Deep Contrastive Representation Learning with Self-distillation

Zhiwen Xiao, *Member, IEEE*, Huanlai Xing, *Member, IEEE*, Bowen Zhao, Rong Qu, *Senior Member, IEEE*, Shouxi Luo, *Member, IEEE*, Penglin Dai, *Member, IEEE*, Ke Li, and Zonghai Zhu

***Abstract*—Recently, contrastive learning (CL) is a promising way of learning discriminative representations from time series data. In the representation hierarchy, semantic information extracted from lower levels is the basis of that captured from higher levels. Low-level semantic information is essential and should be considered in the CL process. However, the existing CL algorithms mainly focus on the similarity of high-level semantic information. Considering the similarity of low-level semantic information may improve the performance of CL. To this end, we present a deep contrastive representation learning with self-distillation (DCRLS) for the time series domain. DCRLS gracefully combine data augmentation, deep contrastive learning, and self distillation. Our data augmentation provides different views from the same sample as the input of DCRLS. Unlike most CL algorithms that concentrate on high-level semantic information only, our deep contrastive learning also considers the contrast similarity of low-level semantic information between peer residual blocks. Our self distillation promotes knowledge flow from high-level to low-level blocks to help regularize DCRLS in the knowledge transfer process. The experimental results demonstrate that the DCRLS-based structures achieve excellent performance on classification and clustering on 36 UCR2018 datasets.**

***Index Terms*—Contrastive Learning, Knowledge Distillation, Representation Learning, Time Series Classification, Time Series Clustering.**

I. Introduction

TIME series data is of significant importance to various areas in the real world, such as electroencephalogram (EEG) analysis [1], cardiotocography (CTG) interpretation [2], symbolic sequence classification [3], automated damage detection [4], anomaly detection [5], regime change detection [6], valvular heart diseases detection [7], colorectal polyp diagnosis [8] and motion detection [9]. To achieve accurate time series classification, it is essential for an algorithm to effectively capture both local and global patterns of the input time series [10].

Deep learning algorithms on time series classification have achieved promising performance because they can discover the intrinsic connections among representations by unfolding the internal representation hierarchy of data. For instance, Wang *et al.* [11] tested three deep learning architectures, including fully convolutional network (FCN), residual network (ResNet), and multilayer perceptron (MLP) for time series classification. Fawaz *et al.* [12] proposed a multi-head neural network, called InceptionTime, to extract multi-scaled features from the input. Xiao *et al.* [13] presented a robust temporal feature network integrating an LSTM-based attention network and a temporal feature network for supervised classification and unsupervised clustering. The algorithms above heavily rely on a massive amount of labeled data, making it challenging to adapt them to scenarios with limited labeled data.

Recently, there has been a surge of interest in self-supervised learning as a means of generating effective representations from unlabelled data for use in downstream tasks. Among the most popular trends in this area are algorithms that incorporate contrastive learning (CL) [14], [15], [16]. CL is a discriminative approach to similar group samples closer. For instance, He *et al.* [17] proposed MoCo using a momentum encoder to learn representations of the negative pairs obtained from a memory bank. Chen *et al.* [18] devised SimCLR to replace the momentum encoder with a large batch of antagonistic pairs. Chen *et al.* [19] developed SimSiam ignoring the negative samples for feature extraction. Jin *et al.* [20] designed a multi-scale contrastive approach with self-distillation that achieved decent performance on a number of benchmark datasets. A self-supervised learning paradigm that utilizes single-stage online knowledge distillation to enhance the quality of model representations was proposed [21]. Hu *et al.* [22] designed an unsupervised cross-modal hashing method to achieve decent retrieval performance. Li *et al.* [23] put forward an online clustering approach for instance- and cluster-level CL. Shu *et al.* [24] introduced an anchor-contrastive representation learning approach for semi-supervised skeleton-based action recognition. Xu *et al.* [25] devised an X-invariant contrastive augmentation and representation learning method for skeleton-based action recognition. Lin *et al.* [26] presented a contrastive matching method with momentum distillation to address the bi-level noisy correspondence problem in graph matching. Yu *et al.* [27] designed a contrastive instance learning method with self-distillation for audio-visual violence

