Anomaly Detection in Quasi-Periodic Time Series Based on Automatic Data Segmentation and Attentional LSTM-CNN

Fan Liu, Xingshe Zhou, Jinli Cao, Zhu Wang, Tianben Wang, Hua Wang, and Yanchun Zhang

Abstract—Quasi-periodic time series (QTS) exists widely in the real world, and it is important to detect the anomalies of QTS. In this paper, we propose an **a**utomatic **Q**TS **a**nomaly **d**etection **f**ramework (AQADF) consisting of a two-level clustering-based QTS segmentation algorithm (TCQSA) and a hybrid attentional LSTM-CNN model (HALCM). TCQSA first automatically splits the QTS into quasi-periods which are then classified by HALCM into normal periods or anomalies. Notably, TCQSA integrates a hierarchical clustering and the k-means technique, making itself highly universal and noise-resistant. HALCM hybridizes LSTM and CNN to simultaneously extract the overall variation trends and local features of QTS for modeling its fluctuation pattern. Furthermore, we embed a trend attention gate (TAG) into the LSTM, a feature attention mechanism (FAM) and a location attention mechanism (LAM) into the CNN to finely tune the extracted variation trends and local features according to their true importance to achieve a better representation of the fluctuation pattern of the QTS. On four public datasets, HALCM exceeds four state-of-the-art baselines and obtains at least 97.3 percent accuracy, TCQSA outperforms two cutting-edge QTS segmentation algorithms and can be applied to different types of QTSs. Additionally, the effectiveness of the attention mechanisms is quantitatively and qualitatively demonstrated.

Index Terms—Quasi-periodic time series, anomaly detection, data segmentation, classification, attentional model, LSTM, CNN

1 Introduction

Quasi-periodic time series is a kind of data stream that consists of repetitive similar fluctuation patterns (i.e., quasi-periods). QTS widely exists in real life. Particularly, many physiological signals are essentially QTSs such as the electrical activity of the heart, i.e., electrocardiogram (ECG) [1], [2], [3], the pressure that feet exert to the ground while walking [4], the arterial pressure wave [5], etc. In a QTS, some quasi-periods may show unexpected waveform changes so that they are quite different from normal quasi-periods. The waveform-changed quasi-periods are usually called anomalies or exceptions. Although anomalies do not occur frequently, the damage they lead to are usually very serious because anomalies often indicate that fatal internal changes are happening to the system. For instance, if the electrical activity of the heart become abnormal, that is, the person is suffering from arrhythmias, the heart may be unable to eject enough blood to vital organs like the brain and the heart, which would lead to serious complications such as stroke, heart failure, and cardiovascular disease (CVD) [6], [7], [8]. Specifically, the American Heart Association reported that arrhythmias cause more than 6.7 million deaths in the US and the EU annually [9]. Hence, it is important to design effective models to detect the anomalies of QTSs.

The core idea of most existing QTS anomaly detection methods [1], [2], [6], [7], [8] is to turn the detection of anomalies of QTS into a supervised classification problem. They are usually comprised of two steps: 1) segmenting the QTS into a set of quasi-periods, 2) classifying the quasi-periods into normal periods or anomalies. In the first step, clustering-based QTS partition strategies are usually adopted [1], [2], [10]. Specifically, they first extract the critical points of QTS as candidate points, and then cluster them into several clusters. Finally, the candidate points in the best cluster are chosen as split points (i.e., period points), based on which the QTS is split into quasi-periods. In the second step, usually, a set of features are extracted from the quasi-periods and then fed into supervised classifiers like random forest, AdaBoost, Naïve Bayes [2], support vector machine (SVM) [6], etc., to distinguish anomalies from normal periods.

Although existing methods achieve good performance, they are still troubled by the following two problems.

Problem 1. It is hard for existing clustering-based QTS partition methods to automatically and effectively split different types of QTSs under the same settings. They usually need to manually set the number of clusters before clustering the candidate points, however, it is obviously difficult to preset a proper value when not knowing the waveform of the QTS. Moreover, for different types of QTSs, the number of

- *Fan Liu, Xingshe Zhou, and Zhu Wang are with the Northwestern Polytechnical University, Xi'an 710129, China. E-mail: liufant800@mail.nwpu.edu.cn, {zhouxs, wangzhu}@nwpu.edu.cn.*
- *Tianben Wang is with the Northwest A&F University, Yangling 712100, China. E-mail: wangtb@nwafu.edu.cn.*
- *Jinli Cao is with the La Trobe University, Melbourne 3083, Australia. E-mail: j.cao@latrobe.edu.au.*
- *Hua Wang is with the Victoria University, Melbourne 3011, Australia. E-mail: hua.wang@vu.edu.au.*
- *Yanchun Zhang is with the Victoria University, Melbourne 3011, Australia, and also with the Guangzhou University, Guangzhou 510006, China. E-mail: yanchun.zhang@vu.edu.au.*

Manuscript received 21 Oct. 2019; revised 17 July 2020; accepted 1 Aug. 2020. Date of publication 6 Aug. 2020; date of current version 29 Apr. 2022. (Corresponding author: Fan Liu.) Recommended for acceptance by W. Zhang. Digital Object Identifier no. 10.1109/TKDE.2020.3014806

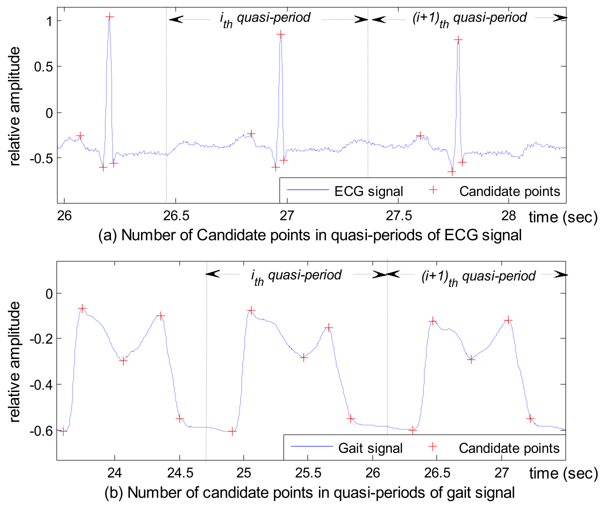

Fig. 1. Candidate points of two types of quasi-periodic time series.

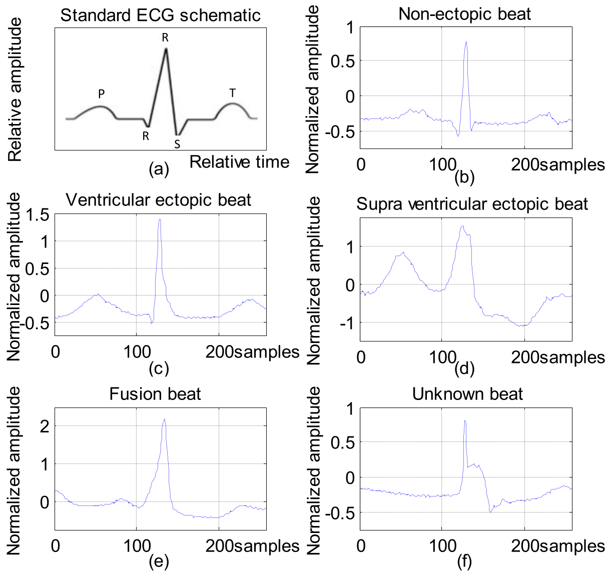

Fig. 2. Standard ECG schematic and different types of arrhythmias.

candidate points in one quasi-period is usually different. Fig. 1 indicates that an ECG quasi-period and a gait quasi-period respectively contain 4 and 5 candidate points. Hence, only when the number of clusters is preset to 4 and 5 respectively for these two types of QTSs can we get the best split points. As a result, it is difficult for existing methods to simultaneously work smoothly on different types of QTS under the same number of clusters. In addition, QTSs from the real world usually contain lots of outliers due to various reasons such as ambient noises and sensor failures. What matters is that the outliers are far from normal samples and hence prone to be clustered into abnormal clusters. Moreover, the number of outliers is unpredictable, so, even if a preset number of clusters works well on some QTSs, it is likely not to perform well on other QTSs of the same type.

Problem 2. Existing quasi-period classification models usually present either low applicability or limited accuracy. Fig. 1 shows that the waveforms of different kinds of QTSs are usually quite different from each other, hence, it is very hard for traditional feature engineering-based classifiers to simultaneously model them accurately by utilizing exactly the same features, resulting in low applicability. Recently, lots of deep learning-based models are designed where explicit feature extraction procedure is avoided [7], [8]. For instance, References [7] and [8] respectively used convolutional neural network (CNN) and long short-term memory network (LSTM) to detect abnormal heartbeats in ECG signal. These models work well, however, based on the analysis below we argue that their performance is limited. Here takes the detection of abnormal heartbeats (i.e., arrhythmia) from ECG as an example. As shown in Fig. 2a, a normal heartbeat mainly consists of three parts, i.e., P, QRS and T waves [11]. From Figs. 2b, 2c, 2d, 2e, and 2f we find that abnormal heartbeats usually show some variations on these three waves. Concretely, P and T waves of ventricular ectopic beat are significantly higher than those of non-ectopic beat, while fusion beat and unknown beat hardly have those waves at all, and the T waves of fusion beat and supra ventricular ectopic beat are even inverted. That is, different arrhythmias have quite different overall variation trends. In addition, the R waves of ventricular ectopic beat, non-ectopic beat and fusion beat are very sharp, while those of supra ventricular beat and unknown beat show drastic fluctuations, which indicates that different arrhythmias also have significant local differences. However, most existing neural network models either utilize LSTM to depict the overall variation trends of QTS [8] or employ CNN to extract the local features of QTS [7], making it hard to accurately model the fluctuation pattern of QTS. Moreover, lesions at different parts of the heart will change the waveforms of specific parts of ECG [11]. For example, the P wave of ventricular fibrillation beat is usually absent or even inverted, and the QRS waves of supra ventricular ectopic beat and unknown beat are usually chaotic [11]. That is, specific variation trends and specific features extracted from specific parts of QTS are more significant in detecting anomalies, and should be paid more attention to. But, ordinary LSTM and CNN can only indiscriminately process the variation trends and local features extracted from different parts of QTS [7], [8], which will inevitably leave out useful hidden information and result in limited performance.

In this paper, we design an **a**utomatic **Q**TS **a**nomaly **d**etection **f**ramework (AQADF) to tackle the above problems. AQADF is comprised of a two-level clustering-based QTS segmentation algorithm (TCQSA) and a hybrid attentional LSTM-CNN model (HALCM). Specifically, TCQSA aims to automatically and accurately split QTSs into successive quasi-periods. HALCM purposes to accurately model the fluctuation pattern of the quasi-periods by exploiting their overall variation trends and local features simultaneously, and further classify the quasi-periods into normal quasi-periods or anomalies. Notably, besides excellent anomaly detection accuracy, AQADF is highly universal and anti-noise and can be directly applied to different types of QTSs. The contributions of this work are four-folds:

First, we propose the TCQSA that can automatically and effectively partition different types of QTSs into high quality quasi-periods. TCQSA contains two levels of clustering. Particularly, the first clustering adopts a hierarchical clustering, aiming to make TCQSA highly universal and able to automatically cluster the candidate points of QTSs without any manual intervention. The second clustering is specifically utilized to eliminate the clusters caused by the outliers of QTSs, which makes TCQSA noise resistant.

Second, to accurately detect the anomalies of QTSs, we build the HALCM by hybridizing a stacked bidirectional LSTM (SB-LSTM) and a two-dimensional CNN (TD-CNN) together. Particularly, SB-LSTM and TD-CNN are utilized to extract the overall variation trends and local features of QTS respectively, as a result, the fluctuation pattern of QTS can be characterized more accurately. Experimental results show that HALCM achieves higher anomaly detection accuracy compared with nine similar networks.

