Expert Systems With Applications 194 (2022) 116502

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Time series classification based on complex network

Hailin Li [a,b], Ruiying Jia [a], Xiaoji Wan [a,*]

[a] College of Business Administration, Huaqiao University, Quanzhou 362021, China
[b] Research Center for Applied Statistics and Big Data, Huaqiao University, Xiamen 361021, China

ARTICLE INFO

Keywords:
Time series classification
Complex network
Random forest
Data mining

ABSTRACT

Time series classification is an important topic in data mining. Time series classification methods have been studied by many researchers. A feature-weighted classification method is proposed based on complex network. There are four stages in the proposed classification algorithm. First, we obtain visible points and edge weights by applying an improved weighted visibility graph algorithm to transform the original time series into weighted complex networks. Next, weighted features are calculated based on visible points and weights, and unable-weighted features are obtained by utilizing NetworkX. After that, we combine two types of features to obtain a total features matrix. We then reconsider the weight of each independent feature by utilizing the importance function of random forest to measure the relative importance of features and construct a importance weighted features matrix. Finally, we utilize a traditional classifier called random forest to cluster the weighted features matrix and generate classification results. Simultaneously, a novel feature weight calculation method is applied during the classification process. We compare the proposed method to other classification methods and the results indicate that the proposed method can improve classification accuracy for time series datasets.

1. Introduction

A time series is a set of observations obtained in chronological order. Each observation is a record of a particular phenomenon at a specific moment, and the record represents a particular indicator of this phenomenon. Time series are closely related to production and daily life, such as temperature changes in a specific area, fluctuations in an individual stock, and voice data. Such time-axis-sorted data are collectively referred to as the time series data (Brillinger, 2001; Li, Wu, et al., 2020; López-Oriona & Vilar, 2021).

Classification (del Campo et al., 2021; Jain, 2021; Johnpaul et al., 2021; Ma et al., 2021) and clustering (Kushwah et al., 2020; Li & Wei, 2020; Li, Wu, et al., 2020; Tavakoli et al., 2020) are perhaps the two most popular tasks in time series data mining. In recent years, many scholars have studied time series classification. Most of approaches of time series classification can be categorized into two groups, distance-based approaches and structure-based approaches. The first category is a classification algorithm based on distance, which utilizes the original time series or sub-sequences to build the classification model. Various combinations of similarity metrics and classification models are designed in this category. Euclidean distance (ED) and dynamic time warping (DTW) distance are the most popular index (Wang et al., 2013). This category includes classifiers such as KNN (Lin et al., 2021; Shokrzade et al., 2021), random forest (Egaji et al., 2021; Li, Zhong, et al., 2020), and decision tree (Gregoriades et al., 2021; Jackins et al., 2021). These methods are accurate but expensive, as they have quadratic time complexity in the time series length (Le Nguyen et al., 2017a). The second category is structure-based approaches (Senin & Malinchik, 2013). They transform raw numeric data into discrete representations, the most popular being the Symbolic Aggregate Approximation (SAX) (Bai et al., 2021; Song et al., 2020). These methods can be implemented effectively, but the classification accuracy is negatively affected due to the loss of feature information.

Complex network is the study of the nature and behavior of the system as a whole produced by the interaction between network individuals. Any complex system that contains a large number of constituent units can be studied as a complex network by abstracting constituent units into nodes and abstracting the interrelationships between units as edges. Networks arguably mimic many real-world networks, such as biological, social, and technological networks (Sporns, 2011; Stam & Reijneveld, 2007). Research on complex networks has been going on for a long time (Albert & Barabási, 2002; Barabási & Albert, 1999; Newman, 2003). Typical applications of the complex networks can be found in internet (Brito et al., 2020), neuroscience (Mota et al., 2017), sociology (Shirado & Christakis, 2017), and economics (Thakkar & Chaudhari, 2021). In addition, the field of complex network classification and clustering is gaining considerable attention in recent years. De Arruda et al. (2016) performed a supervised classification aiming at

* Corresponding author.
E-mail addresses: blihailin@163.com (H. Li), 2076176711@qq.com (R. Jia), wanxiaoji@hqu.edu.cn (X. Wan).

https://doi.org/10.1016/j.eswa.2022.116502
Received 5 September 2021; Received in revised form 13 December 2021; Accepted 1 January 2022
Available online 18 January 2022

discriminating informative from imaginative documents based on a networked model. Xin et al. (2020) proposed a novel framework of complex network classifier (CNC) by integrating network embedding and convolutional neural network to tackle the problem of network classification. Resende and Carneiro (2019) investigated a multi-label solution able to combine existing multi-label classifiers with a high-level classifier based on complex networks measures. Jiang et al. (2016) proposed two scan partition schemes based on complex networks clustering to minimize the capture violations without increasing test-data volume and extra area overhead. Li and Liu (2021) proposed a novel method based on complex networks for multivariate time series. This method relies on the utilization of community detection technology to achieve complete multivariate time series clustering. Li and Du (2021) proposed an MTS clustering method based on a component relationship network (CRN).

Since the 20th century, a very important research topic is to link complex network research methods with time series analysis. By utilizing a variety of complex network algorithms, the original data of time series is transformed into a complex network (Ruiz-Herrera & Torres, 2021; Sun et al., 2020; Zou et al., 2019). The time series is analyzed with the mature topological features of complex network theory to discover the potential dynamic characteristics for subsequent research. There are hundreds of statistical features of complex networks, and it is urgent to study which features can better represent the dynamic characteristics of the original time series.

In application, the critical point of complex network theory for time series is constructing complex networks corresponding to the series. In this regard, domestic and foreign researchers have carried out long-term research and proposed a variety of complex network conversion methods. Zhang and Small (2006) realized the conversion of pseudo-periodic time series into complex networks and analyzed the difference between chaotic signals and Gaussian white noise complex network topology statistical characteristics. Zhang et al. (2008) proposed a similarity network construction algorithm. Lacasa et al. (2008) proposed a visibility graph algorithm, which utilizes the visual relationship between nodes as the rule of edge connection to construct a complex network. The visualization algorithm is the first to transform any one-dimensional time series into a complex network. Lacasa et al. (2015) proposed a multiple visibility graph, creating a precedent for multiple time series analysis based on visibility graph algorithm. Supriya et al. (2016) proposed a weighted visibility graph algorithm, which introduces the idea of weighting into visibility and assigns different weights to each connected edge.

Based on the results of major studies focusing time series classification, the main problems that must be resolved to improve the classification of time series are as follows. (1) Morphological features are the main external manifestations of time series data. The traditional classification algorithm based on distance time series ignores the influence of sequence shape on similarity measure to a certain extent, resulting in more significant information loss. Li, Liu, et al. (2020) proposed the over-stretching and over-compression problems are typical drawbacks of using DTW to measure distances. Commonly used distance metrics, such as Euclidean and DTW, are sensitive to phase shift or amplitude shift and cannot recognize the similarity between two time series with rising, peak, and similar patterns simultaneously, which leads to loss of information (Deng et al., 2020). In practice, time series data are vulnerable to various temporal transformations, such as shift and temporal scaling, however, which are unavoidable in the process of data collection. If a learning algorithm directly calculates the difference between such transformed data based on Euclidean distance, the measurement cannot faithfully reflect the similarity and hence could not learn the underlying discriminative features (He, Agard, et al., 2020). (2) Recent study shows that these SAX-based methods fail to achieve state-of-the-art accuracy. The SAX ignores essential information in a segment, namely the trend of the value change in the segment. Such a miss may cause a wrong classification in some cases since the SAX representation cannot distinguish different time series with similar average values but the different trend (He, Long, et al., 2020). The ignorance of the trend characteristics in the subsequence also led to the loss of SAX information (Sun et al., 2014). A fundamental limitation of SAX: The symbol is mapped from the average value of the fragment, and the boundary distance in the segment is not considered (Bai et al., 2021). (3) The current research on utilizing complex networks to extract time series features for classification needs to be further supplemented, and experimental data should be provided.

In this paper, we propose a novel time series classification algorithm based on complex network topology features. The major contributions of this work can be summarized as follows. (1) We extended an improved weighted visibility graph algorithm proposed by Supriya et al. (2016) and applied it to a wider collection of time series. The introduction of edge weights in the transformation process can retain the dynamic information of the time series to a greater extent. (2) We classified time series from a new perspective and further discussed the relationship between complex network topology features and original time series. (3) After ranking the importance of each dataset feature, we utilized the analytic hierarchy process to calculate each feature's weight. On this basis, this article explores which features have a more significant impact on the classification process. (4) The proposed algorithm can improve the accuracy of classification. In the experimental stage, classification accuracy and time series datasets were utilized to verify the proposed algorithm.

To achieve the improvements discussed above, a novel classification method is proposed. Firstly, we utilize an improved weighted visibility algorithm to transform the original time series into weighted networks and extract the visible points and weights. Second, the known equations such as $c_i = \frac{2E_i}{k_i(k_i-1)}$, $K = \frac{1}{N}\sum_{i=1}^{N} k_i$ are utilized to calculate the weighted features, the NetworkX is utilized to calculate the unable-weighted features. The features importance is obtained by utilizing the importance function of random forest. The AHP is utilized to calculate the feature weight and assign it to the features matrix to get the importance weighted features matrix based on the features importance ranking. Finally, classification results can be obtained based on the importance weighted features matrix by applying random forest or another classification algorithm.

The remainder of this paper is organized as follows. In Section 2, we introduce the basic definitions and related algorithms required for implementing the proposed method. In Section 3, we introduced the weighted visibility graph algorithm originally applied to ECG and described the weighted features. In Section 4, we compare the proposed algorithm to other classification algorithms. In Section 5, we summarize the motivations for CCNF and discuss future research directions.

2. Preliminaries

In this section, we introduce the visibility graph algorithm and complex network features which are the main part of new method and we also introduce random forest. The symbols used in the paper is shown in the Appendix section.

2.1. Visibility algorithm

Lacasa et al. (2008) proposed the visibility graph algorithm which can convert any time series into a complex network. The algorithm takes each sample point of the time series as a node of the network, the connection between the sample points of the time series is utilized as edges of the network. Suppose x, y, and z are three sample points of a certain time series. t_x, t_y, and t_z represent sample indexes, v_x, v_y, and v_z represent sample values. If the following equation is satisfied between the three nodes, there is an edge between x and y.

$$v_z < v_y + (v_x - v_y)\frac{t_y - t_z}{t_y - t_x}. \tag{1}$$

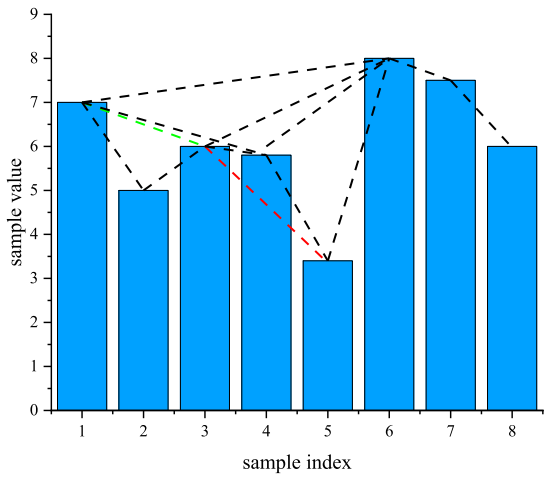


Fig. 1. Distribution of samples.

When the connecting line between the sampled values at two moments is not interrupted by any other sampling points in the middle, there is an edge connection between the two points; on the contrary, there is no edge connection.

The process of the visibility graph algorithm is explained more clearly in the form of graphs. Fig. 1 shows eight sample points. The height of the histogram represents the size of the value, eight nodes are corresponding to the visible graph. When judging whether there is an edge connection between the first node and the third node, a straight line can be utilized to connect these two nodes. According to Fig. 1, the second node is below the line drawn, so the first node and third node are connected(green line). Similarly, For the third node and fifth node, there is a fourth node above the drawn line for the third node and fifth node, so there is no edge connection between the third node and fifth node(red line). After traversing all eight nodes, we can obtain complex network edges. The complex network is visualized as in Fig. 2.

2.2. Complex network features

There are dozens of statistical features of complex networks. The features can be divided into global statistical features, middle-level statistical features, and local statistical features. Considering that there are many features, it is vital to filter out the features that can represent the complex networks well. We selected nine features to describe the complex network, including average_degree, average_shortest_path_length, diameter, global_efficiency, average_clustering_coefficient, average_degree_assortativity_coefficient, density, modularity, and transitivity in this paper. Average_degree, average_shortest_path_length, global_efficiency, diameter, and average_clustering_coefficient are related to weights, and they are unified as weighted features. Density, modularity, average_degree_assortativity_coefficient, and transitivity do not involve weights, and they are unified as unable-weighted features. The term "unable-weighted" means that the feature calculation does not involve the weight of the edges in the network.

Complex networks are usually stored in an adjacency matrix. The adjacency matrix represents the connection relationship between nodes. The adjacency matrix $A = (a_{ij})_N$, $a_{ij} = 0$ indicates no edge between node i and node j, $a_{ij} = 1$ indicates an edge between node i and node j, N represents total number of nodes in the network. After utilizing the visibility graph algorithm to obtain the adjacency matrix, the statistical features of the network can be calculated.

