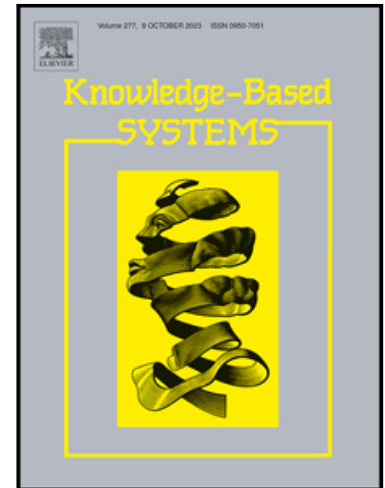


DSDCLNet: Dual-Stream Encoder and Dual-Level Contrastive Learning Network for Supervised Multivariate Time Series Classification

Min Liu , Hui Sheng , Ningyi Zhang , Panpan Zhao , Yugen Yi ,
Yirui Jiang , Jiangyan Dai

PII: S0950-7051(24)00273-9
DOI: <https://doi.org/10.1016/j.knosys.2024.111638>
Reference: KNOSYS 111638



To appear in: *Knowledge-Based Systems*

Received date: 21 September 2023
Revised date: 6 February 2024
Accepted date: 12 March 2024

Please cite this article as: Min Liu , Hui Sheng , Ningyi Zhang , Panpan Zhao , Yugen Yi ,
Yirui Jiang , Jiangyan Dai , DSDCLNet: Dual-Stream Encoder and Dual-Level Contrastive Learning
Network for Supervised Multivariate Time Series Classification, *Knowledge-Based Systems* (2024),
doi: <https://doi.org/10.1016/j.knosys.2024.111638>

This is a PDF file of an article that has undergone enhancements after acceptance, such as the addition of a cover page and metadata, and formatting for readability, but it is not yet the definitive version of record. This version will undergo additional copyediting, typesetting and review before it is published in its final form, but we are providing this version to give early visibility of the article. Please note that, during the production process, errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

**DSDCLNet: Dual-Stream Encoder and Dual-Level Contrastive Learning
Network for Supervised Multivariate Time Series Classification**

Min Liu ¹, Hui Sheng ¹, Ningyi Zhang ¹, Panpan Zhao ¹, Yugen Yi ^{1*},
Yirui Jiang ², Jiangyan Dai ^{3*}

¹ School of Software, Jiangxi Normal University, Nanchang, 330022, China

² School of Water, Energy and Environment, Cranfield University, Bedfordshire
MK43 0AL, UK

³ School of Computer Engineering, Weifang University, Weifang, 261061, China

* Corresponding authors: yiyg510@jxnu.edu.cn and daijy@wfu.edu.cn

Abstract: In recent years, deep learning approaches have shown remarkable advancements in multivariate time series classification (MTSC) tasks. However, the existing approaches primarily focus on capturing the long-term correlations of time series or identifying local key sequence fragments, inevitably neglecting the synergistic properties between global and local components. Additionally, most representation learning methods for MTSC rely on self-supervised learning, which limits their ability to fully exploit label information. Hence, this paper proposes a novel approach termed Dual-Stream Encoder and Dual-Level Contrastive Learning Network (DSDCLNet), which integrates a dual-stream encoder (DSE) and dual-level contrastive learning (DCL). First, to extract multiscale local-global features from multivariate time series data, we employ a DSE architecture comprising an attention-gated recurrent unit (AGRU) and a dual-layer multiscale convolutional neural network (DMSCNN). Specifically, DMSCNN consists of a series of multi-scale convolutional layers and a max pooling layer. Second, to maximize the utilization of label information, a new loss function is designed, which combines classification loss, instance-level contrastive loss, and temporal-level contrastive loss. Finally, experiments are conducted on the UEA datasets and the results demonstrate that DSDCLNet achieves the highest average accuracy of 0.77, outperforming traditional approaches, deep learning approaches, and self-supervised approaches on 30, 23, and 27 datasets, respectively.

Keywords: Multivariate time series classification; Gated recurrent unit; Multiscale convolutional neural network; Dual-stream encoder; Contrastive learning.

1. Introduction

With the development of sensor technology and data acquisition capability, a large amount of multivariate time series (MTS) data containing the temporal evolution process has been collected. MTS refers to a sequence in which multiple correlated variables or features are observed and recorded at different timestamps. Consequently, the exploration of valuable insights within MTS has garnered significant interest in the fields of data mining and machine learning. MTSC presents a formidable challenge in analyzing time series data in diverse domains, such as industrial issue diagnosis [1], medical ailment identification [2], and financial market forecasting [3]. The MTSC is distinct from other classification tasks since it demands to handle inter-correlations and temporal dependencies among variables in MTS simultaneously [4]. Therefore, the core challenge lies in effectively leveraging the local characteristics of these variables while capturing their interactions and temporal dependencies during the classification model construction.

Over the past few decades, MTSC methods have continuously emerged, which is primarily categorized into two main groups: traditional approaches and deep learning approaches. Traditional methods can be further classified into two primary types: distance-based methods and feature-based methods [5]. Distance-based methods normally operate under the assumption that similar time series will exhibit proximity in the feature space. The core of these methods is to calculate the distance or similarity between different time series. For instance, the combination of Dynamic Time Warping (DTW) and Nearest Neighbor represents a widely recognized distance-based classification algorithm [6]. In contrast to distance-based methods, feature-based methods initially extract distinct features from time series data and subsequently employ these extracted features for classification. For example, the WEASEL+MUSE method, which relies on symbolic Fourier approximation, is a classical feature-centric methodology [7]. This method initially transforms the time series data into a sequence of symbols. Subsequently, it utilized Fourier transform to extract relevant features from the symbolic sequence. Finally, these features are then harnessed to perform

classification tasks. Both of these methodological approaches play significant roles in MTSC tasks but come with certain limitations. Specifically, distance-based methods encounter alignment issues when they are applied to lengthy sequences, and they may also increase the computational complexity for high-dimensional data. Feature-based methods require manual feature selection with inevitable subjectivity and adaptive differences that may lose detailed information [8].

Due to the powerful representational capacity and automatic feature learning capability of deep learning models, many deep learning-based MTSC methods have been presented to learn more abstract and meaningful representations [9] from MTS data. For instance, as representative methods, MLSTM-FCN [10] and TapNet [11] have been proven that they can address MTSC tasks effectively. However, their ability to extract spatio-temporal information in MTS is considerably restricted. Recent research has turned to representation learning, whose target is to transform complex time series data into low-dimensional, discriminative representations. Consequently, different classes of time series can be better distinguished in the representation space. Since contrastive learning can learn invariant representations, it achieved satisfactory performance in MTSC. The central concept of contrastive learning is that samples from the same category are closer together and samples from distinct categories farther apart. Recently, some contrastive learning based MTSC models have been proposed, such as TS2Vec [12] and MICOS [13]. TS2Vec is an unsupervised method, which conducted contrastive learning in a hierarchical manner over augmented context views, providing robust contextual representation for each timestamp. Building upon TS2Vec, MICOS introduced a mixed self-supervised, intra-class, and inter-class supervised contrastive learning approach for representation learning in MTS. However, both of them focus on extracting local information through convolutional neural networks (CNN) and cannot maximize the use of label information. Furthermore, their classification performance depends on the data augmentation.

To solve the above challenges, this paper introduces an innovative approach termed Dual-Stream Encoder and Dual-Level Contrastive Learning Network (DSDCLNet), leveraging the strengths of dual-stream encoder (DSE) and dual-level

contrastive learning (DCL). The DSE structure includes an attention-gated recurrent unit (AGRU) and a dual-layer multiscale convolutional neural network (DMSCNN). The AGRU module consists of a gated recurrent unit (GRU) and an attention mechanism that can capture sequential patterns and long-term dependencies in MTS to focus on more discriminative information. DMSCNN is a network architecture that includes two tiers of Multi-Scale Convolutional Neural Network (MSCNN). This architecture is designed to exploit multi-scale local and global information from time series data. Then, to effectively utilize the labeling information, we introduce the DCL module, in which the loss is a weighted sum of classification loss, instance-level contrastive loss, and temporal-level contrastive loss. Instance-level contrastive learning (ICL) considers samples from the same class as positive samples. In Temporal-level Contrastive Learning (TCL), different representations of the same sequence at the same time step are treated as positive samples, while negative samples consist of different sequences belonging to the same class. This construction compels the model to make the feature representations of same time steps more similar, and those of different time steps or sequences more dispersed, aiding in capturing temporal dynamic changes. Consequently, DSDCLNet can extract rich feature information from MTS with scant domain knowledge and does not need to generate contrastive learning sample pairs through complicated data augmentation [14]. In summary, the main contributions are as follows:

- (1) Introduction of an innovative MSCNN architecture capable of efficiently extracting both global and local feature information.
- (2) Development of a dual-stream structure, combining the strengths of AGRU and DMSCNN, designed to capture intricate temporal relationships inherent in MTS data.
- (3) Introduction of a DCL framework that leverages data labels to contrast samples at both the instance level and the temporal level, leading to enhanced classification performance.
- (4) Extensive experimental evaluations demonstrating the superior performance of DSDCLNet compared to several state-of-the-art methods. Ablation studies further

underline the advantages of the dual-level contrastive loss approach.

Table 1 provides the full names corresponding to the abbreviations used in this paper.

Table 1 Abbreviations and full names of methods and modules.

Abbreviation	Full Name
MTS	Multivariate Time Series
MTSC	Multivariate Time Series Classification
DSE	Dual-Stream Encoder
DCL	Dual-level Contrastive Learning
AGRU	Attention-Gated Recurrent Unit
DMSCNN	Dual-layer MultiScale Convolutional Neural Network
DTW	Dynamic Time Warping
CNN	Convolutional Neural Network
MSCNN	MultiScale Convolutional Neural Network
GRU	Gated Recurrent Unit
ICL	Instance-level Contrastive Learning
TCL	Temporal-level Contrastive Learning
RNN	Recurrent Neural Network
BN	Batch Normalization
ReLU	Rectified Linear Unit

2. Related Work

This section provides a brief overview of the latest developments in MTSC tasks.

2.1 Problem Definition

Definition 1 (Multivariate time series, MTS): An MTS $x = \{s_1, s_2, \dots, s_T\} \in R^{T \times V}$ is a data sequence consisting of multiple variables or features observed simultaneously in chronological order, where $s_t \in R^V$, V denotes the number of variables, t represents the timestamp of the current moment, and T is the length of the time series.

Definition 2 (Multivariate time series classification, MTSC): Given an MTS dataset containing N samples denoted as $X = \{x_1, x_2, \dots, x_N\} \in R^{N \times T \times V}$ with corresponding labels $Y = \{y_1, y_2, \dots, y_N\}$, where N denotes the number of samples. The MTSC task is to train a classifier $f(X) \rightarrow Y$ to predict the class labels of MTS.

2.2 Multivariate time series classification

MTSC methods can be classified into three primary groups: distance-based, feature-based, and deep learning-based. DTW [15] was considered as one of the most crucial similarity measurement methods. Mei et al. [16] first developed a metric learning model based on LogDet divergence and triplet constraint to learn the Mahalanobis matrix. Shokoohi-Yekta et al. [17] introduced an innovative independent and dependent DTW strategies by dynamically selecting distances based on training data thresholds and employing a scoring function for decision-making. Among feature-based methods, Li et al. [18] proposed a common principal component analysis approach to extract low-dimensional features from MTS data. Baydogan et al. [19] devised a symbolic MTS representation that utilized a tree-based symbol generation strategy to measure the interrelationships among all attributes of MTS. Zhao et al. [20] introduced a regularized shapelet learning framework, enhancing time series classification speed and accuracy through the inclusion of fused lasso regularization. Although the above methods have been successfully applied to MTSC tasks, they still cannot avoid extensive data preprocessing and feature engineering. Moreover, capturing time-dependent and multi-dimensional interaction information is challenging.