Third, to further improve the anomaly detection performance, we design three attention mechanisms for HALCM. To be specific, we embed a trend attention gate (TAG) into the LSTM to finely tune the variation trends extracted from different parts of QTSs according to their true importance. Into the CNN, a feature attention mechanism (FAM) and a location attention mechanism (LAM) are embedded to enhance the effects of key features extracted at specific parts of QTSs. As a result, HALCM can obtain a more accurate feature representation of the fluctuation pattern of QTSs.

Fourth, experimental results on 4 public datasets show that HALCM surpasses 4 state-of-the-art QTS classification models, and achieves at least 97.3 percent accuracy, 98.5 percent sensitivity and 96.3 percent specificity. Furthermore, TCQSA exceeds 2 cutting-edge QTS segmentation algorithms and is able to obtain higher-quality quasi-periods. In addition, the architecture of HALCM is proved to be the optimum compared with 9 similar ones. Also, the effectiveness of the attention mechanisms is quantitatively and qualitatively testified.

This article is an extension of our conference paper [3], and their differences can be briefly summarized as follows. 1) AQADF is a universal computational framework aiming to detect the anomalies within different types of QTSs, but the older one is designed to identify arrhythmias based on ECG signal. 2) AQADF is able to divide QTSs into successive quasi-periods but the older one is not, hence, AQADF can detect the anomalies within QTSs automatically while the older one must rely on ready-made quasi-periods. 3) In Section 3, this article carefully formulated some basic terms and the problems to be solved. 4) AQADF is evaluated on 4 more datasets of several fields such as medical and astronomical fields, and compared with more baselines. 5) This article conducted more experiments to evaluate how well TCQSA can divide QTSs, and how accurately HALCM can detect anomalies based on the quasi-periods manually annotated or obtained by employing different QTS segmentation methods. In addition, the architecture of HALCM is optimized, and the effectiveness of the designed attention mechanisms are assessed qualitatively and quantitatively. 6) Some figures, algorithms, tables and contents are added or modified into better formats when needed to improve the readability. 7) The whole paper is deeply reorganized to make it more understandable and logically reasonable.

2 Related Work

In this section, the related work is organized into three categories. Concretely, we first elaborate the methods used to split QTSs into quasi-periods. Then, we discuss the models designed to classify the quasi-periods into normal periods or anomalies. Finally, we will introduce some state-of-the-art attentional models employed in other research field because as far as we know, our HALCM is the first attentional neural network designed for QTS anomaly detection.

2.1 QTS Segmentation Methods

Existing QTS segmentation algorithms can be grouped into two categories. The first category divides QTSs into a set of equal-length subsequences (i.e., quasi-periods) by utilizing sliding window techniques [12], [13], [14], [15]. Although this kind of methods do not need too much time and space overhead, they completely destroy the quasi-periodicity of the QTSs and hence make the obtained subsequences unaligned, especially when the length of quasi-periods varies considerably. Moreover, it is hard for them to detect minor anomalies if the length of the subsequences is much longer than that of the anomalies, which is because that in this situation normal subsequences and abnormal subsequences will not present too much difference. The second category is based on clustering technique, by which a group of split points are selected and utilized to partition the QTSs into successive quasi-periods. It avoids the shortcomings of the first type of methods. Huang *et al.* [1] clustered the vertex angles of the peak points of the QTS into a set of clusters, and then regard the points in the cluster with the smallest difference as the split points. But, this method is likely to split a quasi-period into fragments if there are two or more peaks having similar vertex angles in a quasi-period. Tang *et al.* [10] set an automatically updatable threshold and the valley points of QTSs smaller than the threshold were used as the split points. However, the threshold is vulnerable to outliers and especially irregularly changed subsequences. Ma *et al.* [2] clustered the inflexions of QTSs by leveraging the k-means++ algorithm with a four-value feature vector extracted for each inflexion, and the inflexions in the cluster with the highest average silhouette value were selected as the split points. But, it requires to preset the number of clusters manually, and is susceptible to the outliers of QTSs. Oppositely, our TCQSA is completely automated and anti-noise, and can be directly applied to various types of QTSs.

2.2 Quasi-period Classification Models

Earlier, classifying quasi-periods often adopted traditional pattern recognition paradigm. Since ECG is one of the most representative and significant QTSs in real life, here we review some ECG-oriented quasi-period classification methods [2], [16], [17]. Desai *et al.* [16] mined the non-linearity of ECG as features and fed them into different kinds of machine learning-based classifiers. Similarly, Ma *et al.* [2] directly input seven simple statistical features extracted from each quasi-period into traditional classifiers. In [17], the quasi-period was decomposed into several sub-bands and two of them were fed into a probabilistic neural network. In spite of the obtained good performance, the above methods heavily rely on hand-crafted features, leading to low universality. Recently, many LSTM-based or CNN-based quasi-period classification models were proposed with no hand-crafted features required [7], [18]. In [19], a model consisting of two streams of CNNs with different kernel sizes was designed to extract multi-scale features from the quasi-periods. Yildirim *et al.* [8] added a wavelet decomposition layer prior to a bidirectional LSTM so that the fluctuation pattern of QTS can be modeled from multiple time-frequency resolutions. In order to obtain higher performance, researchers integrated LSTM and CNN. In [20], a LSTM was appended to a CNN with three convolutional layers. In [21], CNNs and LSTMs were independently used to classify the quasi-periods, and their classification results were finally fused by a set of predefined inference rules. In this study, HALCM hybridizes a SB-LSTM and a TD-CNN with three attention mechanisms embedded in, which is proved to obtain higher classification performance.

2.3 Cases of Attentional Neural Network Models

Recently, varieties of attention mechanisms have been successfully applied to fields like activity recognition [22], [23], image analysis [24], [25], recommendation system [26], [27], etc. For instance, to accurately recognize the actions in videos, Yeung *et al.* [22] designed an attentional LSTM to tune the outputs corresponding to the current frame by utilizing the temporal relations between itself and its neighbors. To identify actions based on sequential skeleton frames, Xie *et al.* [23] built a temporal attentional recalibration model to recalibrate the extracted temporal features by focusing on more informative skeleton frames. To answer picture-related questions, ABC-CNN [25] paid more attention to the image regions showing closer relation to the intent of the questions. To recommend images for users in social networks, Wu *et al.* [26] used a hierarchical attention model to attentively weight different interest vectors of the users so as to better understand users' preferences. To generate effective cross-network recommendations, Perera *et al.* [27] embedded a time aware gate into a LSTM to model the relevancy of time intervals between user interactions. As far as we know, our HALCM is the first attentional neural network model designed for QTS anomaly detection.

3 Problem Formulation

We first define the terms frequently used, and then formulate the problems to be solved.

Definition 1. *A* time series, *denoted as* $TS = (v_1, \ldots, v_N)$, *is a time-stamped value sequence where* v_i $(1 \leq i \leq N)$ *represents the* $\mathrm{i_{th}}$ *value of* TS, *i.e., the value at time* $\mathrm{t_i}$. *Particularly,* $|TS| = N$ *denotes the number of values contained in* TS.

Definition 2. *If a smaller time-series* $SS = (v_{s1}, \ldots, v_{sM})$ *is comprised of* M *consecutive values and* SS⊆TS, *then* SS *is regarded as a* subsequence *of* TS.

Definition 3. *A* quasi-periodic time series *is actually a special type of time series which is approximately periodic. To distinguish QTS from ordinary time series, we use letter* q *instead of letter* v *to denote the elements of QTS, i.e.,* $QTS = (q_1, \ldots, q_N)$. *An ordinary time series is regarded as a QTS only if* $\exists Q = \{q_{d1}, \ldots, q_{dk} | q_{di} \in QTS,\ i \in [1, k]\}$ *that can partition* QTS *(i.e., itself) into successive subsequences which simultaneously meet the two conditions below. Notably, the strategy that utilizes the Q to split the QTS is not fixed, and researchers can decide it by themselves. For ease of discussion, here we truncate the values between* q_{d_i} *and* $q_{d_{i+1}},\ i \in [1, k-1]$, *i.e., any two adjacent elements of the Q, as a subsequence. Particularly, the value of* $d_i, i \in [1, k]$, *i.e., the subscript of* q_{d_i}, *is exactly the subscript of the element of the QTS corresponding to* q_{d_i}.

Condition 1. Any two subsequences possess almost the same length. $\forall 1 \leq i < j < k$, $m_i = |d_{i+1} - d_i|$ and $m_j = |d_{j+1} - d_j|$ denote the length of the i_{th} and j_{th} subsequences respectively, then we have $|m_i - m_j|\xi_1$, where ξ_1 is a small positive value.

Condition 2. Any two subsequences have visually similar waveforms. Here we employ $ss_i = (q_{d_i}, q_{(d_i)+1}, \ldots, q_{d_{i+1}})$ and $ss_j = (q_{d_j}, q_{(d_j)+1}, \ldots, q_{d_{j+1}})$, $\forall 1 \leq i < j < k$ to represent the i_{th} and the j_{th} subsequences, then we think they are visually similar if $dist(ss_i,\ ss_j) \leq \xi_2$, where *dist()* denotes the euclidean Distance between them and ξ_2 is a small positive value.

Definition 4. *Any element of Q, i.e.,* $q_{d_i} \in$Q *is referred to as a* period point *(or a* split point*) of the QTS, where the Q is defined in definition 3. Generally, these period points correspond to the most critical points of the QTS.*

Definition 5. *Each subsequence obtained from the QTS by utilizing its period points is called a* quasi-period *of the QTS. For example, if the QTS partition strategy mentioned in definition 3 is used, the segment between two adjacent period points is regarded as a quasi-period.*

Definition 6. *If a quasi-period has almost the same fluctuation pattern as most others', then we view it as a* normal period, *otherwise it is called as an* anomaly. *Notably, there is no clear criterion to judge whether a quasi-period is an anomaly or not, and in fact it mainly depends on specific tasks. Taking the detection of abnormal heartbeats as an example, as shown in Fig. 2, we often view the non-ectopic beats as normal periods and the others as anomalies since the former represent the normal state of the heart while the latter are some kind of morbid.*

In this paper, we use the supervised classification-based anomaly detection strategy to detect the anomalies of QTSs, which can be divided into two sub-problems which are respectively formulated as below:

Problem 1 (QTS partition). *Given a QTS,* $qts = \{q_1, \ldots, q_N\}$, *this problem is to select a set of high-quality period points,* $Q = \{q_{d1}, \ldots, q_{dk} | q_{di} \in QTS,\ i \in [1, k]\}$, *to segment* qts *into a set of quasi-periods. Particularly, the value of* k *usually corresponds to the amount of the approximately periodic subsequences contained in* qts, *and hence is not presettable.*

Problem 2 (Anomaly detection). *Given* N *quasi-periods and their class labels,* $\{(X_i,\ \mathrm{y}_i)\}_{i=1}^{N}$ *where* $X_i = \{v_{i1}, \ldots, v_{im}\}$ *and* $\mathrm{y_i}$ *denote the* $\mathrm{i_{th}}$ *quasi-period and its class label. Particularly, class labels are to denote whether a quasi-period is normal or an anomaly, or which type of anomaly the quasi-period belongs to. This problem is to construct a classifier based on* $\{(\mathrm{X}_i,\ \mathrm{y}_i)\}_{i=1}^{N}$ *which can determine the class labels of unlabeled quasi-periods as accurately as possible.*

4 Methodology

4.1 Overview of the Proposed Framework

The framework of AQADF is shown in Fig. 3. As the quasi-periods are viewed as the basic detection units, we first use

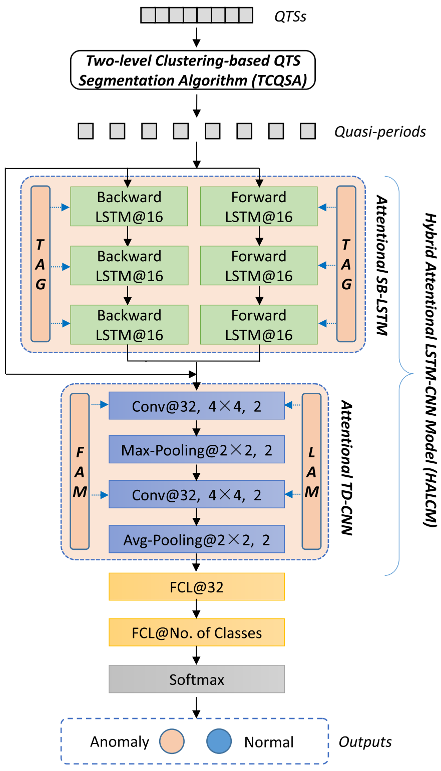

Fig. 3. The proposed automatic QTS anomaly detection framework (AQADF). In each layer, the contents behind the symbol '@' represent the number of neurons (if applicable), the size of the convolution filter or the size of the pooling layer (if applicable), and the stride of the filter or the stride of the pooling layer (if applicable), respectively.

