance between the test transformation sequences and features selected for different classes by using edit similarity (Definition 5). The output label of the test graph is the same as the label of the graph-shapelet pattern, which has maximum similarity to the test time-variant graphs (lines 12–17).

The proposed method is inspired by shapelet features in a time series. Since the difference between time series and time-variant graphs is that the time-variant graph is constituted by a graph at each time point instead of a numerical value, it is reasonable that we convert the given time-variant graphs to time series. As we find graph-shapelets based on the shapelets from converted time series, the graph-shapelet patterns are a discriminative segment in the time-variant graph. Consequently, we can only consider the graph-shapelet instead of the whole graph sequence.

Algorithm 3 Graph-Shapelet Pattern Classifier

Input:
    $\mathcal{G}$: a set of n time-variant graphs;
    l: Length of the graph-shapelets;

Output:
    L_t: The label of a test time-variant graph g_t;
    // **Training Phase:**

1: $graph\_shapelt \leftarrow \emptyset$;
2: $candidates \leftarrow \text{SubTVG}(\mathcal{G}, l)$; // Algorithm 1
3: $max\_sim \leftarrow 0$;
4: $ts \leftarrow \text{GraphSequenceCompiler}(candidates)$; // Algorithm 2
5: **for each** $ts_i \in ts$ **do**
6:    $d_i \leftarrow \sum_j Sim(ts_i, ts_j), ts_j \in ts(i \neq j)$;
7:    **if** $d_i > max\_sim$ **then**
8:      $max\_sim \leftarrow d_i$;
9:      $graph\_shapelets \leftarrow ts_i$;
10:    **end if**
11: **end for**
    // **Test Phase:**
12: $ts(g_t) \leftarrow \text{GraphSequenceCompiler}(g_t)$; // Algorithm 2
13: **for** each $graph\_shapelet$ in $graph\_shapelets$ **do**
14:    calculate $Sim(graph\_shapelet, g_t)$;
15: **end for**
16: L_t is the label with maximum similarity;
17: **return** L_t

F. Time Complexity Analysis

The proposed graph-shapelet pattern-based time-variant graph classification algorithm consists of: 1) converting time-variant graphs to graph transformation sequences and 2) finding frequent transformation subsequences as graph-shapelet patterns for classification. We assume m is the length of the query time-variant graphs, l is the length of the graph-shapelet pattern, r is the average length of the transformation sequences between two graphs, and n is the size of the time-variant graph data set. The algorithm first transfers time-variant graphs to time series, which scans each graph (length is m) over all the time-variant graph samples (length is n). Then, the algorithm extracts segments with length l (the same length with graph-shapelet patterns) to find shapelets from the time series. Therefore, the time complexity for graph-shapelet mining is $O(ml * n)$. After finding the graph-shapelets, the algorithm generates the corresponding graph transformation sequences. Since the algorithm generates every possible operation in the transformation sequences one by one, the entire graph classification time complexity is $O(mrl * n)$.

V. EXPERIMENTS

A. Data Sets

1) Synthetic Time-Variant Graph Data: In our experiments, we first create synthetic information propagation to validate algorithm performance on data with known groundtruth properties. We use a well-known model, namely Small World [54], to generate synthetic information propagation graphs for testing and comparison. This model generates a ring of N vertices

where each vertex is connected to its α nearest neighbors in the ring ($\alpha/2$ on each side, which means α must be even). Then, for each edge in the graph, the target vertex with probability p is rewired. The rewired edge cannot be a duplicate or self-loop. In particular, we set $\alpha = 4$ and $p = 0.1$ in our experiments for synthetic time-variant graph generation. After generating the first graph $g_0^{(0)}$, we simulate the propagation from $g_0^{(0)}$ to generate the other graphs in the first time-variant graph sequence g_0 using the following method. First, we randomly select a root vertex r with a nonzero out-degree from $g_0^{(0)}$. The vertex r is then added to the initially empty list I and all outgoing edges $(r; s)$ are added to the initially empty first-in-first-out queue, where the first element added to the queue will be the first one to be removed. We choose an edge from the candidate set each time and calculate the time delay for the edge until the time delay exceeds a given time window (we use [0,1000] as the time window). When calculating time delay of propagation, we follow a power-law distribution exponent $\beta = 1.5$, which corresponds to the power-law degree exponent of the Weibo network. Finally, we separate graph $g_0^{(1)}, \ldots, g_0^{(i)}$ with time delay 100, 200, 300, 400, 500, 600, and 700 (e.g., we form the first graph in a time-variant graph when the propagation time delay is 100). Thus, the average length in the synthetic data is 7 since we have 7 graphs in each time-variant graph. The whole data set is generated by repeating the above steps to generate 200 graph sequences. If the number of vertices in a graph sequence is more than 100 (a user-defined threshold based on the characteristics of the data set), we label it as an outbreak class, otherwise as a nonoutbreak class. The learning task is to predict whether a time-variant graph will be an outbreak or not.