Currently, deep learning-based approaches [21] have gained prominence for addressing MTSC tasks. These approaches leverage neural network models to acquire data representations from MTS data, among which CNN [22] and Recurrent Neural Network (RNN) [23] are the most widely used models. Wu et al. [24] proposed a network called MCTNet, which combines CNN and Transformer encoders, addressing the data-intensive nature of Transformer through a squeeze and excitation convolution encoder structure. MCTNet leveraged the unique strengths of CNN and self-attention to more accurately capture the underlying deep information in MTS. MLSTM-FCN [10] enhanced the performance by combining Long Short-Term Memory Networks (LSTM) and CNN to extract long-term dependencies and local feature information without any data pre-processing operations. Chen et al. [25] proposed a dual-attention network known as DA-Net, which employed channel attention and self-attention mechanisms to

extract local and global features. Li et al. [26] presented ShapeNet, a model that employs a cluster-based ternary loss function to acquire distinctive shape features. However, while the methods mentioned above have demonstrated promising outcomes, they still suffer from time-intensive training, limitations in effectively capturing multi-scale local-global information, and suboptimal utilization of label information.

2.3 Contrastive Time Series Representation Learning

Contrastive learning has found broad applications in representation learning across diverse domains, yielding impressive outcomes. Furthermore, recent research endeavors have extended the use of contrastive learning to address the task of time series representation learning. For example, Yang et al. [27] introduced the TimeCLR framework, which combined the advantages of DTW and InceptionTime [28] for representation learning in one-dimensional time series. Contrastive learning has not only been applied to univariate time series but has also been widely applied in MTS. Pöppelbaum et al. [29] proposed a self-supervised learning method for time series analysis based on SimCLR contrastive learning [30]. Nonnenmacher et al. [31] introduced the ExpCLR algorithm for contrastive representation learning, utilizing continuous or discrete expert features. Eldele et al. [32] proposed a novel temporal and contextual contrast-based learning structure to learn representations from unlabeled data through contrastive learning. Zhang et al. [33] introduced a strategy to integrate time-frequency consistency into pre-training using self-supervised contrastive estimation, which made time-based and frequency-based representations and their local neighborhoods close in the latent space. Wickstrøm et al. [34] incorporated an augmentation technique to propose a self-supervised framework that merged data points to generate novel samples. TS2Vec [12] conducted contrastive learning in a hierarchical manner to learn representations for each timestamp. The Contrastive learning method is a self-supervised framework without making use of label information. Nevertheless, label information for MTS data is valuable to MTSC task. Therefore, label information is introduced into contrastive learning method to improve performance. Recently, MICOS [13] was developed to extract intricate features, which

utilized spatiotemporal channels and integrated supervised contrastive learning to maximize the utilization of label information. MICOS only incorporated self-supervised contrastive loss, inter-class and intra-class supervised contrastive losses together, but neglected classification loss, thereby not fully harnessing the potential of label information.

3. The proposed Method

In this section, the architecture of the Dual-Stream Encoder and Dual-Level Contrastive Learning Network (DSDCLNet) is first provided. Subsequently, the details of Dual-Stream Encoder Structure (DSE) and Dual-Level Contrastive Learning (DCL) are described respectively. Finally, the classification model and loss function of DSDCLNet are provided.

3.1 Overall Structure of DSDCLNet

Fig.1 illustrates the overall framework of DSDCLNet, which consists of three main modules: encoder, contrastive learning, and fusion classification. First, a Dual-Stream Encoder comprising an attention-gated recurrent unit (AGRU) block and a dual-layer multiscale convolutional neural network (DMSCNN) block is designed to extract multi-scale local-global features. In this encoder, the AGRU block utilizes both the gated recurrent unit (GRU) and an attention mechanism to capture sequential patterns and establish long-term dependencies to extract global features. Meanwhile, the DMSCNN block is used to obtain deeper local features, which employs a dual-layered multiscale convolutional neural network (MSCNN) approach to acquire multi-scale feature information. Specifically, an attention mechanism is utilized to integrate the outputs from various layers of MSCNN modules for further enhancing the representation ability of the learned feature. Then, the contrastive learning module is designed to enhance feature robustness through supervised contrastive learning at both the instance and time-step levels. Moreover, we design a loss function that combines the contrastive learning loss with the cross-entropy classification loss, resulting in a more robust feature representation learning process. Finally, a classification model is

employed to implement the MTSC task.

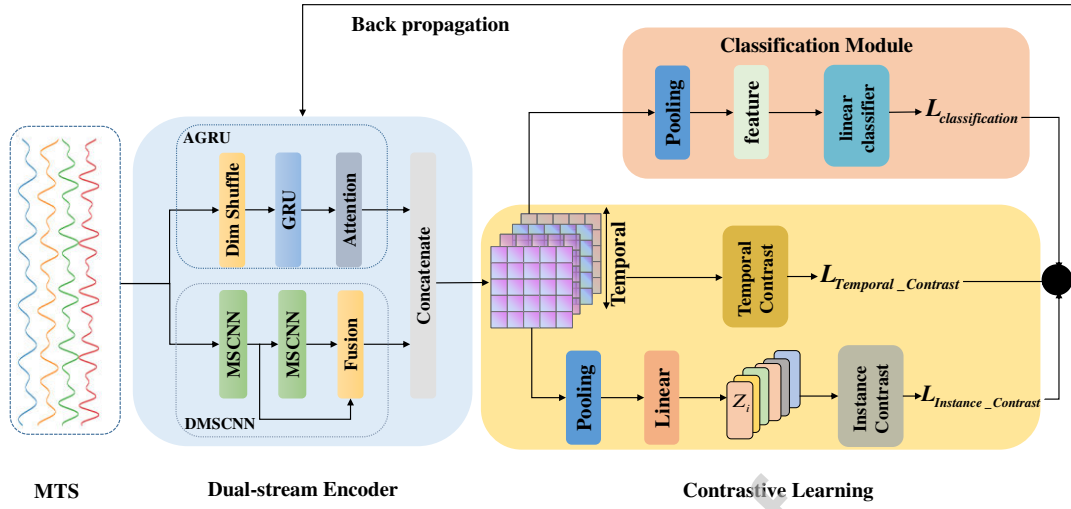


Fig. 1. The overall diagram of DSDCLNet.

3.2 Dual-Stream Encoder Structure

3.2.1 Attention-Gated Recurrent Unit

The Attention-Gated Recurrent Unit (AGRU) module is shown in Fig.1, which generates the feature representation by three stages, including matrix transposition, temporal coding, and attention weighting. First, the matrix transposition is performed on the input MTS data, as depicted in Eq. (1):

$$X' = F_{trans}(X) \quad (1)$$

where $F_{trans}(\cdot)$ denotes the permutation function. Then, $X' = [x'_1, x'_2, \dots, x'_T] \in R^{N \times V \times T}$ is input into the GRU module to obtain the feature representation $H = [h_1, h_2, \dots, h_T] \in R^{T \times d}$, where d is the dimension of the output. The detailed operations of GRU are shown in Eqs. (2) to (5) [35]:

$$z_t = \sigma(W_z \cdot x'_t + U_z \cdot h_{t-1} + b_z) \quad (2)$$

$$r_t = \sigma(W_r \cdot x'_t + U_r \cdot h_{t-1} + b_r) \quad (3)$$

$$h'_t = \tanh(W_h \cdot x'_t + U_h \cdot (r_t \square h_{t-1}) + b_h) \quad (4)$$

$$h_t = (1 - z_t) \square h_{t-1} + z_t \square h'_t \quad (5)$$

where x'_t is the input at the current time step, h_t represents the hidden state at the present step, z_t is the output of the update gate, r_t is the output of the reset gate, σ represents the Sigmoid function, $\square$ denotes element-wise multiplication, W_z , W_r , W_h , U_z , U_r , and U_h are the weight matrices, and b_z , b_r , b_h are the bias vectors. The GRU effectively captures temporal dependencies and contextual information of the input data by utilizing the update gate, reset gate, and hidden state.

Next, to capture the relationship between the various feature channels for improving the representation of MTS data, the feature representation H is sent into the attention module, which performs the adaptive weighting of the features from multiple channels. The attention module consists of two linear layers and a Softmax layer, which accomplishes the adaptive weighting of features from different channels. In this module, H generates the attention score M , which measures the importance of different channels through a two-level linear layer mapping, as shown in Eq. (6).

$$M = RELU(F_{laynorm}(F_{linear}(H))) \quad (6)$$

Then, the global attention weights for each channel are obtained by averaging operations over the time dimension of M' and the normalized operation is carried out by the softmax function [36], as follows:

$$W = Softmax(Mean(M')) \quad (7)$$

Finally, the output feature of AGRU module (Out_{AGRU}) is obtained by the dot-multiplication operation between H and the attention weight matrix as follows:

$$Out_{AGRU} = H \square W \quad (8)$$

Notably, the attention mechanism boosts the capacity of this model for more essential features.

3.2.2 Dual-layer Multiscale Convolutional Neural Network

The Dual-layer Multiscale Convolutional Neural Network (DMSCNN) module contains two MSCNN layers and a Fusion layer, where the initial layer takes X as the input. The MSCNN module is composed of three branches, and each branch is

constructed by a series of operations such as 2D convolution, 1D convolution, and max pooling, as shown in Fig.2.

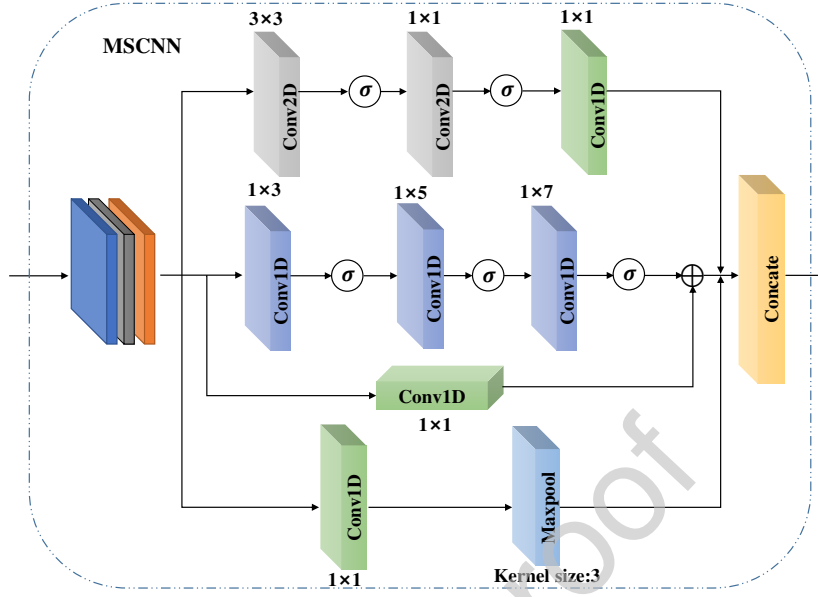


Fig. 2. The structure of MSCNN.

In the first branch, two 2D convolutions are first used to build the relationship between observed variables of MTS data, and the operation process of 2D convolution is shown in Fig. 3.

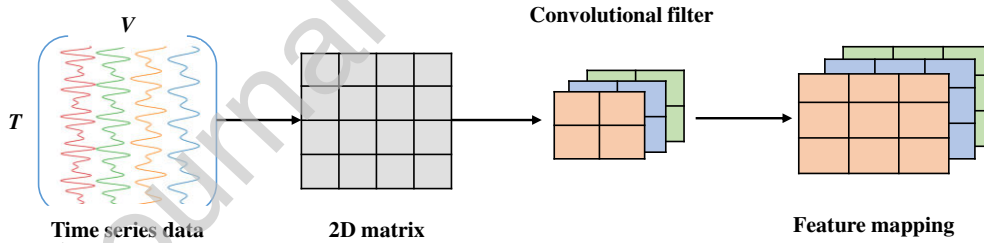


Fig.3. Schematic diagram of 2D convolution.

The multivariate time series data is first converted into a 2D matrix. Then, a 2D convolution operation, similar to that used for images, can be applied to generate a feature mapping of the time series data [37].

The output of 2D convolution can be described as follows:

$$Out_{con2d1} = RELU(BN(W_{1,2} \otimes RELU(BN(W_{1,1} \otimes F_{unsqueeze}(X) + b_{1,1})) + b_{1,2})) \quad (9)$$

where $F_{unsqueeze}(\square)$ denotes the dimension expansion function, $W_{1,1}$ and $W_{1,2}$, $b_{1,1}$ and $b_{1,2}$ are the weight and biases of different convolutional layers, $\otimes$ indicates the

convolutional operation. *BN* and *RELU* are the batch normalization and activation function, respectively. The introduction of nonlinear mapping makes the output features have higher discriminative ability. Then, a 1D convolution with size 1×1 is performed to achieve a more compact feature representation by reducing the feature dimensionality, as shown in Eq. (10):

$$Out_{con2d} = RELU(W_{1,3} \otimes Out_{con2d1} + b_{1,3}) \quad (10)$$

where $W_{1,3}$ and $b_{1,3}$ denote the weights and bias of the convolutional layer.