The node's degree is defined as the number of nodes connected to the node. In an unweighted network, the degree of node i is the sum of the elements in the i row or i column of the adjacency matrix $A = (a_{ij})_N$.

$$k_i = \sum_{j=1}^{N} a_{ij} \qquad i \in [1, N]. \tag{2}$$

The average_degree of the complex network is defined as:

$$K = \frac{1}{N} \sum_{i=1}^{N} k_i. \tag{3}$$

The degree reflects the degree of association between each node and other nodes in the complex network. The average degree is the average value of the degree of all nodes in the complex network, reflecting the tightness of the complex network.

Average_shortest_path_length is a vital network feature. In an unweighted complex network, the shortest path l_{ij} refers to the number of shortest path edges between node i and node j. If two nodes are directly connected, the shortest path between them is 1. If two nodes are not directly connected and there is an intermediate point between them, then their distance is increased by 1, so analogy. The maximum distance between any two nodes in the network is called the diameter of the network. The average_shortest_path_length is also called feature path, which refers to the average of the shortest paths between all pairs of nodes. The average shortest path is computed as follows:

$$L = \frac{1}{N(N-1)} \sum_{i,j \in G(i \neq j)} l_{ij}, \tag{4}$$

where G represents the complex network where node i and node j are located. Unweighted network diameter is computed as follows:

$$D = \max(l_{ij}). \tag{5}$$

The global_efficiency measures the network's effectiveness and is the average value of the inverse of the distance between all pairs of nodes. The global efficiency is computed as follows:

$$E = \frac{1}{N(N-1)} \sum_{i,j \in G(i \neq j)} \frac{1}{l_{ij}}. \tag{6}$$

In real social networks, there is often such a small group form, two friends and their respective friends are likely to know each other, and then there will be an edge connection. This phenomenon is called the degree of clustering. The local clustering of each node in G is the fraction of triangles that actually exist over all possible triangles in its neighborhood. The average clustering coefficient of a graph G is the mean of local clustering. In an unweighted complex network, there are k_i edges connected to it for a certain node i in the network, and the number of actually connected edges is E_i. The clustering coefficient of node i is computed as follows:

$$c_i = \frac{2E_i}{k_i(k_i - 1)}. \tag{7}$$

The average clustering coefficient of the entire complex network is computed as follows:

$$C = \frac{1}{N} \sum_{i=1}^{N} c_i. \tag{8}$$

Degree_assortativity_coefficient measures the similarity of network connections based on nodes degree which is computed as

$$m_{ij} = \frac{\sum_{r=1}^{N} a_{ir} a_{jr}}{k_i + k_j - \sum_{r=1}^{N} a_{ir} a_{jr}}, \tag{9}$$

where a_{ir} denotes whether node i and node r are connected, If they are connected, the value is 1, otherwise it is 0. This rule also applies to node j and node r.

The average_degree_assortativity_coefficient of the complex network is defined as:

$$M = \frac{2}{N(N-1)} \sum_{i,j \in pair} m_{ij}. \tag{10}$$

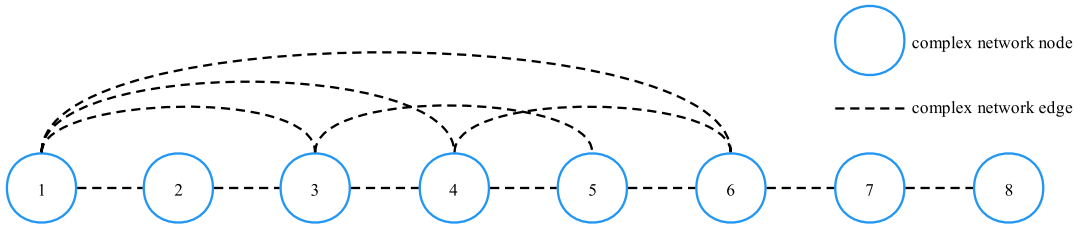


Fig. 2. Visibility graph of samples.

Graph density is utilized to measure the integrity of the network and reflects the closeness of the connections between the nodes. Density is defined as the ratio of the actual number of edges in the network to the maximum possible number of edges. Graph density is computed as follows:

$$\rho = \frac{\text{sum}(A)}{2N(N-1)}, \tag{11}$$

where $\text{sum}(A)$ represents twice the number of edges of the G.

A community is a collection of nodes in the network. The nodes in a community are closely connected, but there are almost no edges with nodes outside the organization. Modularity measures the strength of a complex network community structure. The greater the modularity, the more reasonable the corresponding community division results. Modularity is computed as follows:

$$Q = \frac{1}{\text{sum}(A)} \sum_{i,j \in G} \left(a_{ij} - k_i k_j / (\text{sum}(A))\right) \delta(C_i, C_j), \tag{12}$$

where $a_{ij} = 1$ represents there is an edge between node i and node j, otherwise there is no edge. C_i is the label that node i belongs to a certain community, $\delta(C_i, C_j) = 1$ if and only if $C_i = C_j$.

The transitivity is the fraction of all possible triangles present in G. Possible triangles are identified by the number of "triads" (two edges with a shared vertex). The transitivity is computed as follows:

$$T = 3\frac{\#triangles}{\#triads}, \tag{13}$$

where $\#triangles$ represents the number of triangles in the network, $\#triads$ represents the number of triples in the network.

2.3. Random forest

Random forest is an algorithm based on ensemble learning proposed by Breiman (2001). It is a classifier that utilizes multiple decision trees to train and predict samples. The algorithm adopts a returnable sampling method from b samples. The node of the decision tree in each lesson selects $mtry$ attributes from all p attributes and utilizes the Gini index to generate a non-pruned cart decision tree. The final classification result is obtained by merely voting on the classification results of the decision trees. The random forest breaks through the limitation of a single classifier. Simultaneously, because of its randomness, it has a higher tolerance for abnormal points and noise, so it can effectively avoid over-fitting problems and has a higher generalization ability. The classification performance of some classic classifiers, such as random forest, KNN, and SVM, depends on the problem.

Univariate Time Series (UTS) is a time series that only considers one dimension, and does not consider the influence of other dimensions on this dimension. Suppose an UTS dataset with n objects can be denoted as $Z = \{X^1, X^2, \ldots, X^b\}$, B is the features label set of Z, including p features. There are a total of $ntree$ trees in a random forest, and the number of variables randomly sampled when constructing a decision tree branch is $mtry$. The random forest firstly extracts a training subset Z^* of scale b from the original training set Z according to Bootstrap(step 2). And then, $mtry$ features are randomly selected

Algorithm 1 Classifier random forest, $Label = \text{RF}(Z, ntree, mtry, B, p, X)$.

Input: Dataset $Z = \{X^1, X^2, ..., X^b\}$, number of decision trees $ntree$, variables used in the binary tree $mtry$, features label collection B, variables number p, test sample X;

Output: Labels of X;

1: **for** $e \in$ ntree **do**
2: $Z^* = \text{sample}(Z, p, replace = \text{True})$;
3: $B^* = B[\text{sample}(1 : p, mtry)]$;
4: $DT_e = \text{DecisionTree}(Z^*, \text{B}^*, mtry)$;
5: **end for**
6: $RF = \{DT_e, e = 1, 2, ..., ntree\}$;
7: $\{DT_1(X), DT_2(X), ..., DT_{ntree}(X)\} = RF(X)$;
8: $Label(X) = \arg\max_{h \in H} \sum_{e=1}^{ntree} invote(DT_e(X) = h)$;
9: return $Label(X)$.

from the original features label set B as the sub-features space(step 3). According to the training subset and sub-features space, a decision tree DT_e is generated according to the decision tree algorithm(step 4). The above steps are repeated $ntree$ times to generate $ntree$ decision trees. When a new instance X is classified, the sample is assigned to all decision trees. The random forest utilizes the categories with more decision trees as the overall classification results(steps 7–9). H is the set of classification labels of X by random forest. The class label with the most frequent occurrence in H is selected by voting as the final classification result of X.

2.4. NetworkX

NetworkX (Erd et al., 2021) is a powerful tool for analyzing complex networks based on the python language which contains graphs and complex network algorithms widely utilized. The NetworkX can utilize a standardized data format to store the network's edges and weights, visualize complex networks and perform features statistics. It has a wide range of applications in sociology, neural networks, traffic flow, and other fields. In this article, we utilize the NetworkX to calculate the average_degree_assortativity_coefficient, density, modularity, and transitivity. Suppose $visiblePoints$ is the set of visible points in the complex network G. The complete calculation process is presented in Algorithm 2. D represents density, $part$ represents divided communities, Q represents modularity, M represents average_degree_assortativity_coefficient, T represents transitivity. Steps 1–2 indicate creating an empty complex network G and adding each pair of $visiblePoints$ to the complex network G. Steps 3–9 indicate that the average_ degree_assortativity_coefficient, density, modularity, and transitivity functions of NetworkX are sequentially called to calculate the corresponding four features.

Algorithm 2 Obtain unable-weighted complex network features matrix, UWFM = UCNF(*visiblePoints*).

Input: *visiblePoints*;
Output: Unable-weighted complex network features matrix UWFM = $\{M, D, Q, T\}$;
1: Set G = NetworkX.Graph();
2: **for** (t_i, t_j) in *visiblePoints* **do**
3: G.add_edge(t_i,t_j);
4: **end for**
5: M = NetworkX.average_degree_assortativity_coefficient(G);
6: D = NetworkX.density(G);
7: *part* = community.best_partition(G);
8: Q = community.modularity(*part*, G);
9: T = NetworkX.transitivity(G);
10: return (M,D,Q,T).

2.5. Features importance evaluation

There are often hundreds of previous features in a dataset. How to choose the features that have the greatest impact on the classification results in order to reduce the number of features when building the model or increase the role of certain features in the classification process? We are more concerned about the problem. The feature importance evaluation of random forest is to measure the average contribution value of each feature in the dataset and compare the magnitude. The size of the contribution is usually measured by the Gini index and the error rate of OOB data (that is, the data that is not collected during the random forest sampling process). The importance of a feature is computed as the (normalized) total reduction of the criterion brought by that feature. It is also known as the Gini importance. Some research shows that when the variables are continuous and uncorrelated, the importance estimate based on the Gini index is unbiased; when the signal-to-noise ratio is low, the accuracy of the importance estimate based on the Gini index is also higher than the error rate of OOB data. Considering that the features involved in this article are all continuous, we chose the Gini index as the evaluation method.

The complete calculation process is presented in Algorithm 3. Suppose a dataset has p features: $F_1, F_2, \ldots, F_p$, and U categories, I_j is defined as the importance of the jth feature, GI_m represents the Gini index at the m node on the decision tree DT_e. The purity measure of split at node m is $GI_m = \sum_{u=1}^{U} p_u(1-p_u) = 1 - \sum_{u=1}^{U} {p_u}^2$, where p_u represents the probability of correctly classifying the sample at node m. The Gini index change caused by the pth feature at node m on the eth decision tree is $I_{jm}^{Gini} = GI_m - (GI_l + GI_r)$ (step 5), where GI_l and GI_r respectively represent the Gini index of the left and right branches after branching. The Gini index of all nodes in the eth decision tree DT_e for the pth feature is $I_{DT_e j}^{Gini} = \sum_m I_{jm}^{Gini}$ (step 7). Suppose there are *ntree* trees in total. The average change (Gini index) of the impurity of node splitting in all decision trees of the random forest for the pth feature is $I_j^{Gini} = \sum_{i=0}^{ntree} I_{DT_i j}^{Gini}$ (step 9). Steps 11–13 indicate that all the obtained importance scores are normalized. The importance of features can be calculated by utilizing the Algorithm 3, which lays the foundation for subsequent features weighting.

2.6. Analytic hierarchy process

The analytic hierarchy process (AHP) was put forward by Saaty (2004). It is a decision analysis method of hierarchical weight based on network system theory and multi-objective comprehensive evaluation method. The analytic hierarchy process utilizes mathematical methods to calculate the weights that reflect the relative importance of the elements at each level. It calculates and sorts the relative weights of the elements according to the total ordering between each level.

Algorithm 3 Features importance evaluation, I = FIE(U, *ntree*,p).

Input: number of categories U, number of trees *ntree*, features number p;
Output: The importance of p features I ;
1: **for** each $j \in [1, p]$ **do**
2: **for** each $e \in [1, ntree]$ **do**
3: **for** each $m \in DT_e$ **do**
4: $GI_m = \sum_{u=1}^{U} p_u(1-p_u) = 1 - \sum_{u=1}^{U} {p_u}^2$;
5: $I_{jm}^{Gini} = GI_m - (GI_l + GI_r)$;
6: **end for**
7: $I_{DT_e j}^{Gini} = \sum_m I_{jm}^{Gini}$;
8: **end for**
9: $I_j = \sum_{i=0}^{ntree} I_{DT_i j}^{Gini}$;
10: **end for**
11: **for** each $j \in [1, p]$ **do**
12: $I_j = \frac{I_j}{\sum_{i=1}^{p} I_j}$;
13: **end for**

Algorithm 4 Analytic hierarchy process, w = AHP(O, d).

