Time2Graph+: Bridging Time Series and Graph Representation Learning via Multiple Attentions

Ziqiang Cheng, Yang Yang*, Shuo Jiang, Wenjie Hu, Zhangchi Ying, Ziwei Chai, Chunping Wang

Abstract—Time series modeling has attracted great research interests in the last decades. Among the literature, *shapelet*-based models aim to extract representative subsequences, and could offer explanatory insights in the downstream tasks. But most of those works ignore the seasonal effects on the subsequences, as well as the evolutionary characteristics of shapelets. In order to capture the shapelet dynamics and evolutions, in this paper, we propose a novel framework of bridging time series representation learning and graph modeling, with two different implementations. We first formulate the process of extracting time-aware shapelets by directly adding time-level attentions, then introduce the key idea of transforming time series data into shapelet evolution graphs, to model the shapelet evolutionary patterns. A straightforward solution is to enumerate all possible shapelet transitions among adjacent time series segments, and apply a random-walk-based graph embedding algorithm to learn time series representations (*Time2Graph*). We further extend Time2Graph by adopting graph attention mechanism to refine the procedure of modeling shapelet evolutions, namely *Time2Graph+*. Specifically, we transform each time series data into a unique unweighted shapelet graph, and use GAT to automatically capture the correlations between shapelets. Experimental results on three real-world datasets show the significant improvements of *Time2Graph+* over *Time2Graph* and 17 baseline methods, and observational analysis demonstrates the effectiveness and interpretability brought by both time-level and graph-level attentions. Furthermore, the success of online deployment of *Time2Graph+* model in State Grid of China validates the whole framework in the real-world application. Codes and documentations are available at https://github.com/petecheng/Time2GraphPlus.

Index Terms—Time Series Modeling, Time-Aware Shapelets, Graph Neural Networks, Graph Attention Networks

1 Introduction

Time series representation learning has attracted great research efforts in the last decades, and has been applied in many real-world applications [1, 2, 3, 4]. Those works aim to discover the temporal relationships within chronologically arranged data, and the key issue is how to extract representative features from time series. In the literature, related works can be categorized into feature engineering, kernel-based time series embedding [5], and deep sequence models [6, 7]. While these methods have achieved good performance under various scenarios [8, 9], some of them have also been subject to criticism regarding their drawbacks, i.e., feature engineering methods may need sufficient expert knowledge and are not generalizable across different data domains, and deep sequence models may raise concerns due to their lack of interpretability, etc.

To better understand the intrinsic properties of time series, shapelet-based models are proposed to unearth the repeated or important sequential patterns. *Shapelets*, the representative subsequences that may reflect the characteristics of the key waveforms [10], are expected to offer directly interpretable and explanatory insights behind the time series data, and shapelet-based frameworks have proven to be promising in various practical domains [11, 12, 13]. For example, under the scenario of the empty-nest elderly[1] recognition in State Grid[2], the target is to recognize empty-nest elderly based on users' electricity usage during a period of time, where shapelets of the electricity consumption records can potentially capture the individual electrical power consuming patterns. Most existing works considered shapelets as static [10, 15, 16, 17, 18], however, representative subsequences are often dynamic, which is reflected in two respects:

- First, the same shapelet appearing at different time slices may have different impacts. For instance, the empty-nest elderly may use air conditions less frequently when it is very hot or cold than others due to their frugal lifestyles, so a user who has lower electricity consumptions during this period is more likely to be the elderly; but the same pattern appearing at spring or autumn may not be able to distinguish these two groups of people, since it would be common for all users. We refer to the subsequences of a time series that are able to reflect their representativeness at different time slices as *time-aware shapelets*.
- Second, the way in which the shapelets evolve over time is also vital to fully understand the time series data. In fact, shapelets with small values at a particular time can hardly distinguish an elderly from a common user who indeed consumes a low level of electrical power during that period. It is more reasonable to analyze the electricity

- *Ziqiang Cheng, Yang Yang, Wenjie Hu and Ziwei Chai were with the College of Computer Science and Technology, Zhejiang University, China.*
- *Yang Yang was the corresponding author. E-mail: yangya@zju.edu.cn.*
- *Shuo Jiang and Zhangchi Ying were with the State Grid Zhejiang Electric Power Corporation, China.*
- *Chunping Wang was with FinVolution Group, China.*

Manuscript received Nov. 2020; revised Apr. 2021.

[1] Refers to those elderly with no children or whose children have all lived away from township, left them staying alone. It may raise public safety concerns. since the empty-nest elderly could probably misuse the electrical appliances, or are unable to deal with electricity-related emergencies. As reported in 2013 [14], in China, about 50% (over 100 million) of the total elderly population were the empty-nest.

[2] A Chinese state-owned electric utility corporation, and the largest utility company in the world.

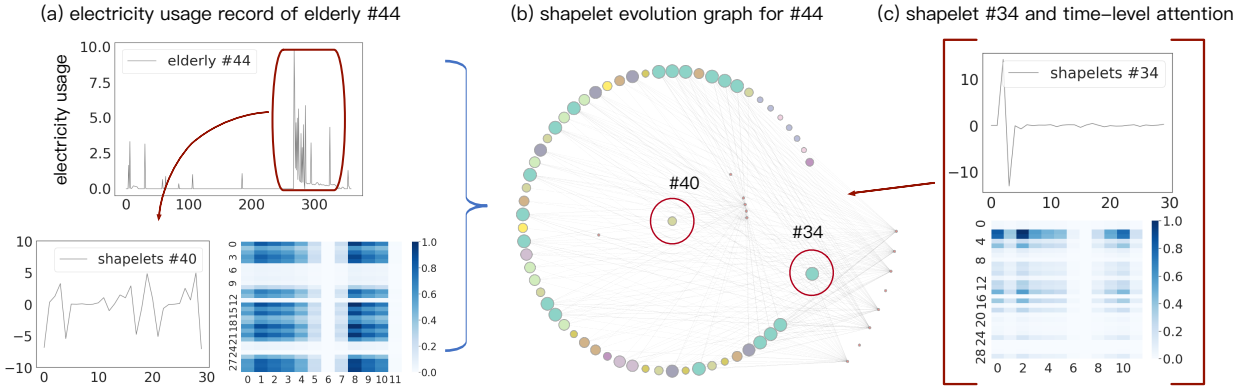


Fig. 1: Illustration of *Time2Graph+* framework in the scenario of user electricity consuming. (a) shows a one-year electricity consumption of an empty-nest elderly user, along with the assigned shapelet #40. After time-aware shapelets being extracted, it constructs the shapelet graph for each single sequence, and captures the evolutionary patterns of shapelets using graph attention networks (b), and (c) visualizes one typical shapelet #34 and its time-level attentions. Note that in (b), the node size is proportional to its in-degree, from which the node color is mapped; and the edge width is proportional to the attention score between two nodes.

consuming behavior over a long range of time, to check whether the user actually has a habit of using less electrical power. In other words, an important clue here is, how *the subsequence patterns evolve over time*.

Motivated by the abovementioned intuitions, we propose a novel framework, *Time2Graph* [19], to model shapelet dynamics . It first extracts parameterized time-aware shapelets from a large pool of candidates under a specific criterion, by introducing timing factors when defining the distance between sequences; then constructs the shapelet evolution graph, in which nodes denote the extracted shapelets, and edges reflect the transition relationships between vertices. Finally, *Time2Graph* deploys Deepwalk to learn the hidden features of shapelets from shapelet evolution graph, and concatenates or aggregates those shapelet embeddings to obtain the time series representations which can be applied in various downstream tasks. Although some previous works [20, 21, 22, 23] have proposed to convert time series data into various graphs, *Time2Graph* is believed to be the pioneer work to study "shapelet dynamics" of time series from the perspective of graphs; and intuitively translates the problem of time series representation learning into graph embedding. Experimental results on several benchmark and real-world datasets demonstrate its effectiveness, but there are several limitations on the implementation of *Time2Graph*:

- **L1**: In the process of constructing the shapelet evolution graph, it first conducts segmentations, assign "similar" shapelets to each segment, and recognizes adjacent appearances of shapelet pairs as their "possible transitions". Since we would never know the ground truth of which shapelets should be assigned to each segment, the dynamic transitions between adjacent shapelets may be incorrectly counted, and be biased on the observed time series data.
- **L2**: Another disadvantage of *Time2Graph* is that, some weak transitions between shapelets might be magnified, since it simply sums up all the counted weighted edges over the entire set of time series, and noisy connections between some pairs of vertices are probably much stronger than they should be after the normalization on edge weights.
- **L3**: Last but not least, *Time2Graph* only constructs a uniform shapelet evolution graph from all time series data to extract frequent shapelet transitions, while infrequent ones may reflect unique transition patterns of a time series.