TCQSA to split the QTSs into a set of quasi-periods. Then, the quasi-periods are further classified into normal periods or anomalies by HALCM. To be specific, a quasi-period is first fed into the attentional SB-LSTM to extract its overall variation trends. Then, the outputs along with the original quasi-period are fed into the attentional TD-CNN to further mine its local features. Finally, AQADF uses two fully connected layers (FCLs) and a softmax function to get the final classification results. At below, the details of TCQSA, HALCM and the training of HALCM will be elaborated.

4.2 Two-Level Clustering-Based QTS Segmentation Algorithm (TCQSA)

From Fig. 1 we find that the critical points of QTS inherit the quasi-periodicity of QTS and hence periodically appear at specific locations. Based on this finding, we propose the TCQSA which converts the partition of QTS into a problem of clustering the critical points of QTS. Specifically, it first extracts all the critical points of QTS as candidate points by utilizing a data compression technique. Then, it employs a two-level clustering-based method to cluster the candidate points into a set of clusters where the critical points in the same cluster are at almost the same location of each quasi-period. Afterwards, the candidate points in the best cluster are selected as the period points by which the QTS is finally split into successive quasi-periods. Particularly, TCQSA is a completely automated method and does not require any manual settings. Besides, TCQSA is anti-noise and is able to get high quality quasi-periods from noisy data. Moreover, TCQSA is highly universal and can be directly applied to various types of QTSs under the same settings.

4.2.1 Extraction of Candidate Points

Intuitively, the best candidate points should be the points containing the most critical information of the QTS. Therefore, in this paper, we utilize Douglas-Peucker (DP) algorithm [28], a widely used data compression technique, to extract the critical points as candidate points. The procedure is described as follows:

Step 1. For a QTS, use line segment $\overline{p_f p_l}$ to represent the QTS where p_f and p_l refer to the first point and the last point of the QTS, respectively.

Step 2. Between p_f and p_l, find the point p_m which has the maximum perpendicular distance to $\overline{p_f p_l}$.

Step 3. Compare the perpendicular distance between p_m and $\overline{p_f p_l}$, i.e., $dist(p_m, \overline{p_f p_l})$, with a pre-defined small positive threshold $\boldsymbol{\lambda}$. If $dist(p_m, \overline{p_f p_l}) \leq \boldsymbol{\lambda}$, then DP algorithm ends and the QTS is finally represented by $\overline{p_f p_l}$.

Step 4. If $dist(p_m, \overline{p_f p_l}) > \boldsymbol{\lambda}$, recursively process the subsequences $\{p_1, \ldots, p_m\}$ and $\{p_m, \ldots, p_n\}$ by using step 1-3, and mark p_m as a candidate point.

The optimal value of $\boldsymbol{\lambda}$ depends on the waveform of QTS, therefore, to make the optimal value of $\boldsymbol{\lambda}$ for different types of QTSs as equal as possible (to enhance the universality of TCQSA), we standardize the QTSs by utilizing the Z-score method [29] before applying the DP algorithm.

4.2.2 Selection of Period Points

As discussed in Section 1, when clustering the candidate points, existing methods usually need to preset the number of clusters, which results in low universality. In addition, the quality of the obtained period points is quite susceptible to data noise. To tackle this issue, we propose a two-level clustering-based approach. Particularly, the first-level clustering adopts a hierarchical clustering to primarily cluster the candidate points into a set of clusters, which not only does not need any manual settings but also makes TCQSA can be directly applied to different types of QTSs with no need to modify any parameters. In addition, the second-level clustering is used specifically to eliminate the clusters formed by the outliers of QTSs, in order to make TCQSA noise resistant.

Cluster the Candidate Points of QTS. To avoid presetting the number of clusters and make TCQSA highly universal, we utilize Louvain algorithm [30], [31] to cluster the candidate points. It is a famous community detection technique and has been widely applied in various fields such as social networking services [32], human-computer interaction [33], etc., due to its high modularity, rapid convergence properties, and especially no need to preset any parameters.

The input of the Louvain algorithm is a weighted graph $G = (V, E)$, where V is a set of vertexes and E is a set of edges. For two vertexes u and $v \in V$, the edge between them, i.e., $e(u, v)$, has weight $w_{u,v}$. Louvain algorithm aims to partition G into a set of disjoint communities (i.e., clusters), denoted as C, which satisfies 1) $\cup sc_i = V$, $\forall c_i \in C$ and 2) $c_i \cap sc_j = \varnothing$, where c_i represents the i_{th} community. To adapt to Louvain algorithm, the candidate points must first be converted into a weighted graph. Concretely, for the i_{th} candidate point p_i, a vector $vec_i = (vdist1_i, vdist2_i, tdist1_i, tdist2_i)$ is created where

$vdist1_i = v_i - v_{i-1}$, $vdist2_i = v_{i+1} - v_i$, $tdist1_i = t_i - t_{i-1}$, $tdist2_i = t_{i+1} - t_i$, v_i and t_i denote the value and the timestamp of p_i respectively [2]. Regarding each candidate point as a vertex, we define the weight of $e(u, v)$, i.e., $w_{u,v}$, as:

$$w_{u,v} = \frac{1}{dist(vec_u, vec_v)} \tag{1}$$

where *dist()* computes the euclidean Distance between vec_u and vec_v. Particularly, the reciprocal of the distance is used to keep the weight and the distance negatively correlated, so that the more similar the candidate points, the larger the weights between them, which is helpful to put similar candidate points into the same cluster. To get better clustering results, we employ the Z-score method [29] to standardize the obtained vectors before computing the weights.

Algorithm 1: Two-level Clustering-based QTS Segmentation Algorithm

```
Input: CPoints, i.e., the candidate points of the QTS
Output: Ppoints, i.e., the period points used to split the QTS.
1.  G = getGraph(CPoints); // turns CPoints into graph G
2.  COM = convert(G); // concert G into communities COM
3.  for each community in COM, denoted as COM_i do
4.      for each community in COM other than COM_i, denoted as COM_j do
5.          tentatively move COM_i into COM_j
6.          if the modularity of COM is not improved then
7.              revoke the above move
8.          end if
9.      end for
10. end for
11. if some communities were successively moved then
12.     go to line 3 to start other round of movement
13. end if
14. F = getFeature(COM); // compute the features of each community
15. C' = k-means(COM,F); // cluster COM using k-means based on F
16. t = getThreshold(C'); // compute the threshold t based on C'.
17. for each element of F, denoted as F_i do
18.     if F_i < t then
19.         remove COM_i from COM, where F_i is derived from COM_i
20.     end if
21. end for
22. AMC = getAMC(COM); // compute the AMC of each community
23. AMC_max = getMAX(AMC); // get the maximum AMC
24. PPoints = vertexes in the community corresponding to AMC_max;
```

The process of clustering the candidate points is shown in Algorithm 1 (*line 1-13*). First, each vertex is initialized as a community. Then, each community is tentatively and orderly moved into another community. If the move cannot improve the quality of the communities (i.e., the clustering results), it will be revoked. This process continues until the quality does not improve any more. Namely, a community is finally moved into the community corresponding to the largest quality improvement. Particularly, to evaluate the quality of the communities effectively, the modularity [30], [31], abbreviated as MOD, is adopted as the metric, which is defined as follows:

$$MOD = \sum_{c_i \in C} \left(\frac{\Sigma_{in}^{c_i}}{2m} - \left(\frac{\Sigma_{tot}^{c_i}}{2m} \right)^2 \right), \tag{2}$$

where $\Sigma_{in}^{c_i} = \Sigma w_{u,v}$ ($\forall u, v \in c_i$ and $e(u, v) \in E$) represents the sum of the weights of the edges contained in community c_i; $\Sigma_{tot}^{c_i} = \Sigma w_{u,v}$ ($\forall u \in c_i$ or $v \in c_i$, and $e(u, v) \in E$) is the sum of the weights of the edges that can connect at least one vertex of community c_i; and $m = \sum_{u,\ v \in V} w_{u,\ v} (\forall u, v \in C)$ denotes the sum of the weights of all the edges and plays a role of a normalization factor. If the weights of inter-community edges and intra-community edges were relatively small and large respectively, $\Sigma_{tot}^{c_i}$ and $\Sigma_{in}^{c_i}$ would be relatively small and large respectively, then the obtained MOD would be large.

Remove the clusters caused by outliers. Due to the complexity of real problems, QTSs is usually full of outliers. Problematically, these outliers tend to be extracted as candidate points and further be clustered as clusters that contain only few candidate points, making it hard to select the best cluster. To be specific, clusters caused by outliers usually have only very few vertexes so that it is easier for them to obtain higher quality than normal clusters when evaluated by using common metrics such as MOD and silhouette value [2], which will lead to suboptimal or even wrong period points. To solve this issue, the second-level clustering is proposed to eliminate the clusters caused by outliers, which is shown in Algorithm 1 (*line 14-21*) and described as below.

Let $C = \{c_1, \ldots c_M\}$ represent the clusters generated by the first-level clustering. The second-level clustering first categorizes the elements in C into several clusters according to their quality. Then, it set an appreciate threshold based on the clustering results. Finally, the clusters in C whose quality is smaller than the threshold is thought to be caused by outliers. Concretely, for c_i, we regard the number of its vertexes as its only feature to denote its quality since normal clusters usually contain much more vertexes than clusters caused by outliers. Thus, we obtain a feature set $F = (f_1, \ldots, f_M)$ where f_i denotes the quality of c_i. Based on F, we then use the k-means technique [29] to cluster C into several clusters. It is notable that there is no strict limitation on the number of clusters, actually, it can be set at will to a certain degree. For example, 3, 4, 5, 6 or even more clusters are acceptable since we just need to roughly rather than precisely categorize the elements of C according to their quality. Merely, as for different values, we may need to modify the implementation of the threshold as appropriate since it relies on the clustering results to some extent. But, once the threshold is set properly, it can be applied to different types of QTSs, namely, it does not destroy the universality of the TCQSA. To make the logic of the second-level clustering simpler, here we cluster C into 3 clusters denoted as $C\prime = \{c\prime_1, c\prime_2, c\prime_3\}$. Suppose the centroids of $c\prime_1$, $c\prime_2$ and $c\prime_3$ are in descending order, then $c\prime_1$, $c\prime_2$ and $c\prime_3$ can approximately represent the clusters made up of high quality candidate points, medium quality candidate points and outliers respectively. Notably, from extensive experiments on several public datasets, we find that occasionally $c\prime_2$ contains very few clusters made up of outliers. Hence, to eliminate the clusters caused by outliers as many as possible and reserve normal clusters as many as possible, the threshold is set as $t = (Vc\prime_2 + Vc\prime_3)/2$, where $Vc\prime_2$ and $Vc\prime_3$ are the centroid values of $c\prime 2$ and $c\prime 3$ respectively. Finally, in C, the clusters whose feature value is smaller than t are discarded and the rest are further used to select the period points from.