2) Real-World Time-Variant Graph Data: We evaluate the performance of the proposed method on two real-world data sets, including phone call graph obtained from MIT [55] and information propagation graph data crawled from Sina Weibo, which is one of the biggest social media platforms of China. Akin to a hybrid of Twitter and Facebook, Weibo users can post messages, and upload pictures and videos for sharing in real-time. Users can also comment and forward posted messages

The phone call time-variant graph contains the dynamics of 75 students/faculty members in the MIT Media Laboratory, and 25 incoming students at the MIT Sloan Business School adjacent to the Media Laboratory. The experiment was designed to study community dynamics (classified as personal behavior or interpersonal interactions), by tracking a sufficient number of people using their personal mobile phones. In our experiments, a unique ID is assigned to each person participating in the communication and an edge links two persons if they communicate via phone call in a day. In this way, we obtain a daily graph $g^{(j)}$. According to the daily vertex degree (threshold is set to 5), each person is either a hub person or a nonhub person. A person with a high vertex degree is considered to be a hub person in the community. We obtain a set of weekly time-variant graph data, i.e., $\mathcal{G}$, and the total number of weeks (number of sequences) is 40. We randomly sample $|V| = (1 \sim 20)$ persons, and assign the label 1 if the weekly phone call includes a hub person (i.e., a hub person

usually triggers interpersonal interactions), otherwise the label 0 is applied. The average length of each sequence is 7. The Weibo time-variant graph data set is obtained from a research community in Weibo by crawling 200 Weibo message propagations from a research community within one month. We sample 50 users and assign each of them a vertex ID (varying from 1 to 50). The propagation of the information from users follows a temporal order. At each time point, all vertices reached by the Weibo message form a graph. For a new Weibo message, the forwarding reforwarding propagation graphs collected in a time period form a time-variant graph (i.e., a time-variant graph), and our learning goal is to use these time-variant graphs to predict whether the time-variant graph for a new Weibo message will break out or not. In our experiments, we obtain 200 time-variant graphs from the propagation of 200 Weibo messages at different time periods, and the average length of each sequence is 15. We define a time-variant graph as an outbreak if the total number of vertices is more than 100.

B. Experimental Settings

1) Baseline Approaches: Because no existing approaches are available for time-variant graph classification, we compare the proposed graph-shapelet patterns-based algorithm (gShapelet for short) with the following benchmark methods in terms of prediction accuracy and running time to evaluate classification performance.

1) *Frequent Subsequence-Based Method:* This type of method converts all the training time-variant graphs to graph transformation sequences and discovers frequent subsequences as features for classification. This method resembles the work in [52], the difference being that [52] does not consider classification problems.
2) *Frequent Subgraph-Based Method:* Because we cannot obtain stable subgraphs from time-variant graphs, we mine frequent subgraphs from the last graph in a time-variant graph as features via gSpan [35], a well used frequent subgraph method.
3) *gShapelet_n:* When a time-variant graph is transformed into a univariate time series, gShapelet_n counts the number of vertices as statistical magnitude.
4) *gShapelet_e:* To transform a time-variant graph into a univariate time series, gShapelet_e counts the number of edges as statistical magnitude.

2) Evaluation Measures: We randomly select 60% time-variant graphs for training, and the remaining time-variant graphs are used for testing. When using frequent subsequence or subgraph-based methods, we need to set the support threshold σ. Given a set of data $\mathcal{G} = \{g_i | g_i = \langle g_i^{(1)} \cdots g_i^{(m_i)} \rangle\}$, a support value σ of a transformation subsequence and subgraph in graph sequence g' are defined as

$$\sigma = \frac{\left| \{g_i | g_i \in \mathcal{G}, \mathrm{ts}(g') \sqsubseteq \mathrm{ts}(g_i)\} \right|}{|\mathcal{G}|} \text{ for subsequences}$$

$$\sigma = \frac{\left| \{g_i | g_i \in \mathcal{G}, g' \sqsubseteq g_i\} \right|}{|\mathcal{G}|} \text{ for subgraphs.}$$
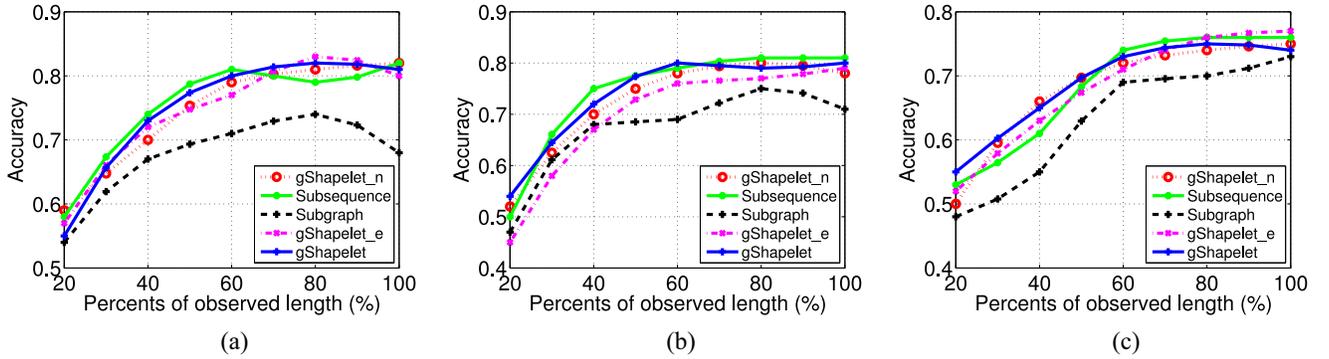
Fig. 5. Accuracy comparisons with respect to different time stages on both synthetic and real-world time-variant graph data sets. (a) Synthetic time-variant graph. (b) Phone call time-variant graph. (c) Weibo time-variant graph.
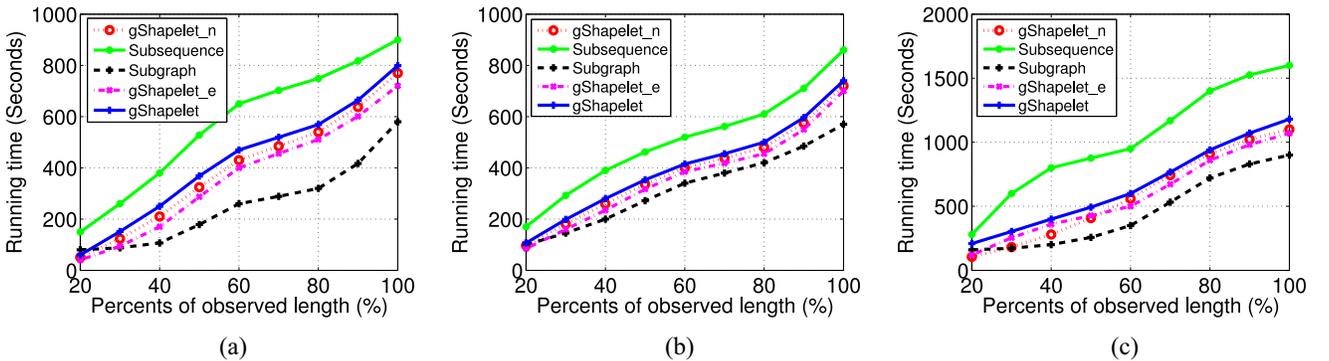