After all, the way of constructing the shapelet evolution graph in *Time2Graph* may bring several biases in learning shapelet embeddings, and cause performance drops in the downstream tasks. To address those potential shortcomings, we then propose *Time2Graph+*, an extension of *Time2Graph* that applies graph attention networks to model the shapelet evolutions, where we *learn* the evolutionary patterns of shapelets via graph attentions rather than approximately counting on adjacent transitions. Specifically, instead of constructing a uniform shapelet evolution graph for all time series data, we first transform each time series into a unique unweighted graph (**L3**), since different sequences may have various evolutionary patterns, i.e., graph structures; then apply graph attention mechanisms to capture the transition patterns between shapelets, by assigning learnable dynamic edge weights which can be naturally interpreted as the shapelet transition probability (**L1** and **L2**). We design an end-to-end supervised framework to learn the global attention parameters, by translating the original time series classification problem into the graph-level classification (Sec. 4.3.2). To validate the effectiveness of *Time2Graph+* framework, we conduct extensive experiments on both public and real-world datasets: *Time2Graph+* achieves competitive performance on the benchmark datasets in *UCR Time Series Archive*, and significantly outperforms *Time2Graph* in three real-wold scenarios (averaged +3% in terms of F1), as well as 17 baseline methods ranging from distance-based and shapelet-based models to deep sequence models. In addition, visualizations of time-level and graph-level attentions demonstrate the interpretability of *Time2Graph+*, which can bring in extra explanatory insights with the help of expert knowledge.

Fig. 1 gives an overview of *Time2Graph+*. Fig. 1a shows a one-year electricity usage record of an empty-nest elderly user (#44), formally defined as "at most two people at home with at least one person above 65 years old". The electricity consumption records are the typical time series which are segmented by months, thus the original sequence

is divided into twelve segments. *Time2Graph+* model first learns time-aware shapelets, and assigns each month with several patterns that are close to the corresponding segment under some threshold. For instance, we present the assigned shapelet #*40* for the subsequence marked by a *red box*, along with its time-level attentions in the bottom of Fig. 1a, where dark areas indicate that the corresponding time step is more important compared to light areas. *Time2Graph+* then aims to model the shapelet evolutions via graphs (Fig. 1b,c): it constructs the shapelet evolution graph for each time series, and captures the individual shapelet evolutionary patterns by applying graph attention mechanism. The graph structure along with the attention scores on edges reveal the shapelet evolutionary patterns, as shown in Fig. 1b: there are several centered vertices with large degrees in the graph, and the top-2 are shapelet-$\#34$ and $\#40$ (marked with red circles); this means that for the elderly $\#44$, the most typical shapelet transition patterns are evolving from or into these two centered shapelets. Furthermore, extra explanatory insights could be inferred from constructed graphs and time-level attentions with the help of expert domain knowledge, as shown in Fig. 1c: shapelet $\#34$ is the shape of a steep fluctuation following by a very flat sequence; its time-level attention matrix emphasizes that the steep fluctuation are with large attention weights, and such shapes are much more representative in spring and autumn (especially at Mar. and Nov.). It might be a typical electricity consumption pattern of the empty-nest elderly users who consumes an unstable amount of electricity during the period of changes of seasons, and then falls into a relatively flat state. More detailed case studies can be referred in Sec. 5.5. Overall, *Time2Graph+* is able to extract representative subsequences in the time series data, and to mine their evolutionary patterns from the perspective of graph modeling. Finally, we summarize our contributions of this paper to the field as follows:

- We give an overview of bridging the domain of time series representation learning and graph embedding, and the key idea of transforming time series data into a shapelet graph to model the subsequence dynamics.
- We illustrate two ways of constructing the shapelet evolution graph (Sec. 4): counting possible shapelet transitions (*Time2Graph*, Sec. 4.2) and learning those transition patterns via graph attentions (*Time2Graph+*, Sec. 4.3). We analyze the potential biases of both methods, and compare them in the downstream time series classification tasks.
- We validate the effectiveness of *Time2Graph* framework based on three public and three real-world datasets. Experimental results show that *Time2Graph+* achieves notably better performance when compared with 17 state-of-the-art baselines, and visualizations of both time- and graph-level attentions demonstrate the extra interpretability (Sec. 5).
- Furthermore, we deploy the *Time2Graph+* model in a real-world scenario: recognizing the empty-nest elderly in State Grid of China, in Jinhua, Zhejiang province. The success of model deployment and application again confirms the validity of our proposed framework (Sec. 6).

2 Related Works

Time series modeling has attracted extensive research efforts over a wide range of fields, such as image alignment [24], speech recognition [25], and motion detection [26], etc. One important technique is Dynamic Time Warping (DTW) [27], which aims to find the appropriate warping path of alignments between time series data, and many applications have been proposed based on this metric [28, 29].

Traditional time series classification models try to extract efficient features from sequences and develop a well-trained classifier, such as BoP [30], TSF [31], EE [32], etc. But the major challenge is that there are often no explicit features in sequences [33], then many works focused on time series representation learning [34]: models based on DTW and traditional embedding techniques [5] aim to project original time series data into feature-vector space; symbolic representations (SR) [35, 36, 37] transform time series using symbols such as characters from a given alphabet; shapelet-discovery-based models [10, 17, 18, 38, 39], from another perspective, try to find typical subsequences based on certain criteria such as information gain. Since exploring significant shapelets from many candidates is very time-consuming, [18] proposed a fast shapelet discovery algorithm based on SR and hashing. Inspired by the great advances in deep learning, many deep sequential models have been proposed for time series classification. Deep learning approaches for TSC often adopt RNN-based and stacked CNN layers to extract time series features, followed with an output layer for predictions. They can be roughly categorized into two main groups [40]: the generative [41, 42] and the discriminative [43, 44] models, and many recent works also utilize unlabeled data under semi-supervised setting [45, 46].

Another relevant domain to this paper is graph representation learning (also known as graph or network embedding). [47, 48] give detailed surveys of this literature, and categorize graph embedding methodologies into three groups:

- Matrix factorization, that utilizes the adjacent or the Laplacian matrix of the graph, to obtain the hidden feature vectors of each row (entry) [49, 50].
- Random-walk-based frameworks generate node sequences from graphs by random walks to capture local structural information, e.g., DeepWalk [51], node2vec [52], and DynamicTriad [53], etc. These works inherit the idea of Word2Vec [54], regarding each node as a word, a node sequence as a sentence, and learn the node representations under the framework of CBOW or Skip-to-Gram.
- Deep learning models design DNNs to learn node features. For example, SDNE [55] adopts auto-encoders to optimize both 1- and 2-order proximity. Besides, DNNs are specifically designed on graphs, which are widely known as GNNs, among which convolutional GNNs are the most popular [56]: they generalize the operation of "convolution" from grid data to graphs, and consist of two main directions: spectral- and spatial-based. Spectral-based approaches formulate graph convolutions by introducing graph convolution kernels from the view of graph signal processing [57], such as ChebNet [58], GCN [59], etc., and spatial-based models instead define graph convolutions as information propagation between nodes, including MPNN [60], GraphSAGE [61], etc. Motivated by the attention mechanisms that are adopted in deep sequence models [62], Graph Attention Networks (GAT) is proposed to automatically learn the edge weights (attention scores) between each pair of connected nodes [40].

3 Notations and Preliminaries

A time series set $T = \{\boldsymbol{t}_1, \cdots, \boldsymbol{t}_{|T|}\}$, where each $\boldsymbol{t}$ contains n chronologically arranged elements , i.e., $\boldsymbol{t} = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n\}$. A *segment* $\boldsymbol{s} = \{\boldsymbol{x}_i, \cdots, \boldsymbol{x}_j\}$ of $\boldsymbol{t}$ is a contiguous subsequence; if $\boldsymbol{t}$ can be divided into m segments of equal length l, then we have $\boldsymbol{s}k = \{\{\boldsymbol{x}_{l*(k-1)+1}, \cdots, \boldsymbol{x}_{l*(k-1)+l}\}$. We denote the dissimilarity between two segments $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ as $d(\boldsymbol{s}_i, \boldsymbol{s}_j)$, and in time series modeling, time warping techniques are often adopted to find an appropriate alignment for the given pair of sequences, where an alignment is defined as

Definition 1. **Alignment.** Given two segments $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ with length l_i and l_j respectively, an alignment $\boldsymbol{a} = (\boldsymbol{a}_1, \boldsymbol{a}_2)$ is a pair of two index sequences of length p, satisfying that

$$1 = \boldsymbol{a}_k(1) \leq \cdots \leq \boldsymbol{a}_k(p) = l_k,\ \boldsymbol{a}_k(n+1) - \boldsymbol{a}_k(n) \leq 1,$$
$$\text{for } k = i, j, \text{ and } 1 \leq n \leq p-1$$

We denote all possible alignments $\boldsymbol{a}$ for two segments $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ as $\mathcal{A}(\boldsymbol{s}_i, \boldsymbol{s}_j)$, then one popular time-warping based metric, *DTW (Dynamic Time Warping)*, can be illustrated as Eq. (1), where $\tau(\boldsymbol{s}_i, \boldsymbol{s}_j|\boldsymbol{a})$ is a predefined dissimilarity for two sequences under the alignment $\boldsymbol{a}$ [27]. We refer to the alignment achieving the minimum in Eq. (1) as $\boldsymbol{a}^*$.

$$d_{DTW}(\boldsymbol{s}_i, \boldsymbol{s}_j) = \min_{\boldsymbol{a} \in \mathcal{A}(\boldsymbol{s}_i, \boldsymbol{s}_j)} \tau(\boldsymbol{s}_i, \boldsymbol{s}_j|\boldsymbol{a}) \tag{1}$$