To further extract information between time steps, a multiscale one-dimensional convolution operation with the aim of capturing relationships among different time steps is employed to extract comprehensive temporal feature information, i.e., the second branch in Fig.2. Furthermore, to enhance the training efficiency and accelerate neural network convergence, a residual block is constructed by performing an element-level summation operation between the convolutional layer's output and channel-adjusted X . The process is depicted in Eq. (11):

$$Out_{1d} = W_{1d_3} \otimes (W_{1d_2} \otimes (W_{1d_1} \otimes X + b_{1d_1}) + b_{1d_2}) + b_{1d_3} + F_{1 \times 1}(X) \quad (11)$$

where W_{1d_1} , W_{1d_2} , and W_{1d_3} denote the weights of different convolutional layers, and b_{1d_1} , b_{1d_2} , and b_{1d_3} denote the bias terms of different convolutional layers. $F_{1 \times 1}$ is the 1×1 convolution of the adjusted channel.

In the third branch, a 1D convolution with size 1×1 is performed to adjust channel and then a maximum pooling operation is conducted to extract the global features, as shown in Eq. (12).

$$Out_{maxpool} = Maxpool(F_{1 \times 1}(X)) \quad (12)$$

Subsequently, we concatenate the output features of three branches as the output features of MSCNN, as shown in Eq. (13):

$$Out_{MSCNN} = Concatenate(Out_{con2d}, Out_{1d}, Out_{maxpool}) \quad (13)$$

where $Concatenate(\square)$ is the concatenation operation.

To efficiently integrate shallow and deep feature information, a fusion layer is

designed to produce the final output, which introduces the attention mechanism to weighted combine the outputs of MSCNN. The details are formulated in Eqs. (14) to (16):

$$h = W \cdot \text{Concate}(Out_{MSCNN1}, Out_{MSCNN2}) + b \quad (14)$$

$$g_1, g_2 = \text{Sigmoid}(h) \quad (15)$$

$$Out_{DMSCNN} = Out_{MSCNN1} \cdot g_1 + Out_{MSCNN2} \cdot g_2 \quad (16)$$

where Out_{MSCNN1} and Out_{MSCNN2} are represented as feature representations of two different MSCNN layers. Therefore, the Dual-Stream Encoder can encode the input multivariate time series data X into Out_{AGRU} and Out_{DMSCNN} .

3.3 Dual-Level Contrastive Learning

Similar to computer vision and natural language processing tasks, collecting and labelling the MTS data is intricate and expensive. Furthermore, the limited number of labeled training data will also lead to potential inductive bias in traditional deep learning networks [38]. To address this issue, a dual-level contrastive learning (DCL) is introduced to enhance the discriminability among feature representations of different classes. First, for the instance-level learning of rich class-specific features, the time series belonging to the same class within the same batch are regarded as positive samples. In contrast, time series of different classes are treated as negative samples. As shown in Eq. (17), minimizing the loss of instance-level contrastive learning (ICL) can make time series with the same label be close with each other while keeping a greater distance between time series with different labels [39].

$$L_{Instance_loss} = - \sum_{i=1}^B \frac{1}{B_{y_i} - 1} \sum_{j=1}^B l_{i \neq j} l_{y_i = y_j} \ln \left[\frac{\exp(cs_{i,j} / r)}{\exp(cs_{i,j} / r) + \sum_{k=1}^B l_{y_i \neq y_k} \exp(cs_{i,k} / r)} \right] \quad (17)$$

where B is the batch size, y_i and y_j denote the labels of anchored time series samples i and j , and B_{y_i} denotes the number of time series samples labelled as y_i in the same batch.

If $i \neq j$, then $l_{i \neq j} = 1$ and vice versa, $l_{i=j} = 0$. $l_{y_i = y_j}$ and $l_{y_i \neq y_k}$ are similarity indicator

functions. $cs_{i,j} = f_i^T f_j / (\|f_i\| \|f_j\|)$ is the cosine similarity between time series samples i and j . f_i and f_j are the low-dimensional features of the encoder outputs of samples i and j . r is the temperature hyperparameter.

ICL makes the model pull similar instances closer and push dissimilar instances apart in the learned feature space. It also promotes the model to learn more abundant and valid feature representations, which improves classification performance, especially in tasks with limited or complex data. Moreover, when considering clusters of similar instances, ICL has the potential to enhance the resilience of the model to input data variations and noise. Therefore, ICL accentuates both the similarities and differences within the data, contributing to improving the generalization capabilities of the model when dealing with unseen data.

Second, the uniqueness of time series data lies in the explicit temporal dependencies among data points. It implies that time-step data may be influenced by its preceding and succeeding time-step data. Therefore, temporal information revealing specific temporal patterns and dynamic changes is essential for categorization in the field of time series. Moreover, considering time series data as standalone samples may result in a partial utilization of the inherent temporal information. To address this issue and learn dynamic feature representations over time, we additionally applied a contrastive learning method called temporal-level contrastive learning (TCL). In TCL, different representations of the same multivariate time series at the same time step are considered as positive samples, while representations from different input multivariate time series belonging to the same class are treated as negative samples. can be defined as follows:

$$L_{Temporal_loss} = - \sum_{i=1}^B \frac{1}{B_{y_i} - 1} \sum_{j=1}^B l_{y_i=y_j} \log \frac{\exp(f_{i,t} \cdot f_{p,t})}{\sum_{y_i \neq y_j} \exp(f_{i,t} \cdot f_{j,t}) + \sum_{t \neq s} \exp(f_{i,t} \cdot f_{p,s})} \quad (18)$$

By minimizing the loss of TCL, the model can explore the associations and variations of the data in the temporal dimension.

Third, the loss of both ICL and TCL complements each other. For instance, if we have a time series dataset collected from various stocks, the loss of ICL can promote

the model to learn the unique characteristics of each stock, including aspects such as volatility, price fluctuations, and so on. The loss of TCL can help the model mine the dynamic trends of stock prices over time, such as cyclical fluctuations. Therefore, combining the two losses of TCL and ICL can improve the representation capacity of the model, as shown in Eq. (19):

$$L_{contraloss} = \sum_i L_{Instance_loss}^i + \sum_i \sum_t L_{Temporal_loss}^{(i,t)} \quad (19)$$

3.4 Classification Module

The two outputs of DSE are contacted to form feature representation, as follows:

$$Out_{Encoder} = Concat(Out_{AGRU}, Out_{DMSCNN}) \quad (20)$$

Then, it maps to a low-dimensional embedding representation using two linear layers, as follows:

$$Out_{Low_dim} = F_{Linear}(F_{Linear}(Out_{Encoder})) \quad (21)$$

Finally, a linear layer is utilized to obtain the classification result, as follows:

$$y' = F_{Linear}(Out_{Low_dim}) \quad (22)$$

3.5 Loss Function

The total loss function of DSDCLNet is formulated in Eq. (23):

$$L_{total} = \lambda \cdot L_{Contraloss} + L_{classification_loss} \quad (23)$$

where λ is the hyperparameter, and $L_{classification_loss}$ is defined as shown in Eq. (24):

$$L_{classification_loss} = \frac{1}{N} \sum_{i=1}^N \sum_{j=1}^M y_{i,j} (\log p(y'_{i,j})) \quad (24)$$

The implementation process of DSDCLNet is presented in Algorithm 1.

Algorithm 1: The classification process of DSDCLNet

Input: training set: $\{X_{train}, Y_{train}\}$, test set: X_{test} , batch size: B , DSE structure: f_{DSE}

Output: Labels of the test set: Y_{test}

```

1: for all  $i \in [0, epoch)$  do
2:   for sampled batch  $\{X_k\}_{k=1}^B$  in  $X_{train}$  do
3:      $X'_k \leftarrow \text{Transpose}(X_k)$ 
4:      $Out_{AGRU_k} \leftarrow \text{Attention}(\text{GRU}(X'_k))$ 
5:      $Out_{DMSCNN_k} \leftarrow \text{Integration}(g_1[\text{MSCNN}(X_k)], g_2[\text{MSCNN}(X_k)])$ 
6:      $Out_{encoder_k} \leftarrow \text{Concat}(Out_{AGRU_k}, Out_{DMSCNN_k})$ 
7:      $Out_{low\_dim_k} \leftarrow \text{Linear}(\text{AdaptiveAvgPool1d}(Out_{encoder_k}))$ 
8:     use Eq. (17) to calculate  $L_{Instance\_loss}^i \leftarrow \text{ICL}(\{Out_{low\_dim}\}_1^B, \{Y_{train}\}_1^B)$ 
9:     use Eq. (18) to calculate  $L_{Temporal\_loss}^{(i,t)} \leftarrow \text{ICL}(\{Out_{encoder}\}_1^B, \{Y_{train}\}_1^B)$ 
10:    use Eqs. (23)-(24) to calculate  $L_{classification\_loss}$  and  $L_{total}$ 
11:    update  $f_{DSE}$  to minimize  $L_{total}$ 
12:   end for
13: end for
14:  $Z_{train} \leftarrow f_{DSE}(X_{train})$ ,  $Z_{test} \leftarrow f_{DSE}(X_{test})$ 
15: construct a classifier with  $Z_{train}$  and  $Y_{train}$ 
16: classify  $Z_{test}$  with the classifier
17: return  $Y_{test}$ 

```

4 Experiments

In this section, we carry out comprehensive experiments to evaluate the performance of DSDCLNet. We first introduce the UEA [40] datasets, evaluation metrics, and the methods used for comparison. Then, we delve into the convergence of the proposed model and the stability of the hyperparameters, as well as comparative and ablation experiments. Finally, we conclude this section with an analysis that visualizes the learned features and assesses computational costs.

4.1 Experimental Setting

4.1.1 Datasets

We perform various experiments on UEA datasets to assess the performance of DSDCLNet. The datasets contain a variety of fields, including Human Activity Recognition (HAR), Motion Classification (Motion), Electrocardiogram (ECG),

Electroencephalogram/Magnetoencephalogram (ECG/MEG), Audio Spectroscopy (AS), and Others (Other). The feature dimensions of dataset range from 2 to 1345, its time steps from 8 to 17984, its categories from 2 to 39, and the sizes of data are from 27 to 50000. More details are provided in Table 2.

Table 2 Details of the datasets.

Dataset	Classes	Dimensions	Length	Train	Test	Type	Abbreviation
PenDigits	10	2	8	7494	3498	Motion	PD
CharacterTrajectories	20	3	182	1422	1436	Motion	CT
ArticulatoryWordRecognition	25	9	144	275	300	Motion	AWR
MotorImagery	2	64	3000	278	100	ECG/MEG	MI
SelfRegulationSCP1	2	6	896	268	293	ECG/MEG	SCP1
SelfRegulationSCP2	2	7	1152	200	180	ECG/MEG	SCP2
FingerMovements	2	28	50	316	100	ECG/MEG	FM
StandWalkJump	3	4	2500	12	15	ECG	SWJ
AtrialFibrillation	3	2	640	15	15	ECG	AF
BasicMotions	4	6	100	40	40	HAR	BM
Libras	15	2	45	180	180	HAR	Lb
Cricket	12	6	1197	108	72	HAR	Ck
NATOPS	6	24	51	180	180	HAR	NTP
RacketSports	4	6	30	151	152	HAR	RS
UWaveGestureLibrary	8	3	315	120	320	HAR	UGL
Epilepsy	4	3	206	137	138	HAR	Ep
Heartbeat	2	61	405	204	205	AS	Hb
SpokenArabicDigit	10	13	93	6599	2199	AS	SAD
JapaneseVowels	9	12	29	270	370	AS	JV
LSST	14	6	36	2459	2466	Other	LS
PEMS-SF	7	963	144	267	173	Other	PS
DuckDuckGeese	5	1345	270	60	40	AS	DDG
EigenWorms	5	6	17984	128	131	Motion	EW
ERing	6	4	65	30	30	HAR	ER
EthanolConcentration	4	3	1751	261	263	Other	EC
FaceDetection	2	144	62	5890	3524	ECG/MEG	FD
HandMovementDirection	4	10	400	320	147	ECG/MEG	HMD
Handwriting	26	3	152	150	850	HAR	HW
InsectWingbeat	10	200	78	30000	20000	AS	IW
Phoneme	39	11	217	3315	3353	AS	Ph

4.1.2 Evaluation metrics

In the experiments, we take the classification accuracy as a primary evaluation metric. In addition, we also assess the performance of the DSDCLNet using several

additional metrics, including the average accuracy, the total best accuracy, and the one-to-one comparison of win, draw, or loss. The notion of one-to-one wins, draws, or losses pertains to instances where the accuracy of DSDCLNet surpasses, matches, or falls behind the corresponding benchmark model accuracy across various datasets. A higher number of wins, fewer draws, and fewer losses indicate a more robust performance of the method.