Fig. 6. Average CPU time with respect to different time stages on both synthetic and real-world time-variant graph data sets. (a) Synthetic time-variant graph. (b) Phone call time-variant graph. (c) Weibo time-variant graph.

C. Experimental Results

The four algorithms are compared under different observation lengths of the time-variant graphs, i.e., 20%, 40%, 60%, 80%, and 100% in each time-variant graph. In Figs. 5 and 6, we report the algorithm performance in terms of classification accuracy and running time, respectively.

1) Effectiveness Results: The results in Fig. 5 show that classification accuracy increases with respect to observation time, and the results tend to be stable as the time elapses. This is mainly because more vertices are observed which leads to more accurate shapelet or subsequence or subgraph patterns. Once the participant length reaches 60%, the algorithms are likely to find the best graph-shapelet patterns, so the accuracy tends to become stable. These results imply that graph-shapelet patterns can be used for early prediction.

As we can observe from Fig. 5, the graph-shapelet-based approaches have comparable accuracy with the subsequence-based method and perform much better than the subgraph-based method. This is because the subsequence-based method mines frequent features from the entire time-variant graphs. This kind of approach ensures classification accuracy, but it requires more running time. The graph-shapelet-based methods first find discriminative segments from the time-variant graphs, and then find features from the segments instead of the whole time-variant graphs. This method results in a significant reduction in running time. Consequently, as with on-line

and off-line methods, accuracy is traded for running time performance, especially on large-scale data sets. Our aim is to make the proposed methods more efficient under comparable accuracy. The proposed graph-shapelet methods outperform the subgraph-based method because the latter does not consider the temporal correlations during subgraph mining, and the distance between subgraph patterns may change irregularly over time.

2) Running Time Comparison: Fig. 6 shows the running time of the methods, where a subgraph-based method has the lowest running time while a subsequence-based method is the most time consuming. The reason is that the subsequence-based method needs to search the entire time-variant graphs to find discriminative substructures whereas the subgraph-based method only mines frequent subgraphs from a single graph. There is no significant difference among *gShapelet*, *gShapelet_n*, and *gShapelet_e*, because the only difference in these three methods is when transforming a time-variant graph into a univariate time series.

The classification accuracy and running time results show that the graph-shapelet pattern-based method has higher accuracy but competitive running time compared with subgraph-based approaches, while graph-shapelet pattern-based method has much lower running time but similar accuracy performance. Thus, we have to tradeoff the accuracy and running time. Considering the improvement of accuracy, the graph-shapelet pattern-based method is more reasonable

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.
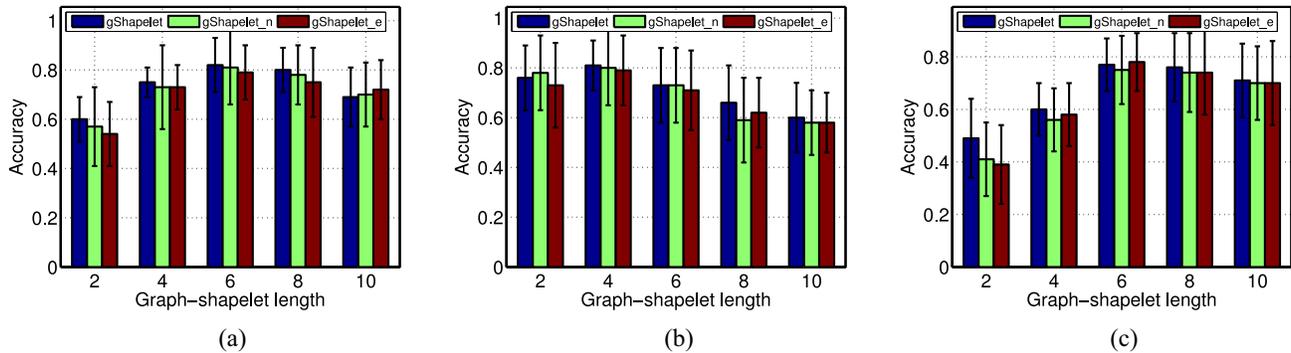