We further measure the dissimilarity between a segment $\boldsymbol{s}$ and a time series $\boldsymbol{t} = \{\boldsymbol{s}_1 \cdots \boldsymbol{s}_m\}$. Inspired by the literature that we often say a segment $\boldsymbol{s}$ is close to $\boldsymbol{t}$ if there exists some segment $\boldsymbol{s}'$ in $\boldsymbol{t}$ between which the distance of $\boldsymbol{s}$ is rather small, we define the distance between $\boldsymbol{s}$ and $\boldsymbol{t}$ as

$$D(\boldsymbol{s}, \boldsymbol{t}) = \min_{1 \leq k \leq m} d(\boldsymbol{s}, \boldsymbol{s}_k) \tag{2}$$

Based on these definitions, previous works have proposed novel methods to extract typical subsequences, i.e., shapelets, to distinguish the *representative power* of segments:

Definition 2. **Shapelet.** A shapelet $\boldsymbol{v}$ is a segment that is representative of a certain class. It can separate T into two disjoint subsets, one that is close to $\boldsymbol{v}$ and another far from $\boldsymbol{v}$ by some specific criterion, such that positive and negative time series samples can be put into different groups. The criteria can be formulated as

$$\mathcal{L} = -g(S_{pos}(\boldsymbol{v}, T), S_{neg}(\boldsymbol{v}, T)) \tag{3}$$

$\mathcal{L}$ measures the dissimilarity between positive and negative samples towards the shapelet $\boldsymbol{v}$. $S_*(\boldsymbol{v}, T)$ denotes the set of distances with respect to a specific group T_*, i.e., positive or negative class; the function g takes two finite sets as input, returns a scalar value to indicate how far these two sets are, and it can be defined as *information gain* [10], or some dissimilarity measurements on sets, i.e., *KL* divergence.

4 Proposed Approaches

In this section, we illustrate the general idea of Time2Graph, a novel time series representation learning framework that transforms time series data to graphs, with time-aware shapelets as nodes and shapelet transitions as edges, and discuss two different implementations in detail. After introducing the time-aware shapelets (Sec. 4.1), we first briefly review *Time2Graph* [19] (Sec. 4.2), then propose an extension of Time2Graph, namely *Time2Graph+* (Sec. 4.3), that applies the graph attention mechanism to model the relationships between each pair of nodes in the shapelet evolution graph.

4.1 Time-Aware Shapelet Extraction

The previous definition of shapelets ignores that subsequences may have different representative powers at different time. For example, low consumption of electrical power in spring is normal, whereas it is a strong signal for identifying some specific group of users in summer, when high temperatures often lead to high electricity usage. Therefore, we consider time-aware shapelets in this paper. We design time-level attentions for quantitatively measuring the timing effects of shapelets at different scales. Specifically, we introduce the *local attention* $\boldsymbol{w_n}$ to denote the importance of the k-th element of a particular shapelet, then the distance between a shapelet $\boldsymbol{v}$ and a segment $\boldsymbol{s}$ is redefined as

$$\hat{d}(\boldsymbol{v}, \boldsymbol{s}|\boldsymbol{w}) = \left(\sum_{k=1}^{p} \boldsymbol{w}_{\boldsymbol{a}_1^*(k)} \cdot (\boldsymbol{v}_{a_1^*(k)} - \boldsymbol{s}_{\boldsymbol{a}_2^*(k)})^2\right)^{\frac{1}{2}} \tag{4}$$

The intuitive explanation of Eq. (4) is to project the weight $\boldsymbol{w}$ onto the *DTW* alignment path. On the other hand, at a *global level*, we aim to measure the timing effects across segments.It is inspired by the fact that shapelets may represent different meaning at different time steps, and it is straightforward to measure such deviations by adding segment-level weights. We set a *global attention* $\boldsymbol{u}_m$ to capture the cross-segments influence, then the distance between a shapelet $\boldsymbol{v}$ and a time series $\boldsymbol{t}$ can be rewritten as

$$\hat{D}(\boldsymbol{v}, \boldsymbol{t}|\boldsymbol{w}, \boldsymbol{u}) = \min_{1 \leq k \leq m} \boldsymbol{u}_k \cdot \hat{d}(\boldsymbol{v}, \boldsymbol{s}_k|\boldsymbol{w}) \tag{5}$$

where $\boldsymbol{t} = \{\boldsymbol{s}_1, \cdots, \boldsymbol{s}_m\}$. Eq. (5) denotes the two-level time-aware distance between a shapelet $\boldsymbol{v}$ and a time series $\boldsymbol{t}$, and the parameters $\boldsymbol{w}, \boldsymbol{u}$ associated with each specific shapelet can be learned separately under some criteria. Given a classification task, we establish a supervised learning method to select the most important time-aware shapelets and to learn their time-level attentions $\boldsymbol{w}_i$ and $\boldsymbol{u}_i$ for each shapelet $\boldsymbol{v}_i$. In particular, we have a pool of segments as shapelet candidates, and a set of time series T with labels. For each shapelet candidate $\boldsymbol{v}$, we modify Eq. (3) as

$$\hat{\mathcal{L}} = -g(S_{pos}(\boldsymbol{v}, T), S_{neg}(\boldsymbol{v}, T)) + \lambda||\boldsymbol{w}|| + \epsilon||\boldsymbol{u}|| \tag{6}$$

where λ and ϵ are the weight of 2-norm penalties. In practice, g is set as the KL-divergence, because it intuitively compares the similarity of two sets and is differentiable while the commonly-used choice, information gain, is not. We further assume that the given parameterized distance sets both follow some particular distributions, e.g., Gaussian Distribution; then g and its gradients can be easily computed by a closed-form solution. After learning the time-level attentions from shapelet candidates, we select the top K shapelets with minimal loss in Eq. (6).

4.2 Basic model: Time2Graph

4.2.1 Shapelet Evolution Graph

After obtaining shapelets, many works use BoP [63] or similar methods to represent the time series, but these algorithms

ignore the correlations between shapelets. Therefore, we propose the concept of *Shapelet Evolution Graph* as follows:

Definition 3. **Shapelet Evolution Graph.** It is a directed and weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ in which $\mathcal{V}$ consists of $\mathcal{K}$ vertices, each denoting a shapelet, and each directed edge $e_{ij} \in \mathcal{E}$ is associated with a weight w_{ij}, indicating the occurrence probability of shapelet $\boldsymbol{v}_i \in \mathcal{V}$ followed by another shapelet $\boldsymbol{v}_j \in \mathcal{V}$ in the same time series.

Graph Construction. We assign each segment $\boldsymbol{s}_i$ to several shapelets that are the most closest to $\boldsymbol{s}_i$ according to the *time-aware dissimilarity*. Then a problem naturally rises as how far the distance would be considered as *closest*? One simple but effective solution is to predefine a threshold δ such that distances less than δ are treated as close, and in practice, we can determine δ by experimental statistics on the training dataset. For convenience, we denote those shapelets assigned to segment $\boldsymbol{s}_i$ as $\boldsymbol{v}_{i,*}$, and say that $\boldsymbol{v}_{ij}$ is the j-th assignment of $\boldsymbol{s}_i$. For the purpose of measuring how reasonable our assignment is, we standardize assignment probability $\boldsymbol{p}_{i,j}$ as

$$\boldsymbol{p}_{i,j} = \frac{\max(\hat{d_{i,*}}(\boldsymbol{v}_{i,*}, \boldsymbol{s}_i)) - \hat{d_{i,j}}(\boldsymbol{v}_{i,j}, \boldsymbol{s}_i)}{\max(\hat{d_{i,*}}(\boldsymbol{v}_{i,*}, \boldsymbol{s}_i)) - \min(\hat{d_{i,*}}(\boldsymbol{v}_{i,*}, \boldsymbol{s}_i))} \tag{7}$$

where $\hat{d_{i,*}}(\boldsymbol{v}_{i,*}, \boldsymbol{s}_i) = \boldsymbol{u}_*[i] * \hat{d}(\boldsymbol{v}_{i,*}, \boldsymbol{s}_i | \boldsymbol{w}_*)$ (Eq. (4)), with the constraint that $\hat{d_{i,*}} \leq \delta$. So the shapelets set $\boldsymbol{v}_{i,*}$ is assigned to segment $\boldsymbol{s}_i$ with probability $\boldsymbol{p}_{i,*}$, and $\boldsymbol{v}_{i+1,*}$ is assigned to $\boldsymbol{s}_{i+1}$ with probability $\boldsymbol{p}_{i+1,*}$ respectively, where $\boldsymbol{s}_i$ is followed by $\boldsymbol{s}_{i+1}$ in a single time series. Then, for each pair (j, k), we create a weighted edge from shapelet $\boldsymbol{v}_{i,j}$ to $\boldsymbol{v}_{i+1,k}$ with weight $\boldsymbol{p}_{i,j} \cdot \boldsymbol{p}_{i+1,k}$. Finally, we merge all duplicated edges as one by summing up their weights.