Select the best period points. To evaluate the quality of the clusters effectively, we design a new metric named average modularity of a cluster (AMC). For a cluster c_i, its AMC is defined as:

$$AMC_{c_i} = \frac{1}{|c_i|}\left(\frac{\Sigma_{in}^{c_i}}{2m} - \left(\frac{\Sigma_{tot}^{c_i}}{2m}\right)^2\right), \tag{3}$$

where $|c_i|$ is the number of candidate points in c_i; $\Sigma_{in}^{c_i}$, $\Sigma_{tot}^{c_i}$ and m remain the same as in Eq. (2). $\Sigma_{in}^{c_i}$ refers to the overall similarity degree of the candidate points in c_i, while $\Sigma_{tot}^{c_i}$ denotes the overall dissimilarity degree between the candidate points in c_i and in other clusters, thus, a high quality cluster usually presents smaller $\Sigma_{in}^{c_i}$ but larger $\Sigma_{tot}^{c_i}$ after the negative effects of $|c_i|$ is eliminated, which further results in a larger AMC. Finally, the candidate points in the cluster with the largest AMC are chosen as the final period points, as shown in Algorithm 1 (*line 22-24*).

4.2.3 Segmentation of QTS

Given a group of period points, there are two commonly used strategies to split the QTS into quasi-periods. The first is to truncate a certain length-fixed subsequence for each period point as a quasi-period where the period point is located at the center of the quasi-period. The second extracts the subsequence between two adjacent period points as a quasi-period. In this study, both of them are tried and the details are given in Section 5.1.1. In addition, the QTS originally has a set of labels to indicate the state of its disjoint subsequences, however, those labels are usually independent from the obtained quasi-periods. Therefore, we annotate the quasi-periods by ourselves. To be specific, a quasi-period is labeled by using the nearest original label from the central element of itself.

4.3 Hybrid Attentional LSTM-CNN Model (HALCM)

To model the fluctuation pattern of quasi-periods more accurately and thus to achieve higher anomaly detection performance, we design a hybrid LSTM-CNN model so as to simultaneously exploit the overall variation trends and local features of QTS. Furthermore, we design three attention mechanisms (i.e., TAG, FAM and LAM) and embed them into LSTM and CNN respectively to finely tune their outputs so that the feature representation yielded by HALCM is able to better represent the fluctuation pattern of QTS.

4.3.1 Hybrid LSTM-CNN

As a potent variant of recurrent neural network (RNN), LSTM mitigates the vanishing gradient problem of RNN, and hence is particularly skilled at learning the long-term dependencies of time series. For a given input x_t at time t, LSTM employs three gates, i.e., forget gate f_t, input gate i_t and output gate o_t, to jointly control its cell state c_t and output h_t as follows:

$$f_t = \sigma\big(W_{xf} \cdot x_t + W_{hf} \cdot h_{t-1} + b_f\big), \tag{4}$$
$$i_t = \sigma(W_{xi} \cdot x_t + W_{hi} \cdot h_{t-1} + b_i), \tag{5}$$
$$o_t = \sigma(W_{xo} \cdot x_t + W_{ho} \cdot h_{t-1} + b_o), \tag{6}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_{xc} \cdot x_t + W_{hc} \cdot h_{t-1} + b_c), \tag{7}$$
$$h_t = o_t \odot tanh(c_t), \tag{8}$$

where $\boldsymbol{W}$ and $\boldsymbol{b}$ are transition matrixes and offsets respectively Particularly, f_t discards the useless information existing in LSTM's past cell state, i_t decides what information should be added to LSTM's current cell state, and o_t decides what should be output. Due to these gates, LSTM can model the long-term variation pattern of time series more easily.

CNN is famous for utilizing hierarchical neuron layers to extract spatial features from input data, and thereby has been widely applied to many research areas, especially image analysis [34]. Concretely, the convolutional layers (CLs) can sequentially extract varieties of features from each part of the feature map while keeping their spatial relationship. In addition, the pooling layers of CNN can reduce the resolution of the feature maps without destroying the spatial invariance of the extracted features. Hence, CNN is skilled at mining the local features of the data.

As mentioned in Section 1, abnormal quasi-periods always show differences on both overall variation trend and local variation pattern in comparison with normal quasi-periods. Hence, to characterize the fluctuation pattern of QTS more effectively, we propose HALCM by hybridizing LSTM and CNN together, with LSTM and CNN respectively used to extract the overall variation trends and local features of the QTS. The architecture and settings of HALCM are shown in Fig. 3, which is determined by trial and error. The model mainly consists of a SB-LSTM and a TD-CNN. Particularly. SB-LSTM has three layers and each of them has two LSTM cells with opposite directions, so as to capture the forward and backward variation trends of the QTS simultaneously. TD-CNN has two CLs, one max-pooling layer and one average-pooling layer. The CLs aim to mine the local features from the input while the pooling layers aim to reduce the dimension of the feature maps and thus enable CNN to extract higher-level features without enlarging the size of the convolution kernels. Given a quasi-period $x = x_1, x_2, \ldots, x_N$, at each time step, SB-LSTM will yield two sets of outputs, i.e., forward results and backward results. Both of them are 1-D vectors where the i_{th} element is actually the outputs of the i_{th} neuron of the LSTM cell. In order to retain as much useful information as possible, we fuse the outputs of each time step into a matrix Y:

$$Y = \begin{bmatrix} Y_{1,f} & Y_{1,b} \\ \vdots & \vdots \\ Y_{N,f} & Y_{N,b} \end{bmatrix}, \tag{9}$$

where $Y_{i,f}$ and $Y_{i,b}$ respectively are the forward results and backward results of x_i. Particularly, the i_{th} row of Y represents the variation trends extracted from $x_1,\ldots x_i$. Namely, Y is a fine-grained step-wise representation of the variation trends of the QTS, and hence contains more details on how the QTS fluctuates than $Y_{N,f}$ and $Y_{N,b}$ combined. Hence, we feed Y into TD-CNN as input. Finally, two FCLs and a softmax layer are appended to TD-CNN to get the final classification results [3], where the number of neurons in the last FCL equals to the number of classes of the dataset.

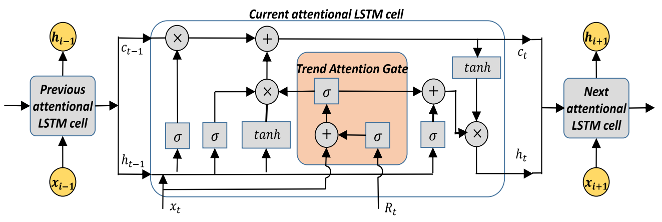

Fig. 4. Attentional LSTM cell equipped with trend attention gate (TAG).

4.3.2 Attentional LSTM

As analyzed in Section 1, variation trends of different parts of a quasi-period usually show different ability in detecting anomalies. However, Eqs. (4), (5), (6), (7), (8) show that ordinary LSTM is unable to distinguishably treat the input at different time steps since it does not know any contextual information of the input, which leads to limited anomaly detection performance. To solve this problem, an attentional LSTM is proposed by embedding a novel TAG into the ordinary LSTM. Due to TAG, the attentional LSTM is able to finely modulate the outputs according to the true significance of the relative location and surrounding waveform of the input, and hence obtain a better understanding of the overall variation trends of the quasi-period. Roughly, TAG first designs a uniform vector format that can denote the relative location and surrounding waveform of the inputs at different time steps simultaneously. Then, TAG assigns appropriate weights to the created vector to indicate its power in detecting anomalies. Last, the weighted vector is utilized to tune the outputs. Fig. 4 shows the structure of TAG, which is implemented as follows.

TAG: For a quasi-period $x = x_1, x_2, \ldots, x_N$, the middle N' points are truncated as the inputs of the attentional LSTM, denoted as x'. Then, for the input at time t, i.e., x'_t, we formulize its relative location and surrounding waveform as:

$$R_t = [0_1, \ldots, 0_{t-1}, x_t, \ldots, x_{t+L}, 0_{t+L+1}, \ldots, 0_N], \tag{10}$$

where segment $x_t, \cdots, x_{t+L}$ is the surrounding waveform of x'_t that the attentional LSTM can refer to when calculating the outputs, and $L = N - N'$ denotes the length of the surrounding waveform. Note that R_t has the same length as x, and x'_t is actually $x_{t+L/2}$, i.e., the central element of segment $x_t, \cdots, x_{t+L}$. As a result, the location of x'_t in R_t can perfectly indicate its location in x. In addition, the zeroes in R_t not only shield the influences that the other parts of the quasi-period impose on x'_t, but also conduce to building length-fixed vectors. Next, the outputs of TAG, denoted as T_t, i.e., the weights representing the importance of R_t in detecting abnormal quasi-periods are calculated as below:

$$T_t = \sigma\big(W_{xt} \cdot x'_t + \sigma(W_{rt} \cdot R_t) + b_t\big). \tag{11}$$

Finally, the attentional LSTM utilizes T_t to tune the outputs corresponding to x'_t by modifying Eqs. (6) and (7) into:

$$o_t = T_t \odot \sigma\big(W_{xo} \cdot x'_t + W_{ho} \cdot h_{t-1} + b_o\big), \tag{12}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot T_t \odot \tanh\big(W_{xc} \cdot x'_t + W_{hc} \cdot h_{t-1} + b_c\big), \tag{13}$$

where symbols c_t, f_t i_t, o_t, h_{t-1}, $\boldsymbol{W}$ and $\boldsymbol{b}$ denote exactly the same meaning as in Eqs. (4), (5), (6), (7), (8). Notably, T_t plays its role in three ways. First, as shown in Eq. (12), T_t is able to directly modulate the outputs of o_t, which is conducive to generating more reasonable outputs. Second, as shown in Eq. (13), the useful information contained in x'_t, i.e., $\tanh(W_{xc} \cdot x_t + W_{hc} \cdot h_{t-1} + b_c)$ is tuned by not only the input gate i_t, but also the weights T_t. That is, the effects of x'_t on the cell state can be adjusted by T_t. Third, the effects of T_t can be further sequentially transferred to the subsequent cell states, i.e, $c_{t+1}, c_{t+2}, \cdots$, so that the variation trend information mined from x'_t can be retained for a long time.

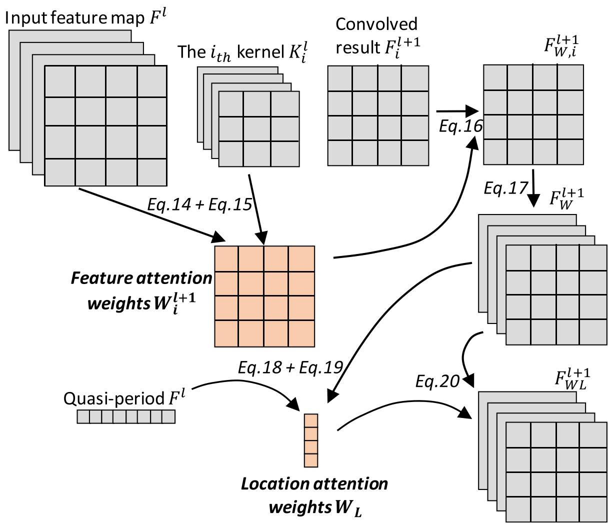
Input feature map F^l
The i_{th} kernel K_i^l
Convolved result F_i^{l+1}
$F_{W,i}^{l+1}$
Eq.16
Eq.17
F_W^{l+1}
Eq.14 + Eq.15
Feature attention weights W_i^{l+1}
Eq.18 + Eq.19
Eq.20
F_{WL}^{l+1}
Quasi-period F^l
Location attention weights W_L

Fig. 5. Attentional convolutional layer embedded with feature attention mechanism (FAM) and location attention mechanism (LAM).