Manuscript received XX, XX; revised XX, XX. This work was partially supported by the National Natural Science Foundation of China (No. 62172342 and No.62202392), the Natural Science Foundation of Hebei Province (No. F2022105027), the Natural Science Foundation of Sichuan Province (No. 2022NSFSC0568, No. 2022NSFSC0944, and No. 2023NSFSC0459), and the Fundamental Research Funds for the Central Universities, P. R. China. (Corresponding Author: Huanlai Xing)

Z. Xiao, H. Xing, B. Zhao, S. Luo, P. Dai, K. Li and Z. Zhu are with the School of Computing and Artificial Intelligence, Southwest Jiaotong University, Chengdu, Tangshan Institute of Southwest Jiaotong University, Tangshan, and Engineering Research Center of Sustainable Urban Intelligent Transportation, Ministry of Education, China (Emails: xiao1994zw@163.com; hxx@home.swjtu.edu.cn; cn16bz@icloud.com; sxluo@swjtu.edu.cn; penglindai@swjtu.edu.cn; keli@swjtu.edu.cn; zzhu@swjtu.edu.cn).

R. Qu is with the School of Computer Science, University of Nottingham, Nottingham NG7 2RD 455356, UK (rong.qu@nottingham.ac.uk).

detection. Wang *et al.* [28] introduced a CL-based transformer model with label-free distillation for feature extraction in remote sensing.

CL has been applied to the time series field. For example, Mohsenvand *et al.* [29] extended the SimCLR model [18] to capture representations from EEG data. Eldele *et al.* [30] introduced an unsupervised time series representation learning framework via temporal and contextual contrasting to learn representations from unlabeled data. Yue *et al.* [31] designed a universal framework for understanding time series representations at an arbitrary semantic level. Different from computer vision (CV), CL in the time series domain develops slowly. The following lists some notable limitations in this domain.

First, the time series domain lacks large-scale datasets, e.g., unlike ImageNet in CV, to provide models with rich semantic features in the data [30], [31]. As a result, time series models usually have little prior knowledge to help handle real-world tasks, such as anomaly detection.

Second, data augmentation is crucial for CL to understand the similarity of different views from the same sample and the similarity with the views from different samples [14]. Unlike image data, time series data is a series of time-ordered data points associated with single or multiple time-dependent variables [13]. Typical augmentation methods for image data, such as crop, are not directly applicable to time series domain. Meanwhile, time series data augmentation has not received enough research attention [29], [30], [31].

Third, a deep neural network aims at learning multiple levels of feature representations with increasing abstraction. Its performance is dependent on high- and low-level semantic information obtained from the data [32]. As known, the quality of the obtained high-level semantic information is based on that of the low-level semantic information already extracted. On the other hand, most learning models update their parameters using the backpropagation (BP) [33] method. High-level semantic information influences low-level semantic information to a certain extent according to BP. *Hence, low-level semantic information is also important and should be considered in the CL process*. To the best of our knowledge, most of the existing CL-based algorithms in the time series field, such as SimCLR on EEG [29] and temporal and contextual contrasting [30], mainly focus on the similarity of high-level semantic information. Ignoring the similarity of low-level semantic information may lead to a deteriorated CL performance. Unfortunately, this issue has not been well studied in the literature.

To tackle the problems above, we propose a deep contrastive representation learning with self-distillation (DCRLS). To be specific, DCRLS gracefully integrates data augmentation, deep contrastive learning, and self distillation to explore the representations hidden in time series data. Data augmentation provides different views from the same sample as the input of DCRLS. The feature extractor for DCRLS is a residual network (ResNet) containing three residual blocks. Deep contrastive learning provides the contrast similarity of high-level semantic information and that of low-level semantic information between peer residual blocks to mine instance-level features between different perspectives from the same sample. Inspired by BYOT [34], self distillation transfers knowledge from high-level to low-level blocks for self regularization purpose. Unlike BYOT's KL divergence-based knowledge distillation (KD) loss function, DCRLS uses the L_2 function to measure the difference between high-level and low-level feature vectors. Fig. 1 illustrates the overview of DCRLS.