4.1.3 Description of comparison methods

We compare DSDCLNet with three traditional MTSC methods (ED_I , DTW_I , and DTW_D [39]), six deep learning methods (MLSTM-FCN [10], WEASEL+MUSE [7], TapNet [11], ShapeNet [26], supervised TS2Vec [13], and MICOS [13]), as well as five self-supervised methods (T-loss [41], TNC [42], TS-TCC [32], TST [43], and TS2Vec (Self-supervised) [12]). In the following, we provide a summarized description of these methods.

ED_I , DTW_I and DTW_D . ED_I , DTW_I and DTW_D represent the nearest neighbor classifiers utilizing the Euclidean distance, the summed DTW distances across dimensions, and the computed DTW distances for multidimensional data, respectively.

MLSTM-FCN. Based on the LSTM-FCN [44] model, the channel attention [45] is added to generate potential features.

WEASEL+MUSE. It begins by creating multivariate feature vectors by applying the sliding window technique to each dimension of the time series data. Then, feature selection methods are employed to eliminate irrelevant features. Finally, a classifier is used for classification.

TapNet. The model extracts low-dimensional features from MTS data using a randomized group alignment approach combined with a multilayer convolutional network. To address the challenge of limited training labels, the model incorporates an attention prototype network.

ShapeNet. The model selects shape templates by embedding shape template candidates of different lengths into a unified space and utilizes a clustering triad loss during training. This strategy enhances accuracy by reducing the distance between

similar shape templates in the embedding space.

TS2Vec (supervised). The model transforms an unsupervised approach into a supervised one by modifying the loss function to incorporate a supervised comparison loss. In this method, each anchor point considers all samples from the same category as positives and regards samples from different categories as negatives.

MICOS. The model is an MTSC framework that relies on a hybrid supervised contrastive learning approach. It analyses spatiotemporal channels to extract more complex spatiotemporal features from MTS data. The model combines self-supervised, intra-class, and inter-class supervised contrastive learning techniques to calculate the training loss.

T-loss. The method is an unsupervised representation learning of time series. It employs an encoder composed of dilated convolutions to generate embedded representations and is trained using a triplet loss.

TNC. This is a self-supervised framework that aims to learn generalized representations of non-stationary time series. The method leverages the local smoothness of a signal's generative process to define time neighborhoods with stationary properties. The framework ensures that the distribution of signals from within a neighborhood is distinguishable from the distribution of signals outside the neighborhood using a debiased contrastive objective in the encoding space.

TS-TCC. TS-TCC achieves unsupervised learning of discriminative representations from time-series data using strong-weak data augmentations, temporal contrastive learning, and contextual contrastive modules. It has demonstrated efficiency in few-labeled and transfer learning scenarios.

TST. This is a framework for multivariate time series representation learning that is based on the Transformer encoder. It includes unsupervised pre-training and achieves superior performance across various domains.

TS2Vec (Self-supervised). TS2Vec is a universal framework that learns representations for time series through hierarchical contrastive learning on augmented context views. It has demonstrated significant superiority across various tasks.

4.1.4 Training settings

In all datasets, the output dimension of the GRU network is set to 128, the number and output dimensions of MSCNN are set to 2 and 64, respectively. The Adam optimizer with 300 optimization iterations and a typical learning rate of $1e-3$ is utilized for DSDCLNet training. The hyperparameter λ , which is set to 0.6, plays a crucial role in the loss function. The source code was implemented using PyTorch 2.0.1. The experiments were executed on the server with an NVIDIA A100-SXM4 GPU using CUDA 11.8.

Table 3 The parameter setting of DSDCLNet.

Parameters	Values	Description
<i>GRU_outdim</i>	128	Output dimensions of the GRU layer
<i>MSCNN_num</i>	2	Number of MSCNN layers
<i>MSCNN_outdim</i>	64	Output dimensions of the MSCNN layer
<i>lr</i>	$1e-3$	Learning rate
λ	0.6	Hyperparameters in the loss function

4.2 The convergence of DSDCLNet

The model convergence tests are conducted on six datasets, including AWR, JV, Lb, Hb, UGL, and Ep, which belong to different types. Figure 4 illustrates a consistent trend in the loss of the DSDCLNet model. The loss function initially decreases rapidly, followed by a gradual slowdown of the decline. Eventually, the loss function converges to a lower value and stabilizes. The results presented in Fig. 4 demonstrate that our proposed DSDCLNet converges rapidly, thereby validating the efficacy of our dual-level contrastive loss and classification loss.

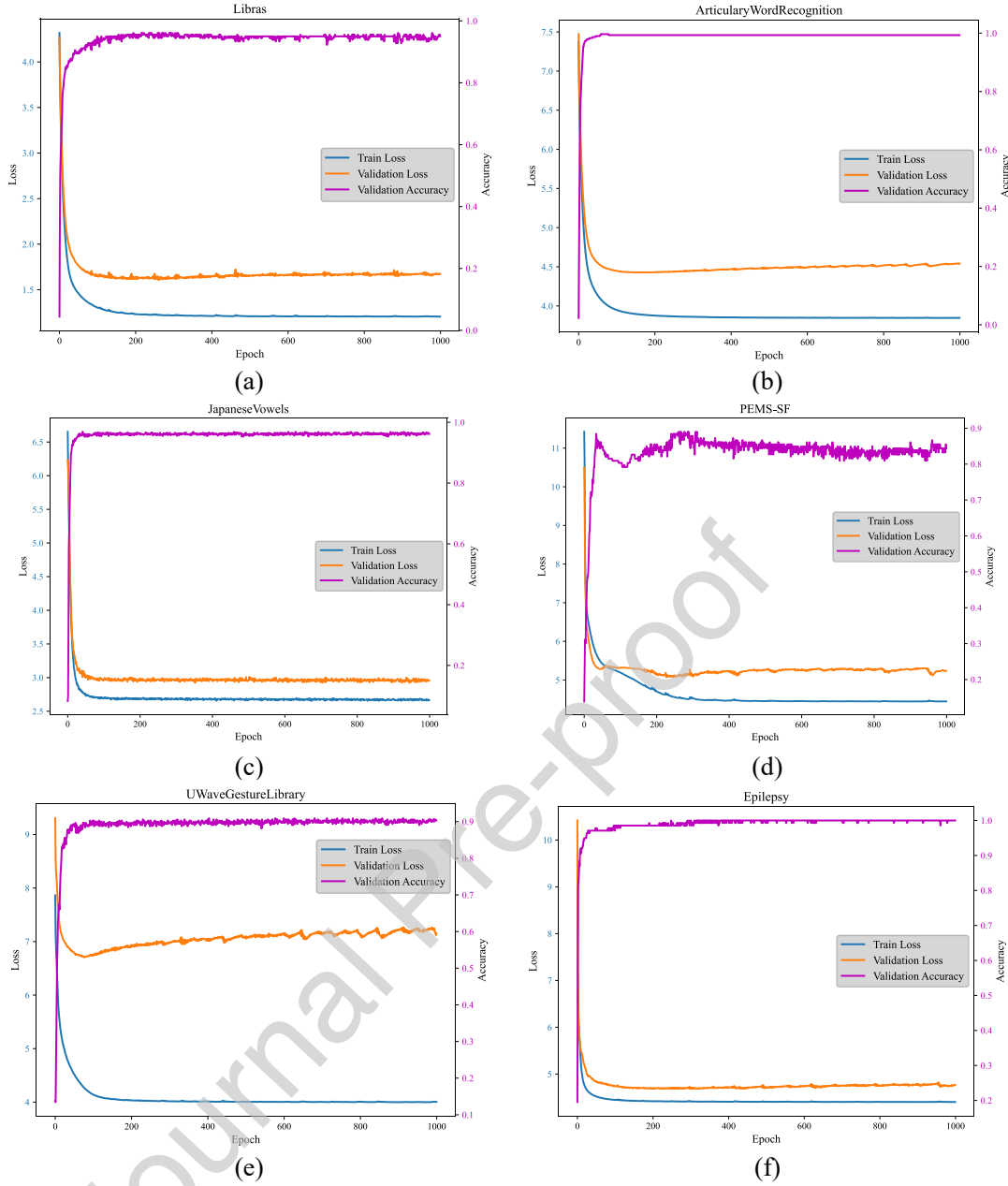


Fig. 4. Convergence of DSDCLNet on six datasets.

4.3 Hyperparameter stability

To evaluate the parameter stability of DSDCLNet, ten datasets (including Bm, UGL, SCP2, SWJ, Ep, AWR, NTP, Lb, Hb, and JV) are randomly selected for the cross-validation. Four parameters are examined: the number of model iterations (*epoch*), the learning rate (*lr*), the quantity of MSCNNs (*MSCNN_num*), and the proportion of the contrastive loss (λ). Figure 5 demonstrates the classification accuracy of DSDCLNet with various hyperparameters. Fig. 5 (a) shows the classification accuracy for epochs ranging from 100 to 600 with the green curve indicating the average accuracy trend.

There is no discernible pattern in the accuracy changes at different *epochs*, but the best performance is achieved at 300. Fig. 5 (b) illustrates the classification effect for learning rate (*lr*) ranging from $1e-1$ to $1e-6$, with performance grows gradually improves as the learning rate decreases, reaching the best result at $1e-3$. Fig. 5 (c) shows that the classification accuracy initially increases and then decreases as the number of MSCNN increases, with the best result achieved when *MSCNN_num* is 2. There is no obvious variation trend in Fig. 5 (d) as λ changes, but the average accuracy is highest when λ is 0.6. Therefore, DSDCLNet achieves optimal performance with an *epoch* of 300, *lr* of $1e-3$, *MSCNN_num* of 2, and λ of 0.6.

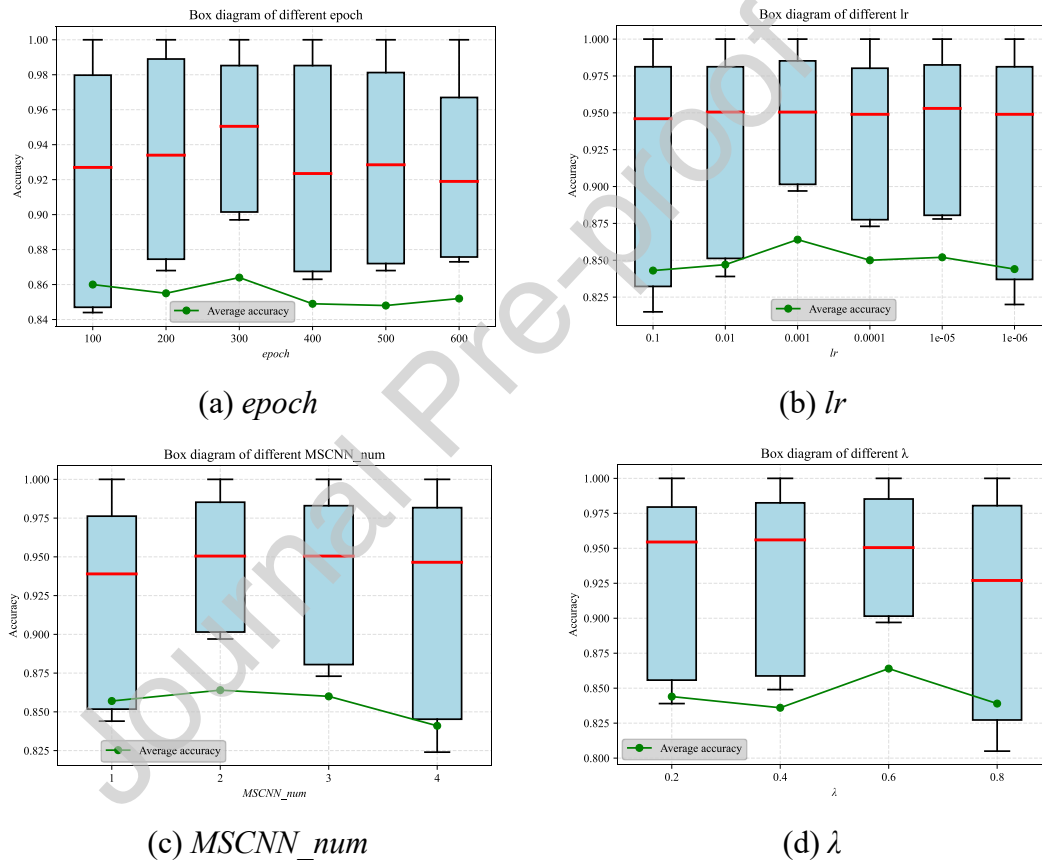


Fig. 5. The Boxplot of classification accuracy with different hyperparameters.