4.3.3 Attentional CNN

As elaborated in Section 1, specific features mined from specific locations of the quasi-period show higher importance in detecting anomalies. However, although ordinary CNN can extract various features from different parts of the quasi-period, it does not know which features are more important, which limits its performance due to the less important features. To solve this issue, we embed a FAM and a LAM into the CL of CNN to modulate the features based on their types and locations respectively. The workflows of FAM and LAM are shown in Fig. 5, and the details are as follows.

FAM: $F^l \in \mathbb{R}^{H \times W \times C}$, $K_i^l \in \mathbb{R}^{H' \times W' \times C}$ and $F_i^{l+1} \in \mathbb{R}^{H'' \times W''}$ are respectively used to denote the feature map, the i_{th} kernel, and the i_{th} convolved result (i.e., the features extracted by K_i^l from F^l) of the l_{th} CL. Since each kernel corresponds to a certain feature type, the core idea of the FAM is to tune the convolution results based on the feature maps and the kernels. Concretely, we first reshape F^l and K_i^l into $F^l = [f_1, \cdots, f_{H \times W \times C}]$ and $K_i^l = [k_1, \cdots, k_{H' \times W' \times C}]$ by flattening their widths, heights and channels, where f_i, $k_i \in \mathbb{R}^1$. Next, for F_i^{l+1}, we calculate its feature attention weights W_i^{l+1} by utilizing a FCL and a softmax layer as follows:

$$f^i = \tanh\big(W_f \cdot F^l + W_k \cdot K_i^l + b_f\big), \tag{14}$$

$$W_i^{l+1}(j) = \frac{\exp\Big(f_j^i\Big)}{\sum_{j=1}^{j=H'' \times W''} \exp\Big(f_j^i\Big)}, \; W^{l+1} \in \mathbb{R}^{1 \times \left(H'' \times W''\right)}, \tag{15}$$

where $W_f \in \mathbb{R}^{(H\times W\times C)\times(H''\times W'')}$, $W_k \in \mathbb{R}^{(H'\times W'\times C)\times(H''\times W'')}$ are matrixes utilized to transform F^l and K_i^l to $\mathbb{R}^{1\times(H''\times W'')}$. Then, W_i^{l+1} is reshaped to $\mathbb{R}^{H''\times W''}$ and the weighted convolution results $F_{W,i}^{l+1}$ corresponding to K_i^l are computed as:

$$F_{W,i}^{l+1} = F_i^{l+1} \odot W_i^{l+1}, \qquad F_W^{l+1} \in \mathbb{R}^{H''\times W''}, \tag{16}$$

where the weights derived from the i_{th} kernel are fused into the i_{th} convolution results. Finally, we calculate the feature weighted convolution results of the l_{th} CL as:

$$F_W^{l+1} = \left[F_{W,1}^{l+1}, \ldots, F_{W,C''}^{l+1}\right], \quad F_W^{l+1} \in \mathbb{R}^{H''\times w''\times C''}, tag17 \tag{17}$$

where C'' refers to the number of the kernels of the l_{th} CL.

LAM: As stated before, the input of TD-CNN is a matrix where the i_{th} row corresponds to the i_{th} point of the quasi-period. In addition, convolution and pooling operations do not affect the spatial relationship of the extracted features, thus, the order of the rows of the feature maps can still represent the relative locations of the points of the quasi-period. Therefore, the core idea of LAM is to tune the feature values of F_W^{l+1} according to the rows they are in.

We first compute the location attention weights W_L for F_W^{l+1}. Usually, the number of rows of F_W^{l+1} is different from the number of points of the original quasi-period. Hence, to know the true locations in the original quasi-period from which the features are extracted, we incorporate the original quasi-period into the computation of W_L. Specifically, let $S \in \mathbb{R}^{1\times N}$ represent the quasi-period and transform F_W^{l+1} into $F_{W'}^{l+1} \in \mathbb{R}^{H''\times(W''\times C'')}$, then, W_L is computed as below:

$$l = \tanh\left(F_{W'}^{l+1} \cdot W_l + S \cdot W_s + b_l\right), \tag{18}$$

$$W_L(j) = \frac{\exp(l_j)}{\sum_{j=1}^{j=H''} \exp(l_j)}, \quad W_L \in \mathbb{R}^{H''\times 1}, \tag{19}$$

where $W_l \in \mathbb{R}^{(W''\times C'')\times 1}$, $W_s \in \mathbb{R}^{N\times H''}$ are used to transform $F_{W'}^{l+1}$ and S into shape of $\mathbb{R}^{H''\times 1}$, so that it can correspond to each row of F_W^{l+1}. Particularly, the incorporation of S makes the LAM able to decide which rows of F_W^{l+1} are more useful. Finally, the W_L is fused into F_{WL}^{l+1} so as to obtain the location weighted feature map F_{WL}^{l+1} as below:

$$F_{WL}^{l+1} = W_L \odot F_W^{l+1}, \qquad F_{WL}^{l+1} \in \mathbb{R}^{H''\times w''\times C''}, \tag{20}$$

with the i_{th} row of F_W^{l+1} multiplied by the i_{th} value of W_L.

4.3.4 *Training of the Model*

We formalize the class labels using one-hot encoding, and compute the loss of the model by using the categorical cross-entropy defined below:

$$\mathcal{L} = -\sum\nolimits_{i=1}^{\tilde{N}} y_i \log\left(\widehat{y_i}\right), \tag{21}$$

where $\tilde{N}$ denotes the number of instances in a mini-batch, y_i and $\widehat{y_i}$ represent the true label and predicted label of the i_{th} instance respectively. We utilize LeakyRelu function to activate neurons [7]. The trainable parameters of HALCM are initialized randomly, and updated in each iteration by using Adam optimizer [8]. To prevent overfitting, a dropout layer with a keep rate of 0.95 is appended to the second pooling layer of TD-CNN. We initialize the learning rate as 0.002 and exponentially decay it every 200 iterations using a decay factor of 0.9. Each dataset is randomly divided into training set, validation set and test set by 0.7:0.1:0.2. The epoch is set as 20.

5 Experimental Evaluation

5.1 Dataset Description and Data Preprocessing

To get robust evaluation results, the AQADF is validated on the following four public datasets.

1. MIT-BIH arrhythmia dataset (mitdb) [35] consists of 48 half-hour excerpts of two-lead ECG signals (sampled at 360 Hz, and only the first lead is used). It includes five different types of heartbeats, i.e., non-ectopic (N), ventricular ectopic (V), supraventricular ectopic (S), fusion (F) and unknown (Q).
2. Gait in Neurodegenerative disease dataset (gaitndd) [8] consists of 64 subjects' force signals under the foot while walking. It is collected by utilizing force-sensitive resistors, and comprised of 16 healthy control (HC) subjects, 15 Parkinson's disease (PD) patients, 20 Huntington's disease (HD) patients, and 13 amyotrophic lateral sclerosis (ALS) patients. In this study, the left-foot force signals which are sampled at 300Hz are used for the experiments.
3. Wrist PPG during Exercise dataset (wrist) [5] records eight subjects' motion parameters of the wrist during four types of exercises, i.e., walking (W), running (R) on a treadmill, and riding exercise bike with high resistance (RH) or low resistance (RL). The z-axis data of the gyroscope sampled at 256 Hz is used in this paper. Particularly, the subject s6 is excluded due to its poor data quality.
4. MGH/MF waveform dataset (mghdb) [1] has 250 patients' hemodynamic and electrocardiographic waveforms collected in critical care units. This study uses the arterial pressure waves (sampled at 360 Hz) of the patients whose waveform patterns are annotated as hypertension (H1) or hypotension (H2), including 11 H1 and 8 H2. Since the signals are too long and some parts of them are missing, we truncate 200000 samples (from 600000 to 800000) from each recording. Exceptionally, for the recording named 172, we truncate samples 1 to 200000 since its length is too short.

As stated in Section 4.2.3, we use two different strategies to split QTS into quasi-periods. For the 1st and 3rd datasets, we respectively truncate 256 samples and 192 samples for each period point by considering the sampling rate of the signals, the normal range of heart rate and the normal velocity of movement. As to the 2nd and the 4th datasets, the subsequence between two adjacent period points is truncated as a quasi-period. To reduce the computational overhead, each quasi-period is further down sampled into 64 samples by using a polyphase filter (for the 1st and the 3rd datasets) or cubic spline interpolation method [36] (for the 2nd and the 4th datasets). The details of the obtained quasi-periods are given in Table 1. Notably, dataset mitdb is quite imbalanced, which will reduce the generalization ability of the proposed HALCM. To tackle this problem, we synthesize

TABLE 1
Details of the Datasets Used in This Study

Datasets	*Num. of samples* [a]	*Class* [b]	*Num. of instances*
mitdb	650000	*Non-ectopic (N)*	83516
		Ventricular ectopic (V)	2492
		Supraventricular ectopic (S)	6192
		Fusion (F)	714
		Unknown (Q)	7781
gaitndd	90000	*Healthy control (HC)*	3863
		Parkinson's disease (PD)	3290
		Huntington's disease (HD)	4351
		amyotrophic lateral sclerosis (ALS)	2408
wrist	93889 (averaged)	*Walking on a treadmill (W)*	998
		Running on a treadmill (R)	878
		Riding exercise bike with high resistance (RH)	913
		Riding exercise bike with low resistance (RL)	1036
mghdb	200000 (truncated)	*Hypertension (H1)*	8155
		Hypotension (H2)	5810

[a]*The value means the number of samples in each QTS recording.* [b]*The italic classes are viewed as the positive instances*, i.e., normal periods.

new quasi-periods for the classes with fewer instances by utilizing Synthetic Minority Over-sampling Technique [37], which guarantees the synthesized quasi-periods follow the same distribution as the original ones. Finally, each class in dataset mitdb has the same number of quasi-periods.

5.2 Baselines and Evaluation Metrics

In this study, different baselines and metrics are employed for different experimental purposes.

To evaluate TCQSA, the methods in [2] and [1] are used as baselines which are also clustering-based. Ma *et al.* [2] employs the k-means++ algorithm to cluster the candidate points, which needs to preset the number of the clusters. Huang *et al.* [1] designs a clustering points on a line (CPOL) algorithm to cluster the vertex angles of the peak points (i.e., candidate points whose value are bigger than its two adjacent neighbors) of the QTSs. Both of them choose the points in the top-quality cluster as period points.

To evaluate HALCM, the methods in [2], [6], [7] and [8] are used as baselines. The first two are feature engineering-based. Specifically, Ma *et al.* [2] extracts basic statistics such as the maximum and the minimum from quasi-periods as features, while Elhai *et al.* [6] extracts linear and non-linear features like higher order spectra (HOS) as features. Then, they feed the features into traditional classifiers, i.e, Naïve Bayes [2] and support vector machine [6]. The last two are deep learning-based models into which the quasi-periods are directly fed as input. The model in [7] is a 9-layer CNN consisting of three 1-demensional CLs, three max-pooling layers and three FCLs. Yildirim *et al.* [8] designs a stacked bidirectional LSTM with a wavelet transform layer used to decompose the input into multiple time-frequency resolutions. Notably, the baselines are redone on the datasets obtained in this study with the optimal parameters adopted for the sake of fair comparison.