4.2.2 Representation Learning

Finally, we learn hidden representations for both shapelets and time series based on shapelet evolution graph. We first employ an existing graph embedding algorithm (DeepWalk) to obtain vertex (shapelet) embeddings $\boldsymbol{\mu} \in \mathbb{R}^B$, where B is the embedding size. Note that a path in $\mathcal{G}$ intuitively reflects possible transitions between shapelets. Next, for a time series $\boldsymbol{t}$ with assigned shapelets $\{\boldsymbol{v}_{\{1,\cdots,m\},*}\}$ and corresponding assignment probabilities $\{\boldsymbol{p}_{\{1,\cdots,m\},*}\}$, we retrieve the embedding $\boldsymbol{\mu}(\boldsymbol{v}_{i,j})$ of each shapelet $\boldsymbol{v}_{i,j}$ multiplied by assignment probability $\boldsymbol{p}_{i,j}$, and sum them up for each segment. If there exists a segment $\boldsymbol{s}$ that we cannot assign any shapelets to it, the embeddings of $\boldsymbol{s}$ would be left as $\mathbf{0}$. It is reasonable since shapelet embeddings are always non-zero guaranteed by graph embedding models (embeddings are often bound to be normalized), so segments without shapelet assignments are distinguished by empty values. By far, we get segment representations, and finally concatenate all those m segment embeddings to obtain the representations $\boldsymbol{\Phi}$ for time series $\boldsymbol{t}$ as $\boldsymbol{\Phi} = \|_{i=1}^{m} \sum_j \boldsymbol{p}_{i,j} \cdot \boldsymbol{\mu}(\boldsymbol{v}_{i,j})$. The representation vector of time series can then be applied as features for various time series classification tasks, by the way of feeding the embedding features into an outer classifier. The algorithm and implementation details can be referred in [19].

4.3 Extension: Time2Graph+

Based on the time-aware shapelets (Sec. 4.1), here we propose *Time2Graph+*, an extension of *Time2Graph* by utilizing graph attentions to model shapelet evolutions.

4.3.1 Backgrounds and intuitions

In *Time2Graph* [19] (Sec. 4.2), we manually count all possible shapelet transitions, and add edges $\langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle$ with the weight as the product of shapelet assignment probabilities of node $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ upon adjacent segments. Although this intuitive way is expected to capture some transition paths or evolutionary patterns of shapelets, there are several unavoidable flaws:

Biased transition probability estimation. Recall that we regard normalized shapelet transition counts as shapelet transition probabilities: for all adjacent segments $\boldsymbol{s}_i$ and $\boldsymbol{s}_{i+1}$, we add edges between each pair of assigned shapelets $\boldsymbol{v}_{i,j}$ and $\boldsymbol{v}_{i+1,k}$, with the weight as the product of their assignment probability. But such edge weights are just estimations of transition probabilities, which may be inaccurate due to the limited observed data and the intuitive transition counting.

Magnified weak transitions. Another flaw of the way we construct the shapelet evolution graph in *Time2Graph* is, some weak transitions could be magnified if distance threshold δ is not predefined appropriately. Note that there may exist some "centered" shapelets which can be assigned to all segments under the threshold δ, then some shapelet pair, denoted by $\langle \boldsymbol{v}_c, \boldsymbol{v}_j \rangle$ which contains one of those centered shapelets $\boldsymbol{v}_c$, may repeat much more times during the enumeration. Although the product of assignment probability of $\langle \boldsymbol{v}_c, \boldsymbol{v}_j \rangle$ on one single adjacent segment pair is rather small, once we sum it over many times, it is more likely to be a strong edge in the shapelet evolution graph. The magnified weak transitions could be the significant noises in the graph, which would affect learning the shapelet hidden representations, and cause performance drops in the downstream tasks.

Inconsistent transition patterns. In *Time2Graph*, we construct a uniform shapelet evolution graph from all time series data to observe frequent shapelet transitions. But we ignore the point that 1) some infrequent transitions may indeed reflect the characteristics of some specific sequences, but contribute little to the final embeddings in *Time2Graph*, and 2) shapelet transition probabilities may be different between various time series. After all, shapelet transition patterns are likely to be inconsistent across different samples.

4.3.2 Model specification

To address the abovementioned concerns, and motivated by recent advances of GNNs and graph attention mechanism, we apply graph attention networks (GAT) to model the shapelet transitions. A straightforward motivation here is that, the main drawbacks of the way we construct the shapelet evolution graph in *Time2Graph* lies in the estimation of transition probabilities, which is reflected by the edge weights, while edge weights can be intuitively regarded as learnable attention scores between node (shapelet) pairs. So we do not need to count the estimated transitions; instead, we can use graph attentions to model the shapelet evolutionary patterns, and design an end-to-end framework to learn the attention parameters. Besides, benefited from graph attentions, we can initialize a unique shapelet evolution graph for each time series data, which helps model "personalized" transition patterns of different sequences.

Definition 4. **Parameterized Shapelet Evolution Graph.** It is a directed and unweighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{H}, \mathcal{E} | \boldsymbol{\Theta})$ parameterized by $\boldsymbol{\Theta}$. $\mathcal{V}$ consists of K vertices, each denoting

a time-aware shapelet; $\mathcal{H} = h_{\{1,\cdots,|V|\}}$, where $h_i \in R^{|F|}$ is the initial feature of node v_i with dimension as $|F|$. Each directed edge $e_{ij} \in \mathcal{E}$ indicates the transition relationships between v_i and v_j, and the weight w_{ij} of e_{ij} is computed by $f(i, j|\mathcal{H}, \mathcal{E}, \mathbf{\Theta})$, where f denotes a scalar function based on the graph attention mechanism.

Graph Initialization. We first sum up the edge weights for each time series sample rather than over the whole dataset, since we want to construct a single graph for each time series. Besides, we remove the global distance threshold δ; instead, count all possible transitions among assigned shapelets, and use another hyper-parameter p (percentile) to conduct edge pruning, i.e., removing edges whose weights are under $p\%$ among all edges. There are two main reasons that we choose percentile p rather than δ: 1) global distance threshold may not appropriate since the data scale may differ a lot across samples; and 2) percentile directly controls the graph density ($p\%$ edges are retained). Finally, we discard the edge weights after pruning edges by percentile p, and only consider their structural information as the unweighted edge set $\mathcal{E}$.

As for the initial node features, since each node represent a shapelet, the original shapelet sequence and some basic statistics are the trivial choice. But it may not work in our setting, because we represent each time series as an independent graph with the same nodes and these features are the same across different samples. A better solution is to embed intrinsic information of the original time series into the graph, thus we define the distance-based features of shapelet $\boldsymbol{v}_i$ for the time series $t = \{s_1, \cdots, s_m\}$ as

$$h_i = \text{concate}(d(v_i, s_j)), 1 \leq j \leq m \tag{8}$$

Learning shapelet transitions via graph attentions. After initializing the unweighted shapelet evolution graph for each time series, we then apply Graph Attention Networks (GAT) [40] to model the transition relationships between each pair of nodes, and translate the time series classification to a graph classification task to learn model parameters. Formally, given a shapelet evolution graph $\mathcal{G}(\mathcal{V}, \mathcal{H}, \mathcal{E}|\Theta)$, the attention score between the node pair v_i and v_j can be defined as

$$\begin{aligned} e_{ij} &= \text{LeakyReLU}(\boldsymbol{a}^T[\boldsymbol{W}h_i||\boldsymbol{W}h_j]) \\ \alpha_{ij} &= \text{softmax}(e_{ij}) = \frac{\exp(e_{ij})}{\sum_{k\in N_i}\exp(e_{ik})} \end{aligned} \tag{9}$$

$\Theta = \{\boldsymbol{W}, \boldsymbol{a}\}$ is the parameter of self-attention mechanism implemented by a single-layer FNN; e_{ij} is the attention coefficient that measures the importance of node v_j to v_i, and normalized as α_{ij} such that 1) comparable across different nodes; 2) consistent with the scale of transition probability, i.e., the sum of transition probabilities from node v_i to all nodes is equal to 1. We then conduct graph-level classifications to learn the attention parameters as in [40]:

- we use multi-head attentions to model various transition patterns, then node-level hidden features can be written as

$$\hat{h_i} = \|_{k=1}^{n}\sigma(\sum\nolimits_{j\in N_i}\alpha_{ij}^k\boldsymbol{W}^k h_j) \tag{10}$$

- the graph-level representation for a single time series is the averaged node representations:

$$\hat{\mathcal{H}} = \sum_{i=1}^{K}\frac{1}{K}h_i = \sum_{i=1}^{K}\frac{1}{K}\|_{k=1}^{n}\sigma(\sum\nolimits_{j\in N_i}\alpha_{ij}^k\boldsymbol{W}^k h_j) \tag{11}$$

- the predicted graph label can be obtained by adopting a softmax and MLP layer on graph representations, and finally we use cross-entropy loss to learn the attention parameters: (with y as the time series label)

$$\mathcal{L} = \text{CrossEntropy}(\text{softmax}(\sigma(MLP(\hat{\mathcal{H}}))), y) \tag{12}$$

Besides, the graph representations of the original time series can also be applied in various downstream tasks, by adopting a well-trained outer-classifier as in *Time2Graph*.

5 EXPERIMENTS

5.1 Experimental Setup

We use three public datasets, *Earthquakes* (EQS), *WormsTwoClass* (WTC) and *Strawberry* (STB) from the *UCR Time Series Archive* [64], along with three real-world datasets, Electricity Consumption Records (*ECR*) and Elderly Electricity Records (*EER*) from State Grid of China, and Network Traffic Flow (*NTF*) from *China Telecom.*, to validate our proposed model. Table 1 shows the overall statistics of those six datasets:

Metric \ Dataset	EQS	WTC	STB	ECR	NTF	EER
#(time series)	461	258	983	60,872	5,950	4,807
positive ratio(%)	25.3	42.2	64.3	2.3	6.4	23.3

TABLE 1: Overall statistics of 5 datasets in the experiments.