4.4 Classification performance evaluation

4.4.1 Comparison with traditional methods

The experiment compares DSDCLNet with three traditional methods, ED_I , DTW_I , and DTW_D . Table 4 displays the classification results of the traditional methods on the original data and the data after feature normalization. The experimental results of the

comparison methods are referred to the original papers. Table 4 demonstrates the superiority of DSDCLNet compared to ED_I , DTW_I , and DTW_D , with the total best accuracy of DSDCLNet is 30, indicating that DSDCLNet outperforms the comparison methods on all datasets. Furthermore, DSDCLNet exhibits an average accuracy that exceeds highest average accuracy of compared methods by approximately 10.9% and surpasses the lowest by about 18.5%.

Table 4 The comparison results of DSDCLNet and traditional baseline methods.

Datasets	Normalized			Normalize			DSDCLNet
	ED_I	DTW_I	DTW_D	ED_I	DTW_I	DTW_D	
PD	0.973	0.939	0.977	0.973	0.939	0.977	0.981
CT	0.964	0.969	0.990	0.964	0.969	0.989	1.000
AWR	0.970	0.980	0.987	0.970	0.980	0.987	0.993
MI	0.510	0.390	0.500	0.510	0.482	0.500	0.630
SCP1	0.771	0.765	0.775	0.771	0.765	0.775	0.884
SCP2	0.483	0.533	0.539	0.483	0.533	0.539	0.594
FM	0.550	0.520	0.530	0.550	0.520	0.530	0.610
SWJ	0.200	0.333	0.200	0.200	0.333	0.200	0.533
AF	0.267	0.267	0.200	0.267	0.267	0.220	0.467
BM	0.675	1.000	0.975	0.676	1.000	0.975	1.000
Lb	0.833	0.894	0.872	0.833	0.894	0.870	0.950
Ck	0.944	0.986	1.000	0.944	0.986	1.000	1.000
NTP	0.850	0.850	0.883	0.850	0.850	0.883	0.989
RS	0.868	0.842	0.803	0.868	0.842	0.803	0.901
UGL	0.881	0.869	0.903	0.881	0.868	0.903	0.925
Ep	0.667	0.978	0.964	0.666	0.978	0.964	1.000
Hb	0.62	0.659	0.717	0.619	0.658	0.717	0.932
SAD	0.967	0.96	0.963	0.967	0.959	0.963	0.99
JV	0.924	0.959	0.949	0.924	0.959	0.949	0.973
LS	0.456	0.575	0.551	0.456	0.575	0.551	0.675
PS	0.705	0.734	0.711	0.705	0.734	0.711	0.855
DDG	0.275	0.550	0.600	0.275	0.550	0.600	0.725
EW	0.550	0.603	0.618	0.549	N/A	0.618	0.856
ER	0.133	0.133	0.133	0.133	0.133	0.133	0.922
EC	0.293	0.304	0.323	0.293	0.304	0.323	0.453
FD	0.519	0.513	0.529	0.519	N/A	0.529	0.582
HMD	0.279	0.306	0.231	0.278	0.306	0.231	0.473
HW	0.371	0.509	0.607	0.200	0.316	0.286	0.635
IW	0.128	N/A	0.115	0.128	N/A	N/A	0.262
Ph	0.104	0.151	0.151	0.104	0.151	0.151	0.300
Average accuracy	0.591	0.658	0.643	0.585	0.661	0.651	0.770
Total best accuracy	0	1	1	0	1	1	30
Average rank	5.100	4.317	3.700	5.300	4.583	3.933	1.067

To statistically compare the differences between DSDCLNet and traditional MTSC methods, we employ the pairwise post-hoc analysis method proposed by Benavoli et al. [46]. Initially, we apply the Wilcoxon signed rank test to rectify Holm's alpha at a significance level of 5%. Next, we compute the comparison of mean ranks. Finally, we utilize critical difference (CD) plots to characterize the results. In Fig. 6, the $ED_{I,U}$, $DTW_{I,U}$, and $DTW_{D,U}$ represent the ED_I , DTW_I , and DTW_D methods without normalization, while $ED_{I,N}$, $DTW_{I,N}$, and $DTW_{D,N}$ denote the ED_I , DTW_I , and DTW_D techniques after normalization. Fig. 6 shows the average ranking of DSDCLNet is 1.0667, which is lower than that of the conventional methods. It indicates that DSDCLNet significantly outperforms the ED_I , DTW_I , and DTW_D methods. Moreover, the methods connected by a line do not show a statistically significant difference between them.

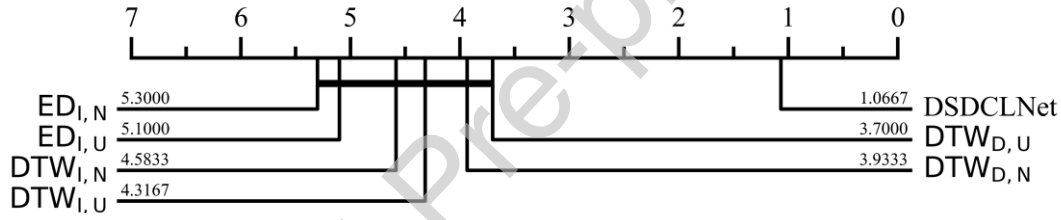


Fig. 6. CD of DSDCLNet on UEA datasets.

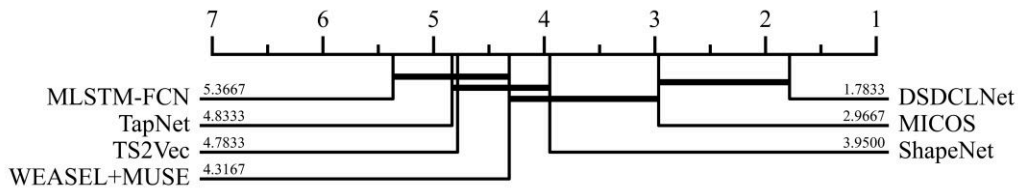
4.4.2 Comparison with deep learning methods

Table 5 compares the classification performance of DSDCLNet with six other deep learning methods. DSDCLNet achieves a significant improvement on five datasets, including AF, FM, Hb, Lb, and PS. Specifically, DSDCLNet outperforms the top-performing results of the comparative methods by 6.7%, 3%, 16.6%, 6.1%, and 4.6% on the corresponding datasets. Furthermore, based on the overall best accuracy, DSDCLNet outperforms other methods by achieving the highest accuracy on 23 datasets, while alternative methods excel in only 1 to 9 datasets. In terms of average accuracy, DSDCLNet shows the highest performance (0.770), which represents a 2.8% improvement compared to MICOS (0.742), which is the highest average accuracy among the compared methods. Furthermore, compared with other deep learning methods, DSDCLNet achieves a higher winning rate with lower draws and losses.

Table 5 The comparison results of DSDCLNet and deep learning methods.

Datasets	MLSTM-FCN	WEASEL+MUSE	TapNet	ShapeNet	TS2Vec	MICOS	DSDCLNet
PD	0.978	0.948	0.980	0.977	0.981	0.981	0.981
CT	0.985	0.990	0.997	0.980	0.992	0.994	1.000
AWR	0.973	0.990	0.987	0.987	0.980	0.990	0.993
MI	0.500	0.500	0.590	0.610	0.460	0.500	0.630
SCP1	0.874	0.710	0.652	0.782	0.741	0.799	0.884
SCP2	0.472	0.460	0.550	0.578	0.556	0.578	0.594
FM	0.580	0.490	0.530	0.580	0.540	0.570	0.610
SWJ	0.067	0.333	0.400	0.533	0.533	0.533	0.533
AF	0.267	0.333	0.333	0.400	0.200	0.333	0.467
BM	0.950	1.000	1.000	1.000	1.000	1.000	1.000
Lb	0.856	0.878	0.850	0.856	0.867	0.889	0.950
Ck	0.917	1.000	0.958	0.986	0.958	1.000	1.000
NTP	0.889	0.870	0.939	0.883	0.922	0.967	0.989
RS	0.803	0.934	0.868	0.868	0.901	0.941	0.901
UGL	0.891	0.916	0.894	0.906	0.913	0.891	0.925
Ep	0.761	1.000	0.971	0.987	0.942	0.971	1.000
Hb	0.663	0.727	0.751	0.756	0.737	0.766	0.932
SAD	0.990	0.982	0.983	0.975	0.978	0.981	0.990
JV	0.976	0.973	0.965	0.984	0.976	0.989	0.973
LS	0.373	0.590	0.568	0.590	0.545	0.667	0.675
PS	0.699	0.773	0.751	0.751	0.775	0.809	0.855
DDG	0.675	0.575	0.575	0.725	0.660	0.740	0.725
EW	0.504	0.890	0.489	0.878	0.840	0.901	0.856
ER	0.941	0.964	0.895	0.574	0.896	0.941	0.922
EC	0.373	0.430	0.323	0.323	0.285	0.247	0.453
FD	0.545	0.545	0.556	0.602	0.517	0.523	0.582
HMD	0.365	0.365	0.378	0.338	0.297	0.649	0.473
HW	0.286	0.605	0.357	0.451	0.579	0.621	0.635
IW	0.167	N/A	0.208	0.250	0.179	0.218	0.262
Ph	0.110	0.190	0.175	0.298	0.231	0.276	0.300
Average accuracy	0.648	0.723	0.682	0.714	0.699	0.742	0.770
Total best accuracy	1	4	1	3	3	9	23
Average rank	5.367	4.317	4.833	3.950	4.783	2.967	1.783
Ours 1-to-1-Wins	27	23	29	24	25	21	-
Ours 1-to-1-Draws	1	4	1	3	4	4	-
Ours 1-to-1-Losses	2	3	0	3	1	5	-

CD plots are also employed to statistically assess the differences between DSDCLNet and other deep learning methods across various datasets, depicted in Fig. 7. The results demonstrate that DSDCLNet outperforms all other deep learning comparison methods except MICOS, with statistically significant differences.

**Fig. 7.** CD plots of DSDCLNet and deep learning methods on UEA datasets.

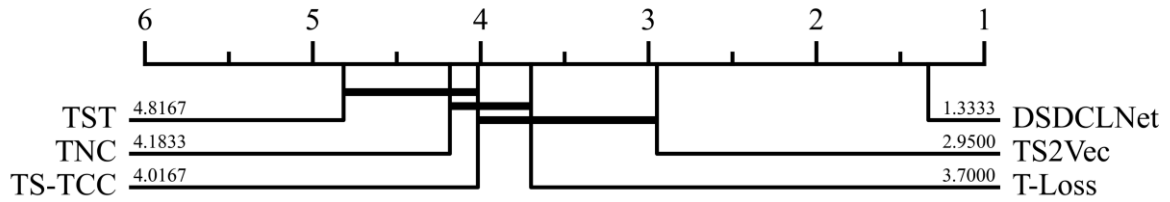
4.4.3 Comparison with self-supervised methods

Table 6 demonstrates the comparison of DSDCLNet with existing self-supervised models TST, TNC, TS-TCC, T-Loss, and TS2Vec. The results indicate that DSDCLNet outperforms all self-supervised baselines in terms of average accuracy, total best accuracy, and 1-to-1 wins. DSDCLNet exhibits significant improvement compared to other comparative methods on MI, SCP1, FM, AF, Lb, NTP, RS, LS, PS, and ER datasets. However, in terms of overall accuracy, DSDCLNet achieves the highest average accuracy (0.770) among all methods, surpassing others in 27 datasets according to the Total best accuracy. Meanwhile, DSDCLNet achieves the most wins (27 times) and relatively fewer draws or losses, indicating a significant performance advantage.