Fig. 7. Comparisons with varying graph-shapelet length on both synthetic and real-world time-variant graph data sets. (a) Synthetic time-variant graph data. (b) Phone call time-variant graph data. (c) Weibo time-variant graph data.

for time-variant graph learning. This is because graph-shapelet pattern-based methods consider the transformation relationships between changing graphs, and graph-shapelet patterns are able to learn the discriminative structures of data belonging to different classes. In addition, graph-shapelet patterns are usually much shorter than the original time series, which avoids comparisons being made on the entire data set and results in more efficient running time performance.

3) Analysis of gShapelet Algorithm: Fig. 7 shows the results of gShapelet using both vertices and edges as the graph statistics, respectively, compared to gShapelet_n only using vertices as the graph statistics and gShapelet_e only using edges as the graph statistics, with respect to various graph-shapelet pattern lengths. We observe that the highest classification accuracy is achieved when *len* is 6, 4, 6 for the Synthetic, MIT, and Weibo time-variant graph data, respectively.

Fig. 8 shows a simple yet interesting example extracted from the MIT phone call time-variant graph data. In the first sequence, "normal" (nonhub) persons 1 and 2 connect with hub person 2, which leads to an increased number of participants. On the other hand, communications generated by normal persons have a relatively stable number of users. This implies that the first sequence is likely to contain interesting information, and the potential graph-shapelet patterns can be explored from the corresponding graph transformation sequence.

VI. DISCUSSION

Intuitively, since the outbreak of an information propagation graph is justified by the number of vertices influenced by the propagation, alert readers may wonder why we transfer time variant-graphs as transformation sequences and further identify graph-shapelet patterns for classification, or why we do not directly use the change in the number of vertices as features for classification. Our hypothesis is that outbreak in information propagation is driven by some special graph structure, rather than simply being determined by vertices which have a large number of degrees. Indeed, an information propagation may not be outbreak even if it reaches some vertices with very high degrees (i.e., popular users). This is consistent with the
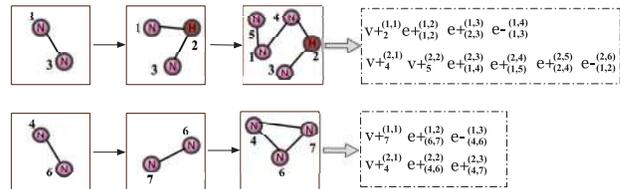

Fig. 8. Two graph transformation sequences extracted from the MIT phone call time-variant graph data. The symbol "N" represents a normal (nonhub) person and "H" represents a hub person. The first graph sequence shows weekly phone call time-variant graph data contain a hub person, while the second graph sequence shows all the participants as normal persons.

research in [56], which observe that information propagation is not mainly spread by vertices with a large number of degrees; rather, it is spread by vertices with a medium number of degrees.

To further validate our hypothesis, we use a real-world case study from Weibo time-variant graph data to demonstrate the discovered graph-shapelet patterns. Fig. 9 is an example of information propagation outbreak which was crawled every half an hour. The Weibo time-variant graph information propagation concerns a news message propagated in a University community regarding an interesting institutional documentary. At different stages, the propagation of the news forms variant graphs. The diffusion of one message in a propagation stage constitutes one time-variant graph, and the diffusion of messages in the community forms a time-variant graph dataset. The message diffusion includes outbreak and nonoutbreak time-variant graphs, and our aim is to classify each information propagation graph record into the correct class (outbreak versus nonoutbreak). A naive method of predicting time-variant graph outbreak is to count the volume of vertices, but the time-variant graph is not only influenced by the vertices but by the structures connecting the vertices, such as who is sending the message, who resends the message, when the message is sent or resent, etc., which is precisely captured by time-variant graphs. The general graph classification uses each graph separately in a certain period to discover features. In reality, a diffusion graph will influence subsequent graphs. For time-variant graph classification, we consider the variation between different graphs. Because a graph-shapelet pattern