Input: Judgment matrix O, matrix rank d;
Output: Feature weight w;
1: **for** each $i \in [1, d]$ **do**
2: **for** each $j \in [1, d]$ **do**
3: $o_{ij} = o_{ij} / \sum_{q=1}^{d} o_{qj}$;
4: **end for**
5: **end for**
6: **for** each $i \in [1, d]$ **do**
7: $o_i = \sum_{j=1}^{d} o_{ij}$;
8: $w_i = o_i / \sum_{q=1}^{d} o_q$;
9: **end for**
10: $\lambda_{\max} = \frac{1}{d} \sum_{i=1}^{d} \frac{(Ow)}{w_i}$;
11: $CI = \frac{\lambda_{\max} - d}{d-1}$;
12: Select random consensus index RI;
13: $CR = \frac{CI}{RI}$;
14: **if** $CR < 0.1$ **then**
15: **for** each $i \in [1, d]$ **do**
16: $w_i = w_i / \sum_{i=1}^{d} w_i$;
17: **end for**
18: **end if**

Suppose a specific d elements has relative importance according to a certain criterion. The ith element is compared with the element j in the remaining d-1 elements, and their relative importance is respectively denoted as o_{ij} based on the assumption of the rule of scaling. This d-rank matrix is called the judgment matrix, denoted as O = $(o_{ij})_{d \times d}$. The complete calculation process is presented in Algorithm 4. Firstly, after constructing the judgment matrix O, the columns are normalized as $o_{ij} = o_{ij} / \sum_{q=1}^{d} o_{qj}$(steps 1–5). Then, the elements of the matrix O are added in rows to obtain a vector w, and the vector w is normalized (steps 6–9). w_i is denoted as $w_i = o_i / \sum_{q=1}^{d} o_q$. Steps 10–13 indicate calculating the maximum eigenvalue $\lambda_{\max}$ and the consistency check coefficient CR. Since $Ow = \lambda w$, $\lambda_{\max} = \frac{1}{d} \sum_{i=1}^{d} \frac{(Ow)}{w_i}$. After defining the consistency index $CI = \frac{\lambda_{\max} - d}{d-1}$, the random consistency index RI is introduced. The ratio of the consistency index CI to the random consistency index RI of the same rank is called the consistency ratio CR, which is denoted as $CR = \frac{CI}{RI}$. We usually utilize the consistency

ratio CR as the standard for the consistency test of the judgment matrix O. If the CR is smaller, the consistency of the matrix is better. Generally set a critical value of 0.1, when $CR < 0.1$, the judgment matrix O has passed the consistency test, and the eigenvalues and eigenvectors are both acceptable. the standardized w is the weights of the d elements (steps 14–18).

3. Classification based on complex network

Classification based on complex networks has become one of the hot research directions. The visibility graph algorithm and horizontal visibility algorithms do not consider the influence of weights on features in transforming time series into complex networks, so more information will be lost, which will eventually negatively affect the classification results.

In this section, we propose a novel algorithm that combines a weighted visibility graph method and random forest. First, to reduce the impact of information loss, we introduce a simple algorithm to transform the original time series. Next, some equations are proposed to generate weighted features, which are the basis of the final classification results. After that, we calculate each feature's weight through the analytic hierarchy process and assign it to the features matrix. Finally, we propose a classification method called CCNF. The proposed classification algorithm combines complex network features, features weights, and random forest and performs well on time series datasets.

Algorithm 5 Improved weighted visibility graph, $(visiblePoints, new\_weight) = \mathrm{IWVG}(X)$.

Input: Time series $X = \{x_1, x_2, ..., x_n\}$;
Output: visible points set $visiblePoints$ edge weights set new_weight;

1: **for** each $i \in [1, n]$ **do**
2: $ts = ts \cup [i, x_i]$;
3: **end for**
4: **for** i,j in combinations(ts,2) **do**
5: $(t_i, v_i) = i$;
6: $(t_j, v_j) = j$;
7: $connect$ = True;
8: $medium = ts[t_i, t_j - 1]$;
9: **for** t_z, v_z in $medium$ **do**
10: **if** $v_z > v_j + (v_i - v_j)(\mathrm{float}(t_j - t_z)/(t_j - t_i))$ **then**
11: $connect$ = False;
12: **end if**
13: **end for**
14: **if** $connect$ **then**
15: $visiblePoints = visiblePoints \cup (t_i, t_j)$;
16: **end if**
17: **for** (t_i, t_j) in $visiblePoints$ **do**
18: $w_{ij} = \left|\arctan\left(\frac{v_j - v_i}{t_j - t_i}\right)\right|$;
19: $new\_weight = new\_weight \cup (w_{ij})$;
20: **end for**
21: **end for**
22: return new_weight, $visiblePoints$.

3.1. Improved weighted visibility algorithm

In the process of weighted complex network construction, the choice of weights largely determines the performance of features. Supriya et al. (2016) proposed a weighted visibility graph algorithm for ECG data, reflecting the time series features more precisely. On this basis, we apply this algorithm to the public time series collection. According to the principle of weighted visibility graph construction, the w_{xy} of the edge a_{xy} between node x and node y is computed as follows:

$$w_{xy} = \begin{cases} \left|\arctan\left(\frac{v_y - v_x}{t_y - t_x}\right)\right| & (x < y) \\ 0 & (x \geq y) \end{cases}. \tag{14}$$

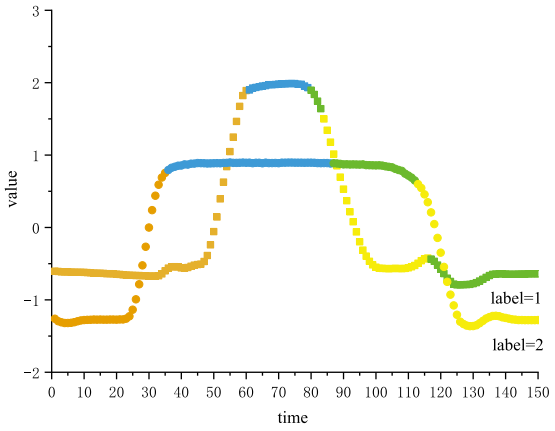


Fig. 3. Original time series (GunPoint dataset).

The angles of different complex network nodes have apparent differences, which reflect the size of the original time series fluctuations. The complete calculation process is presented in Algorithm 5. Suppose the time series X is denoted as $\{x_1, x_2, \ldots, x_n\}$. Steps 1–3 indicate the creation of key–value pairs of X, the node x_i in X and i is composed of (i, x_i) and added to ts. Steps 4–16 indicate that after the node tuples in ts are combined in pairs, Eq. (1) is utilized to determine whether there is an edge connection between the two points of a tuple in turn. The $medium$ is the set of all nodes located between node i and node j. The relationship between nodes i, j and z in the $medium$ is sequentially judged to determine whether there is a connection between the three nodes according to Eq. (1). If there is an edge connection, the tuple (t_i, t_j) will be added to the $visiblePoints$. Finally, the weight of each group nodes w_{ij} in $visiblePoints$ is calculated according to Eq. (14) and added to new_weight (steps 17–21).

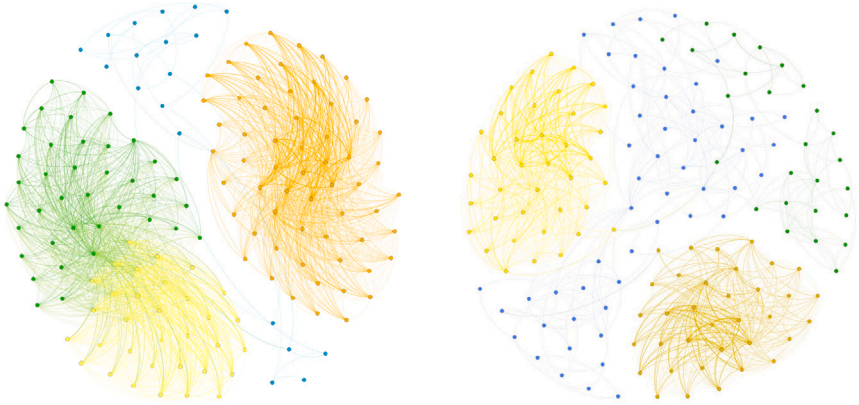
In order to more clearly illustrate the process of converting a time series into a complex network utilizing the weighted visibility graph algorithm, we have selected two time series with different labels from the GunPoint dataset. According to Fig. 3, the two time series have obvious differences in shape. The label = 1 sequence is above the label = 2 sequence, and the peak distributions of the two are also different.

According to Fig. 4, there are obvious differences in the overall features of the complex networks formed by the proposed weighted visibility graph algorithm. The complex network (a) is mainly divided into three communities. The community nodes are closely connected, and the nodes outside the community have scattered distribution. The complex network (b) is mainly divided into two communities. The connections within the community are relatively loose, and many nodes are floating outside the community. The differences of these complex network structures are finally reflected in the values of the features. According to Fig. 4(a), the connection between nodes is closer, and each node has more connected edges, so the graph density value and the average degree value are larger. For the network diameter, most nodes in Fig. 4(b) do not form communities and are far away, so the network diameter value is smaller than that in Fig. 4(a).

3.2. Weighted complex network features

Compared with the unweighted network, the weighted network assigns different weights to the edges in order to describe the connection strength between nodes. The calculation of the five features of average_degree, average_shortest_path_length, diameter, global_efficiency, and average_clustering involves the edges weights. Therefore, the calculation of w_ij needs to be added based on Eq. (3)–(6) and (8).

The point_intensity integrates the features of degree and edge weight. In a weighted complex network, the strength s_i of node i refers to the



(a) Complex network with label = 1. (b) Complex network with label = 2.

Fig. 4. Complex networks(GunPoint dataset).

sum of the edges' weights, which are directly connected to node i. It is computed as follows:

$$s_i = \sum_{j \in N_i} w_{ij}, \tag{15}$$

where N_i represents the set of nodes connected to node i, w_{ij} represents the weight of the edge between node i and node j.

Average_point_intensity which refers to the average point intensity of all nodes in the network is computed as follows:

$$S = \frac{1}{N} \sum_{i=1}^{N} s_i. \tag{16}$$

Utilizing edges weights to calculate the average shortest path and network diameter can better reflect the network features in a weighted complex network. The weight is divided into similarity weight and dissimilarity weight. In this article, we choose similarity weight, the weighted distance is proportional to the weight value. For the entire network G, the weighted average_ shortest_path_length is computed as:

$$L = \frac{1}{N(N-1)} \sum_{i,j \in G(i \neq j)} dis_{ij}, \tag{17}$$

where dis_{ij} represents the minimum value of the sum of weighted path weights between node i and node j.

The diameter of weighted complex network is computed as follows:

$$D = \max(dis_{ij}). \tag{18}$$

The global_efficiency in a weighted complex network measures the network's effectiveness. It represents the average value of the inverse of the weighted distance between all pairs of nodes. The global efficiency is computed as follows:

$$E = \frac{1}{N(N-1)} \sum_{i,j \in G(i \neq j)} \frac{1}{dis_{ij}}. \tag{19}$$

In a weighted complex network, the clustering coefficient of nodes takes into account the influence of edges weights on the clustering degree, which is a supplement to traditional calculation methods. On the basis of the definition of sub-intensity, it is computed as follows:

$$c_i = \frac{2}{k_i(k_i-1)} \sum_{j,k} \left(\tilde{w}_{ij} \tilde{w}_{ik} \tilde{w}_{jk}\right)^{1/3}, \tag{20}$$

where $\tilde{w}_{ij}$ represents the weight of the edge connected to node i divided by the maximum weight in the network.

The average_clustering_coefficient of the entire weighted complex network is computed as follows:

$$C = \frac{1}{N} \sum_{i=1}^{N} c_i. \tag{21}$$

3.3. Features weighting based on importance ranking

The importance of each feature that affects the final classification effect is measured by constructing a random forest and utilizing FIE algorithm. According to the feature importance ranking results, features weights are assigned for subsequent classification tasks. However, due to the characteristics of random forest, when the features are solved based on the time series training dataset, the importance of each feature obtained in multiple experiments for the same data is different. But it can be seen from the observation of the results of multiple experiments, the value of the importance of each feature fluctuates within a specific range. Therefore, when measuring the importance of the features, we averaged several experiments since it overfits to the experiments considered.

If the importance value of a feature is larger, it means that the impurity of the feature is reduced more when the node is divided, which is beneficial to subsequent classification. For the more critical features, the original feature value is given a greater weight to improve its influence in the entire classification process, theoretically, the final classification effect can be optimized.

Calculating the importance of features requires the total features matrix (TFM) as input data. The TFM is composed of a weighted features matrix WFM and an unable-weighted features matrix UWFM. A total of five weighted features are calculated by Eq. (16)–(19) and (21). A total of four unable-weighted features are calculated by Algorithm 2. We utilize the TFM as the input parameter of FIE to obtain the average importance ranking of the nine features after ten trials. After that, the judgment matrix O is constructed based on the importance of the features. Finally, we utilize the judgment matrix O as the input parameter of the AHP algorithm to obtain the weight vector of nine features. The importance weighted features matrix is obtained by assigning w to the TFM.