Our major contributions are summarized as follows:

- We present DCRLS that ensembles data augmentation, deep contrastive learning, and self distillation techniques, to mine the intrinsic connections between the internal representational hierarchy of data for downstream tasks.
- Unlike other CL-based algorithms, DCRLS considers the contrast similarity of high-level semantic information and that of low-level semantic information between peer residual blocks via deep contrastive learning.
- DCRLS is applied to time series classification and clustering with 36 UCR2018 datasets considered. On classification, DCRLS outperforms six existing classification algorithms on 28 datasets in terms of mean accuracy, 'win'/'tie'/'lose' measure, and AVG_rank; on clustering, DCRLS wins on 8 out of 36 datasets when compared with 13 clustering algorithms with respect to average rand index, 'win'/'tie'/'lose' measure, and AVG_rank.

The remainder of this paper is structured as follows. In Section II, we review related work on time series classification and clustering. In Section III, we provide a detailed description of the problem formulation and our proposed approach, DCRLS. Section IV presents an analysis of our experimental results, and finally, we draw our conclusion in Section V.

II. Related Work

This section reviews time series classification and clustering algorithms.

A. Time Series Classification

Time series classification algorithms can be roughly classified into two categories: traditional and deep-learning algorithms.

1) Traditional Algorithms: Distance- and feature-based algorithms are two main streams for time series classification. Combining the nearest neighbor (NN) and dynamic time warping (DTW) is a commonly used as distance-based method. This method focuses on calculating the similarities between spatial features of data, e.g., DTW_I, DTW_A, DTW_C and DTW_D [10]. Besides, researchers have devoted their effects on ensemble approaches for time series classification. Lines and Bagnall [35] introduced an elastic ensemble (EE) algorithm hybridizing 11 1-NN-based elastic distances for various time series classification tasks. Lines *et al.* [36] proposed a collective of transformation-based ensemble (COTE) algorithm that used multiple standard classifiers to achieve decent performance. The hierarchical vote collective of transformation-based ensembles (HIVE-COTE) [37], HIVE-COTE 2.0 [38], and local cascade ensemble (LCE) [39] are also ensemble-based.

Feature-based algorithms aims at capturing representative features from time series data. For example, Karlsson *et al.*