The description of three public datasets can be found on the *UCR Archive* [64], and here we briefly introduce the two real-world datasets as follows:

Electricity Consumption Records (ECR). It is provided by the State Grid Corporation of China and contains the daily electricity consumption records (K·Wh) of 60,872 users over the span of one year (2017). For every user, it records the daily total, on-peak and off-peak electricity usage. Some users (namely electricity theft) may take unauthorized actions on the electricity meter or power supply lines to cut costs, and there are a total of 1,433 (2.3%) users who have been manually confirmed as having stolen electrical power. Given users and their electricity consumption record, the task is to determine which users have stolen electrical power during the year [65].

Network Traffic Flow (NTF). This dataset is provided by China Telecom, the major mobile telecommunications service provider in China. It consists of 5,950 network traffic series, each of which describes the hourly inflow and outflow of different servers, from 6, April to 15, May in 2017. When an abnormal flow goes through server ports and some process is suddenly dead, an alarm state (label) is recorded by the operating system; there are 383 (6.4%) servers with abnormal flow series. The goal is to use the daily network traffic data to detect whether there are abnormal flows.

Elderly Electricity Records (EER). This dataset is also provided by the State Grid Corporation of China. It contains daily electricity consumption records (K·Wh) of 4,807 citizen users among the whole year of 2018, similar to the format of ECR dataset. Besides, a large-scale on-site investigations targeted on those users were conducted by staffs in State Grid, from which we collected labels of whether each user belongs to empty-nest elderly or not. Overall, the proportion of elderly users is 23.3% (1,119 cases). Detailed descriptions

Datasets / Methods	Public Dataset			Real-World Dataset								
	EQS	WTC	STB	ECR			NTF			EER		
	Accuracy			Prec	Recall	F_1	Prec	Recall	F_1	Prec	Recall	F_1
NN-ED (13.67)	68.22	62.41	95.60	18.71	10.48	13.44	37.71	46.35	41.59	29.79	35.94	32.58
NN-DTW (12.67)	70.31	68.16	95.53	15.52	18.15	16.73	33.20	43.75	37.75	30.39	33.10	31.69
NN-WDTW (13.17)	69.50	67.74	95.44	15.52	18.15	16.73	35.29	46.86	40.27	30.39	33.10	31.69
NN-CID (14.17)	69.41	69.56	95.51	18.18	13.71	15.63	32.56	43.75	37.33	30.67	32.74	31.67
DDTW (12.67)	70.79	70.92	95.60	18.78	13.71	15.85	30.48	42.71	35.58	29.09	34.16	31.42
XGBoost (origin) (10.33)	74.82	62.34	95.92	38.36	19.48	25.86	71.43	17.86	28.57	33.45	34.88	34.15
XGBoost (feature) (7.67)	75.54	64.94	**97.03**	**56.82**	16.23	25.25	80.00	21.43	33.80	39.18	34.16	36.50
BoP (9.5)	74.80	**74.42**	96.45	14.86	4.44	6.83	43.40	47.92	45.55	20.00	1.07	2.03
TSF (11.83)	74.67	68.51	96.27	26.32	2.02	3.75	57.52	33.85	42.62	43.94	10.32	16.71
EE (11.83)	73.50	71.74	95.88	10.18	33.47	15.62	42.98	27.08	33.23	31.40	32.74	32.06
SAXVSM (7.0)	73.76	72.10	96.97	21.59	42.74	28.69	30.19	50.00	37.65	29.22	38.79	33.33
LS (15.5)	74.22	73.57	92.49	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
FS (11.83)	74.66	70.58	91.66	10.45	**79.84**	18.48	63.55	35.42	45.49	37.50	2.14	4.04
LPS (12.0)	66.78	74.26	96.35	17.00	24.19	19.97	24.17	30.21	26.85	24.41	25.62	25.00
MLP (16.0)	70.29	59.86	96.58	0.00	0.00	0.00	0.00	0.00	0.00	25.42	32.38	28.48
LSTM (13.67)	74.82	42.86	63.84	13.64	31.86	19.11	7.22	16.67	10.08	26.77	82.21	40.38
VAE (12.83)	71.22	62.34	71.35	19.02	14.11	16.20	59.79	30.21	40.14	27.05	82.21	40.70
Shapelet-Seq (14.33)	75.53	55.84	78.10	14.37	66.94	23.66	18.45	**61.98**	28.44	**46.15**	8.54	14.41
Time2Graph (**2.67**)	**79.14**	72.73	96.76	30.10	40.26	34.44	71.52	56.25	62.97	28.67	71.53	40.94
Time2Graph+-static (4.83)	76.98	70.13	95.95	27.30	50.00	35.32	75.00	53.57	62.50	30.65	**73.31**	43.23
Time2Graph+ (**2.83**)	77.70	71.43	96.49	35.94	44.81	**39.88**	**97.62**	48.81	**65.08**	32.80	66.19	**43.87**

TABLE 2: Comparison of classification performance on the public and real-world datasets (%). The best performance of each dataset is bold and with an underline. Numbers following model names are the averaged rankings w.r.t accuracy or F1 scores.

of this dataset and the real-world application of empty-nest elderly recognition can be referred in Sec. 6.

We compare our proposed *Time2Graph+* model with several groups of the state-of-the-art baselines:

Distance-based Models. Previous work has stated that in most time series classification tasks, 1-NN-based methods are hard to beat [34, 66]. As for the distance metric applied in 1-NN, we use Euclidean Distance (ED), Dynamic Time Warping (DTW), Weighted DTW (WDTW) [28], Complexity-Invariant Distance (CID) [67] and Derivative DTW (DDTW) [29].

Feature-based Models. We extract some statistical features (mean, standard deviation, etc.), or just take the raw time series as input, and use the same outer classifier as which *Time2Graph+* uses (*XGBoost* (origin/feature)), to validate the effectiveness of learned representations. Besides, several feature-based algorithms have been proposed for time series classifications. In this paper, we choose some typical algorithms to compare with our model: Bag-of-Patterns (BoP) [30], Time Series Forest (TSF) [31], Elastic Ensembles (EE) [32], and Vector Space Model using SAX (SAXVSM) [68].

Shapelet-based Models. Another typical group of related baselines extracts *shapelets* to capture the representative patterns of the original time series data, including Learn Time Series Shapelets (LS) [11], Fast Shapelets (FS) [18] and Learned Pattern Similarity (LPS) [39].

Deep learning models. We consider three commonly-used deep models, MLP, LSTM and VAE, due to their efficacy in feature-representation tasks and processing time series data.

***Time2Graph+* variants.** We first compare *Time2Graph+* with Time2Graph, to check the validity of adopting graph attentions; we then compare *Time2Graph+* with its derivatives by modifying some key components to see how they fare: a) we sample the most possible shapelet sequence (each segment is assigned with highest assignment probability) for each time series, and use *LSTM* to conduct end-to-end classifications, denoted as *Shapelet-Seq*; b) we learn shapelets without considering timing factors, and embed them in the same way of *Time2Graph+*, referred as *Time2Graph+*-static.

We use *XGBoost* [69] as the outer classifier, and 5-fold nested cross-validation for fine-tuning on hyper-parameters. For baselines, parameters are also tuned if we can easily access the API from source codes; otherwise, we use the default settings. Other implementation details are listed in the corresponding project homepage: https://github.com/petecheng/Time2Graph (*Time2Graph*), and https://github.com/petecheng/Time2GraphPlus (*Time2Graph+*).

5.2 Comparison Results

Table 2 shows the comparison results for classification tasks. All three public datasets from *UCR Archive* use accuracy as evaluation metric since they are balanced, and for those three real-world datasets, which are relatively imbalanced, we show the prediction precision, recall and F1 score.

We conclude from Table 2 that *Time2Graph+* achieves competitive performances on all datasets with an averaged ranking of 2.83 (rank 1 on all real-world datasets), and discuss in detail by comparing with different groups of baselines.

- We first contrast *Time2Graph+* with distance-based models represented by *NN-DTW*. Previous works point out that distance-based models are almost "hard to beat" on time series classifications, but in our real-world scenarios, they perform a little bit poorly due to the great complexity of evolutionary patterns among those time series data.
- Secondly, performances of feature-based models are also of great sense. It results from that time series of some specific data sources are more sensitive to the statistical features, up- or down-trend, etc., which could be well captured by these feature-based models. As for the real-world datasets, some feature-based baselines perform relatively better than others, but *Time2Graph+* always beats them all on the F1 score, since it not only considers statistical features, but also

extracts intrinsic evolutionary patterns of time series data, which is directly reflected from the comparison between *Time2Graph+* and *XGBoost(features)*.