Aiming at how to construct the judgment matrix by utilizing feature importance ranking, we proposed a new index system. After utilizing the FIE algorithm to obtain the importance I of nine features, we obtain the $order\_I$ by ranking the I. The order of biggest importance value is nine, and the order of smallest importance value is one. The relative importance of the ith feature and the jth feature is defined as:

$$o_{ij} = \frac{order\_I(i)}{order\_I(j)}. \tag{22}$$

The ranking index of the ith feature and the ranking index of the jth feature are compared as the relative importance of the two importance. Therefore, we can construct a judgment matrix O relative to the new scale criterion. The importance of p features obtained by the FIE algorithm is ranked as $order\_I$.

Algorithm 6 Features weighting based on importance ranking, (IWFM, $feature\_weight$) = FWIR(TFM, U, $ntree$, p, d).

Input: Total Features matrix TFM, number of categories U, number of trees $ntree$, features number p, matrix rank d;
Output: Importance weighted features matrix IWFM;

1: **for** each $h \in [1, 10]$ **do**
2: $I_{1h}, I_{2h}, ..., I_{ph}$ = FIE($U, ntree, p$);
3: **for** each $j \in [1, p]$ **do**
4: $I_j = I_j \cup (I_{jh})$;
5: **end for**
6: **end for**
7: **for** each $j \in [1, p]$ **do**
8: $I = I \cup (\text{mean}(I_j))$
9: **end for**
10: $order\_I$ = order(I);
11: **for** each $i \in [1, p]$ **do**
12: **for** each $j \in [1, p]$ **do**
13: $o_{ij} = \frac{order\_I(i)}{order\_I(j)}$
14: **end for**
15: **end for**
16: $feature\_weight$ = AHP(O, d);
17: IWFM = $feature\_weight$ * TFM;
18: return (IWFM, $feature\_weight$).

The complete calculation process is presented in Algorithm 6. Total features matrix TFM, number of categories U, number of trees $ntree$, and features number p are the input parameters of the FWIR algorithm. Steps 1–9 indicate that after repeating the FIE algorithm ten times, the average importance value of each feature I is obtained. Step 10 indicates utilizing the order function to get the ranking of the importance of the nine features in I. Steps 11–15 indicate comparing the importance rankings in I to construct the judgment matrix O. Further, the judgment matrix O is utilized as the input parameter of the AHP algorithm to calculate the feature weight vector $feature\_weight$. The $feature\_weight$ is assigned to the total feature matrix TFM to obtain the importance weighted features matrix IWFM.

3.4. Classification method

In this section, we propose a novel classification method based on complex network called CCNF. CCNF combines complex network features and random forest during the classification process. Additionally, CCNF combines weighted features and features importance because the former can reduce information loss and the latter can enhance the influence of certain features in the classification process.

An UTS dataset with b objects can be denoted as $Z = \{X^1, X^2, \dots, X^b\}$ and $X^i = [x_{1i}, x_{2i}, \dots, x_{ni}]^{\mathrm{T}}$. We utilize Algorithm 5 to calculate the $visiblePoints$ and new_weight. Second, weighted features matrix WFM is computed utilizing Eq. (16)–(19) and (21). Based on NetworkX, we can utilize Algorithm 2 to calculate unable-weighted features matrix UWFM. We merge WFM and UWFM to get TFM. To highlight the effects of different features, we calculate the importance of each feature $I = \{I_1, I_2, \dots, I_p\}$ according to Algorithm 3, where I_l denotes the importance of the lth feature. Considering the randomness of random forest, we repeat the experiment ten times to obtain the average importance of all features and sort them. Next, based on Eq. (22) and the importance factor $I = \{I_1, I_2, \dots, I_p\}$ obtained previously, we can compute the judgment matrix O. We obtain $feature\_weight$ utilizing the AHP algorithm. To introduce the aforementioned features effects, we utilize the $feature\_weight$ to combine the features matrix TFM into a importance weighted features matrix IWFM. In this study, we considered IWFM as primary data and utilized random forest to generate final classification results. The specific classification process is illustrated in Fig. 5.

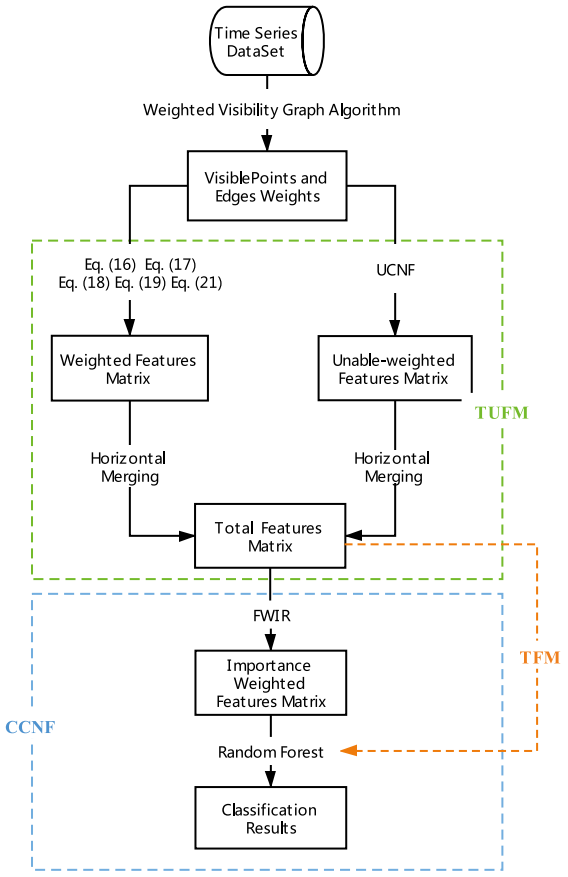


Fig. 5. CCNF procedure.

According to Fig. 5, the entire CCNF process can be divided into four phases. The first phase utilizes an improved weighted visibility graph algorithm to convert a single dimensional time series into a complex network to obtain visible points and edges weights. The second phase generates a total features matrix, combining the weighted features matrix WFM generated by Eq. (16)–(19) and (21), and the unable-weighted features matrix UWFM generated by calling NetworkX. The third stage is to utilize the FWIR algorithm to calculate the weight of each feature and generate importance weighted features matrix IWFM. The final phase is result generation. In this study, we utilized random forest to obtain classification results. According to different application fields, different classification algorithms can be chosen in the final phase, such as KNN and SVM.

The complete calculation process is presented in Algorithm 7. For an UTS dataset $Z = \{X^1, X^2, \dots, X^b\}$, CCNF requires the features label set B of Z, the number of decision trees $ntree$, the number of categories U, the number of features p, and the number of variables used in the binary tree $mtry$ as input parameters. CCNF initially calculates $visiblePoints$ and new_weight according to Algorithm 5. Next, two different features matrices are calculated utilizing Eq. (16)–(19) and (21), and Algorithm 2. We combine these two matrices as a single total features matrix TFM. Base on the TFM, importance weighted features matrix IWFM and $feature\_weight$ can be obtained by utilizing Algorithm 6. For the test sample X, the total features matrix TFM_X of X is obtained by repeating steps 1–4. We considered the IWFM and TFM_X as raw data and utilized random forest to generate classification results.

4. Experimental evaluation

To verify the rationality of the proposed algorithm and the quality of classification results, we conducted experiments on 35 time series

Algorithm 7 CCNF: time series classification based on complex network features.

Input: Train dataset $Z = \{X^1, X^2, ..., X^b\}$, features label collection B, number of decision trees $ntree$, number of variables used in the binary tree in the node $mtry$, number of categories U, features number p, test sample X;

Output: Classification result $Lable$;

1: Calculate $visiblePoints$ and new_weight according to Algorithm 5;
2: Calculate weighted features matrix WFM according to Eq. (16)–(19) and (21);
3: Calculate unable-weighted features matrix UWFM according to Algorithm 2;
4: Combine WFM and UWFM as TFM;
5: Calculate importance weighted features matrix IWFM and eigenvector, (IWFM, $feature\_weight$) = FWIR(TFM, U, $ntree$, p);
6: Calculate TFM_X of X by repeating steps 1–4;
7: $Lable$ = RF(IWFM, $ntree$, $mtry$, B, p, TFM_X);
8: return $Lable$.

Table 1
Datasets and parameters.

Name	Train	Test	Length	Class
Beef	30	30	470	5
ToeSegmentation1	40	288	277	2
Coffee	28	28	286	2
DistalPhalanxOutlineAgeGroup	400	139	80	3
ECG200	100	100	96	2
GunPoint	50	150	150	2
MoteStrain	20	1252	84	2
ProximalPhalanxTW	400	205	80	6
ProximalPhalanxOutlineAgeGroup	400	205	80	3
Ham	109	105	431	2
DistalPhalanxOutlineCorrect	600	276	80	2
Herring	64	64	512	2
Beetfly	20	20	512	2
Car	60	60	577	4
MiddlePhalanxTWOutlineAgeGroup	400	154	80	3
ShapeletSim	20	180	500	2
SonyAIBORobotSurface1	20	601	70	2
Earthquakes	322	139	512	2
Wine	57	54	234	2
SyntheticControl	300	300	60	6
MiddlePhalanxOutlineCorrect	600	291	80	2
ECGFiveDays	23	861	136	2
BirdChicken	20	20	512	2
ToeSegmentation2	36	130	343	2
TwoLeadECG	23	1139	82	2
Plane	105	105	144	7
ArrowHead	36	175	251	3
CBF	30	900	128	3
DiatomSizeReduction	16	306	345	4
FaceFour	24	88	350	4
SonyAIBORobotSurface2	27	953	65	2
Symbols	25	995	398	6
ItalyPowerDemand	67	1029	24	2
MiddlePhalanxTW	399	154	80	6
DistalPhalanxTW	400	139	80	6

(Table 1) datasets and compared classification results between CCNF and other algorithms.

4.1. Evaluation index

In this study, we utilized a classical evaluation index to compare the results of different methods, namely accuracy. The accuracy is the most common indicator for evaluating classification algorithms and it has recently been utilized as the only evaluation indicator (del Campo et al., 2021; Geler et al., 2020; Lahreche & Boucheham, 2021). A true positive (TP) decision means the number of positive examples correctly classified: the number of actual positive and classified samples as positive by the classifier. A false positive (FP) decision means that the number of positive examples erroneously classified: the number of examples that are negative but are classified as positive by the classifier; A false negative (FN) decision means the number of negative examples that are erroneously classified: the number of actual positive examples but are classified as negative by the classifier; A true negative (TN) decision means the number of correctly classified as negative examples: the number of actual negative examples and classified examples as negative by the classifier.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}. \tag{23}$$

To verify the performance of the proposed method, we applied it to benchmark datasets from the UCR databases. It should be noted that the choice of datasets is fair and there is no preference for any method. The datasets originate from different domains, such as spectro, image, and sensor. They comprise different univariate time series with different number of classes, length, training size, and testing size. Each dataset consists of two subsets, the training set and the testing set.

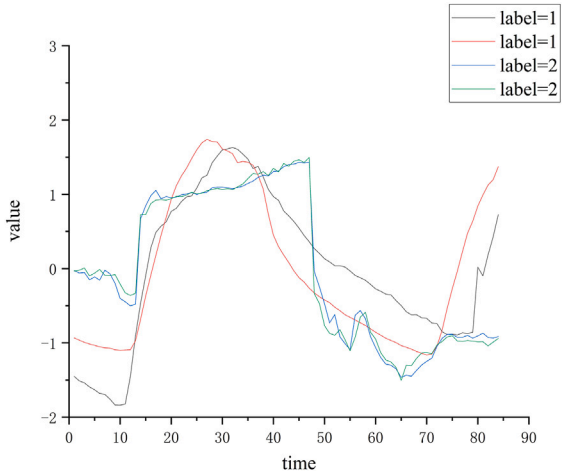
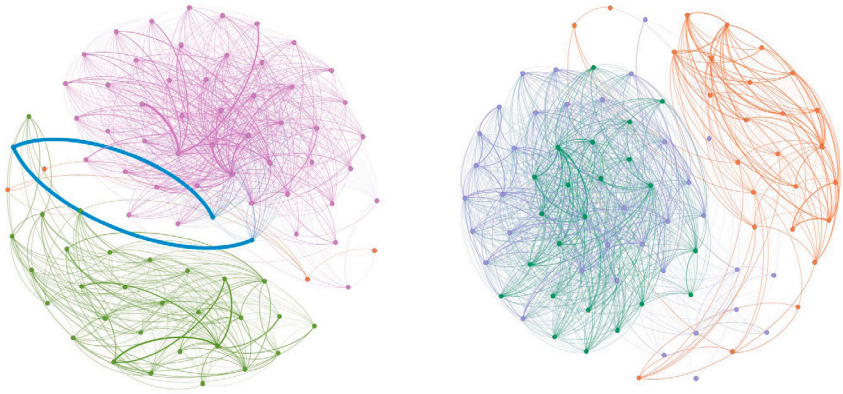


Fig. 6. Original time series (MoteStrain dataset).