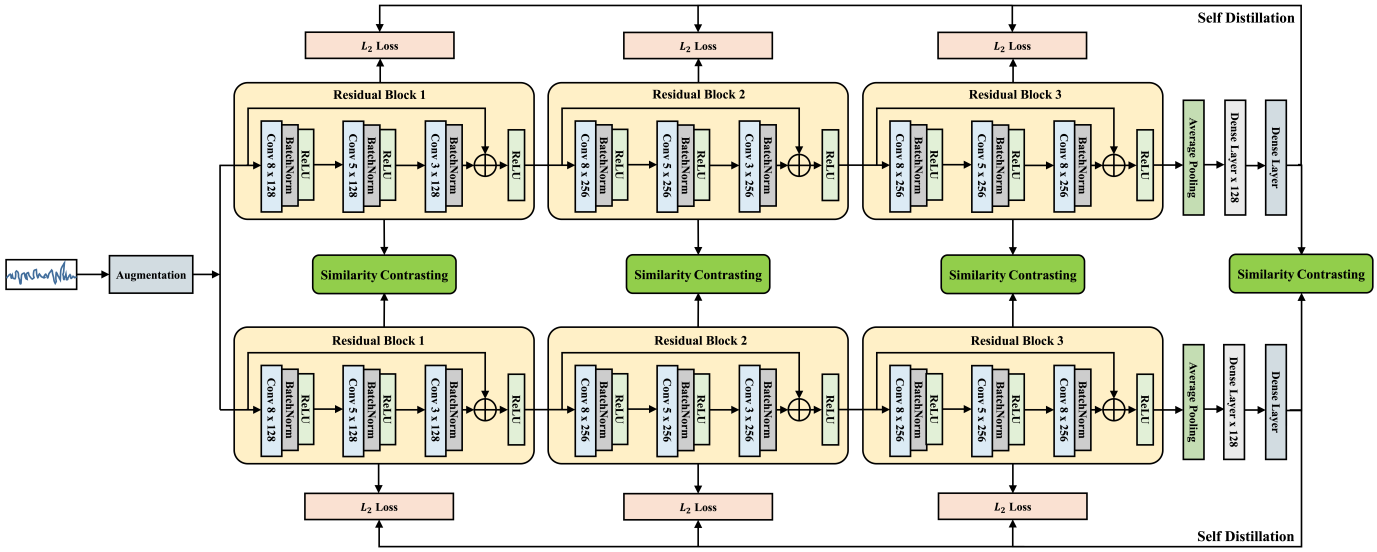


Fig. 1. The overview of DCRLS. The feature extractor is a residual network (ResNet) including three residual blocks, an average pooling layer, and two dense (i.e., fully-connected) layers. Each residual block consists of a 1-dimensional convolutional neural network (Conv), a batch normalization (BatchNorm) module, and a rectified linear unit (ReLU) function, where "Conv 8x128" stands for a 1-dimensional Conv with a kernel size of 8 and a channel number of 128.

[40] designed a generalized random shapelet forest algorithm with shapelets and bag-of-words methods used for feature extraction. The learned pattern similarity [41], hidden state conditional random field [42] and interpretable representation learning [43] are feature-based as well.

2) Deep Learning Algorithms: Deep learning algorithms learn the underlying relationships among data representations by constructing an internal hierarchy of data representation. Currently, there are two branches of work in the community: single-network-based and dual-network-based models. A single-network-based model is based on one (usually hybridized) network that concentrates on the significant connections within the hierarchy. For example, Xiao *et al.* [44] introduced a multi-process collaborative architecture using a multi-head capsule network to extract multi-scale features from the input. Other examples of single-network-based models include MLP, ResNet, FCN [11], InceptionTime [12], ResNet with random vector functional link [45], residual channel attention network [46], CNN-based federated distillation learning [47], deep echo state network [48], ROCKET [49], MiniRocket [50], and ConvTimeNet [51]. In contrast, dual-network-based models typically consist of two parallel networks: one for local-feature extraction and one for global-relation extraction. The former typically uses CNNs to extract local features, while the latter focuses on capturing the connections among the extracted features. Examples of dual-network-based models include RTFN [13], robust neural temporal search [52], FCN-LSTM [53], ResNet-Transformer [54], SelfMatch [55] and attentional prototype network [56].

B. Time Series Clustering

Time series clustering algorithms are used to group similar data together for better analysis of its structure [57]. Two main streams in the time series clustering community are traditional and deep learning algorithms. Traditional clustering algorithms use dissimilarity measure or time series representation methods to distinguish the differences in data structures. For example, Yang *et al.* [58] introduced a combination of discriminative analysis and $l_{2,1}$-norm minimization for unsupervised feature selection. Li *et al.* [59] proposed an unsupervised feature selection algorithm for non-negative spectral analysis. The robust unsupervised feature selection [60], robust local learning method [61], K-spectral centroid clustering [62], DTW-based barycenter averaging method [63], K-shape [64], U-shape [64] and fuzzy-based clustering [65] all belong to traditional clustering methods. On the contrary, deep learning clustering ones provide rich representations for standard clustering, e.g., the deep temporal clustering [66], unsupervised deep embedding algorithm [67], improved deep embedding algorithm [67], deep temporal clustering representation algorithm [68], and ClusterGAN [69].