CD plots are also used to statistically assess the differences between DSDCLNet and self-supervised methods across various datasets, as shown in Fig. 8. The results indicate that DSDCLNet exhibits statistically significant differences from all self-supervised methods, reaching a level of statistical significance.

Table 6 The results of DSDCLNet and the other self-supervised methods.

Datasets	TST	TS-TCC	TNC	T-Loss	TS2Vec	DSDCLNet
PD	0.560	0.974	0.979	0.981	0.989	0.981
CT	0.975	0.985	0.967	0.993	0.995	1.000
AWR	0.977	0.953	0.973	0.943	0.987	0.993
MI	0.500	0.610	0.500	0.580	0.510	0.630
SCP1	0.754	0.823	0.799	0.843	0.812	0.884
SCP2	0.550	0.533	0.550	0.539	0.578	0.594
FM	0.560	0.460	0.470	0.580	0.480	0.610
SWJ	0.267	0.333	0.400	0.333	0.467	0.533
AF	0.067	0.267	0.133	0.133	0.200	0.467
BM	0.975	1.000	0.975	1.000	0.975	1.000
Lb	0.656	0.822	0.817	0.883	0.867	0.950
Ck	1.000	0.917	0.958	0.972	0.972	1.000
NTP	0.850	0.822	0.911	0.917	0.928	0.989
RS	0.809	0.816	0.776	0.855	0.855	0.901
UGL	0.575	0.753	0.759	0.875	0.906	0.925
Ep	0.949	0.957	0.957	0.971	0.964	1.000
Hb	0.746	0.751	0.746	0.741	0.683	0.932
SAD	0.923	0.970	0.934	0.905	0.988	0.990
JV	0.978	0.930	0.978	0.989	0.984	0.973
LS	0.408	0.474	0.595	0.509	0.537	0.675
PS	0.740	0.734	0.699	0.676	0.682	0.855
DDG	0.620	0.380	0.460	0.650	0.680	0.725
EW	0.748	0.779	0.840	0.840	0.847	0.856
ER	0.874	0.904	0.852	0.133	0.874	0.922
EC	0.262	0.285	0.297	0.205	0.308	0.453
FD	0.534	0.544	0.536	0.513	0.501	0.582
HMD	0.243	0.243	0.324	0.351	0.338	0.473
HW	0.225	0.498	0.249	0.451	0.515	0.635
IW	0.105	0.264	0.469	0.156	0.466	0.262
Ph	0.085	0.252	0.207	0.222	0.233	0.300
average accuracy	0.617	0.668	0.670	0.658	0.704	0.770
Total best accuracy	1	1	1	2	1	27
Average rank	4.817	4.017	4.183	3.700	2.950	1.333
Ours 1-to-1-Wins	28	28	28	27	27	-
Ours 1-to-1-Draws	1	1	0	2	0	-
Ours 1-to-1-Losses	1	1	2	1	3	-

**Fig. 8.** CD plots of DSDCLNet on UEA datasets.

4.5 Ablation experiments

Firstly, we evaluate the impact of DCL by comparing the performance of the DSDCLNet model with a modified version that removes the contrastive learning component. The experimental results are presented in Fig. 9, which clearly demonstrates a performance decline when contrastive learning is omitted. The main reason is that the proposed model is guided by the contrastive learning component to learn more discriminative feature representations by comparing the similarity between pairs of samples.

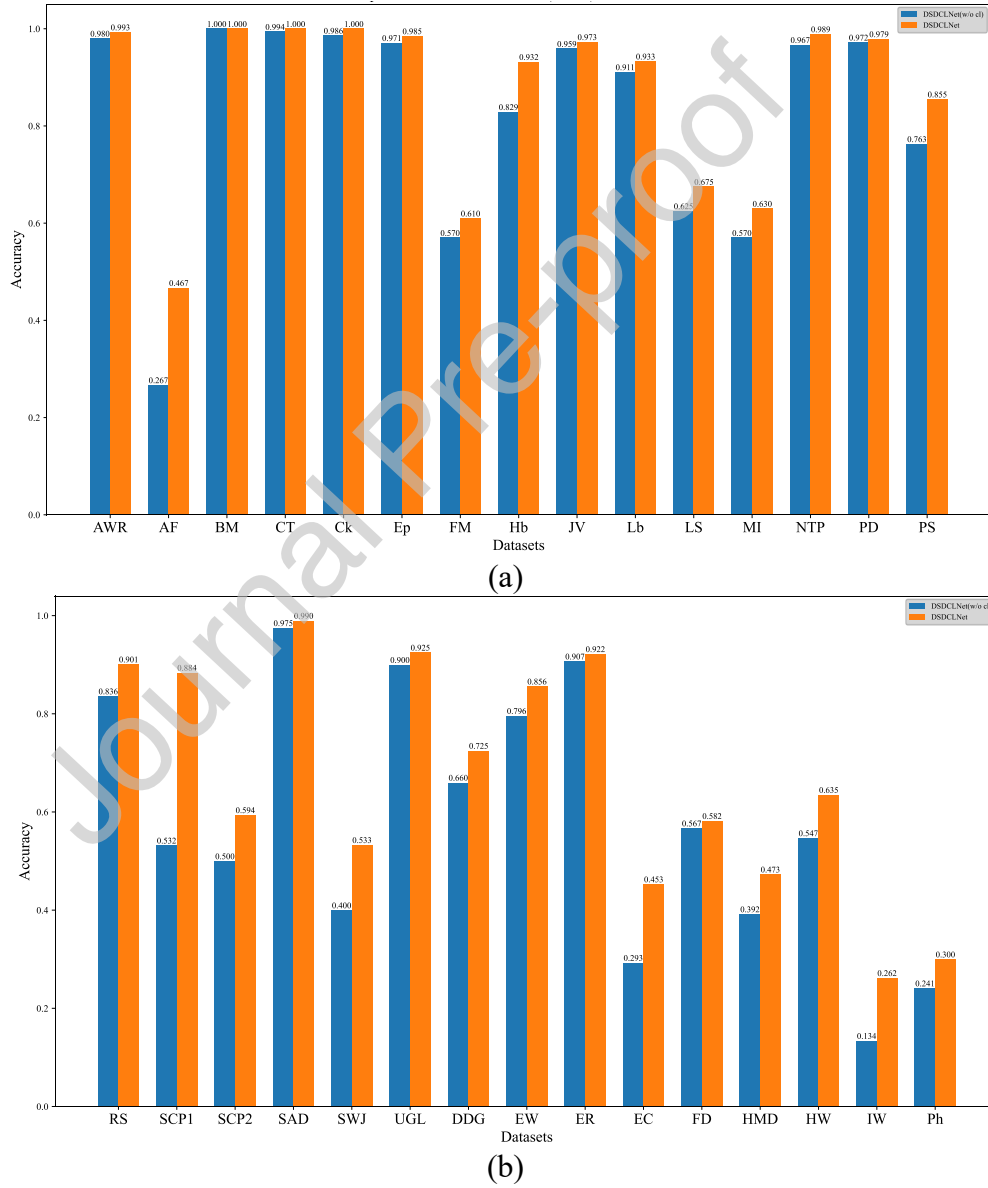


Fig.9. The comparison of classification accuracy of DSDCLNet and DSDCLNet (w/o DCL).

Next, four components such as AGRU, DMSCNN, ICL, and TCL are removed

from the DSDCLNet model. The impact of each module is then validated. From Table 7, although BM, Ck, and SWJ datasets exhibit comparable performance even without specific components, and RS dataset even outperform our proposed model when the DMSCNN components is removed, we observe that removing each of the four components leads to a decrease in the average accuracy by 6.8%, 22.7%, 6.2% and 5.6%, respectively. Hence, it can be inferred that AGRU, DMSCNN, instance-level and time-level contrast play essential roles in DSDCLNet.

Table 7 The results of DSDCLNet and DSDCLNet without each component.

Datasets	w/o AGRU	w/o DMSCNN	w/o ICL	w/o TCL	DSDCLNet
BM	1.000	0.725	1.000	1.000	1.000
CT	0.990	0.494	0.991	0.999	1.000
Ck	0.986	0.833	0.986	1.000	1.000
Ep	0.971	0.754	0.993	0.971	1.000
FM	0.550	0.530	0.550	0.540	0.610
Hb	0.698	0.898	0.805	0.859	0.932
JV	0.965	0.900	0.951	0.959	0.973
AWR	0.980	0.940	0.973	0.980	0.993
Lb	0.878	0.094	0.889	0.883	0.950
LS	0.493	0.527	0.600	0.596	0.675
MI	0.500	0.490	0.500	0.530	0.630
NTP	0.961	0.828	0.967	0.969	0.989
PD	0.970	0.850	0.963	0.965	0.981
PS	0.775	0.792	0.809	0.838	0.855
RS	0.816	0.914	0.862	0.855	0.901
AF	0.133	0.333	0.400	0.200	0.467
SCP1	0.850	0.498	0.798	0.729	0.884
SCP2	0.567	0.500	0.500	0.500	0.594
SAD	0.953	0.832	0.973	0.972	0.990
SWJ	0.467	0.200	0.333	0.533	0.533
UGL	0.797	0.375	0.888	0.872	0.925
DDG	0.679	0.620	0.673	0.712	0.725
EW	0.776	0.613	0.789	0.712	0.856
ER	0.893	0.167	0.919	0.889	0.922
EC	0.301	0.289	0.298	0.263	0.453
FD	0.567	0.568	0.571	0.566	0.582
HMD	0.527	0.405	0.311	0.460	0.473
HW	0.501	0.079	0.540	0.578	0.635
IW	0.284	0.185	0.135	0.186	0.262
Ph	0.236	0.062	0.282	0.293	0.300
Average accuracy	0.702	0.543	0.708	0.714	0.770

Then, to validate the effectiveness of the two-dimensional convolution on the proposed MSCNN module, we replace the two-dimensional convolution with one-dimensional convolution. The classification results are shown in Fig. 10. The results

indicate that utilizing a two-dimensional convolution model leads to superior classification performance.

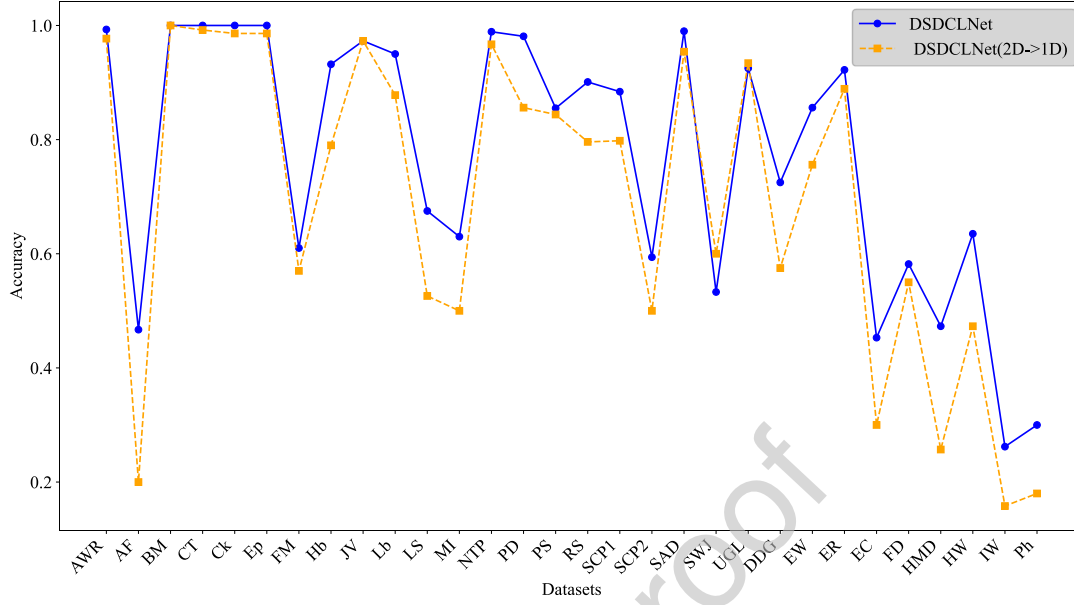


Fig. 10. Comparison of DSDCLNet and DSDCLNet (2D->1D) on UEA Datasets.