4.2. Instance analysis

The purpose of case analysis is to calculate complex network features to verify the degree to which the features represent the original time series. To clarify the classification process of CCNF, four time series were selected from the MoteStrain dataset to illustrate the whole process. Two of the sequences have a label of 1, and the other two sequences have a label of 2.

According to Fig. 6, the two time series with label = 1 are displayed as an "N" shape as a whole, the slope between the nodes is relatively large. The two time series with label = 2 show irregular fluctuations, and the slope between nodes is small. We utilized the weighted visibility graph algorithm to convert the four time series into complex networks. We named the complex network converted from the time series of label = 1 (black line) as c1, and named the complex network converted from the time series of label = 1 (red line) as c2. Correspondingly, the networks converted from the two time series with label = 2 are named c3 and c4 in turn. According to Fig. 7, the two complex networks with label = 1, c1 and c2, are closely connected internally, and there are many edges between nodes. The two networks are each divided into three communities. The nodes in the community are closely connected, and a few nodes float outside the community. It can be seen that the morphological distributions of the two complex networks with label = 1 are similar. According to Fig. 8, the two complex networks with label = 2, c3 and c4, are loosely connected



(a) Complex network c1 (lable = 1). (b) Complex network c2 (lable = 1).

Fig. 7. Complex networks(lable = 1).

(a) Complex network c3 (lable = 2). (b) Complex network c4 (lable = 2).

Fig. 8. Complex networks(lable = 2).

internally, and there are few edges between nodes. The two networks are divided into three communities, but each community contains few nodes, and most of the nodes are outside the community. It can be seen that the morphological distributions of the two complex networks with label = 2 are similar.

This article utilizes the above four time series to illustrate the proposed CCNF process. Firstly, the improved visibility algorithm is utilized to obtain the set of visible points and edges weights corresponding to the four series. Based on the $visiblePoints$ and new_weight obtained by conversion of each time series, Eq. (16)–(19) and (21) are utilized to calculate the weighted features matrix. According to the $visiblePoints$, the UWFM algorithm is utilized to calculate the unable-weighted features matrix. After the two matrices are combined, the total features matrix is obtained, as shown in Table 2. From the above analysis, it can be seen that the networks of Fig. 7 are denser than the networks of Fig. 8 as a whole, and the nodes have more edges, which mean that the networks have more stronger compressive ability and connectivity. The five features of two networks with label = 1 average_point_intensity, average_shortest_path_length, density, and global_ efficiency are greater than those of label = 2. It can be seen that the features of the complex network reflect the morphological characteristics of the network to a large extent and further reflect the dynamic characteristics of the original time series. Since features values can clearly distinguish the differences between complex networks, they can play a better role in subsequent classification process.

We utilize the FIE algorithm to calculate the importance of the nine features involved in the paper based on Table 2. Considering the randomness of random forests, the FIE algorithm is executed ten times to ensure the accuracy of the results. The importance of the nine features is shown in Table 3. The importance of different network features in the network classification process is different, which can be directly seen from Table 3. Therefore, if each feature is treated equally, it does not conform to the above-mentioned situation. Drawing on the ideas of features space weighting (Chai & Yan, 2018; Liang et al., 2020; Zhao et al., 2018), we have enough motivation to try to use the Gini index analogy of information entropy and information gain to assign weights to the features matrix. The features importance calculated by random forest is just the normalized calculation of the increment of Gini index. It is also reasonable to assign the features importance as a weight to the features matrix.

Table 2
Complex network features of four time series from the Motestrain dataset.

Feature	c1	c2	c3	c4
average_point_intensity	7.523	12.847	4.615	3.873
average_shortest_path_length	0.064	0.109	0.041	0.033
diameter	2.890	2.660	4.440	4.220
global_efficiency	0.195	0.357	0.144	0.113
average_clustering_coefficient	0.074	0.149	0.052	0.046
average_degree_assortatity_coefficient	0.080	0.636	0.050	−0.013
density	0.190	0.340	0.119	0.094
modularity	0.327	0.288	0.620	0.625
transitivity	0.589	0.758	0.602	0.539

Table 3
Features importance and ranking.

Feature	Importance	Rank
average_point_intensity	0.124	6
average_shortest_path_length	0.105	3
diameter	0.130	8
global_efficiency	0.122	5
average_clustering_coefficient	0.110	4
average_degree_assortatity_coefficient	0.132	9
density	0.127	7
modularity	0.092	2
transitivity	0.058	1

After obtaining the importance of each feature, the judgment matrix O is constructed based on Eq. (22). According to Table 3, the order of importance of each feature is 6, 3, 8, 5, 4, 9, 7, 2, and 1. In the process of constructing O, the features are compared in pairs. For example, the relative importance of the two features of density and modularity is 7/2. The judgment matrix O is:

$$\mathbf{O} = (o_{ij})_{9\times 9} = \begin{pmatrix} 1.000 & 2.000 & 0.750 & 1.200 & 1.500 & 0.667 & 0.857 & 3.000 & 6.000 \\ 0.500 & 1.000 & 0.375 & 0.600 & 0.750 & 0.333 & 0.429 & 1.500 & 3.000 \\ 1.333 & 2.667 & 1.000 & 1.600 & 2.000 & 0.889 & 1.143 & 4.000 & 8.000 \\ 0.833 & 1.667 & 0.625 & 1.000 & 1.250 & 0.556 & 0.714 & 2.500 & 5.000 \\ 0.667 & 1.333 & 0.500 & 0.800 & 1.000 & 0.444 & 0.571 & 2.000 & 4.000 \\ 1.500 & 3.000 & 1.125 & 1.800 & 2.250 & 1.000 & 1.286 & 4.500 & 9.000 \\ 1.167 & 2.333 & 0.875 & 1.400 & 1.750 & 0.778 & 1.000 & 3.500 & 7.000 \\ 0.333 & 0.667 & 0.250 & 0.400 & 0.500 & 0.222 & 0.286 & 1.000 & 2.000 \\ 0.167 & 0.333 & 0.125 & 0.200 & 0.250 & 0.111 & 0.142 & 0.500 & 1.000 \end{pmatrix}. \quad (24)$$

According to the AHP algorithm, the elements of the judgment matrix O are firstly normalized by column. For example, the elements in the first column of matrix O are $O[:,1] = (1.000, 0.500, 1.333, 0.833, 0.667, 1.500, 1.167, 0.333, 0.167)^{T}$, the sum of 9 elements is 7.500. The normalized $O_z[:,1] = (0.133, 0.067, 0.178, 0.111, 0.08\ 9, 0.200, 0.156, 0.044, 0.022)^{T}$ can be obtained by dividing each element by 7.500. We obtain the standardized matrix O_z according to the above rules. The elements of the matrix O_z are added in rows to obtain the vector $w = (1.200, 0.600, 1.600, 1.000, 0.8\ 00, 1.800, 1.400, 0.400, 0.200)^{T}$ and the normalized vector $w_z = (0.133, 0.067, 0.178\ , 0.111, 0.089, 0.200, 0.156, 0.044, 0.022)^{T}$. The w_z is the eigenvector of the nine features. In order to further test the consistency of the judgment matrix, we need to calculate the maximum eigenvalue $\lambda_{\max}$ and the consistency index CI. The $\lambda_{\max}$ is calculated as:

$$\lambda_{\max} = \frac{1}{d}\sum_{i=1}^{d}\frac{(Ow)}{w_i}, \quad (25)$$

and the CI is calculated as:

$$CI = \frac{\lambda_{\max} - d}{d-1}. \quad (26)$$

It is known that $d = 9$ and $\lambda_{\max} = 9.000$ can be calculated according to Eq. (25). The $CI = 0.000$ is further calculated according to Eq. (26). When $d = 9$, the random consistency index RI is 1.460. According to Eq. (27), the consistency ratio CR is calculated to be 0.000. The CR is less than 0.1, which means that the judgment matrix O satisfies the consistency test, and the calculated eigenvector w has consistency. By assigning the consistent eigenvector w to the features matrix F shown in Table 3, the importance weighted features matrix IWFM is obtained. Finally, the IWFM is utilized as the original data of the random forest classifier to obtain the classification results.

$$CR = \frac{CI}{RI}. \quad (27)$$

The above-mentioned process of calculating the importance of features is included in Algorithm 6. The judgment matrix O, the matrix rank d, and the features matrix F are utilized as input parameters, and the importance weighted features matrix IWFM is output.

Table 4
Decision tree parameters for average accuracy.

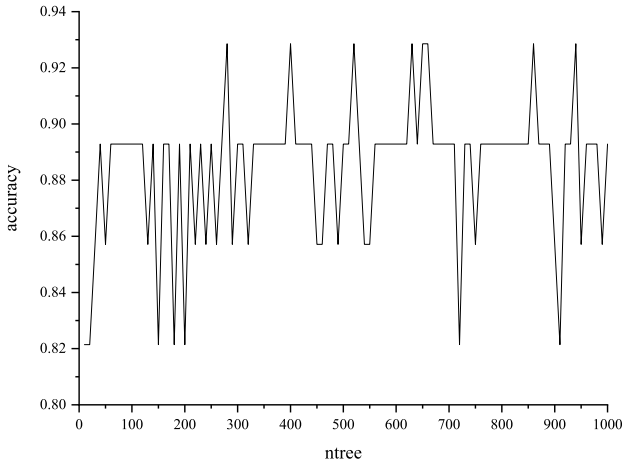
Name	Accuracy	Ntree	Mtry
Beef	0.567	30	7
ToeSegmentation1	0.850	20	3
Coffee	0.929	20	2
DistalPhalanxOutlineAgeGroup	0.833	40	8
ECG200	0.846	180	8
GunPoint	0.858	40	2
MoteStrain	0.838	410	1
ProximalPhalanxTW	0.789	520	3
ProximalPhalanxOutlineAgeGroup	0.854	630	1
Ham	0.714	60	8
DistalPhalanxOutlineCorrect	0.807	90	2
Herring	0.893	70	4
Beetfly	0.850	20	9
Car	0.881	40	3
MiddlePhalanxTWOutlineAgeGroup	0.780	530	1
ShapeletSim	0.856	60	1
SonyAIBORobotSurface1	0.880	10	2
Earthquakes	0.807	130	3
Wine	1.000	30	2
SyntheticControl	0.673	20	2
MiddlePhalanxOutlineCorrect	0.858	30	3
ECGFiveDays	0.907	70	4
BirdChicken	0.850	50	3
ToeSegmentation2	0.908	40	2
TwoLeadECG	0.948	80	2
Plane	1.000	30	2
ArrowHead	0.801	40	2
CBF	0.894	200	1
DiatomSizeReduction	0.875	50	2
FaceFour	0.887	160	3
SonyAIBORobotSurface2	0.869	130	5
Symbols	0.817	110	3
ItalyPowerDemand	0.861	560	3
MiddlePhalanxTW	0.694	10	1
DistalPhalanxTW	0.775	520	1

4.3. Parameter settings

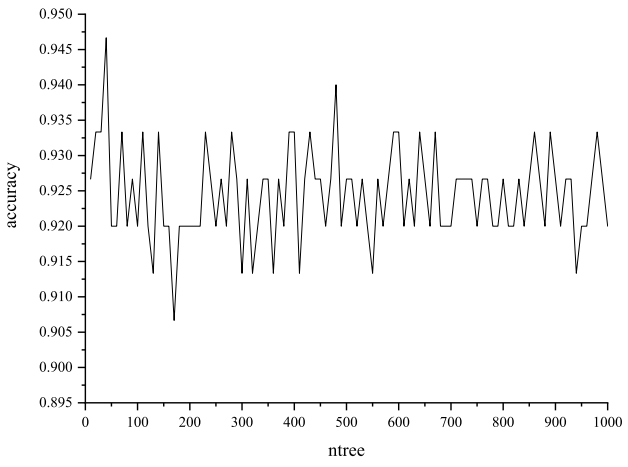
For random forests, the number of decision trees and the number of variables used in the node binary tree will significantly affect the classification results. Therefore, to further explore the specific impact of these two parameters on the classification results, we set the number of decision trees from 10 to 1,000, with a step size of 10. The number of node binary tree features is set from one to nine (the total number of features selected in this article), with a step size of one. By adjusting these two parameters, the classification accuracy is improved. We chose coffee and GunPoint two datasets to illustrate random forest parameters' influence on the final classification results.

In Fig. 9(a), the impact of the number of decision trees on the classification effect is very significant. As the number of decision trees increases from 10 to 1,000, the accuracy rate fluctuates sharply between 0.80 and 0.94. When the number of subtrees is equal to 300, the highest accuracy rate appears for the first time. It can also be seen in Fig. 9(b) that the number of sub-trees has a severe impact on the decision tree. When the number of sub-trees is equal to 80, the highest accuracy rate appears for the first time. Therefore, according to Fig. 9, we initially concluded that the accuracy rate and the number of subtrees have a definite correlation, and further guess that when the number of subtrees is between 10–300, the accuracy rate can achieve the highest value. In Fig. 10(a), as mtry increases from one to eight, the accuracy first increases and then decreases. When mtry=2, it reaches the highest value. In Fig. 10(b), as mtry increases, the accuracy shows a downward trend. When mtry=1, it reaches the highest value. Therefore, according to Fig. 10, we initially guess that the effect of mtry on accuracy differs in different datasets.