III. The Proposed DCRLS

First of all, the problem formulation is introduced. Then, the three key components of DCRLS, namely data augmentation, deep contrastive learning, and self distillation, are described. Finally, the DCRLS-based classification and clustering structures are given.

A. Problem Formulation

Let an arbitrary time equidistant time series sequence denoted by $x_i = \{\{x_{1,1}^{(i)}, ..., x_{1,d}^{(i)}\}, ..., \{\{x_{l,1}^{(i)}, ..., x_{l,d}^{(i)}\}\} \in \mathcal{X}$, where $\mathcal{X} \subseteq \mathbb{R}^{l \times d}$ is the input space, and l and d are the length and number of covariates of x_i, respectively. x_i is univariate if d = 1, and x_i is multivariate, otherwise. The goal is to learn a nonlinear embedding function f_θ, which maps $x_i \in \mathcal{X}$ to its

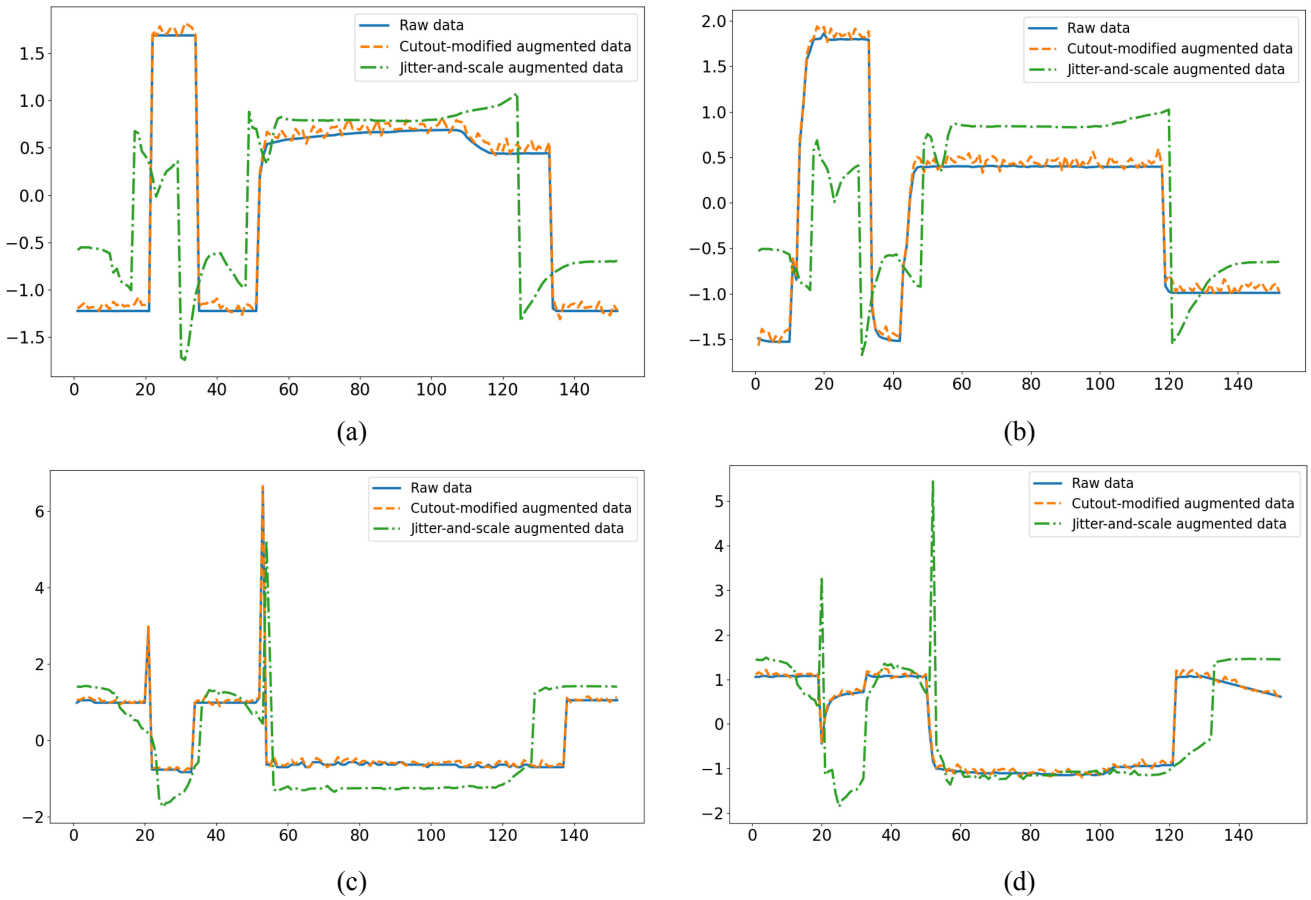


Fig. 2. Example raw data and its jitter-and-scale and Cutout-modified data on the Wafer dataset.

representation r_i that interprets itself for downstream tasks, e.g., time series classification.

B. Data Augmentation

Data augmentation is a widely-used regularization method in deep learning that can significantly improve a model's robustness. DCRLS leverages two augmentation methods to produce different views from the same sample as the input. To be specific, the first method is based on a jitter-and-scale strategy through adding the Gaussian function to the raw data. The second method is a variant of Cutout [70], where a small piece of the raw data is randomly replaced without changing its overall trend. Fig. 2 illustrates examples of the raw data and the resulting jitter-and-scale and Cutout-modified data generated from the Wafer dataset.

C. Deep Contrastive Learning (DeepCL)