Finally, we test the effectiveness of label supervision on supervised time-level contrastive learning by comparing supervised TCL with unsupervised TCL, as shown in Fig. 11. we denote DSDCLNet using Unsupervised Temporal Contrastive Learning as DSDCLNet (UTCL). The experimental results demonstrate that incorporating label information enhances the classification performance of the DSDCLNet model.

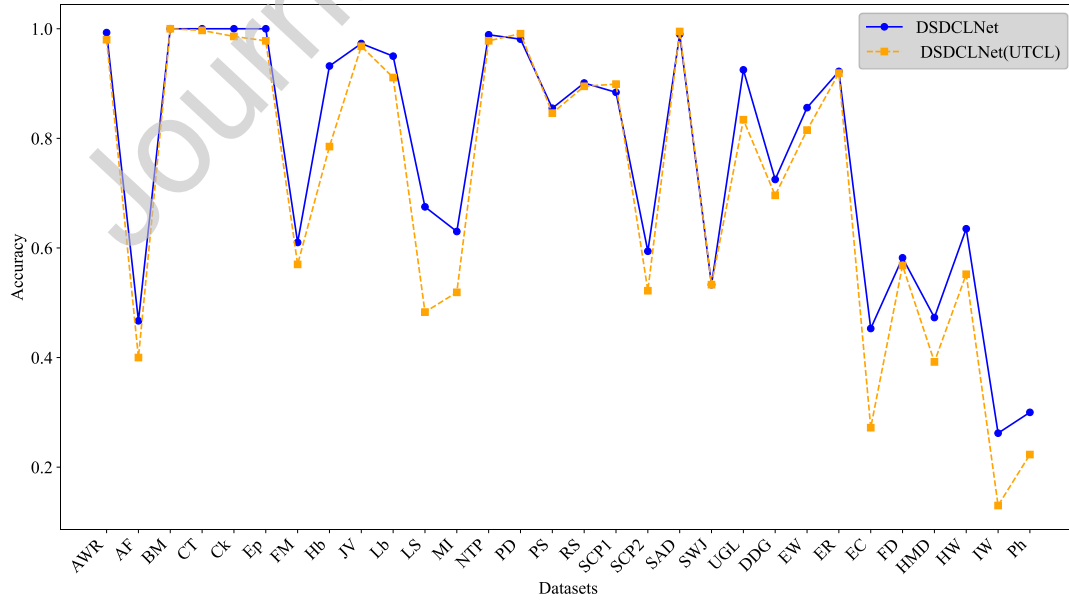
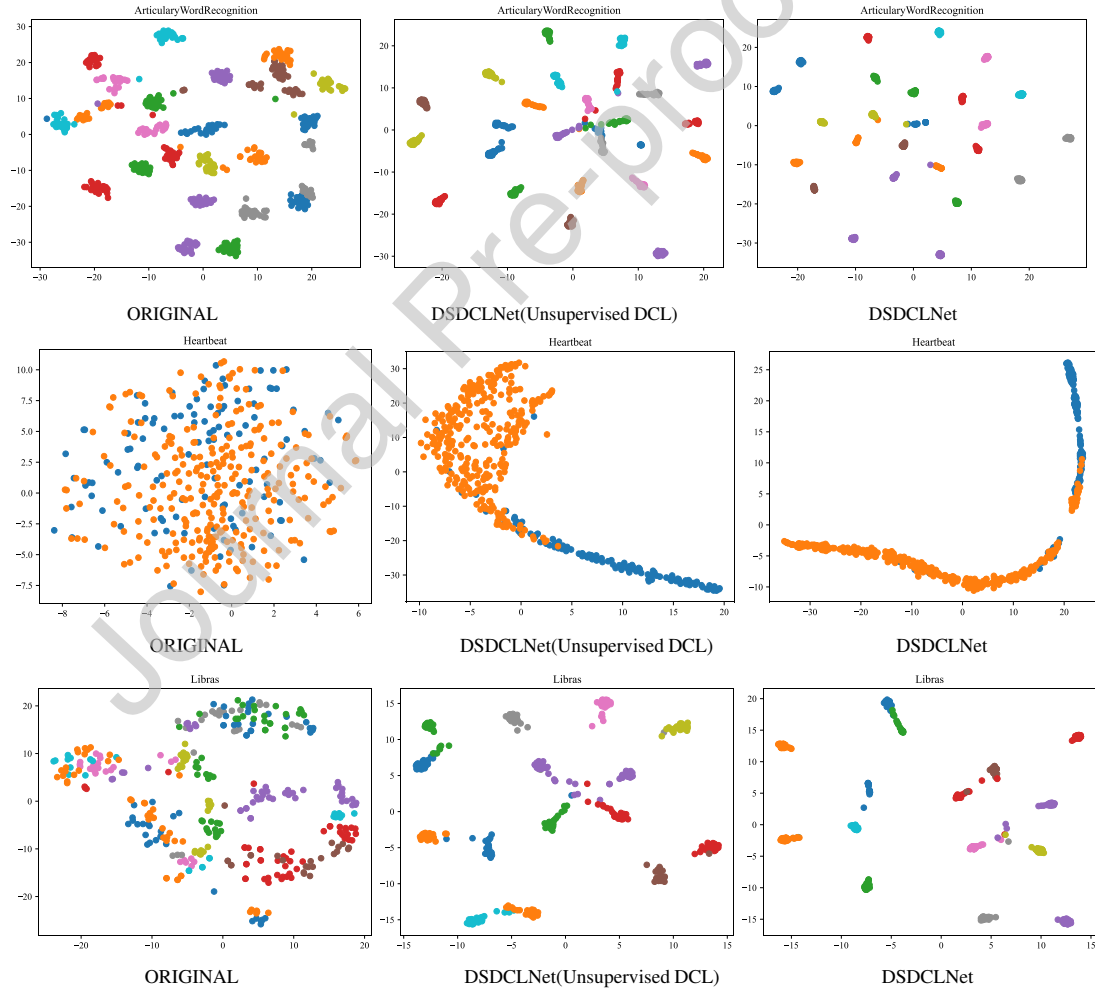


Fig. 11. Comparison of DSDCLNet and DSDCLNet (UTCL) on UEA Datasets.

4.6 Feature visualization

We present visualizations of the features extracted by DSDCLNet on six datasets using the t-SNE algorithm [47]. The results are shown in Fig. 12, where different colors represent different classes of data. The first column displays the feature visualization of the original data. The second column shows the feature visualization learned by DSDCLNet with unsupervised DCL, and the third column shows the feature visualization learned by DSDCLNet with supervised DCL. From this figure, we can see that the dual-level supervised contrastive learning helps increase the distance between data samples from different classes and reduce the distance between data samples from the same class.



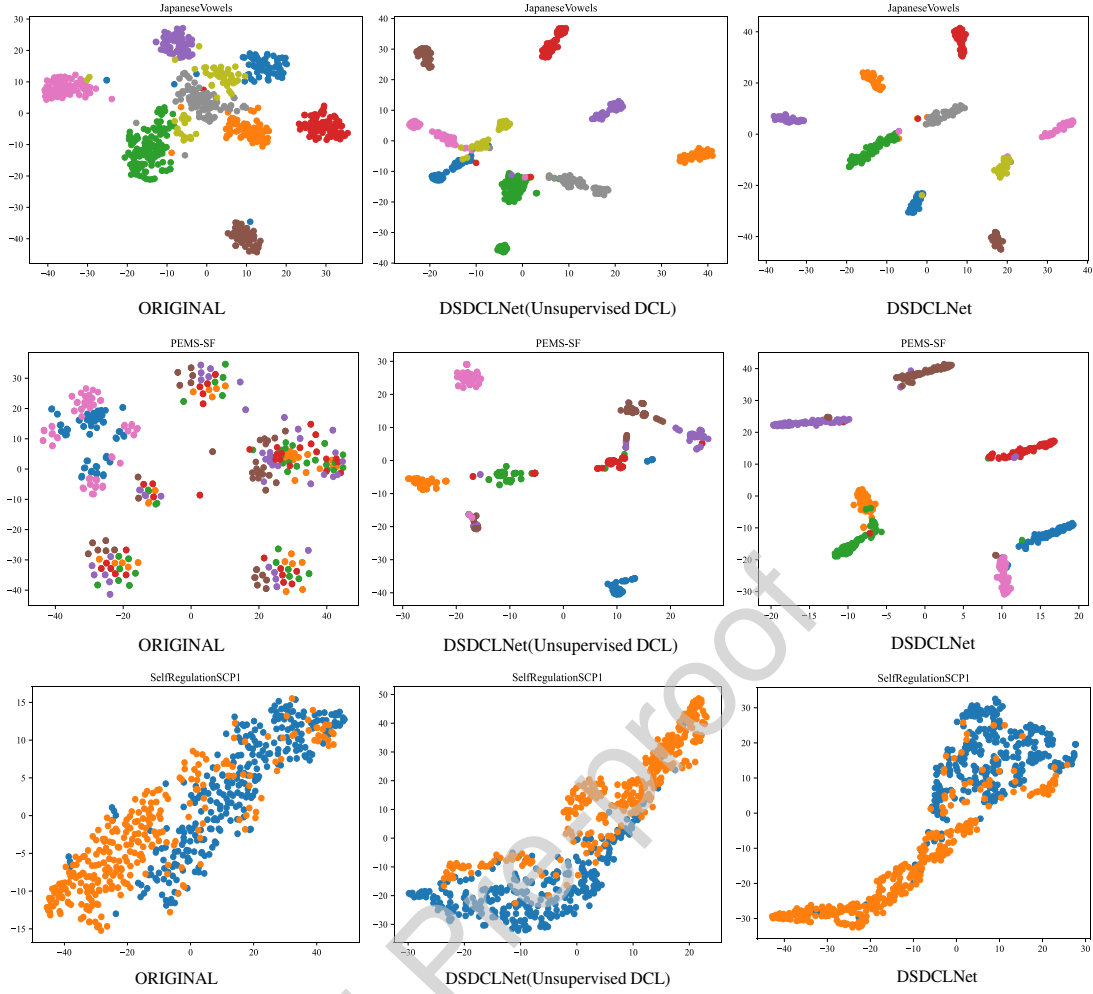


Fig. 12. t-SNE visualization of the learned representation on six datasets.

4.7 Computational cost analysis

We analyze the number of parameters (Param), floating-point operations (FLOPS), and runtime (Time) for TapNet and DSDCLNet on the UEA datasets. It is worth noting that TapNet is the most effective and efficient model for the MTSC task. Table 8 shows that although DSDCLNet has significantly lower parameter count and FLOPS than TapNet, its runtime is longer due to the complex calculations required for contrastive learning at the instance and time levels.

Table 8 Comparison of computational costs.