- Another important group of baselines is shapelet-based model, and Table 2 shows that *Time2Graph+* has absolute advantage over shapelet-based models on the real-world and most of the public datasets, where the extra predictive power is brought by modeling the time-level dynamics and evolution patterns of shapelets. Besides, we notice that *LS* can not even recognize any positive samples in our three real-world datasets due to the limited discriminative power of its extracted subsequences. We then look into the domain of deep learning, i.e., MLP, VAE and LSTM. They do not perform well on public datasets because there are few samples in these three UCR datasets to fit the parameters of deep models, resulting in overfitting on the training set; while for the real-world scenario, imbalanced setting and complicated patterns of time series make those shallow models difficult to work. Since it is not the point of this paper to focus on designing deep neural networks for time series modeling, we do not compare with some complex deep sequential models in our experiments.
- Finally, we compare *Time2Graph+* with its variants. As mentioned in Sec. 1, *Shapelet-Seq* suffers from the size of possible sequences, and when we only sample sequences with the highest probability, its performance fails to match many baselines, since a substantial amount of information is lost during sequence sub-sampling. The performance incrementation from *Time2Graph+*-Static to *Time2Graph+* demonstrates the predictive power brought by time-aware shapelets, and additional interpretability/insights derived from time-level attentions are shown in Sec. 5.4. From Table 2, we can see that the *Time2Graph+* model performs worse than *Time2Graph* (drop 1.2% of accuracy in average) on the public datasets, but outperforms the *Time2Graph* (+3% in terms of F1) on all real-world datasets. This may be owing to the fact that *Time2Graph+* is more complicated than *Time2Graph* as it has much more learnable parameters, then in those public datasets of limited amount of samples, it is hard to fit *Time2Graph+* well, and its performance gets impaired. When it comes to the real-world datasets, as the sample size increases (#(time series) is at least 4.8K), *Time2Graph+* significantly outperforms all baselines on F1 score. Even though some baseline methods achieve higher precision or recall, all of them seem to encounter biases on positive or negative samples.

In summary, *Time2Graph+* is better at finding representative subsequence patterns, as well as capturing evolutionary characteristics in the real-world time series data.

5.3 Parameter Analysis

We examine the sensitivities of three important hyperparameters in *Time2Graph+*: number of selected shapelets K, segment length l for time series segmentation, and percentile p for graph construction. Due to space limitations, we only present the results of EER dataset, as shown in Fig. 2:

- K should be large enough to capture a sufficient number of evolutionary patterns; while when K is too large, it will bring in less representative shapelets as noise (Fig. 2a).
- Another parameter that should be tuned is the segment length l. Fig. 2b shows that it achieves the best performances when l is 30, which is exactly number of days in a month. It seems not to be a coincidence that, in ECR dataset, the best segment length is also 30, while the optimal choice for NTF is 24, i.e., number of hours in a day. We conclude that the segment length l should be carefully selected based on the physical characteristics of original data source.
- Fig. 2c illustrates how percentile p affect model performances during graph constructions. As it controls the sparsity of shapelet evolution graphs, *Time2Graph+* performs poor when p is very small or large (the graphs are too sparse or dense). Besides, the optimal value for p varies across different datasets, i.e., 30 for EER, 80 for ECR, etc. So percentile p should be fine-tuned since the sparsity of shapelet graphs is a domain-specific property.

Note that number of shapelets K is linear to the model's time complexity, and segment length l is quadratic. The running time of *Time2Graph+* actually is longer than most of the baselines, while within a tolerable range ($\sim$2 hours for EER dataset on a GeForce RTX 2080 with 12GB memory).

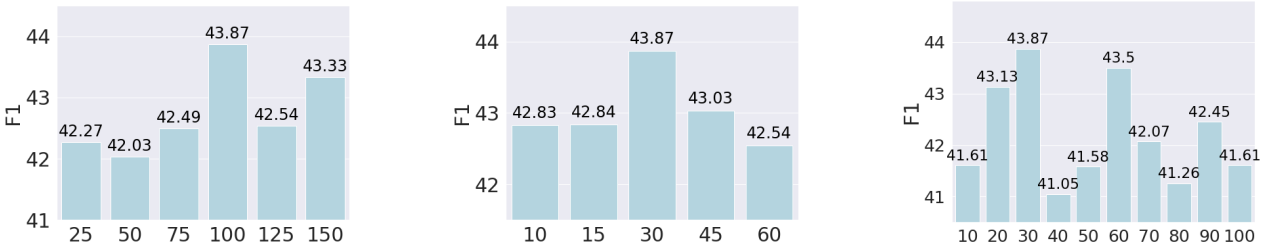
5.4 Case Study of Time-Aware Shapelets

In the following two sections, we are going to conduct several case studies and show some exciting observations to explore the interpretability of *Time2Graph+*, and we use EER dataset as the example since we have deployed our proposed framework in this real-world application, and much domain knowledge is available here from experts.

The first question is, do the shapelets we extracted indeed have varying levels of discriminatory power? As shown in Fig. 3a, the training loss grows much slower at the right end, and the KL divergence of distributions of distances between positive and negative samples towards the top (rank 1-50) shapelets on the test set is statistically significantly [3] larger than that for the bottom (rank 51-100) shapelets ($p = 7.9 * 10^{-6}$). It reflects the effectiveness of these selected shapelets to some extent. Furthermore, we show each shapelet's mean value and standard deviation (std) in Fig. 3b. Again, the std of top shapelets are statistically significantly larger than those of bottom ones ($p = 3.8 * 10^{-3}$) , while the mean values across shapelets exhibit very little difference; this suggests that typical patterns intend to be unstable.

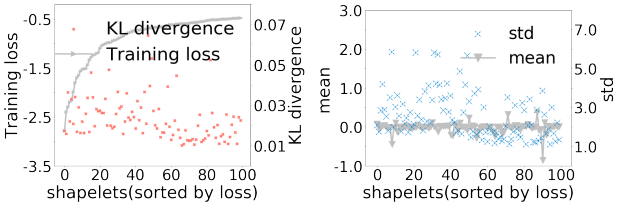
And to make further illustration, we compare the top-1 shapelet extracted by *LS* (a popular baseline) and *Time2Graph+* in Fig. 4a, b. The scale and trends of these two shapelets differ a lot, and Fig. 4b provides additional information towards time-aware shapelets in *Time2Graph+*: this particular shapelet matters in winter and summer (from month-level time attentions, i.e., global timing factor), and weights more especially at the peak and valley of time series data during the month (from day-level attentions, i.e., local timing factor). Along with the case study in Sec. 1 (Fig. 1), we can conclude that such clue obtained from the shapelet shape and time-level attentions is the distinct advantage of our proposed model on the interpretability.

[3] We conduct the two-sided t-test experiments on two groups of shapelets equally divided by their training loss (rank 1-50 and 51-100), and the null hypothesis is that these two groups are expected to have identical average of test loss (KL-divergence).



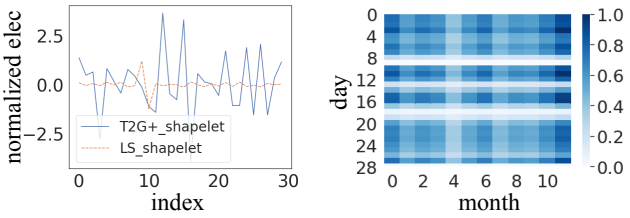
(a) number of shapelets K (b) segment length l (c) percentile p

Fig. 2: Parameter analysis. Sensitivity of hyperparameters in *Time2Graph+* on the EER dataset: (a) number of extracted shapelets K, i.e., number of nodes in the shapelet evolution graph; (b) segment length l and (c) percentile p% during the process of graph construction. When focusing on the specific parameter, we fix others as the optimal values.



(a) loss vs. divergence (b) shapelet scale and std

Fig. 3: Shapelet analysis. (a) shows the training loss and KL-divergence between positive and negative samples on test set; (b) shows the mean and standard deviation (std) of shapelets.



(a) top-1 shapelet comparison with Learning Shapelets (LS) (b) time-level attention scores of shapelet #0

Fig. 4: Shapelet analysis. (a) compares the rank-1 shapelet between *Learn Shapelets (LS)* and *Time2Graph+* (*T2G+*), and (b) visualizes the timing factors of shapelet-#0 in the *Time2Graph+*.

5.5 Case Study of the Shapelet Evolution Graphs

We finally take a deeper look into the constructed shapelet evolution graphs in *Time2Graph+*. It is important to unearth the distinctive characteristics of those graphs, helping better understand the evolutionary patterns of time series, as well as the significant benefits brought by the way we transform time series into graphs. We take a case study of an empty-nest elderly user and a normal user, with the corresponding user id as #44 and #5, as shown in Fig. 5. There are several significant differences between these two users:

- The scales of the original electricity consumption curve differ a lot; user-#44 has the peak of ∼10 kW·h for a day, while the highest record of the daily electricity usage of user-#5 reaches ∼80 kW·h.
- The constructed shapelet evolution graphs of these two users have very different graph structures, as visualized by the graph-level attentions. Although there are many similar isolated nodes on the ring of the circles, the centered nodes and their connected vertices are quite different (Sec. 1). For example, centered nodes for the elderly-#44 are #40 and #34, while #43, 94, 74 for the normal user-#5.
- Furthermore, the typical centered shapelets in the graph may encode the individual time-level dynamics, according to the shapelet shapes and its corresponding time attention matrix. For example, the pattern of shapelet #34, that a rapid rise and fall, following with a long time of flat value, has a higher weight in January, March, September and October, when the electricity curve of user #44 exactly exhibits such kind of evolution. This may contribute to better understand the habit of electrical power consuming of a single user, which could help the downstream applications such as the real-time electricity usage anomaly detections.