DeepCL aims at understanding the contrast similarity of high-level semantic information and that of low-level semantic information between peer residual blocks to capture the discriminative representations from unlabeled data. As illustrated in Fig. 1, there are four parts involving CL, including Residual Block 1, Residual Block 2, Residual Block 3, and the output of ResNet (i.e. the whole model).

CL understands the similarity of different views from the same sample via a contrasting loss function [18]. To be specific, it obtains two feature vectors for every sample from the two augmented views above. Assume there are N input samples. The DeepCL method thus obtains $2N$ feature vectors. Let z_t^i be the i-th feature vector, $i \in \{1, 2, ..., 2N\}$. Let $z_t^{i^+}$ denote the positive sample of z_t^i that comes from the other augmented view of the same input. $(z_t^i, z_t^{i^+})$ is a positive pair. In the meanwhile, we consider $(2N-2)$ feature vectors from other inputs within the same batch as the negative samples of z_t^i. In other words, z_t^i has $(2N-2)$ negative pairs with its negative samples. We use the contrasting loss function in [18], [29], $\mathcal{L}_{CL}$, to maximize the similarity between the positive pair and minimize that between negative pairs, as defined in Eq. (1).

$$\mathcal{L}_{CL} = -\sum_{i=1}^{2N} log \frac{exp(sim(z_t^i, z_t^{i^+})/\tau_{CL})}{\sum_{m=1}^{2N} \mathbb{1}_{[m \neq i]} exp(sim(z_t^i, z_t^m)/\tau_{CL})} \quad (1)$$

where, $sim(u, v) = \frac{u^T v}{||u|| ||v||}$ represents the dot product between l_2 normalized u and v (i.e., cosine smilarity). $\mathbb{1}_{[m \neq i]} \in \{0, 1\}$ is an indicator function, equal to 1 iff $[m \neq i]$ and τ_{CL} is a temperature parameter of $\mathcal{L}_{CL}$. In this paper, we set $\tau_{CL} = 1.0$ (more details are found in Section IV-C).

TABLE I
DETAILS OF THE 36 UCR2018 DATASETS SELECTED. NOTE: THE LAST COLUMN REPRESENTS THE NUMBER OF PARAMETERS OF DCRLS ON SPECIFIC DATASETS.