Datasets	TapNet			DSDCLNet		
	Param(M)	FLOPS(G)	Time (S)	Param(k)	FLOPS(G)	Time (S)
AWR	1.980	101.865	83.472	361.425	1.328	91.844
AF	1.353	5.872	86.037	347.147	2.417	25.796
BM	1.133	9.381	99.393	352.980	0.053	243.703
CT	1.781	653.397	162.704	351.972	0.640	378.376
Ck	2.004	280.429	120.613	354.588	3.910	194.550
Ep	1.175	69.715	18.849	348.756	0.466	58.136
FM	1.116	23.862	104.989	383.554	0.529	82.781
Hb	1.433	249.592	114.921	430.018	7.180	272.373
JV	1.105	5.066	232.404	382.346	0.144	82.976
Lb	1.512	15.830	303.020	349.559	0.357	56.963
LS	1.487	163.590	352.600	354.990	0.309	587.632
MI	2.035	127.252	39.797	434.242	13.659	3196.915
NTP	1.259	20.825	173.827	378.726	0.365	58.762
PD	0.969	23.149	630.378	348.554	0.066	1592.122
PS	5.192	372.818	115.605	1741.824	23.836	1036.976
RS	1.097	5.187	128.616	352.980	0.186	47.571
SCP1	1.463	652.239	91.164	352.578	7.806	283.386
SCP2	1.594	569.808	25.207	353.986	10.224	310.676
SAD	0.804	83.754	265.879	364.042	0.449	2876.604
SWJ	2.305	86.953	95.949	349.963	7.858	182.986
UGL	1.386	173.862	39.272	349.560	1.944	67.585
DDG	2.891	339.879	79.622	2238.000	7.698	532.909
EW	9.765	105.131	112.851	353.181	19.576	1056.340
ER	1.180	4.223	31.080	350.566	0.234	79.282
EC	1.468	59.385	185.327	348.756	7.195	741.080
FD	0.475	101.819	1699.537	431.426	1.289	3637.385
HMD	1.299	121.019	131.827	358.612	1.874	102.083
HW	1.998	183.325	222.757	353.178	0.611	145.985
IW	0.592	73.672	2056.340	387.978	0.633	29151.920
Ph	1.987	142.170	1446.271	367.055	0.190	2205.780

5 Conclusions

This paper proposes a DSDCLNet model, which is based on the Dual Stream Encoder (DSE) and Dual-Level Contrastive Learning (DCL). The DSE utilized AGRU and DMSCNN to efficiently exploit dynamic patterns and long-term dependencies of MTS data to extract multi-scale local-global features. Furthermore, DCL constructed the supervised contrastive learning at both temporal and instance levels and designed a

weighted loss function based on dual-level contrastive loss and classification loss. This approach maximizes the utilization of labeled data and enhances the classification performance. Compared to other state-of-the-art methods, DSDCLNet achieves the highest average classification accuracy of 0.770 and wins 23 out of 30 on MTS datasets. In future work, we will apply the DSDCLNet model to other domains and further optimize the contrastive learning techniques for semi-supervised learning.

Acknowledgments

This work is supported in part by grants from the National Natural Science Foundation of China (Nos. 62062040 and 62006174), the Outstanding Youth Project of Jiangxi Natural Science Foundation (No. 20212ACB212003), the Jiangxi Province Key Subject Academic and Technical Leader Funding Project (No. 20212BCJ23017), the Jiangxi Province Graduate Innovation Project (No. YC2022-S275), the Major Program of National Social Science Foundation of China (No. 20&ZD068).

Reference

- [1] Hsu C Y, Liu W C. Multiple time-series convolutional neural network for fault detection and diagnosis and empirical study in semiconductor manufacturing. *Journal of Intelligent Manufacturing*, 2021, 32: 823-836.
- [2] El-Sappagh S, Abuhmed T, Islam S M R, et al. Multimodal multitask deep learning model for Alzheimer's disease progression detection based on time series data. *Neurocomputing*, 2020, 412: 197-215.
- [3] Cheng D, Yang F, Xiang S, et al. Financial time series forecasting with multi-modality graph neural network. *Pattern Recognition*, 2022, 121: 108218.
- [4] Hsieh T Y, Wang S, Sun Y, et al. Explainable multivariate time series classification: a deep neural network which learns to attend to important variables as well as time intervals. *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. 2021: 607-615.
- [5] Bianchi F M, Scardapane S, Løkse S, et al. Reservoir computing approaches for representation and classification of multivariate time series. *IEEE Transactions on*

Neural Networks and Learning Systems, 2020, 32(5): 2169-2179.

[6] Bankó Z, Abonyi J. Correlation based dynamic time warping of multivariate time series. *Expert Systems with Applications*, 2012, 39(17): 12814-12823.

[7] Schäfer P, Leser U. Multivariate time series classification with WEASEL+ MUSE. *The 3rd ECML/PKDD Workshop on AALTD*, 2018, 637-646.

[8] Yang C, Wang X, Yao L, et al. Attentional gated Res2Net for multivariate time series classification. *Neural Processing Letters*, 2023, 55(2): 1371-1395.

[9] Ruiz A P, Flynn M, Large J, et al. The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 2021, 35(2): 401-449.

[10] Karim F, Majumdar S, Darabi H, et al. Multivariate LSTM-FCNs for time series classification. *Neural Networks*, 2019, 116: 237-245.

[11] Zhang X, Gao Y, Lin J, et al. Tapnet: Multivariate time series classification with attentional prototypical network. *Proceedings of the AAAI Conference on Artificial Intelligence*. 2020, 34(04): 6845-6852.

[12] Yue Z, Wang Y, Duan J, et al. Ts2vec: Towards universal representation of time series. *Proceedings of the AAAI Conference on Artificial Intelligence*. 2022, 36(8): 8980-8987.

[13] Hao S, Wang Z, Alexander A D, et al. MICOS: Mixed supervised contrastive learning for multivariate time series classification. *Knowledge-Based Systems*, 2023, 260: 110158.

[14] Wen Q, Sun L, Yang F, et al. Time series data augmentation for deep learning: A survey. *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI-21)*, 2021, 4654-4660.

[15] Li H. Time works well: Dynamic time warping based on time weighting for time series data mining. *Information Sciences*, 2021, 547: 592-608.

[16] Mei J, Liu M, Wang Y F, et al. Learning a mahalanobis distance-based dynamic time warping measure for multivariate time series classification. *IEEE Transactions on Cybernetics*, 2015, 46(6): 1363-1374.

[17] Shokoohi-Yekta M, Hu B, Jin H, et al. Generalizing DTW to the multi-dimensional

case requires an adaptive approach. *Data Mining and Knowledge Discovery*, 2017, 31: 1-31.

[18] Li H. Accurate and efficient classification based on common principal components analysis for multivariate time series. *Neurocomputing*, 2016, 171: 744-753.

[19] Baydogan M G, Runger G. Learning a symbolic representation for multivariate time series classification. *Data Mining and Knowledge Discovery*, 2015, 29: 400-422.

[20] Zhao H, Pan Z, Tao W. Regularized shapelet learning for scalable time series classification. *Computer Networks*, 2020, 173: 107171.

[21] Ismail Fawaz H, Forestier G, Weber J, et al. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 2019, 33(4): 917-963.

[22] Liu C L, Hsaio W H, Tu Y C. Time series classification with multivariate convolutional neural network. *IEEE Transactions on Industrial Electronics*, 2018, 66(6): 4788-4797.

[23] Yu W, Kim I Y, Mechefske C. Analysis of different RNN autoencoder variants for time series classification and machine prognostics. *Mechanical Systems and Signal Processing*, 2021, 149: 107322.

[24] Wu Y, Lian C, Zeng Z, et al. An aggregated convolutional transformer based on slices and channels for multivariate time series classification. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2023, 7(3): 768-779.

[25] Chen R, Yan X, Wang S, et al. DA-Net: Dual-attention network for multivariate time series classification. *Information Sciences*, 2022, 610: 472-487.

[26] Li G, Choi B, Xu J, et al. Shapenet: A shapelet-neural network approach for multivariate time series classification. *Proceedings of the AAAI Conference on Artificial Intelligence*. 2021, 35(9): 8375-8383.

[27] Yang X, Zhang Z, Cui R. TimeCLR: A self-supervised contrastive learning framework for univariate time series representation. *Knowledge-Based Systems*, 2022, 245: 108606.

[28] Ismail Fawaz H, Lucas B, Forestier G, et al. Inceptiontime: Finding alexnet for time series classification. *Data Mining and Knowledge Discovery*, 2020, 34(6): 1936-1962.

- [29] Pöppelbaum J, Chadha G S, Schwung A. Contrastive learning based self-supervised time-series analysis. *Applied Soft Computing*, 2022, 117: 108397.
- [30] Chen T, Kornblith S, Norouzi M, et al. A simple framework for contrastive learning of visual representations, *International Conference on Machine Learning*. PMLR, 2020: 1597-1607.
- [31] Nonnenmacher M T, Oldenburg L, Steinwart I, et al. Utilizing expert features for contrastive learning of time-series representations. *International Conference on Machine Learning*. PMLR, 2022: 16969-16989.
- [32] Eldele E, Ragab M, Chen Z, et al. Time-Series Representation Learning via Temporal and Contextual Contrasting. *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI-21)*, 2021, 2352-2359.
- [33] Zhang X, Zhao Z, Tsigikaridis T, et al. Self-supervised contrastive pre-training for time series via time-frequency consistency. *Advances in Neural Information Processing Systems*, 2022, 35: 3988-4003.
- [34] Wickstrøm K, Kampffmeyer M, Mikalsen K Ø, et al. Mixing up contrastive learning: Self-supervised representation learning for time series. *Pattern Recognition Letters*, 2022, 155: 54-61.
- [35] Tan Q, Ye M, Yang B, et al. Data-gru: Dual-attention time-aware gated recurrent unit for irregular multivariate time series. *Proceedings of the AAAI Conference on Artificial Intelligence*. 2020, 34(01): 930-937.
- [36] Niu Z, Zhong G, Yu H. A review on the attention mechanism of deep learning. *Neurocomputing*, 2021, 452: 48-62.
- [37] Wang J, Cheng S, Tian J, et al. A 2D cnn-lstm hybrid algorithm using time series segments of EEG data for motor imagery classification. *Biomedical Signal Processing and Control*, 2023, 83: 104627.
- [38] Xuhong L I, Grandvalet Y, Davoine F. Explicit inductive bias for transfer learning with convolutional networks. *International Conference on Machine Learning*. PMLR, 2018: 2825-2834.
- [39] Chen L, Wang F, Yang R, et al. Representation learning from noisy user-tagged data for sentiment classification. *International Journal of Machine Learning and*

Cybernetics, 2022, 13(12): 3727-3742.

[40] Bagnall A, Dau H A, Lines J, et al. The UEA multivariate time series classification archive, 2018. arXiv preprint arXiv:1811.00075, 2018.

[41] Franceschi J Y, Dieuleveut A, Jaggi M. Unsupervised scalable representation learning for multivariate time series. Advances in Neural Information Processing Systems, 2019, 32:1-12.

[42] Tonekaboni S, Eytan D, Goldenberg A. Unsupervised representation learning for time series with temporal neighborhood coding. International Conference on Learning Representations, 2021,1-17.

[43] Zerveas G, Jayaraman S, Patel D, et al. A transformer-based framework for multivariate time series representation learning. Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. 2021: 2114-2124.

[44] Karim F, Majumdar S, Darabi H, et al. LSTM fully convolutional networks for time series classification. IEEE Access, 2017, 6: 1662-1669.

[45] Hu J, Shen L, Sun G. Squeeze-and-excitation networks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 7132-7141.

[46] Benavoli A, Corani G, Mangili F. Should we really use post-hoc tests based on mean-ranks?. The Journal of Machine Learning Research, 2016, 17(1): 152-161.

[47] Van der Maaten L, Hinton G. Visualizing data using t-SNE. Journal of Machine Learning Research, 2008, 9(11): 2579-2605

Declaration of Interest Form

We wish to confirm that there are no known conflicts of interest associated with the manuscript submitted to *Knowledge-Based Systems* and there has been no significant financial support for this work that could have influenced its outcome.

We confirm that the manuscript has been read and approved by all named authors and that there are no other persons who satisfied the criteria for authorship but are not listed. We further confirm that the order of authors listed in the manuscript has been approved by all of us.

We confirm that we have given due consideration to the protection of intellectual property associated with this work and that there are no impediments to publication,

including the timing of publication, with respect to intellectual property. In so doing we confirm that we have followed the regulations of our institutions concerning intellectual property.

Journal Pre-proof



DSPACE

<https://dspace.org/>

DSDCLNet: dual-stream encoder and dual-level contrastive learning network for supervised multivariate time series classification

Liu, Min; Sheng, Hui; Zhang, Ningyi; Zhao, Panpan; Yi, Yugen ...

2024-05-23

Elsevier

<https://doi.org/10.1016/j.knosys.2024.111638>

Liu M, Sheng H, Zhang N, et al., (2024) DSDCLNet: dual-stream encoder and dual-level contrastive learning network for supervised multivariate time series classification. Knowledge-Based Systems, Volume 292, May 2024, Article number 111638