To assess the quality of the clusters, the mean silhouette value (MSV) is used [2] as the metric since the computation of MSV only relies on the original features of the candidate

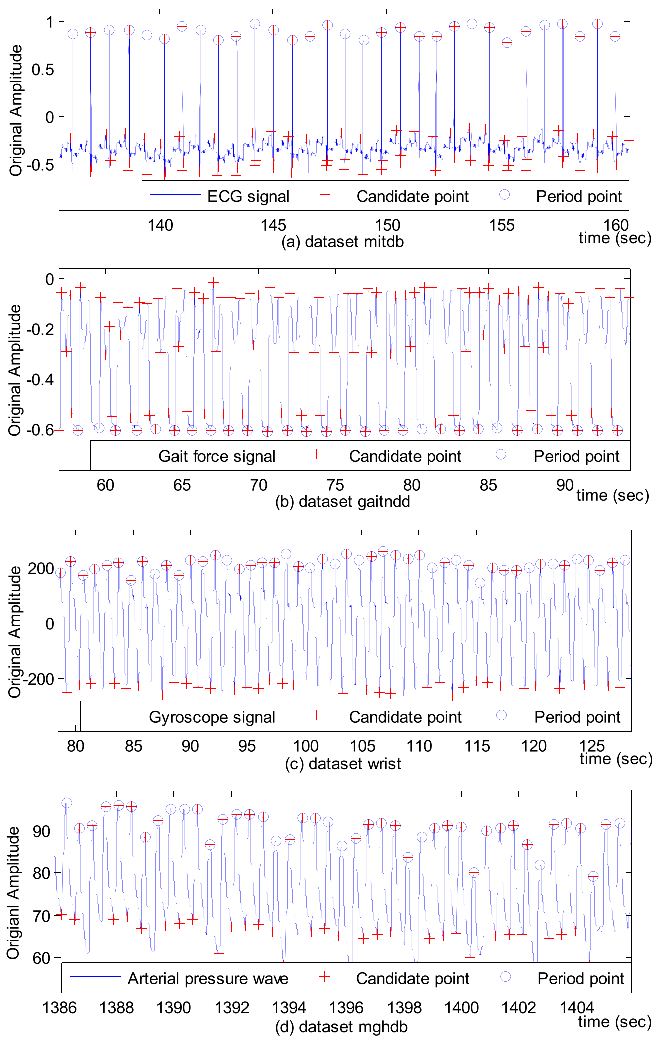

Fig. 6. Period points detected from different types of QTSs.

points, avoiding the effects of the transformation of the features. For a cluster, if the points in itself are closer to each other but are farther to the points in other clusters, then it is viewed as a good cluster and will obtain a larger MSV.

To evaluate the classification results, 3 popular metrics, i.e., accuracy (ACC), sensitivity (SEN) and specificity (SPE) are employed [7]. Particularly, larger ACC indicates better overall classification performance. Larger SEN means that the model is skilled at detecting positive instances, while larger SPE signifies better ability for recognizing negative instances.

5.3 Experimental Results

In this section, we first assess the segmentation of QTSs, and then evaluate the classification of the quasi-periods. Next, we analyze the structure of HALCM, investigate the effects of the attention mechanisms, and optimize the key parameters.

5.3.1 *Segmentation of QTSs*

Fig. 6 shows the period points detected from four different datasets, from which we find that our TCQSA can not only accurately identify the key points of QTSs as candidate points, but also periodically select the best ones from each quasi-period as period points, even if the waveforms of

TABLE 2
Performance of Different QTS Segmentation Algorithms

Dataset [a]	*Method*	*Mean Silhouette Value (MSV)*						
		C1 [b]	*C2*	*C3*	*C4*	*C5*	*C6*	*Mean* [c]
mitdb (100)	*k-means++ [2]*	0.3435	0.8718	0.8707	0.9021	-	-	0.7470
	COPL [1]	0.9021	0.8707	0.8718	0.4805	0.5606	0.3387	0.6707
	TCQSA (ours)	0.6674	0.8707	0.8862	**0.9377**	-	-	**0.8405**
gaitndd (als1)	*k-means++ [2]*	0.8416	0.7839	0.3546	0.6622	0.6511	-	0.6587
	COPL [1]	0.6494	0.7839	0.4731	0.6139	0.0773	0.5777	0.5292
	TCQSA (ours)	0.7134	**0.8891**	-	-	-		**0.8013**
wrist (s5_run)	*k-means++ [2]*	0.8908	0.8869	-	-	-	-	0.8808
	COPL [1]	0.2129	0.1053	0.7997	0.4699			0.3970
	TCQSA (ours)	**0.8963**	0.8911	-	-	-		**0.8937**
mghdb (003)	*k-means++ [2]*	0.8546	0.8821	-	-	-	-	**0.8683**
	COPL [1]	**0.8871**	0.3388	0.6970	0.6696	0.5845	-	0.6354
	TCQSA (ours)	0.8546	0.8819	-	-	-	-	0.8682

[a]*The content in the parentheses is the name of the selected recording.* [b]*The 'C1' means the first cluster obtained.* [c]*The 'Mean' denotes the mean value of the clusters' MSVs.*

these four types of QTSs are quite different from each other. Particularly, the reason why TCQSA has so strong adaptability is that TCQSA does not need to preset the number of clusters, which makes itself almost unaffected by the waveforms of QTSs. It is notable that the parameter λ in the DP algorithm is equally set as 0.3 for these four datasets in this study. The reason why the same λ can work well on QTSs having much different amplitude is that the standardization of the QTSs by using Z-score method can effectively eliminate the amplitude difference of the QTSs, which further improves the adaptability of the proposed TCQSA.

The MSVs of the clusters generated by our TCQSA and the clustering-based QTS segmentation baselines are compared in Table 2. Notably, for the method in [2], we manually set the number of clusters as the number of candidate points in most quasi-periods since in this way it can yield the largest MSVs. Table 2 shows that TCQSA produces the largest MSVs for all the datasets except for mghdb. Nevertheless, for mghdb, the largest MSV of TCQSA is very close to that of the baselines, only with a gap less than 0.01. In particular, the largest MSVs of the four datasets produced by TCQSA are larger than 0.88, which demonstrates the strong adaptability of our TCQSA. In addition, in terms of the mean MSV, TCQSA also surpasses the baselines in most cases. Specifically, for mghdb, the mean MSV of TCQSA is larger than that of COPL by 0.2328 but is just slightly less than that of k-means++. However, for other datasets, TCQSA respectively exceeds k-means++ and COPL by about 0.1 and 0.15 or more, which means that TCQSA can produce better period points more easily.

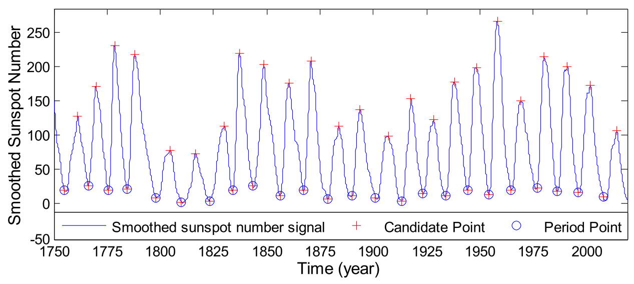

Fig. 7. Period points detected from sunspot number signal.

To further investigate the universality of TCQSA, we test it on a public dataset coming from astronomy domain, i.e., the Sunspot Number dataset provided by the Royal Observatory of Belgium [38]. It records the sunspot number between year 1749 and year 2020 which is quasi periodical from a long-term perspective. As shown in Fig. 7, although the signal fluctuates violently and the quasi-periods present significant differences from each other, TCQSA still can accurately detect its period points, which indicates that TCQSA is of high universality and can achieve good performance on different types of QTSs. Moreover, the MSVs respectively corresponding to the best cluster of TCQSA, k-means++ [2] and COPL [1] are 0.5411, 0.5, 0.4672, which shows obvious superiority of TCQSA.

5.3.2 Classification of Quasi-Periods

The performance of the quasi-period classification models is first evaluated without taking the TCQSA into account. That is, the quasi-periods are created by using the true period points given by the original datasets, aiming to avoid the effects caused by the QTS segmentation algorithms. Table 3 displays the classification results of HALCM and the baselines, which indicates that HALCM produces higher performance for all the datasets in comparison with the baselines. Specifically, all the ACCs, SENs and SPEs of HALCM exceed 97.3, 98.5 and 96.3 percent respectively. Particularly, for dataset mitdb, HALCM even achieves 99.3 percent of ACC, 99.6 percent of SEN and 98.1 percent of SPE. Among the baselines, the method in [8] performs the best, however, its performance is still lower than HALCM by up to 3.1, 3.6 and 3.1 percent in terms of ACC, SEN and SPE respectively. An interesting finding is that the feature engineering-based methods (i.e., [2] and [6]) consistently underperform the deep learning-based models (i.e., [7], [8] and HALCM) except for the SENs of [2] on datasets mitdb and gaitndd, which signifies that deep learning usually has stronger information extraction ability and modeling ability. Particularly, HALCM respectively surpasses the methods in [2]

TABLE 3
Performance of Different QTS Classification Methods Based on Original Quasi-Periods

Classification Method	*Dataset mitdb*			*Dataset gaitndd*			*Dataset wrist*			*Dataset mghdb*		
	ACC (%)	*SEN (%)*	*SPE (%)*	*ACC (%)*	*SEN (%)*	*SPE (%)*	*ACC (%)*	*SEN (%)*	*SPE (%)*	*ACC (%)*	*SEN (%)*	*SPE (%)*
Ma et al. [2]	94.5	96.7	93.9	90.7	93.6	89.9	92.3	93.5	91.9	90.4	91.2	89.3
Elhaj et al. [6]	94.7	95.0	93.6	89.3	90.6	88.9	93.1	94.0	92.8	92.1	92.6	91.3
Acharya et al. [7]	96.3	96.7	94.5	92.3	93.3	92.1	94.3	95.5	93.8	93.5	94.0	92.9
Yildirim et al. [8]	96.7	97.1	95.1	95.0	96.3	94.7	94.5	97.0	93.7	95.0	95.2	94.7
HALCM (ours)	99.3	99.6	98.1	98.0	98.5	96.3	97.3	98.5	96.8	98.1	98.8	97.0

TABLE 4
Performance of HALCM Based on the Quasi-Periods Obtained by Using QTS Segmentation Algorithms

Segmentation Strategy	*Dataset mitdb*			*Dataset gaitndd*			*Dataset wrist*			*Dataset mghdb*		
	ACC (%)	*SEN (%)*	*SPE (%)*	*ACC (%)*	*SEN (%)*	*SPE (%)*	*ACC (%)*	*SEN (%)*	*SPE (%)*	*ACC (%)*	*SEN (%)*	*SPE (%)*
True period points	*99.3*	*99.6*	*98.1*	*98.0*	*98.5*	*96.3*	*97.3*	*98.5*	*96.8*	*98.1*	*98.8*	*97.0*
k-means++ [2]	*96.7*	*97.9*	*96.4*	*94.3*	*95.0*	*94.2*	*93.9*	*94.5*	*93.7*	*95.0*	*96.2*	*93.3*
COPL [1]	*96.8*	*97.7*	*96.6*	*93.8*	*94.5*	*93.6*	*93.5*	*95.5*	*92.8*	*93.7*	*94.4*	*92.9*
TCQSA (ours)	*98.2*	*99.0*	*98.0*	*96.1*	*96.9*	*95.9*	*96.1*	*96.5*	*95.9*	*97.0*	*97.6*	*96.1*

and [7] by up to 7.7 and 8.7 percent for ACC, 7.6 and 7.9 percent for SEN, and 7.7 and 7.9 percent for SPE, which indicates that HALCM can model the fluctuation patterns of QTSs more effectively.

Afterwards, we assess the effects of the QTS segmentation algorithms on the classification results of the quasi-periods in Table 4, where the quasi-periods are created using different QTS segmentation algorithms. Overall, no matter which QTS segmentation algorithm is employed, the final classification performance definitely decrease, which is reasonable because manually marked period points possess higher accuracy than computer detected ones. However, the utilization of TCQSA produces the least performance reduction compared with k-means++ [2] and COPL [1]. Specifically, the reduction caused by TCQSA is smaller than 1.9, 2 and 0.9 percent in terms of ACC, SEN and SPE respectively for all the datasets. In addition, we observe that COPL performs worse than k-means++ in most cases, which corroborates the results of Table 3. Particularly, COPL falls behind TCQSA by up to 3.3 percent of ACC, 3.2 percent of SEN and 3.2 percent of SPE, which signifies that TCQSA can identify the most representative points from each quasi-period as the final period points more consistently.