To further explore the influence of random forest parameters selection on the final classification effect, we list the number of decision trees and the number of binary tree variables corresponding to each

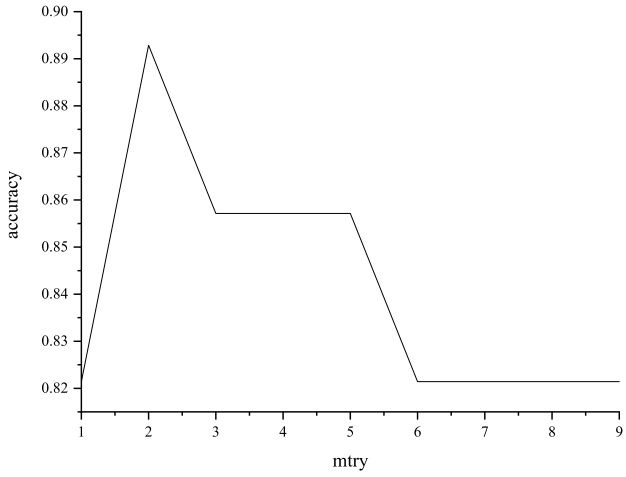


(a) Accuracy changes with ntree (Coffee dataset).

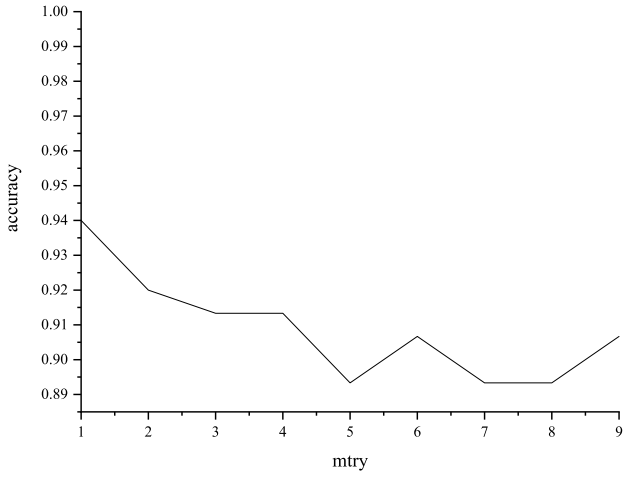


(b) Accuracy changes with ntree (GunPoint dataset).

Fig. 9. Accuracy changes with ntree.



(a) Accuracy changes with mtry (Coffee dataset).



(b) Accuracy changes with mtry (GunPoint dataset).

Fig. 10. Accuracy changes with mtry.

dataset's best classification result in Table 4. According to Table 4, the number of subtrees corresponding to 83 percent of the datasets is less than 300, and most of them are within 100. Besides, it can be seen that the number of binary tree variables corresponding to 80 percent of the datasets is concentrated in between one and five. However, the beef dataset corresponds to seven binary tree variables. In short, although CCNF involves the selection of two parameters, there are rules to follow, that is, the number of subtrees is generally distributed between 10–300, and the number of binary tree variables is generally distributed between one and five, which reduces the difficulty of parameter selection.

4.4. Features importance ranking

After obtaining the features matrix, we utilized the FIE to calculate and rank the importance of each feature and then utilized the AHP to calculate the weights.

For the nine features involved in this article, we utilized the AHP to build a ninth-rank judgment matrix, and output the eigenvector and weight of each feature, as shown in Table 5. According to Table 5, the eigenvectors are 0.415, 1.591, 0.354, 0.530, 3.183, 0.616, 0.455, 0.796, and 1.061. The corresponding weight values of the nine features are 4.613%, 17.681%, 3.929%, 5.894%, 35.361%, 6.842%, 5.052%, 8.840%, and 11.787%. It is clear to see that the weight of average_clustering_coefficient is more significant than that of other features, which means that this feature has the most outstanding contribution to the construction of the decision tree and can better represent the dynamic characteristics of the original time series. The weight of the diameter is the smallest, which means that the distinguishing degree of the feature is insufficient, and the difference in the diameter of the complex network is small, which ultimately leads to the insufficient contribution of the feature in the process of constructing the random forest. By comparing Table 5 with Table 3, we found that the weights of

Table 5
AHP Analytic Hierarchy Process Results.

Feature	Eigenvector	Weight
average_point_intensity	0.415	4.613%
average_shortest_path_length	1.591	17.681%
diameter	0.354	3.929%
global_efficiency	0.530	5.894%
average_clustering_coefficient	3.183	35.861%
average_degree_assortativity_coefficient	0.616	6.843%
density	0.455	5.052%
modularity	0.796	8.840%
transitivity	1.061	11.787%

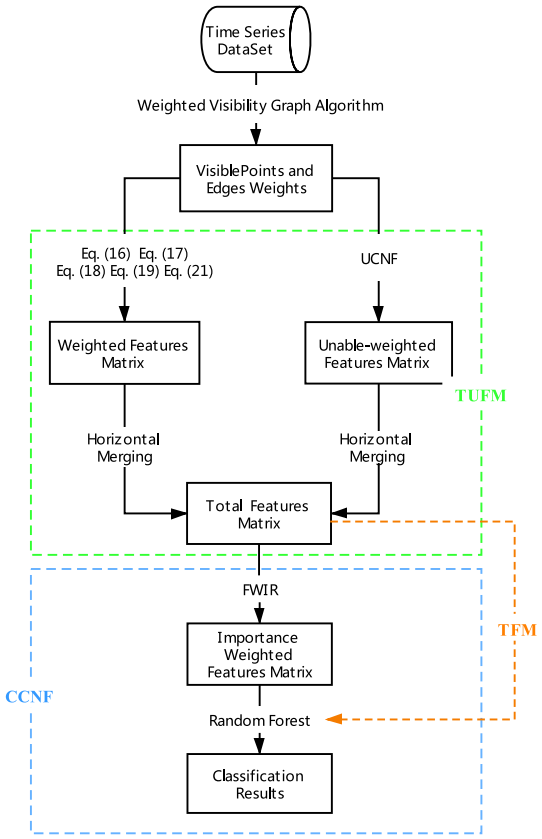


Fig. 11. CCNF procedure.

certain features are significantly different in the above two tables. For example, diameter ranks second in terms of weight in Table 3, but has the smallest weight in Table 5. For the four sample time series in the motestrain, the diameter can better distinguish the network, and the result of Table 5 is a comprehensive consideration of the importance ranking of the nine features of the 35 datasets. Since the importance of diameter in other datasets is not high, the weight of diameter is weakened when the mean value of the features importance ranking of all datasets is finally taken. This difference also shows that the network characteristics always behave different distinctions due to the structure of the complex network . Therefore, it is urgent to screen out features that are broadly distinguishable.

Further, we obtained the maximum eigenvalue $\lambda_{\max}$ by combining the eigenvectors $feature\_weight$ and then utilized the $\lambda_{\max}$ to calculate the CR value. The CR value is 0.001, which means that the judgment matrix O of this study meets the consistency test, and the calculated weights are consistent.

In general, by utilizing the importance function of the random forest and the analytic hierarchy process, we calculate the weight of each feature and lay the foundation for constructing the importance weighted features matrix.

4.5. Efficiency evaluation

According to the above description, extract weighted features, weighted features importance all occur during the classification process of the proposed algorithm. Whether or not these processes improve clustering results requires further verification. An effective algorithm flow should improve algorithm efficiency in each stage, meaning unnecessary calculations should be avoided. Therefore, we compared

Table 6
Accuracy of the classification results after different steps.

Dataset	TUFM	TFM	CCNF
Beef	0.467	0.500	**0.567**
Toesegment1	0.768	**0.850**	**0.850**
Coffee	0.679	**0.929**	**0.929**
DistalPhalanxOutlineAgeGroup	0.815	**0.838**	0.833
ECG200	0.770	0.840	**0.846**
GunPoint	0.853	**0.947**	**0.947**
Motestrain	0.821	0.838	**0.858**
ProximalPhalanxTW	0.758	0.788	**0.789**
ProximalPhalanxOutlineAgeGroup	0.801	0.849	**0.854**
Ham	0.619	0.676	**0.714**
DistalPhalanxOutlineCorrect	0.742	0.805	**0.807**
Herring	0.811	0.892	**0.893**
BeetleFly	**0.850**	**0.850**	**0.850**
Car	0.820	0.870	**0.881**
MiddlePhalanxOutlineAgeGroup	0.775	**0.780**	**0.780**
ShapeletSim	0.750	0.840	**0.856**
SonyAIBORobot Surface1	**0.882**	0.877	0.880
Earthquakes	0.798	0.804	**0.807**
Wine	0.980	**1.000**	**1.000**
SyntheticControl	0.440	0.670	**0.673**
MiddlePhalanxOutlineCorrect	0.697	0.855	**0.858**
ECGFiveDays	0.843	0.906	**0.907**
BirdChicken	0.800	0.800	**0.850**
ToeSegmentation2	0.846	**0.908**	**0.908**
TwoLeadECG	0.809	0.947	**0.948**
Plane	0.990	**1.000**	**1.000**
ArrowHead	0.691	0.680	**0.801**
CBF	0.520	0.687	**0.894**
DiatomSizeReduction	0.792	0.850	**0.875**
FaceFour	0.534	0.659	**0.887**
SonyAIBORobotSurface2	0.697	0.767	**0.869**
Symbols	0.725	0.800	**0.817**
ItalyPowerDemand	0.754	0.782	**0.861**
MiddlePhalanxTW	0.576	0.622	**0.694**
DistalPhalanxTW	0.720	0.773	**0.775**
mean	0.748	0.814	**0.845**
avg rank	2.629	1.829	**1.057**
win rate	0.057	0.257	**0.943**
var	0.016	0.012	**0.008**

the results of each phase of CCNF, as shown in Fig. 11. (1) Total unweighted features matrix(TUFM): By calculating the five unweighted features average_degree, average_shortest_path_length, diameter, global_efficiency, and average_clustering_coefficient, we directly generate the unweighted features matrix. After combining the unweighted features matrix and the unable-weighted features matrix, we utilize random forest to classify the combined matrix and finally obtain the classification results. (2) Total features matrix(TFM): After merging the weighted features matrix and the unable-weighted features matrix, we directly classify the merged matrix as the initial data and no longer assign weights according to the features importance ranking. (3) CCNF: We add edges weights and features importance information to features matrix (this is the complete recommendation method).

As showed in Table 6, the proposed algorithm achieves performance improvements at all stages. By extracting five weighted features and obtaining a weighted features matrix, the classification accuracy increases by almost nine percent. Adding features weights also improves the accuracy of the final result. For most datasets, reconsidering the weight of each feature in the features matrix can contribute to increasing the impact of certain features in the classification process. After utilizing the importance weighted features matrix as the original data of the random forest, the classification accuracy of the two datasets, BirdChicken and beef, has improved the most. In contrast, the classification accuracy of DistalPhalanxOutlineAgeGroup has declined. We consider that the ranking of average importance is more in line with the respective importance rankings of the first two datasets, thereby reasonably magnifying the impact of some features in the classification process. For the latter dataset, the average importance ranking is quite

Table 7
Accuracy of different classification algorithms implemented in last phase.

Dataset	IWFM-CART	IWFM-KNN	IWFM-SVM	CCNF
Beef	0.333	0.367	0.333	**0.567**
Toesegment1	**0.855**	0.750	0.821	0.850
Coffee	0.821	0.857	0.857	**0.929**
DistalPhalanxOutlineAgeGroup	0.808	0.800	0.818	**0.833**
ECG200	0.730	0.750	0.780	**0.846**
GunPoint	0.820	0.887	0.920	**0.947**
Motestrain	0.777	0.841	0.847	**0.858**
ProximalPhalanxTW	0.715	0.783	0.783	**0.789**
ProximalPhalanxOutlineAgeGroup	0.777	0.780	0.732	**0.854**
Ham	0.648	0.610	0.629	**0.714**
DistalPhalanxOutlineCorrect	0.708	0.690	0.782	**0.807**
Herring	0.745	0.680	0.724	**0.893**
BeetleFly	**0.850**	0.804	0.750	**0.850**
Car	0.732	0.820	0.583	**0.881**
MiddlePhalanxOutlineAgeGroup	0.763	0.745	0.773	**0.780**
ShapeletSim	0.667	0.689	0.733	**0.856**
SonyAIBORobotSurface1	0.847	0.731	0.754	**0.880**
Earthquakes	**0.820**	0.755	**0.820**	0.807
Wine	0.960	**1.000**	**1.000**	**1.000**
SyntheticControl	0.573	0.640	0.620	**0.673**
MiddlePhalanxOutlineCorrect	0.678	0.632	0.675	**0.858**
ECGFIVEDays	0.600	0.700	0.851	**0.907**
BirdChicken	0.600	0.300	**0.850**	**0.850**
ToeSegmentation2	0.808	0.569	0.877	**0.908**
TwoLeadECG	0.889	0.341	**0.960**	0.948
Plane	0.877	0.981	0.981	**1.000**
ArrowHead	0.560	0.640	0.723	**0.801**
CBF	0.591	0.639	0.657	**0.894**
DiatomSizeReduction	0.733	0.800	0.779	**0.875**
FaceFour	0.843	0.716	0.625	**0.887**
SonyAIBORobotSurface2	0.774	0.720	0.738	**0.869**
Symbols	0.740	0.722	0.785	**0.817**
ItalyPowerDemand	0.774	0.716	0.764	**0.861**
MiddlePhalanxTW	0.607	0.541	0.607	**0.694**
DistalPhalanxTW	0.723	0.670	**0.783**	0.775
mean	0.736	0.705	0.763	**0.845**
avg rank	2.771	3.057	2.343	**1.114**
win rate	0.057	0.029	0.143	**0.886**
var	0.015	0.023	0.016	**0.008**

different from the respective importance rankings, which weakens the influence of the important original features in the classification process, resulting in a decline in the classification accuracy.