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

12   IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS
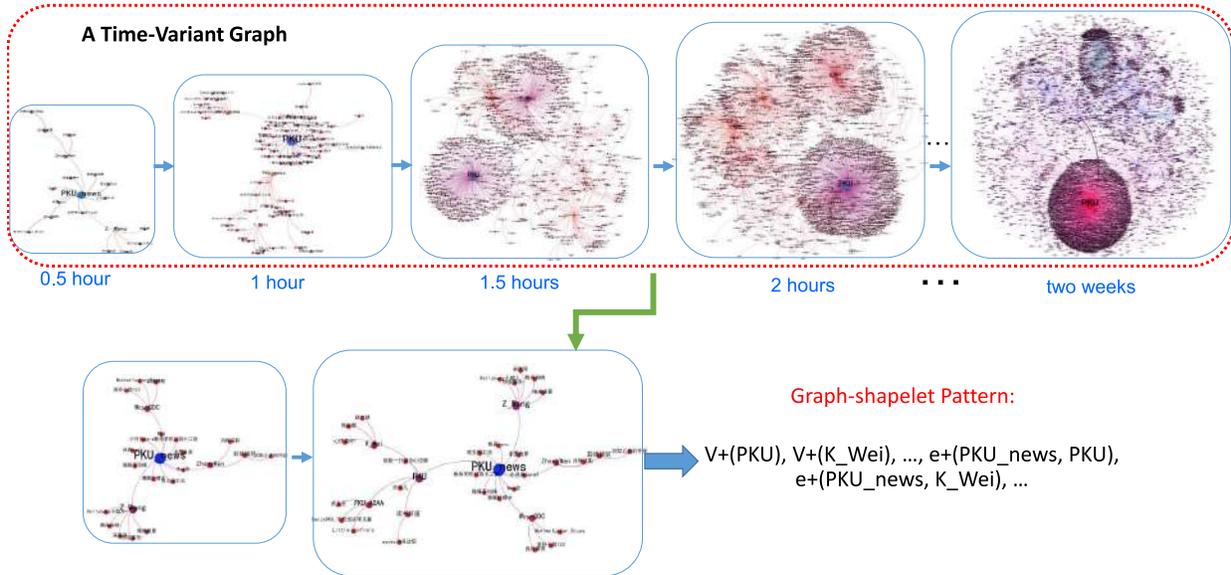

Fig. 9.   An example of a Sina Weibo message propagation as a time variant graph. At each propagation stage, the diffusion (including reached vertices and propagation edges) constitutes a graph. The propagation of the message in temporal order will form a set of temporally related graphs. Each Weibo propagation is regraded as a time-variant graph.

denotes the transformation of vertices and edges between different graphs, it can inherently capture the vertex information, structure information, and temporal variation graphs, for time-variant graph classification.

Fig. 9 shows an example of the mined graph-shapelet patterns. This graph-shapelet pattern example represents the outbreak class and demonstrates that a time-variant graph tends to be outbreak if there is a sequence of special vertex and edge insertions, such as vertex PKU and edge (PKU_news, PKU) which represents Peiking Universality.

VII. Conclusion

In this paper, we formulated a new time-variant graph classification task and proposed the use of graph-shapelet patterns as features to represent time-variant graphs for learning and classification. We argued that existing graph classification methods are all based on static settings, where training graphs are assumed to be independent of one another and the structure of each graph remains unchanged. In reality, many applications involve data with dynamic changing structures. Accordingly, a time-variant graph can be used to represent a sequence of graphs and capture the dynamic evolution of the underlying object. The inherent structural dependency and temporal correlations of the time-variant graphs raise significant challenges, and we advocated a graph-shapelet pattern concept to detect significant changes in the time-variant graph as features. By using graph-shapelet patterns to convert a time-variant graph as a tokenized sequence, we can effectively calculate the distance between two time-variant graphs for classification. Experiments on synthetic and real-world data demonstrated that our method is much more accurate than traditional subgraph feature-based approaches.

The key innovation of this paper compared to existing work in the area is threefold: 1) a new time-variant graph learning task to model dynamic changes in structured data; 2) a unique graph-shapelet pattern as a feature for capturing the structural and temporal dependency of graphs; and 3) a fast time-variant graph classification model using structure features and edit similarity.

Although our proposed graph-shapelet features and algorithms for classifying time-variant graphs achieve faster running time and high accuracy, some limitations exist that need improvement in the future work: 1) new approaches and theoretical studies of converting a time variant graph to a time series needs to be studied for minimum information loss and 2) the proposed algorithm, based on edit similarity, is a straightforward solution for graph classification problems, however, more state-of-the-art graph classification algorithms could be applied to achieve a better and faster performance, such as graph boosting (gboost) method. This paper inspires some interesting directions for future work: 1) the problem could be further extended by considering the graph structure when converting a time-variant graph to a time series and 2) correlations of subgraphs could assist graph stream classification or incremental subgraph features.

References

[1] C. C. Yang, X. Shi, and C.-P. Wei, "Discovering event evolution graphs from news corpora," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 39, no. 4, pp. 850–863, Jul. 2009.

[2] C. Deng, R. Ji, D. Tao, X. Gao, and X. Li, "Weakly supervised multi-graph learning for robust image reranking," *IEEE Trans. Multimedia*, vol. 16, no. 3, pp. 785–795, Apr. 2014.

[3] J. Wu, X. Zhu, C. Zhang, and P. S. Yu, "Bag constrained structure pattern mining for multi-graph classification," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 10, pp. 2382–2396, Oct. 2014.

[4] X. Yuan, J. Guo, X. Hao, and H. Chen, "Traffic sign detection via graph-based ranking and segmentation algorithms," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 45, no. 12, pp. 1509–1521, Dec. 2015.

[5] J. N. K. Liu, Y.-L. He, E. H. Y. Lim, and X.-Z. Wang, "A new method for knowledge and information management domain ontology graph model," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 43, no. 1, pp. 115–127, Jan. 2013.

[6] A. Sztyber, A. Ostasz, and J. M. Kościelny, "Graph of a process— A new tool for finding model structures in a model-based diagnosis," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 45, no. 7, pp. 1004–1017, Jul. 2015.