Overall, the proposed framework *Time2Graph+* illustrated in Fig. 5 is able to capture the representative subsequences along with its time-level dynamics, as well as the shapelet transitions and evolutions, which are reflected by the graph attentions. Those characteristics of time series are expected to be jointly modeled by graph attention networks (node features and graph structures), and the graph-level representations can be directly applied into various downstream tasks. We will introduce a real-world application deployed in State Grid of China in the following section, to further demonstrate the effectiveness of *Time2Graph+*.

6 A Real-World Application

In the real-world scenario of citizens' electricity consuming, an important issue is that it may be very dangerous for some specific groups of people to use electricity alone at home, due to their limited behavioral capacity. One such kind of citizens is the empty-nest elderly, since they are old and live alone, usually have difficulty moving conveniently, and once there are emergency cases related with electricity usage, they are probably unable to properly handle with them. A commonly applied solution is to monitor the electricity usage of those elderly users in real-time, and if there are any abnormal states among their electricity consuming, staffs would take immediate actions to make sure their safety.

However, in practice only a small part of the empty-nest elderly are recognized (labels are often provided by some government departments during on-site investigations), and staffs in State Grid almost have no idea about who are the elderly among millions of users. Besides, it is very costly and time-consuming to check on-site to obtain the labels of

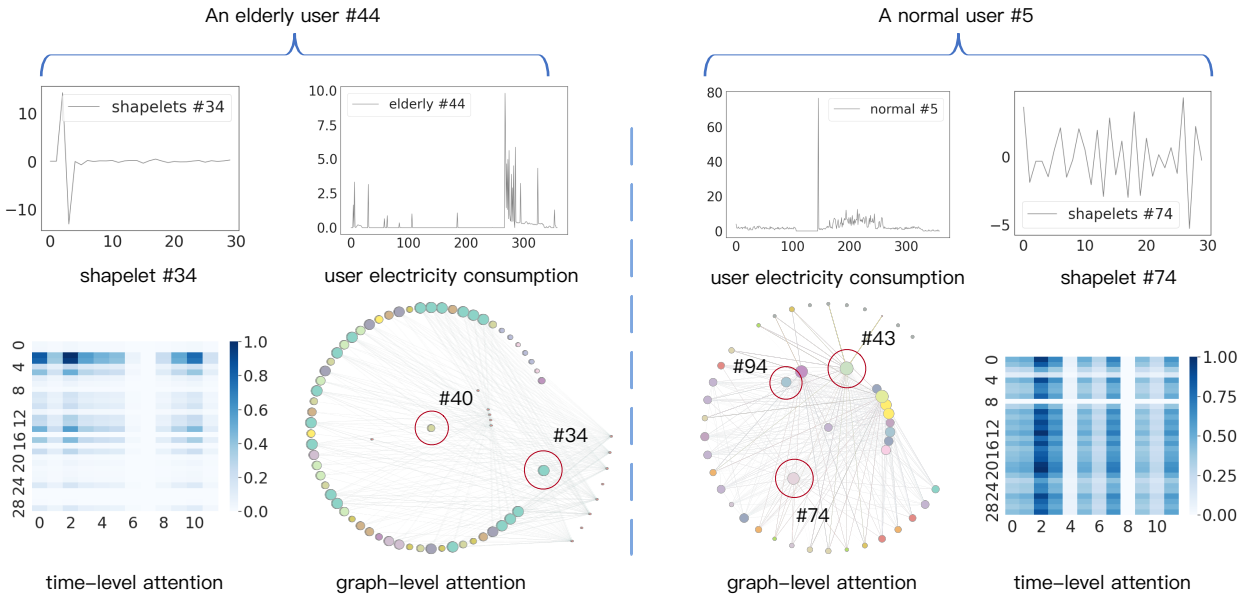


Fig. 5: Case study of shapelet evolution graphs for two different sequences. The left and right part respectively visualize an empty-nest elderly and a normal user, including the whole-year electricity consumption records, typical shapelets and its time-level attentions, and the shapelet evolution graph where the node size and color, along with the edge width are the same as in Fig. 1.

whether a user is the empty-nest elderly or not. So we deploy our proposed *Time2Graph+* model to automatically recognize the empty-nest elderly users in the State Grid Jinhua Power Supply Co. Ltd., a subsidiary of State Grid of China. Our target is to find the elderly as precise as possible given users' electricity consumption records over the past year.

To support some necessary observational studies and the training, validation and evaluation for the model, we need a large pool of labeled users. We first conduct a large-scale on-site questionnaire over 10,000 users who are randomly sampled, and collect ∼6,600+ valid results to tell whether each of them is the empty-nest elderly or not. In total, there are 4,807 users who have continuous electricity-usage records over the past year, and they are split into training and validation sets during the process of model learning (see experiments in Sec. 5). After that, we again randomly sampled ∼800 unlabeled users, among which *Time2Graph+* predicts 39 samples as the empty-nest elderly . Note that here we increase the classification threshold, i.e., predicting less positive samples (the elderly), since in this real-world scenario, precision is the more important metric. Then staffs in the State Grid check those 39 users by on-site investigation, and 25 of them are verified as the elderly, indicating that our model achieves 75.8% in terms of precision. Our framework has been deployed in the online platform in the State Grid to automatically recognize the empty-nest elderly users, and in the future, a complete electricity-usage-safety monitoring system will be built based on these predicted labels.

7 Conclusions and Discussions

In this paper, we discuss the field of bridging the community of time series modeling and graph representation learning. We focus on time series shapelets, i.e., representative subsequences, and introduce the key idea of transforming time series data into shapelet evolution graphs to model the evolutionary dynamics. Based on extracting time-aware shapelets that use time-level attentions to capture the seasonal effects on subsequences, we then utilize the shapelet evolution graphs to learn time series representations. We propose two ways of constructing and modeling the graphs, namely *Time2Graph* and *Time2Graph+* respectively. In *Time2Graph*, we construct a uniform shapelet evolution graph for the whole time series set, and use a random-walk-based graph embedding algorithm to learn shapelet representations. While for *Time2Graph+*, we convert each time series data into a unique unweighted shapelet graph, then adopt graph attention networks to learn the self-attention weights between each pair of connected nodes, and interpret the normalized attention scores as the shapelet transition probabilities. Moreover, weuse graph-level hidden representations aggregated from GAT as the embeddings for the original time series, which can be applied into various downstream tasks. Experimental and observational results on the three real-world datasets demonstrate the improvements and extra interpretability of *Time2Graph+*. Last but not least, we have deployed *Time2Graph+* in State Grid of China in Jinhua, Zhejiang Province, to recognize the empty-nest elderly users. The success of the real-world application furthermore validates the model effectiveness.

Acknowledgments. It is supported by the National Key Research and Development Project of China (2018AAA0101900) and a research funding from Tongdun Technology.

References

[1] T. Z. Tan, C. Quek, and G. S. Ng, "Brain-inspired genetic complementary learning for stock market prediction," in *2005 IEEE Congress on Evolutionary Computation*, vol. 3. IEEE, 2005, pp. 2653–2660.

[2] S. K. Jensen, T. B. Pedersen, and C. Thomsen, "Time series management systems: A survey," *IEEE TKDE*, vol. 29, no. 11, pp. 2581–2600, 2017.

[3] J. Huang, A. Badam, R. Chandra, and E. B. Nightingale, "Weardrive: Fast and energy-efficient storage for wearables," in *2015 {USENIX} Annual Technical Conference ({USENIX}{ATC} 15)*, 2015, pp. 613–625.

[4] N. Laptev, S. Amizadeh, and I. Flint, "Generic and scalable framework for automated time-series anomaly detection," in *KDD'15*, 2015, pp. 1939–1947.

[5] A. Hayashi, Y. Mizuhara, and N. Suematsu, "Embedding time series data for classification," in *MLDM'05 workshop*.

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*.

[7] V. Fortuin, M. Hüser, F. Locatello, H. Strathmann, and G. Rätsch, "Som-vae: Interpretable discrete representation learning on time series," *arXiv preprint:1806.02199*, 2018.

[8] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "Lstm-based encoder-decoder for multi-sensor anomaly detection," *arXiv preprint:1607.00148*, 2016.

[9] M. Johnson, D. K. Duvenaud, A. Wiltschko, R. P. Adams, and S. R. Datta, "Composing graphical models with neural networks for structured representations and fast inference," in *NIPS'16*, 2016, pp. 2946–2954.

[10] L. Ye and E. Keogh, "Time series shapelets: a novel technique that allows accurate, interpretable and fast classification," *DMKD.*, vol. 22, no. 1, pp. 149–182, 2011.

[11] J. Grabocka, N. Schilling, M. Wistuba, and L. Schmidt-Thieme, "Learning time-series shapelets," in *KDD'14*. ACM, 2014, pp. 392–401.

[12] J. Hills, J. Lines, E. Baranauskas, J. Mapp, and A. Bagnall, "Classification of time series by shapelet transformation," *DMKD.*, vol. 28, no. 4, pp. 851–881, 2014.

[13] A. Bostrom and A. Bagnall, "Binary shapelet transform for multiclass time series classification," in *TLSDKCS'17.*, 2017.

[14] Y. Wu and J. Dang, "China report of the development on aging cause," *Social Sciences Acadamic Press*, 2013.

[15] L. Ye and E. J. Keogh, "Time series shapelets: a new primitive for data mining," in *KDD*, 2009, pp. 947–956.

[16] Z. Xing, J. Pei, and P. S. Yu, "Early classification on time series," *KIS.*, vol. 31, no. 1, pp. 105–127, 2012.