In addition, we considered different classification algorithms in the final phase. The traditional classifiers CART, KNN, and SVM are implemented and compared with approaches based on random forest. The comparing methods are namely as importance weighted features matrix IWFM-CART, importance weighted features matrix IWFM-KNN, and importance weighted features matrix IWFM-SVM. The experimental results are listed in Table 7. In Table 7, it is easy to see that the application of random forest works best, and the accuracy index is higher than comparing algorithms. Even so, when it comes to different fields of application, different classification algorithms can still adapt to the calculation framework of CCNF.

4.6. Time complexity analysis

Let b be the number of time series in a dataset and let n be the length of the longest series, k is the number of visible points, p is the number of features. The time complexity of CCNF is the sum of visible points calculation, edge weights calculation, weighted features calculation, unable-weighted features calculation, and result computation. The visible points calculation operation for handling all objects has a complexity of approximately $O\left(bn^2\right)$. Calculating the edges' weights has a complexity of approximately $O(k)$. Obtaining the weighted features requires $O\left(bn^2\right)$. Calculating the unable-weighted features requires $O(k)$. Result computation requires $O\left(ntree\left(blogb\right)\right)$. The total time complexity is approximately $O\left(bn^2 + k + ntree(b\log b)\right)$.

Table 8
Time complexities of different algorithms.

	Time complexity
mWDN	$O(n\log(n)+n)$
SAX-VFSEQL	$O(bn)$
FSH	$O(bn^2)$
CDTW_1NN	$O(b^2+n^2)$
CCNF	$O\left(bn^2 + k + ntree(b\log b)\right)$

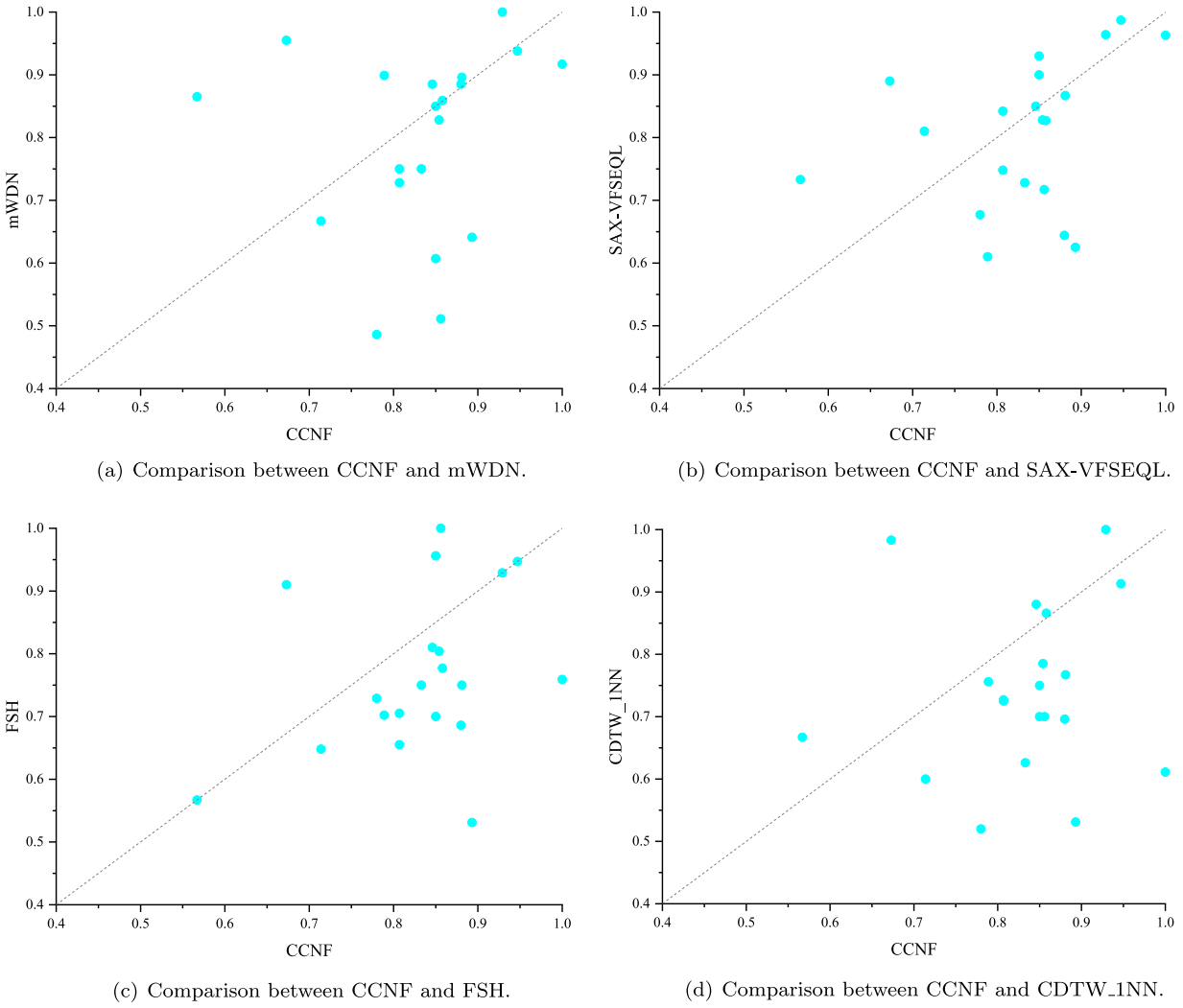
Table 9
Accuracy of the classification results of different algorithms.

Dataset	mWDN	SAX-VFSEQL	FSH	CDTW_1NN	CCNF
Beef	**0.865**	0.733	0.567	0.667	0.567
Toesegment1	0.607	0.930	**0.956**	0.750	0.850
Coffee	**1.000**	0.964	0.929	**1.000**	0.929
DistalPhalanxOutlineAgeGroup	0.750	0.728	0.750	0.626	**0.833**
ECG200	**0.885**	0.850	0.810	0.880	0.846
GunPoint	0.938	**0.987**	0.947	0.913	0.947
Motestrain	0.859	0.827	0.777	**0.866**	0.858
ProximalPhalanxTW	**0.899**	0.610	0.702	0.756	0.789
ProximalPhalanxOutlineAgeGroup	0.828	0.828	0.804	0.785	**0.854**
Ham	0.667	**0.810**	0.648	0.600	0.714
DistalPhalanxOutlineCorrect	0.728	**0.842**	0.655	0.725	0.807
Herring	0.641	0.625	0.531	0.531	**0.893**
BeetleFly	0.850	**0.900**	0.700	0.700	0.850
Car	**0.896**	0.867	0.750	0.767	0.881
MiddlePhalanxOutlineAgeGroup	0.486	0.677	0.729	0.520	**0.780**
ShapeletSim	0.511	0.717	**1.000**	0.700	0.856
SonyAIBORobotSurface1	**0.885**	0.644	0.686	0.696	0.880
Earthquakes	0.750	0.748	0.705	0.727	**0.807**
Wine	0.917	0.963	0.759	0.611	**1.000**
SyntheticControl	0.955	0.890	0.910	**0.983**	0.673
MiddlePhalanxOutlineCorrect	0.743	0.546	0.546	0.766	**0.858**
ECGFIVEDays	0.955	0.955	**0.998**	0.79	0.907
BirdChicken	0.850	**1.000**	0.750	0.700	0.850
ToeSegmentation2	0.836	0.862	0.692	**0.908**	**0.908**
TwoLeadECG	0.923	**0.948**	**0.925**	**0.868**	**0.948**
Plane	0.969	0.991	**1.000**	**1.000**	**1.000**
ArrowHead	0.800	0.789	0.594	0.800	**0.801**
CBF	0.853	0.957	0.940	**0.996**	0.894
DiatomSizeReduction	**0.987**	0.866	0.866	0.935	0.875
FaceFour	0.875	**0.932**	0.909	0.886	0.887
SonyAIBORobotSurface2	0.857	0.816	0.790	0.859	**0.869**
Symbols	0.889	0.887	0.934	**0.938**	0.817
ItalyPowerDemand	**0.966**	0.816	0.917	0.955	0.861
MiddlePhalanxTW	0.549	0.487	0.533	0.507	**0.694**
DistalPhalanxTW	0.672	0.604	0.626	0.633	**0.775**
avg rank	2.486	2.800	3.257	2.971	**2.171**
win rate	0.229	0.200	0.114	0.200	**0.400**
var	0.019	0.019	0.021	0.021	**0.008**

Therefore, the greater the length and the number of the time series, the faster the complexity of the CCNF algorithm will grow. The complexity comparisons of the benchmark algorithms are summarized in Table 8. In Table 8, it is obvious that due to the existence of the time length, the number of objects, and the number of trees, the time efficiency of the algorithms is difficult to see intuitively from the time complexities analysis. But the highest order of the time complexity of the proposed algorithm is the quadratic term, so as the size of the dataset increases, the new algorithm requires longer calculation time. The reason for the high time complexities of the CDTW_1NN, FSH, and CCNF may be the need to traverse the original time series. Among all the steps included in the CCNF, the complexity of complex network transformation is the highest, which is also a key issue in applying the visibility graph algorithm to large-scale data.

4.7. Clustering comparisons

To verify the performance of the proposed algorithm, we compared it to several other algorithms, namely mWDN (Wang et al., 2018), SAX-VFSEQL (Le Nguyen et al., 2017b), FSH (Rakthanmanon & Keogh,



(a) Comparison between CCNF and mWDN.

(b) Comparison between CCNF and SAX-VFSEQL.

(c) Comparison between CCNF and FSH.

(d) Comparison between CCNF and CDTW_1NN.

Fig. 12. Comparisons between CCNF and other algorithms.

2013), and CDTW_1NN (Dau et al., 2018) on the 35 datasets. We have quoted the classification results in the Zhang et al. (2021), and have maintained complete consistency in the datasets selection and evaluation indicator, so there is no preference. These algorithms can be applied to UTS datasets with unequal lengths. A pairwise comparison of the accuracy index results is visualized in Fig. 12. The dashed line was added to make the comparison of the results of the different methods more obvious. A dotted line divides a graph into two diagonal regions, and the more points that the area corresponds to the better the performance of the algorithm. As Fig. 12 shows, the proposed CCNF method is more accurate than all other approaches. The accuracy result is listed in Table 9. The best approach for each dataset is highlighted in bold. Moreover, we also present number of winners, average rank, and variance for each approach. The superiority of our algorithm is obvious. CCNF achieves the highest accuracy on 14 datasets, and the average rank is also the highest. Among several relatively poor datasets, CCNF can obtain close to the best results on the GunPoint, Car, and SonyAIBORobotSurface1 datasets. However, for the ShapeletSim dataset, the FSH algorithm provides superior performance. Meanwhile, the other algorithms generated results similar to those of CCNF. This dataset may benefit from the fast discovering time series shapelets. Overall, the Classification results obtained by CCNF are superior in avg rank, win rate, and variance of accuracy.

5. Conclusions

In this article, we proposed a novel classification method. The proposed algorithm consists of four stages. In the first stage, we choose a new weighted visibility graph algorithm to convert the original time series into a complex network. In the second stage, we extract features from the complex network, including utilizing the NetworkX to obtain unable-weighted features matrix and utilizing the above equation to calculate the weighted features matrix. After that, we combine the two types of features matrix to obtain a total feature matrix. In the third stage, we utilize the importance function of random forest to calculate and rank the importance of each feature, and then utilize the analytic hierarchy process to obtain the features weights and assign them to the total features matrix to obtain the importance weighted features matrix. According to experimental results, these two stages can each improve classification results. Finally, we utilize the random forest classification method to obtain classification results. The main innovations of the proposed algorithm are as follows. (1) The application of a weighted visibility algorithm is extended from pure ECG data to public time series collection, and the performance of effectively reducing information loss in the process of complex network conversion is verified through comparative experiments. (2) Original time series information is converted into a low-dimensional feature space by constructing two types of features matrices. (3) The importance function of random

Table A.1
The symbols in the paper.