[7] L. C. Rêgo and A. M. dos Santos, "Probabilistic preferences in the graph model for conflict resolution," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 45, no. 4, pp. 595–608, Apr. 2015.

[8] M. Deshpande, M. Kuramochi, N. Wale, and G. Karypis, "Frequent substructure-based approaches for classifying chemical compounds," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 8, pp. 1036–1050, Aug. 2005.

[9] H. Shang, Y. Tao, Y. Gao, C. Zhang, and X. Wang, "An improved invariant for matching molecular graphs based on VF2 algorithm," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 45, no. 1, pp. 122–128, Jan. 2015.

[10] X. Kong, W. Fan, and P. S. Yu, "Dual active feature and sample selection for graph classification," in *Proc. Int. Conf. Knowl. Disc. Data Min. (KDD)*, 2011, pp. 654–662.

[11] R. Hong, L. Zhang, C. Zhang, and R. Zimmermann, "Flickr circles: Aesthetic tendency discovery by multi-view regularized topic modeling," *IEEE Trans. Multimedia*, vol. 18, no. 8, pp. 1555–1567, Aug. 2016.

[12] H. Sabirin and M. Kim, "Moving object detection and tracking using a spatio-temporal graph in H. 264/AVC bitstreams for video surveillance," *IEEE Trans. Multimedia*, vol. 14, no. 3, pp. 657–668, Jun. 2012.

[13] X. Kong, P. S. Yu, X. Wang, and A. B. Ragin, "Discriminative feature selection for uncertain graph classification," in *Proc. SIAM Int. Conf. Data Min. (SDM)*, 2013, pp. 82–93.

[14] H. Wang, P. Zhang, I. Tsang, L. Chen, and C. Zhang, "Defragging subgraph features for graph classification," in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manag.*, 2015, pp. 1687–1690.

[15] H. Wang *et al.*, "Incremental subgraph feature selection for graph classification," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 1, pp. 128–142, Jan. 2017.

[16] H. Wang, J. Wu, S. Pan, P. Zhang, and L. Chen, "Towards large-scale social networks with online diffusion provenance detection," *Comput. Netw.*, vol. 114, pp. 154–166, Feb. 2017.

[17] C. C. Aggarwal, "On classification of graph streams," in *Proc. SIAM Int. Conf. Data Min. (SDM)*, 2011, pp. 652–663.

[18] S. Pan, J. Wu, X. Zhu, and C. Zhang, "Graph ensemble boosting for imbalanced noisy graph stream classification," *IEEE Trans. Cybern.*, vol. 45, no. 5, pp. 954–968, May 2015.

[19] H. Seo, J. Kim, and M.-S. Kim, "GStream: A graph streaming processing method for large-scale graphs on GPUs," *SIGPLAN Not.*, vol. 50, no. 8, pp. 253–254, 2015.

[20] L. Ye and E. Keogh, "Time series shapelets: A new primitive for data mining," in *Proc. ACM SIGKDD Int. Conf. Knowl. Disc. Data Min. (KDD)*, 2009, pp. 947–956.

[21] A. Mueen, E. Keogh, and N. Young, "Logical-shapelets: An expressive primitive for time series classification," in *Proc. ACM SIGKDD Int. Conf. Knowl. Disc. Data Min. (KDD)*, 2011, pp. 1154–1162.

[22] J. Grabocka, N. Schilling, M. Wistuba, and L. Schmidt-Thieme, "Learning time-series shapelets," in *Proc. ACM SIGKDD Int. Conf. Knowl. Disc. Data Min. (KDD)*, 2014, pp. 392–401.

[23] L. Bai and E. R. Hancock, "Fast depth-based subgraph kernels for unattributed graphs," *Pattern Recognit.*, vol. 50, pp. 233–245, Feb. 2016.

[24] K. Riesen and H. Bunke, "Graph classification by means of Lipschitz embedding," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 6, pp. 1472–1483, Dec. 2009.

[25] A. Maronidis, A. Tefas, and I. Pitas, "Subclass graph embedding and a marginal fisher analysis paradigm," *Pattern Recognit.*, vol. 48, no. 12, pp. 4024–4035, 2015.

[26] L. Rossi, A. Torsello, and E. R. Hancock, "Unfolding kernel embeddings of graphs: Enhancing class separation through manifold learning," *Pattern Recognit.*, vol. 48, no. 11, pp. 3357–3370, 2015.

[27] C. Ding and L. Zhang, "Double adjacency graphs-based discriminant neighborhood embedding," *Pattern Recognit.*, vol. 48, no. 5, pp. 1734–1742, 2015.

[28] F. Serratosa and X. Cortés, "Interactive graph-matching using active query strategies," *Pattern Recognit.*, vol. 48, no. 4, pp. 1364–1373, 2015.

[29] J. Sang and C. Xu, "Robust face-name graph matching for movie character identification," *IEEE Trans. Multimedia*, vol. 14, no. 3, pp. 586–596, Jun. 2012.

[30] M. Izadi and P. Saeedi, "Robust weighted graph transformation matching for rigid and nonrigid image registration," *IEEE Trans. Image Process.*, vol. 21, no. 10, pp. 4369–4382, Oct. 2012.

[31] B. Li, N. Chen, and U. Schlichtmann, "Statistical timing analysis for latch-controlled circuits with reduced iterations and graph transformations," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 31, no. 11, pp. 1670–1683, Nov. 2012.