5.3.3 Analysis of Model Architecture

First, we analyze the effectiveness of the integration of the SB-LSTM and the TD-CNN by substituting them with other similar substitutions. Specifically, for SB-LSTM, we design a stacked directional LSTM (SD-LSTM), an unstacked bidirectional LSTM (UB-LSTM) and an unstacked directional LSTM (UD-LSTM) as substitutions. For TD-CNN, we design a one-dimensional CNN (OD-CNN) as the substitution. Notably, when building these substitutions, only the number of the LSTM layers, the direction of the LSTM cells, and the dimension of the kernels in CLs are changed. Particularly, we feed only the outputs of the LSTM part corresponding to the last time step into OD-CNN, since OD-CNN can only process one-dimensional inputs. Finally, we obtain ten different combinations of LSTM and CNN in total, and their performance on datasets mitdb and gaitndd is shown in Table 5. When the CNN part is fixed, SB-LSTM and UD-LSTM yields the best and the worst performance respectively, which is because UD-LSTM can only extract simple temporal dependencies of QTSs from just one direction, while SB-LSTM can do it from both directions. Likewise, when we fix the LSTM part, TD-CNN is always superior to OD-CNN, which can be explained that the input of TD-CNN is comprised of the outputs of SB-LSTM at all time steps, hence, it retains more useful information for the classification. Consequently, the best performance comes from the combination of SB-LSTM and TD-CNN, which outperforms the combination of UD-LSTM and OD-CNN by 8.3, 8.7 and 7 percent for mitdb, and 5.2, 6.4 and 3.6 percent for gaitndd, in terms of ACC, SEN and SPE respectively. It is notable that if only SB-LSTM or TD-CNN is employed, the obtained performance is about 90 percent of ACC, which is lower than any combination. That is, mining only the overall variation trends or the local features of QTSs is hard to accurately model its fluctuation pattern.

Table 6 shows the accuracy values of HALCM with different number of LSTM layers and CLs on four datasets. Fixing the number of CLs as two, we find that increasing the number of LSTM layers to three can markedly improve the ACCs up to 98.2 percent (averaged). Particularly, for each additional LSTM layer, the performance gain is about 2-3 percent. However, if the number of LSTM layers continues increasing, the performance will increase only a little, or even decrease sometimes. Similarly, if we fix the number of LSTM layers as three, increasing the number of CLs from one to two can bring an

TABLE 5
Performance of HALCM With Different Combinations of LSTM and CNN Substitutions

Combinations of LSTM and CNN		***mitdb***			***gaitndd***		
		ACC (%)	*SEN (%)*	*SPE (%)*	*ACC (%)*	*SEN (%)*	*SPE (%)*
UD-LSTM	OD-CNN	91.0	90.9	91.1	92.8	93.1	92.7
	TD-CNN	94.6	94.1	96.6	93.1	92.5	93.2
UB-LSTM	OD-CNN	93.7	94.0	92.7	93.4	94.9	93.1
	TD-CNN	94.2	97.0	93.5	94.8	93.9	95.0
SU-LSTM	OD-CNN	95.3	95.4	94.8	94.7	95.1	94.6
	TD-CNN	98.0	98.2	97.3	96.3	97.1	96.1
SB-LSTM	OD-CNN	96.4	96.6	95.4	95.1	97.1	94.6
	TD-CNN	**99.3**	**99.6**	**98.1**	***98.0***	***98.5***	***96.3***
Only SB-LSTM		89.1	88.7	90.5	92.5	91.4	92.8
Only TD-CNN		89.4	90.0	87.1	92.1	93.0	91.9

TABLE 6
Performance of HALCM With Different Number of LSTM Layers and Convolutional Layers

No. of layers[a]	*ACC obtained based on different datasets (%)*				*Mean*
	mitdb	*gaitndd*	*wrist*	*mghdb*	*ACC (%)*
1L[a]+2C[b]	94.2	94.8	92.8	91.5	93.3
2L+2C	98.4	96.5	95.2	94.7	96.2
3L+2C	99.3	98.0	97.3	98.1	98.2
5L+2C	99.2	98.1	97.0	98.5	98.2
8L+2C	99.3	98.1	97.5	98.0	98.2
3L+1C	93.5	94.1	93.2	92.4	93.3
3L+2C	99.3	98.0	97.3	98.1	98.2
3L+3C	99.4	98.3	97.2	98.2	98.3
3L+5C	99.2	98.6	97.5	97.8	98.3
3L+8C	98.9	98.5	96.8	98.0	98.1

[a]*L and C represent LSTM layer and convolutional layer, respectively.*

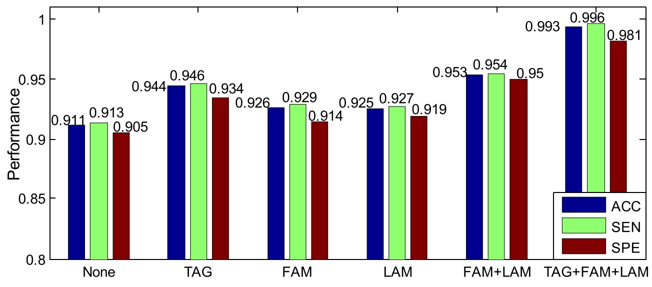

Fig. 8. Performance of models with different attention mechanisms.

improvement of 3.9-5.7 percent of ACC, while continuously increasing it from two to five, and further to eight will bring no increment or even 0.2 percent of decrement in terms of mean ACC. Hence, HALCM is finally set as three LSTM layers plus two CLs to yield as good performance as possible while keep the computational time relatively low.

5.3.4 Effects of Attention Mechanisms

To evaluate the proposed attention mechanisms in detail, we conduct ablation experiments by adding TAG, FAM and LAM to HALCM one by one, and the obtained classification results on mitdb are shown in Fig. 8. It can be find that whichever attention mechanism is utilized, the performance can be improved. Concretely, the increment produced by FAM (1.5 percent of ACC, 1.6 percent of SEN and 0.9 percent of SPE) or LAM (1.4 percent of ACC, 1.4 percent of SEN and 1.4 percent of SPE) is smaller than that produced by TAG (3.3 percent of ACC, 3.3 percent of SEN and 3.1 percent of SPE). Particularly, if FAM and LAM are included at the same time, the increment of ACC, SEN and SPE will increase to 4.2, 4.1 and 4.5 percent respectively, which is larger than that yielded by FAM and LAM combined. In other words, if we know the types and the locations of the extracted features at the same time, the local pattern of QTSs can be modeled more accurately. Moreover, if we utilize TAG, FAM and LAM simultaneously, ACC, SEN and SPE will reach 99.3, 99.6 and 98.1 percent, which exceeds the case without using any attention mechanism by 8.2, 8.3 and 7.9 percent respectively.

We also qualitatively analyze the effects of the attention mechanisms by visualizing the intermediate outputs of the model. Concretely, we view the LSTM layers and CLs as a set of successive feature extractors. Accordingly, their outputs are regarded as the extracted feature vectors, and each of them consists of numerous feature values. Then, we use Principal Component Analysis [39] technique to reduce the dimension of the feature vectors. Particularly, for each feature vector, the top two principle components (PCs) are selected for the visualization. Finally, from dataset mitdb, we randomly select 25 instances for each class, and visualize their PCs orderly obtained from each feature extractor with or without using the attention mechanisms in Fig. 9. From Figs. 9a, 9b, 9c, 9d, 9e, 9f, 9g, 9h, 9i, and 9j, we observe that as the number of feature extractors used increases, the instances of different classes gradually become distinguishable. But, we find that utilizing the attention mechanisms can separate the instances much more clearly than not using them. For example, in Fig. 9f, most of the instances are mingled together, while in the counterpart, i.e., Fig. 9a, these instances do not stick together that tightly. As the effects of the attention mechanisms accumulate (Figs. 9a, 9b, 9c, 9d, 9e), most instances are finally separated completely. Particularly, in Fig. 9e, only few instances of class N, V and S are mingled. While in Fig. 9j, much more instances of class N, V, S, F and Q are still mixed. It is because that the designed TAG, FAM and LAM can fine tune the extracted features based on the categories they belong to. As a result, the intra- and inter-class difference is reduced or enhanced respectively, which makes the instances more distinguishable.

5.3.5 Optimization of Key Parameters

We optimize the key parameters of HALCM. First, the L in Eq. (10), i.e., the length of the surrounding waveform that the LSTM cell can refer to when calculating the output, is a key parameter for extracting the overall variation trends of the QTSs. Fig. 10a illustrates the classification performance on mitdb as L increases from 2 to 16, from which we find that SPE varies more drastically than ACC and SEN. However, all of them increase first, and then reach their optima when L is set to 8. After that, they all decrease gradually. It

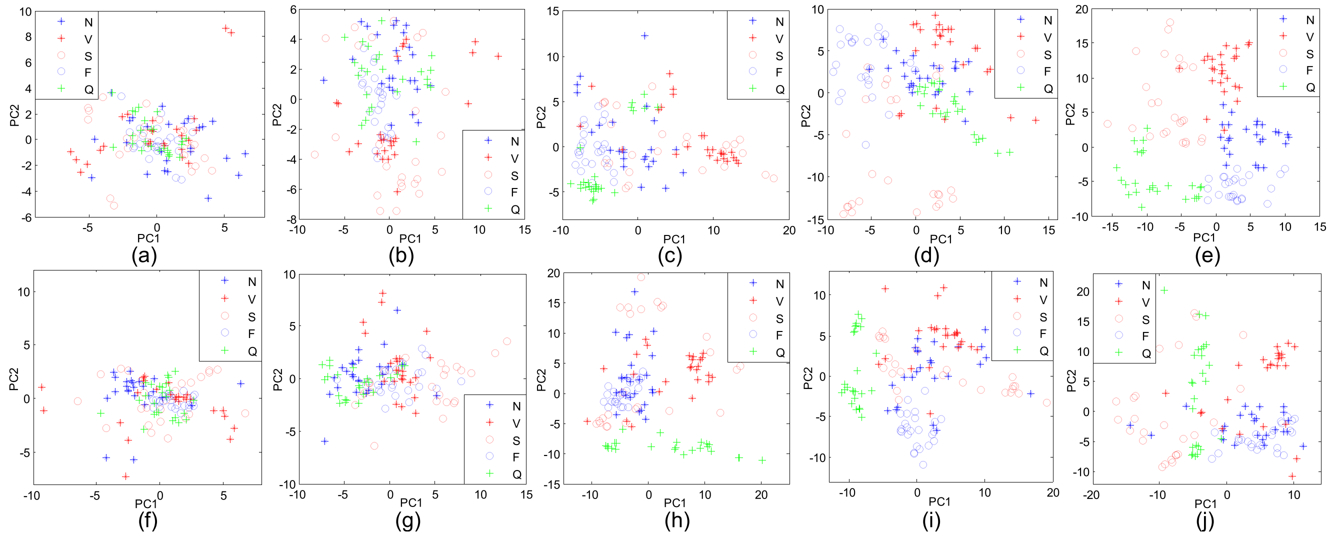

Fig. 9. Visualization of the principle components (PCs) extracted from the outputs of different layers with or without using attention mechanisms. (a)-(e): using the attention mechanisms, (f)-(j): not using the attention mechanisms.