Symbols	Meaning
A	The adjacency matrix.
G	An complex network.
C	The average_clustering_ coefficient of G.
D	The diameter of G.
E	The global_efficiency of G.
Z	Time series dataset.
K	The average_degree of G.
L	The average_shortest_path_length of G.
M	The average_degree_assortativity_coefficient of G.
Q	The modularity of G.
T	The transitivity of G.
S	The average_point_intensity of G.
X	Test sample.
U	The number of categories in the time series dataset.
O	Judgment matrix.
I	The importance of p features.
b	The number of time series in the dataset.
n	The length of time series.
p	The number of features owned by the dataset.
x, y, z	The element of the time series.
k	The number of the visiblePoints of G.

forest is utilized to filter out those features that can better reflect the complex network. Due to the wide application of multivariate time series, we plan to extend the proposed CCNF method from single-dimensional time series to multi-dimensional time series in future work. At the same time, we are also looking for an effective way to improve the conversion complexity of complex networks, so that we can apply CCNF to large-scale data.

CRediT authorship contribution statement

Hailin Li: Conceptualization, Validation, Supervision. **Ruiying Jia:** Methodology, Software, Writing – original draft. **Xiaoji Wan:** Data curation, Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work has been supported by the National Natural Science Foundation of China (71771094, 61300139) and Project of Science and Technology Plan of Fujian Province of China (2019J01067).

Appendix

See Table A.1.

References

Albert, R., & Barabási, A.-L. (2002). Statistical mechanics of complex networks. *Reviews of Modern Physics*, *74*(1), 47.

Bai, B., Li, G., Wang, S., Wu, Z., & Yan, W. (2021). Time series classification based on multi-feature dictionary representation and ensemble learning. *Expert Systems with Applications*, *169*, 114–162.

Barabási, A.-L., & Albert, R. (1999). Emergence of scaling in random networks. *Science*, *286*(5439), 509–512.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32.

Brillinger, D. R. (2001). *Time series: Data analysis and theory*. SIAM.

Brito, S., Canabarro, A., Chaves, R., & Cavalcanti, D. (2020). Statistical properties of the quantum internet. *Physical Review Letters*, *124*(21), Article 210501.

Chai, R., & Yan, T. (2018). Improved granular feature weighted multi-label classification based on fuzzy C-means. *Computer Applications and Software*, *35*, 286–291.

Dau, H. A., Silva, D. F., Petitjean, F., Forestier, G., Bagnall, A., Mueen, A., & Keogh, E. (2018). Optimizing dynamic time warping's window width for time series data mining applications. *Data Mining and Knowledge Discovery*, *32*(4), 1074–1120.

De Arruda, H. F., Costa, L. d. F., & Amancio, D. R. (2016). Using complex networks for text classification: Discriminating informative and imaginative documents. *EPL (Europhysics Letters)*, *113*(2), 28007.

del Campo, F. A., Neri, M. C. G., Villegas, O. O. V., Sánchez, V. G. C., Domínguez, H. d. J. O., & Jiménez, V. G. (2021). Auto-adaptive multilayer perceptron for univariate time series classification. *Expert Systems with Applications*, *181*, 115–147.

Deng, H., Chen, W., Shen, Q., Ma, A. J., Yuen, P. C., & Feng, G. (2020). Invariant subspace learning for time series data based on dynamic time warping distance. *Pattern Recognition*, *102*, Article 107210.

Egaji, O. A., Evans, G., Griffiths, M. G., & Islas, G. (2021). Real-time machine learning-based approach for pothole detection. *Expert Systems with Applications*, *184*, Article 115562.

Erd, F. C., Vignatti, A. L., & da Silva, M. V. (2021). The generalized influence blocking maximization problem. *Social Network Analysis and Mining*, *11*(1), 1–17.

Geler, Z., Kurbalija, V., Ivanović, M., & Radovanović, M. (2020). Weighted kNN and constrained elastic distances for time-series classification. *Expert Systems with Applications*, *162*, Article 113829.

Gregoriades, A., Pampaka, M., Herodotou, H., & Christodoulou, E. (2021). Supporting digital content marketing and messaging through topic modelling and decision trees. *Expert Systems with Applications*, *184*, Article 115546.

He, L., Agard, B., & Trépanier, M. (2020). A classification of public transit users with smart card data based on time series distance metrics and a hierarchical clustering method. *Transportmetrica A: Transport Science*, *16*(1), 56–75.

He, Z., Long, S., Ma, X., & Zhao, H. (2020). A boundary distance-based symbolic aggregate approximation method for time series data. *Algorithms*, *13*(11), 284.

Jackins, V., Vimal, S., Kaliappan, M., & Lee, M. Y. (2021). AI-based smart prediction of clinical disease using random forest classifier and naive Bayes. *The Journal of Supercomputing*, *77*(5), 5198–5219.

Jain, B. (2021). Warped softmax regression for time series classification. *Knowledge and Information Systems*, *63*(3), 589–619.

Jiang, Z., Luo, G., & Shen, K. (2016). Complex networks clustering for lower power scan segmentation in at-speed testing. *IEICE Transactions on Electronics*, *99*(9), 1071–1079.

Johnpaul, C., Prasad, M. V., Nickolas, S., & Gangadharan, G. (2021). Representational primitives using trend based global features for time series classification. *Expert Systems with Applications*, *167*, Article 114376.

Kushwah, V., Wadhvani, R., & Kushwah, A. K. (2020). Trend-based time series data clustering for wind speed forecasting. *Wind Engineering*, *45*(4), 992–1001.

Lacasa, L., Luque, B., Ballesteros, F., Luque, J., & Nuno, J. C. (2008). From time series to complex networks: The visibility graph. *Proceedings of The National Academy of Sciences*, *105*(13), 4972–4975.

Lacasa, L., Nicosia, V., & Latora, V. (2015). Network structure of multivariate time series. *Scientific Reports*, *5*(1), 1–9.

Lahreche, A., & Boucheham, B. (2021). A fast and accurate similarity measure for long time series classification based on local extrema and dynamic time warping. *Expert Systems with Applications*, *168*, Article 114374.

Le Nguyen, T., Gsponer, S., & Ifrim, G. (2017a). Time series classification by sequence learning in all-subsequence space. In *2017 IEEE 33rd international conference on data engineering* (ICDE), (pp. 947–958). IEEE.

Le Nguyen, T., Gsponer, S., & Ifrim, G. (2017b). Time series classification by sequence learning in all-subsequence space. In *2017 IEEE 33rd international conference on data engineering* (pp. 947–958). IEEE.

Li, H., & Du, T. (2021). Multivariate time-series clustering based on component relationship networks. *Expert Systems with Applications*, *173*, Article 114649.

Li, H., & Liu, Z. (2021). Multivariate time series clustering based on complex network. *Pattern Recognition*, *115*, Article 107919.

Li, H., Liu, J., Yang, Z., Liu, R. W., Wu, K., & Wan, Y. (2020). Adaptively constrained dynamic time warping for time series classification and clustering. *Information Sciences*, *534*, 97–116.

Li, H., & Wei, M. (2020). Fuzzy clustering based on feature weights for multivariate time series. *Knowledge-Based Systems*, *197*, Article 105907.

Li, H., Wu, Y. J., & Chen, Y. (2020). Time is money: Dynamic-model-based time series data-mining for correlation analysis of commodity sales. *Journal of Computational and Applied Mathematics*, *370*, Article 112659.

Li, J., Zhong, P.-a., Yang, M., Zhu, F., Chen, J., Liu, W., & Xu, S. (2020). Intelligent identification of effective reservoirs based on the random forest classification model. *Journal of Hydrology*, *591*, Article 125324.

Liang, L., Guo, K., & Sheng, X. (2020). Research on kernel function of support vector machine based on weighted feature subspace. *Science Technology and Engineering*, *20*, 6101–6106.

Lin, G., Lin, A., & Cao, J. (2021). Multidimensional KNN algorithm based on EEMD and complexity measures in financial time series forecasting. *Expert Systems with Applications*, *168*, Article 114443.

López-Oriona, A., & Vilar, J. A. (2021). Quantile cross-spectral density: A novel and effective tool for clustering multivariate time series. *Expert Systems with Applications*, *185*, Article 115677.

Ma, Q., Zheng, Z., Zhuang, W., Chen, E., Wei, J., & Wang, J. (2021). Echo memory-augmented network for time series classification. *Neural Networks*, *133*, 177–192.

Mota, N. B., Copelli, M., & Ribeiro, S. (2017). Thought disorder measured as random speech structure classifies negative symptoms and schizophrenia diagnosis 6 months in advance. *Npj Schizophrenia*, *3*(1), 1–10.

Newman, M. E. (2003). The structure and function of complex networks. *SIAM Review*, *45*(2), 167–256.

Rakthanmanon, T., & Keogh, E. (2013). Fast shapelets: A scalable algorithm for discovering time series shapelets. In *Proceedings of the 2013 SIAM international conference on data mining* (pp. 668–676). SIAM.

Resende, V. H., & Carneiro, M. G. (2019). Towards a high-level multi-label classification from complex networks. In *2019 IEEE 31st international conference on tools with artificial intelligence* (pp. 1140–1147). IEEE.

Ruiz-Herrera, A., & Torres, P. J. (2021). The role of movement patterns in epidemic models on complex networks. *Bulletin of Mathematical Biology*, *83*(10), 1–13.

Saaty, T. L. (2004). Decision making—the analytic hierarchy and network processes (AHP/ANP). *Journal of Systems Science and Systems Engineering*, *13*(1), 1–35.

Senin, P., & Malinchik, S. (2013). Sax-vsm: Interpretable time series classification using sax and vector space model. In *2013 IEEE 13th international conference on data mining* (pp. 1175–1180). IEEE.

Shirado, H., & Christakis, N. A. (2017). Locally noisy autonomous agents improve global human coordination in network experiments. *Nature*, *545*(7654), 370–374.

Shokrzade, A., Ramezani, M., Tab, F. A., & Mohammad, M. A. (2021). A novel extreme learning machine based kNN classification method for dealing with big data. *Expert Systems with Applications*, *183*, Article 115293.

Song, K., Ryu, M., & Lee, K. (2020). Transitional SAX representation for knowledge discovery for time series. *Applied Sciences*, *10*(19), 6980.

Sporns, O. (2011). The human connectome: a complex network. *Annals of The New York Academy of Sciences*, *1224*(1), 109–125.

Stam, C. J., & Reijneveld, J. C. (2007). Graph theoretical analysis of complex networks in the brain. *Nonlinear Biomedical Physics*, *1*(1), 1–19.

Sun, Y., Li, J., Liu, J., Sun, B., & Chow, C. (2014). An improvement of symbolic aggregate approximation distance measure for time series. *Neurocomputing*, *138*, 189–198.

Sun, Z., Sun, Y., Chang, X., Wang, Q., Yan, X., Pan, Z., & Li, Z.-p. (2020). Community detection based on the matthew effect. *Knowledge-Based Systems*, *205*, Article 106256.

Supriya, S., Siuly, S., Wang, H., Cao, J., & Zhang, Y. (2016). Weighted visibility graph with complex network features in the detection of epilepsy. *IEEE Access*, *4*, 6554–6566.

Tavakoli, N., Siami-Namini, S., Khanghah, M. A., Soltani, F. M., & Namin, A. S. (2020). An autoencoder-based deep learning approach for clustering time series data. *SN Applied Sciences*, *2*(5), 1–25.

Thakkar, A., & Chaudhari, K. (2021). A comprehensive survey on deep neural networks for stock market: The need, challenges, and future directions. *Expert Systems with Applications*, *177*, Article 114800.

Wang, X., Mueen, A., Ding, H., Trajcevski, G., Scheuermann, P., & Keogh, E. (2013). Experimental comparison of representation methods and distance measures for time series data. *Data Mining and Knowledge Discovery*, *26*(2), 275–309.

Wang, J., Wang, Z., Li, J., & Wu, J. (2018). Multilevel wavelet decomposition network for interpretable time series analysis. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 2437–2446).

Xin, R., Zhang, J., & Shao, Y. (2020). Complex network classification with convolutional neural network. *Tsinghua Science and Technology*, *25*(4), 447–457.

Zhang, J., & Small, M. (2006). Complex network from pseudoperiodic time series: Topology versus dynamics. *Physical Review Letters*, *96*(23), Article 238701.

Zhang, J., Sun, J., Luo, X., Zhang, K., Nakamura, T., & Small, M. (2008). Characterizing pseudoperiodic time series through the complex network approach. *Physica D: Nonlinear Phenomena*, *237*(22), 2856–2865.

Zhang, H., Wang, P., Fang, Z., Wang, Z., & Wang, W. (2021). ELIS++: a shapelet learning approach for accurate and efficient time series classification. *World Wide Web*, *24*(2), 511–539.

Zhao, L., Sun, W., Sun, Z., & chen, Q. (2018). Brake pad image classification algorithm based on color segmentation and information entropy weighted feature matching. *Journal of Tsinghua University*, *58*, 547–552.

Zou, Y., Donner, R. V., Marwan, N., Donges, J. F., & Kurths, J. (2019). Complex network approaches to nonlinear time series analysis. *Physics Reports*, *787*, 1–97.