[32] S. Wang, X. Guo, X. Mu, Y. Huo, and L. Qi, "Advanced weight graph transformation matching algorithm," *IET Comput. Vis.*, vol. 9, no. 6, pp. 960–966, Dec. 2015.

[33] H. Fei and J. Huan, "Structured sparse boosting for graph classification," *ACM Trans. Knowl. Disc. Data*, vol. 9, no. 1, pp. 1–22, 2014.

[34] M. Thoma *et al.*, "Near-optimal supervised feature selection among frequent subgraphs," in *Proc. SIAM Int. Conf. Data Min. (SDM)*, 2009, pp. 1076–1087.

[35] X. Yan and J. Han, "gSpan: Graph-based substructure pattern mining," in *Proc. IEEE Int. Conf. Data Min. (ICDM)*, 2002, pp. 721–724.

[36] H. Saigo, S. Nowozin, T. Kadowaki, T. Kudo, and K. Tsuda, "gBoost: A mathematical programming approach to graph classification and regression," *Mach. Learn.*, vol. 75, no. 1, pp. 69–89, 2008.

[37] S. Pan, J. Wu, and X. Zhu, "CogBoost: Boosting for fast cost-sensitive graph classification," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 11, pp. 2933–2946, Nov. 2015.

[38] S. Perkins and J. Theiler, "Online feature selection using grafting," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2003, pp. 592–599.

[39] J. Zhou, D. P. Foster, R. A. Stine, and L. H. Ungar, "Streamwise feature selection," *J. Mach. Learn. Res.*, vol. 7, pp. 1861–1885, Jan. 2006.

[40] L. H. Ungar, J. Zhou, D. P. Foster, and B. A. Stine, "Streaming feature selection using IIC," in *Proc. 10th Int. Workshop Artif. Intell. Stat. (AISTATS)*, 2005, p. 357.

[41] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. J. Keogh, "Querying and mining of time series data: Experimental comparison of representations and distance measures," *Proc. VLDB Endowment (PVLDB)*, vol. 1, no. 2, pp. 1542–1552, 2008.

[42] H. Wang and J. Wu, "Boosting for real-time multivariate time series classification," in *Proc. AAAI*, San Francisco, CA, USA, 2017, pp. 4999–5000.

[43] X. Xi, E. Keogh, C. Shelton, L. Wei, and C. A. Ratanamahatana, "Fast time series classification using numerosity reduction," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2006, pp. 1033–1040.

[44] J. Kleinberg, "Bursty and hierarchical structure in streams," in *Proc. ACM SIGKDD Int. Conf. Knowl. Disc. Data Min. (KDD)*, 2002, pp. 91–101.

[45] M. G. Elfeky, W. G. Aref, and A. K. Elmagarmid, "Periodicity detection in time series databases," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 7, pp. 875–887, Jul. 2005.

[46] L. Wei *et al.*, "Assumption-free anomaly detection in time series," in *Proc. Int. Conf. Sci. Stat. Database Manag. (SSDBM)*, 2005, pp. 237–240.

[47] J. Lin, E. Keogh, S. Lonardi, and P. Patel, "Finding motifs in time series," in *Proc. ACM SIGKDD Int. Conf. Knowl. Disc. Data Min. (KDD)*, 2002, pp. 53–68.

[48] D. Yankov, E. Keogh, J. Medina, B. Chiu, and V. Zordan, "Detecting time series motifs under uniform scaling," in *Proc. ACM SIGKDD Int. Conf. Knowl. Disc. Data Min. (KDD)*, 2007, pp. 844–853.

[49] B. Hartmann and N. Link, "Gesture recognition with inertial sensors and optimized DTW prototypes," in *Proc. IEEE Int. Conf. Syst. Man Cybern. (SMC)*, 2010, pp. 2102–2109.

[50] A. Mcgovern, D. H. Rosendahl, R. A. Brown, and K. K. Droegemeier, "Identifying predictive multi-dimensional time series motifs: An application to severe weather prediction," *Data Min. Knowl. Disc.*, vol. 22, nos. 1–2, pp. 232–258, 2011.

[51] Z. Xing, J. Pei, P. S. Yu, and K. Wang, "Extracting interpretable features for early classification on time series," in *Proc. SIAM Int. Conf. Data Min. (SDM)*, 2011, pp. 247–258.

[52] A. Inokuchi and T. Washio, "A fast method to mine frequent subsequences from graph sequence data," in *Proc. IEEE Int. Conf. Data Min. (ICDM)*, 2008, pp. 303–312.

[53] A. Sanfeliu and K.-S. Fu, "A distance measure between attributed relational graphs for pattern recognition," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-13, no. 3, pp. 353–362, May/Jun. 1983.

[54] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[55] N. Eagle, A. S. Pentland, and D. Lazer, "Inferring friendship network structure by using mobile phone data," *Proc. Nat. Acad. Sci. USA*, vol. 106, no. 36, pp. 15274–15278, 2009.

[56] J. Leskovec, M. McGlohon, C. Faloutsos, N. Glance, and M. Hurst, "Cascading behavior in large blog graphs: Patterns and a model," in *Proc. Soc. Appl. Ind. Math. Data Min. (SDM)*, 2007, pp. 551–556.

Haishuai Wang received the Ph.D. degree in computer science from the Center of Artificial Intelligence, University of Technology Sydney, Sydney, NSW, Australia.