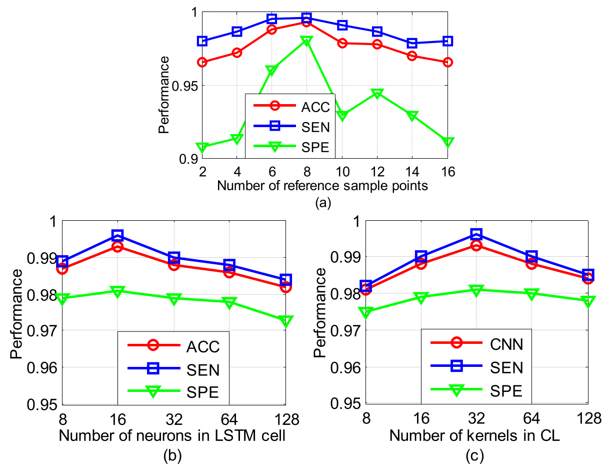

Fig. 10. Optimization of the key parameters of AQADF.

can be explained that too short surrounding waveform cannot provide adequate context information for the corresponding input, while too long surrounding waveform may ruin the information already mined from previous inputs.

Additionally, we analyze the number of neurons in each LSTM cell and the number of kernels in each CL, since they directly affect the amount of hidden information extracted and the time overhead of the model. Figs. 10b and 10c show the effects of these two parameters on the classification results on dataset mitdb. It can be seen that the three metrics keep relatively stable as the parameters vary. In particular, when we set the two parameters as 16 and 32 respectively, all the metrics reach their maximums. Nonetheless, in real applications, the number of neurons and kernels can be reduced appropriately, which can reduce the computational burden significantly but keep the performance still high.

6 Conclusion and Future Work

To detect the anomalies of QTSs, we design the AQADF by combining TCQSA and HALCM. TCQSA employs hierarchical clustering so that it can be applied to different types of QTSs. Besides, the removal of the outliers makes TCQSA more robust. The combination of LSTM and CNN enables HALCM to model the variation pattern of QTSs more comprehensively. Moreover, the usage of the 3 attention mechanisms helps to obtain a better feature representation of the variation pattern of QTSs. Experiments on public datasets show that TCQSA exceeds 2 cutting-edge baselines in MSV. HALCM obtains at least 97.3 percent of ACC on 4 datasets, which precedes the best of 4 baselines by 3.1 percent. Future work aims to design attention mechanisms having simpler structures but higher performance.

Acknowledgments

This work was partially supported by the National Key R&D Program of China (2016YFB1001401), the National Natural Science Foundation of China (61960206008, 61672161), the Innovative Talents Promotion Program of Shaanxi Province (2018KJXX-011), the Northwest A&F University Doctoral Research Project (Z1090219096) and the China Scholarship Council (201706290110).

References

[1] G. Huang *et al.*, “Online mining abnormal period patterns from multiple medical sensor data streams,” *World Wide Web*, vol. 17, no. 4, pp. 569–587, Jul. 2014.
[2] J. Ma *et al.*, “Supervised anomaly detection in uncertain pseudoperiodic data streams,” *ACM Trans. Internet Technol.*, vol. 16, no. 1, Feb. 2016, Art. no. 4.
[3] F. Liu *et al.*, “An attention-based hybrid LSTM-CNN model for arrhythmias classification,” *Proc. Int. Joint Conf. Neural Netw.*, 2019, pp. 1–8.
[4] T. D. Pham, “Texture classification and visualization of time series of gait dynamics in patients with neuro-degenerative diseases,” *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 26, no. 1, pp. 188–196, Jan. 2018.
[5] E. Brophy *et al.*, “A machine vision approach to human activity recognition using photoplethysmograph sensor data,” *Proc. 29th Irish Signals Syst. Conf.*, 2018, pp. 1–6.
[6] F. A. Elhaj *et al.*, “Arrhythmia recognition and classification using combined linear and nonlinear features of ECG signals,” *Comput. Methods Prog. Biomed.*, vol. 127, pp. 52–63, Apr. 2016.
[7] U. R. Acharya *et al.*, “A deep convolutional neural network model to classify heartbeats,” *Comput. Biol. Med.*, vol. 89, pp. 389–396, Oct. 2017.
[8] Ö. Yildirim, “A novel wavelet sequence based on deep bidirectional LSTM network model for ECG signal classification,” *Comput. Biol. Med.*, vol. 96, pp. 189–202, May 2018.
[9] V. Fuster *et al.*, “2011 ACCF/AHA/HRS focused updates incorporated into the ACC/AHA/ESC 2006 guidelines for the management of patients with atrial fibrillation,” *EP Europace*, vol. 57, no. 11, pp. 651–745, Mar. 2011.
[10] L.-A. Tang *et al.*, “Effective variation management for pseudo periodical streams,” in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2007, pp. 257–268.
[11] 2020. [Online]. Available: https://www.aci.health.nsw.gov.au/__data/assets/pdf_file/0007/380284/Arrhythmia_Management.pdf
[12] M. C. Chuah and F. Fu, “ECG anomaly detection via time series analysis,” in *Proc. Int. Symp. Parralel Distrib. Appl.*, 2007, pp. 123–135.
[13] C. Chakraborty, T. Kamiyama, and H. Takahashi, “An efficient anomaly detection in quasi-periodic time series data—A case study with ECG,” in *Proc. Int. Work-Conf. Time Series Anal.*, 2017, pp. 147–157.
[14] Y. Gu, A. McCallum, and D. Towsley, “Detecting anomalies in network traffic using maximum entropy estimation,” in *Proc. 5th ACM SIGCOMM Conf. Internet Meas.*, 2005, pp. 32–32.
[15] E. Keogh, J. Lin, and A. Fu, “HOT SAX: Efficiently finding the most unusual time series subsequence,” in *Proc. 5th IEEE Int. Conf. Data Mining*, 2005, Art. no. 8.
[16] U. Desai *et al.*, “Diagnosis of multiclass tachycardia beats using recurrence quantification analysis and ensemble classifiers,” *J. Mech. Med. Biol.*, vol. 16, no. 1, 2016, Art. no. 1640005.
[17] R. J. Martis, U. R. Acharya, and L. C. Min, “ECG beat classification using PCA, LDA, ICA and discrete wavelet transform,” *Biomed. Signal Process. Control*, vol. 8, no. 5, pp. 437–448, Sep. 2013.
[18] B. Pourbabaee, M. J. Roshtkhari, and K. Khorasani, “Deep convolutional neural networks and learning ecg features for screening paroxysmal atrial fibrillation patients,” *IEEE Trans. Syst., Man, Cybern.: Syst.*, vol. 48, no. 12, pp. 2095–2104, Dec. 2018.
[19] X. Fan *et al.*, “Multiscaled fusion of deep convolutional neural networks for screening atrial fibrillation from single lead short ECG recordings,” *IEEE J. Biomed. Health Inform.*, vol. 22, no. 6, pp. 1744–1753, Nov. 2018.
[20] S. L. Oh *et al.*, “Automated diagnosis of arrhythmia using combination of CNN and LSTM techniques with variable length heart beats,” *Comput. Biol. Med.*, vol. 102, pp. 278–287, Nov. 2018.
[21] F. Zhou, L. Jin, and J. Dong, “Premature ventricular contraction detection combining deep neural networks and rules inference,” *Artif. Intell. Med.*, vol. 79, pp. 42–51, Jun. 2017.
[22] S. Yeung *et al.*, “Every moment counts: Dense detailed labeling of actions in complex videos,” *Int. J. Comput. Vis.*, vol. 126, no. 2-4, pp. 375–389, Apr. 2018.
[23] C. Xie *et al.*, “Memory attention networks for skeleton-based action recognition,” in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 1639–1645.
[24] L. Chen *et al.*, “SCA-CNN: Spatial and channel-wise attention in convolutional networks for image captioning,” in *Proc. 2017 IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 6298–6306.
[25] K. Chen *et al.*, “ABC-CNN: An attention based convolutional neural network for visual question answering,” 2015, *arXiv:1511.05960*.

[26] L. Wu *et al.*, "A hierarchical attention model for social contextual image recommendation," *IEEE Trans. Knowl. Data Eng.*, to be published, doi: 10.1109/TKDE.2019.2913394.
[27] D. Perera and R. Zimmermann, "LSTM Networks for online cross-network recommendations," *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 3825–3833.
[28] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *Can. Cartographer*, vol. 10, no. 2, pp. 112–122, Dec. 1973.
[29] I. B. Mohamad and D. Usman. "Standardization and its effects on K-means clustering algorithm," *Res. J. Appl. Sci. Eng. Technol.*, vol. 6, no. 17, pp. 3299–3303, Sep. 2013.
[30] X. Que *et al.*, "Scalable community detection with the louvain algorithm," *Proc. IEEE Int. Parrallel Distrib. Process. Symp.*, 2015, pp. 28–37.
[31] L. Waltman and N. J. V. Eck, "A smart local moving algorithm for large-scale modularity-based community detection," *Eur. Phys. J. B*, vol. 86, Nov. 2013, Art. no. 471.
[32] J. Ruan and W. Zhang, "An efficient spectral algorithm for network community discovery and its applications to biological and social networks," *Proc. 7th IEEE Int. Conf. Data Mining*, 2007, pp. 643–648.
[33] D. Meunier *et al.*, "Hierarchical modularity in human brain functional networks," *Front. Neuroinform.*, vol. 3, Oct. 2009, Art. no. 37.
[34] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[35] A. L. Goldberger *et al.*, "PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals," *Circulation*, vol. 101, no. 23, pp. e215–e220, Jun. 2000.
[36] S. McKinley and M. Levine, "Cubic spline interpolation," *College Redwoods*, vol. 45, no. 1, pp. 1049–1060. 1998.
[37] N. V. Chawla *et al.*, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jun. 2002.
[38] 2020. [Online]. Available: http://www.sidc.be/
[39] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics Intell. Lab. Syst.*, vol. 2, no. 1-3, pp. 37–52, 1987.

Fan Liu received the BS degree in software engineering from Northwestern Polytechnical University (NPU), China, in 2014. Currently, he is working toward the PhD degree in computer science, in NPU. His research interests include machine learning, data mining, pattern recognition, deep learning as well as their applications in health informatics, and health assessment and healthcare.

Xingshe Zhou is a full professor with NPU, China. He is an executive director of the China Computer Federation and a program committee member of the National Natural Science Foundation of China. He has published more than 200 papers in fields including embedded computing, distributed computing, pervasive computing, and health informatics.

Jinli Cao received the PhD degree from the University of Southern Queensland, Australia. She is a senior lecturer with the Department of Computer Science and Information Technology, La Trobe University, Australia. She has published more than 100 research papers in journals such as the *IEEE Transactions on Parallel & Distributed Systems*, *IEEE Transactions on Knowledge and Data Engineering*, and conferences such as WWW, WISE, DASFAA, etc.

Zhu Wang received the BS, MS, and PhD degrees from NPU, in 2006, 2009 and 2013, respectively. He is an assistant professor with the School of Computer Science, NPU, China. From Nov. 2010 to Apr. 2012, he worked as a visiting scholar at Institute TELECOM SudParis in France. His research interests include social network analysis, pervasive computing, and health informatics.

Tianben Wang received the BS degree from the School of computer science of Northwest A&F University, China, in 2011, and the MS degree and PhD degrees from NPU, in 2013 and 2018, respectively. He is an assistant professor with the School of computer science of Northwest A&F University, China. His current research interests include contactless behavior sensing, ubiquitous computing, health informatics, and intelligent assistive technology. He is now an assistant professor with the College of Mechanical and Electronic Engineering, NWAFU, Yangling, Xi'an, China.

Hua Wang received the PhD degree from the University of Southern Queensland, Australia. He is a full professor at Victoria University, Australia. As a chief investigator, he has been awarded three Australian Research Council Discovery grants. He has published more than 200 papers. His research fields include data security, data mining, web services, and their applications in e-health and e-environment.

Yanchun Zhang received the PhD degree from the University of Queensland, Australia. He is a full professor and the director of the Centre for Applied Informatics at Victoria University, Australia. He has been active in areas of information systems, database, data management and mining and health informatics. He was a member of the Australian Research Council College of Experts (2008–2010).