[17] J. Lines, L. M. Davis, J. Hills, and A. Bagnall, "A shapelet transform for time series classification," in *KDD'12*, 2012.

[18] T. Rakthanmanon and E. Keogh, "Fast shapelets: A scalable algorithm for discovering time series shapelets," in *SDM*, 2013, pp. 668–676.

[19] Z. Cheng, Y. Yang, W. Wang, W. Hu, Y. Zhuang, and G. Song, "Time2graph: Revisiting time series modeling with dynamic shapelets." in *AAAI*, 2020, pp. 3617–3624.

[20] L. Lacasa, B. Luque, F. Ballesteros, J. Luque, and J. C. Nuno, "From time series to complex networks: The visibility graph," *Proceedings of the National Academy of Sciences*, vol. 105, no. 13, pp. 4972–4975, 2008.

[21] Z. Gao and N. Jin, "Complex network from time series based on phase space reconstruction," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 19, no. 3, 2009.

[22] P. Boniol and T. Palpanas, "Series2graph: Graph-based subsequence anomaly detection for time series," *Proceedings of the VLDB Endowment*, vol. 13, no. 12, pp. 1821–1834, 2020.

[23] W. Hu, Y. Yang, Z. Cheng, C. Yang, and X. Ren, "Time-series event prediction with evolutionary state graph," in *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 2021, pp. 580–588.

[24] X. Peng, J. Huang, Q. Hu, S. Zhang, and D. N. Metaxas, "Head pose estimation by instance parameterization," in *ICPR'14*, 2014, pp. 1800–1805.

[25] H. Shimodaira, K.-i. Noma, M. Nakai, and S. Sagayama, "Dynamic time-alignment kernel in support vector machine," in *NIPS'02*, 2002, pp. 921–928.

[26] S. Seto, W. Zhang, and Y. Zhou, "Multivariate time series classification using dynamic time warping template selection for human activity recognition," in *CI'15*, 2015.

[27] M. Müller, "Dynamic time warping," *IRMM'07*.

[28] Y.-S. Jeong, M. K. Jeong, and O. A. Omitaomu, "Weighted dynamic time warping for time series classification," *Pattern Recognition.*, pp. 2231–2240, 2011.

[29] T. Górecki and M. Łuczak, "Using derivatives in time series classification," *DMKD.*, pp. 310–331, 2013.

[30] J. Lin, R. Khade, and Y. Li, "Rotation-invariant similarity in time series using bag-of-patterns representation," *JIIS.*, vol. 39, no. 2, pp. 287–315, 2012.

[31] H. Deng, G. Runger, E. Tuv, and M. Vladimir, "A time series forest for classification and feature extraction," *Information Sciences*, vol. 239, pp. 142–153, 2013.

[32] J. Lines and A. Bagnall, "Time series classification with ensembles of elastic distance measures," *DMKD*, 2015.

[33] Z. Xing, J. Pei, and E. Keogh, "A brief survey on sequence classification," *ACM SIGKDD Explorations Newsletter.*, vol. 12, no. 1, pp. 40–48, 2010.

[34] A. Bagnall, J. Lines, A. Bostrom, J. Large, and E. Keogh, "The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances," *DMKD.*, vol. 31, no. 3, pp. 606–660, 2017.

[35] J. Lin, E. Keogh, L. Wei, and S. Lonardi, "Experiencing sax: a novel symbolic representation of time series," *DMKD.*, vol. 15, no. 2, pp. 107–144, 2007.

[36] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, "A symbolic representation of time series, with implications for streaming algorithms," in *DMKD'03 Workshop*, 2003.

[37] P. Schäfer, "The boss is concerned with time series classification in the presence of noise," *DMKD*.

[38] L. Hou, J. Kwok, and J. Zurada, "Efficient learning of timeseries shapelets," in *AAAI'16*, 2016.

[39] M. G. Baydogan and G. Runger, "Time series representation and similarity based on local autopatterns," *DMKD.*, 2016.

[40] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *International Conference on Learning Representations*, 2018.

[41] D. Rajan and J. J. Thiagarajan, "A generative modeling approach to limited channel ecg classification," in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2018, pp. 2571–2574.

[42] Z. Che, X. He, K. Xu, and Y. Liu, "Decade: a deep metric learning model for multivariate time series," in *KDD workshop on mining and learning from time series*. sn, 2017.

[43] C.-L. Liu, W.-H. Hsaio, and Y.-C. Tu, "Time series classification with multivariate convolutional neural network," *IEEE TIE*, vol. 66, no. 6, pp. 4788–4797, 2018.

[44] S. Lin and G. C. Runger, "Gcrnn: Group-constrained convolutional recurrent neural network," *IEEE TNNLS*, vol. 29, no. 10, pp. 4709–4718, 2017.

[45] X. Zhang, Y. Gao, J. Lin, and C.-T. Lu, "Tapnet: Multivariate time series classification with attentional prototypical network," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, pp. 6845–6852, Apr. 2020.

[46] M. González, C. Bergmeir, I. Triguero, Y. Rodríguez, and J. M. Benítez, "Self-labeling techniques for semi-supervised time series classification: an empirical study," *Knowledge and Information Systems*, vol. 55, no. 2, pp. 493–528, 2018.

[47] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: Methods and applications," *arXiv preprint arXiv:1709.05584*, 2017.

[48] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *KBS.*, 2018.

[49] 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.

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

[51] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *KDD'14*, 2014.

[52] A. Grover and J. Leskovec, "node2vec: Scalable feature

learning for networks," in *KDD'16*, 2016, pp. 855–864.
[53] L.-k. Zhou, Y. Yang, X. Ren, F. Wu, and Y. Zhuang, "Dynamic network embedding by modeling triadic closure process." in *AAAI'18*, 2018.
[54] 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.
[55] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *KDD'16*, 2016, pp. 1225–1234.
[56] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.
[57] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE SPM*.
[58] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in neural information processing systems*, 2016, pp. 3844–3852.
[59] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.
[60] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *ICML'17*, 2017.
[61] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in neural information processing systems*, 2017, pp. 1024–1034.
[62] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.
[63] M. Baydogan, R. Gokce, T. George, and Eugene, "A bag-of-features framework to classify time series," *T-PAMI.*, vol. 35, no. 11, pp. 2796–2802, 2013.
[64] H. A. Dau, E. Keogh, K. Kamgar, C.-C. M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana, Yanping, B. Hu, N. Begum, A. Bagnall, A. Mueen, and G. Batista, "The ucr time series classification archive," October 2018, https://www.cs.ucr.edu/~eamonn/time_series_data_2018/.
[65] W. Hu, Y. Yang, J. Wang, X. Huang, and Z. Cheng, "Understanding electricity-theft behavior via multi-source data," in *WWW'2020*, 2020, p. 2264–2274.
[66] X. Wang, A. Mueen, H. Ding, G. Trajcevski, P. Scheuermann, and E. Keogh, "Experimental comparison of representation methods and distance measures for time series data," *DMKD.*, vol. 26, no. 2, pp. 275–309, 2013.
[67] G. E. Batista, E. J. Keogh, O. M. Tataw, and V. M. De Souza, "Cid: an efficient complexity-invariant distance for time series," *DMKD.*, pp. 634–669, 2014.
[68] P. Senin and S. Malinchik, "Sax-vsm: Interpretable time series classification using sax and vector space model," in *ICDM'13.*, 2013, pp. 1175–1180.
[69] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *KDD'16*, 2016.

Ziqiang Cheng is currently a master student of the College of Computer Science and Technology, Zhejiang University. His main research interests include computational social science, graph modeling and time series modeling. He has published 4 research papers in major international conferences including: WWW, AAAI and WSDM.

Yang Yang received his Ph.D. degree from Tsinghua University in 2016. He is an associate professor in the College of Computer Science and Technology, Zhejiang University. His main research interests include data mining and social network analysis. He has been visiting scholar at Cornell University and Leuven University. He has published over 30 research papers in major international journals and conferences including: KDD, WWW, AAAI, TKDE and TOIS.

Shuo Jiang is a senior engineer working at State Grid Jinhua power supply company. He graduated from the College of Electrical Engineering, Zhejiang University and received a master degree. He is mainly engaged in the technology of electricity measurement and collection. He has participated in 3 standard's preparation and published 2 research papers in Chinese Core Journals.

Wenjie Hu received his master's degree from the College of Computer Science and Technology, Zhejiang University. He is currently an R&D engineer of Alibaba Cloud Intelligence Business Group. His main research interests include graph embedding and time series modeling. He has published 4 research papers in major international conferences including: WWW, AAAI and WSDM.

Zhangchi Ying is an engineer working for Information & Telecommunication Branch, State Grid Zhejiang Electric Power Corporation. He was graduated from the College of Control Science and Engineering, Zhejiang University and received a master degree. He is mainly engaged in the technology of big-data analysis, data management and electricity measurement. He has participated in 2 standard's preparation.

Ziwei Chai is currently a senior student of the College of Computer Science and Technology, Zhejiang University. His main research interests include graph anomaly detection and time series modeling.

Chunping Wang received her Ph.D. degree in Machine Learning from Duke University. She started her professional career in Opera Solutions as a Data Scientist, and currently she is a Principal Data Scientist in FinVolution Group. Her current interests include the mining of both structured and unstructured data via machine learning technology to empower the financial industry.