He was a Post-Doctoral Associate with the Department of Computer Scient and Engineering, Washington University in St. Louis, St. Louis, MO, USA. He is a Research Fellow with the Department of Biomedical Informatics, Harvard University, Cambridge, MA, USA. His current research interests include data mining and machine learning as well as its applications on bioinformatics and social networks.

Dr. Wang is an Associate Editor for IEEE ACCESS and the Program Committee of the SIAM Data Mining Conference 2018 and NIPS ML4H 2017.

Jia Wu (M'16) received the Ph.D. degree in computer science from the University of Technology Sydney, Sydney, NSW, Australia.

He is currently a Lecturer with the Department of Computing, Faculty of Science and Engineering, Macquarie University, Sydney. His current research interests include data mining and machine learning. He has published over 100 top-tier journals, such as the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, IEEE TRANSACTIONS ON CYBERNETICS, *ACM Transactions on Knowledge Discovery Data*, *Pattern Recognition*) and conference papers, such as International Joint Conference on Artificial Intelligence, AAAI Conference on Artificial Intelligence, IEEE International Conference on Data Mining, IEEE International Conference on Data Engineering, SIAM International Conference on Data Mining, ACM Conference on Information and Knowledge Management in the above areas.

Dr. Wu was a recipient of ICDM Best Student Paper Candidate Award.

Xingquan Zhu (SM'12) received the Ph.D. degree in computer science from Fudan University, Shanghai, China.

He is a Professor with the Department of Computer and Electrical Engineering and Computer Science, Florida Atlantic University, Boca Raton, FL, USA. His current research interests include data mining, machine learning, bioinformatics, and computational advertising. He has published over 230 refereed journal and conference papers in the above fields.

Dr. Zhu has been an Associate Editor of the ACM TRANSACTIONS ON KNOWLEDGE DISCOVERY FROM DATA since 2017 and the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING from 2008 to 2012 and since 2014. He is currently serving on the Editor Board of the *International Journal of Social Network Analysis* and *Mining SNAM* since 2010, *Journal of Big Data* since 2013, and *Network Modeling Analysis in Health Informatics* and *Bioinformatics Journal* since 2014. He is a Program Committee Co-Chair of the 14th International Conference on Advanced Data Mining and Applications in 2018, and a Senior Program Committee Member of ACM SIGKDD 2018 and IEEE ICDM 2018. He was a Program Committee Co-Chair for the 14th IEEE International Conference on Bioinformatics and BioEngineering, IEEE International Conference on Granular Computing, 23rd IEEE International Conference on Tools with Artificial Intelligence, and the 9th International Conference on Machine Learning and Applications. He also served as a Conference Co-Chair for ICMLA-2012.

Yixin Chen received the Ph.D. degree in computer science from the University of Illinois at Urbana–Champaign, Champaign, IL, USA, in 2005.

He is a Professor of Computer Science and Engineering with Washington University in St. Louis, St. Louis, MO, USA. His research has been funded by NSF, NIH, DOE, Microsoft, and Memorial Sloan-Kettering Cancer Center. His current research interests include data mining, machine learning, artificial intelligence, and optimization.

Dr. Chen was a recipient of the Best Paper Award at the IDEAL Conference in 2016, the Distinguished Paper Award at the AMIA Conference in 2015, the Best Student Paper Runner-Up Award at the ACM SIGKDD Conference in 2014, the Best Paper Award at the AAAI Conference on Artificial Intelligence in 2010, the IEEE International Conference on Tools for AI in 2005, the Best Paper Award nominations at IEEE ICDM in 2013, the IEEE RTAS in 2012, KDD in 2009, the Early Career Principal Investigator Award from the Department of Energy in 2006 and a Microsoft Research New Faculty Fellowship in 2007. His research on planning has won First Prizes in the International Planning Competitions in 2004 and 2006. He is an Associate Editor for *ACM Transactions of Intelligent Systems and Technology*, *Annals of Mathematics and Artificial Intelligence*, and *Journal of Artificial Intelligence Research*. He was an Associate Editor for the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING from 2008 to 2012.

Chengqi Zhang (SM'95) received the Ph.D. degree from the University of Queensland, Brisbane, QLD, Australia, in 1991 and the D.Sc. degree (higher doctorate) from Deakin University, Geelong, VIC, Australia, in 2002.

Since 2001, he has been a Professor of Information Technology with the University of Technology Sydney, Sydney, NSW, Australia, where he has been the Director of the UTS Priority Investment Research Center for Quantum Computation and Intelligent Systems since 2008. Since 2005, he has been the Chairman of the Australian Computer Society National Committee for Artificial Intelligence. He has published over 200 research papers, including several in first-class international journals, such as the Artificial Intelligence, IEEE, and ACM Transactions. He has published six monographs and edited 16 books, and has attracted 11 Australian Research Council grants. His current research interests include data mining and its applications.

Dr. Zhang has been serving as an Associate Editor for three international journals, including the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING from 2005 to 2008. He served as the General Chair, the PC Chair, or the Organizing Chair for five international conferences, including ICDM 2010 and WI/IAT 2008. He is the General Co-Chair of KDD 2015 in Sydney and the Local Arrangements Chair of IJCAI-2017 in Melbourne, and a fellow of the Australian Computer Society.