No.	Dataset	Train	Test	Class	SeriesLength	Type	Parameter (M)
1	Arrow	36	175	3	251	Image	2.564104
2	Beef	30	30	5	470	Spectro	2.678507
3	BeetFly	20	20	2	512	Image	2.696969
4	BirdChicken	20	20	2	512	Image	2.696969
5	Car	60	60	4	577	Sensor	2.732370
6	ChlorineConcentration	467	3840	3	166	Sensor	2.520499
7	Coffee	28	28	2	286	Spectro	2.581031
8	DiatomSizeReduction	16	306	4	345	Image	2.613354
9	DistalPhalanxOutlineAgeGroup	400	139	3	80	Image	2.476381
10	DistalPhalanxOutlineCorrect	600	276	2	80	Image	2.475353
11	ECG200	100	100	2	96	ECG	2.483561
12	ECGFIveDays	23	861	2	136	ECG	2.504081
13	GunPoint	50	150	2	150	Motion	2.511263
14	Ham	109	105	2	431	Spectro	2.655416
15	Herring	64	64	2	512	Image	2.696969
16	Lighting2	60	61	2	637	Sensor	2.761094
17	Meat	60	60	3	448	Spectro	2.665165
18	MiddlePhalanxOutlineAgeGroup	400	154	3	80	Image	2.476381
19	MiddlePhalanxOutlineCorrect	600	291	2	80	Image	2.475353
20	MiddlePhalanxTW	399	154	6	80	Image	2.479465
21	MoteStrain	20	1252	2	84	Sensor	2.477405
22	OSULeaf	200	242	6	427	Image	2.657476
23	Plane	105	105	7	144	Sensor	2.513325
24	ProximalPhalanxOutlineAgeGroup	400	205	3	80	Image	2.476381
25	ProximalPhalanxTW	400	205	6	80	Image	2.479465
26	SonyAIBORobotSurface1	20	601	2	70	Sensor	2.470223
27	SonyAIBORobotSurface2	27	953	2	65	Sensor	2.467658
28	SwedishLeaf	500	625	15	128	Image	2.513341
29	Symbols	25	995	6	398	Image	2.642599
30	ToeSegmentation1	40	228	2	277	Motion	2.576414
31	ToeSegmentation2	36	130	2	343	Motion	2.610272
32	TwoPatterns	1000	4000	4	128	Simulated	2.502033
33	TwoLeadECG	23	1139	2	82	ECG	2.476379
34	Wafer	1000	6164	2	152	Sensor	2.596467
35	Wine	57	54	2	234	Spectro	2.554355
36	WordSyonyms	267	638	25	270	Image	2.596467

The loss function of DeepCL, $\mathcal{L}_{DCL}$, is defined in Eq. (2):

$$\mathcal{L}_{DCL} = \mathcal{L}_{CL}^{Resblk1} + \mathcal{L}_{CL}^{Resblk2} + \mathcal{L}_{CL}^{Resblk3} + \mathcal{L}_{CL}^{output} \quad (2)$$

where, $\mathcal{L}_{CL}^{Resblk1}$, $\mathcal{L}_{CL}^{Resblk2}$, $\mathcal{L}_{CL}^{Resblk3}$ and $\mathcal{L}_{CL}^{output}$ are the contrasting loss functions of Residual Block 1, Residual Block 2, Residual Block 3, and the output of ResNet, respectively.

D. Self Distillation

Self-distillation, a special teacher-student model, transfers its knowledge from high-level to low-level blocks [34]. Let V_1, V_2, V_3 represent the output feature vectors of the first, second and third residual block, respectively. To match each vector with ResNet's soft label (i.e., output feature vector), q, a temporal classifier, ψ, is employed. This classifier comprises an average pooling layer and a dense (i.e., fully-connected) layer. After passing through the classifier, V_i's output is represented as q_i, which is defined in Eq. (3).

$$q_i = f_{softmax}(\psi(V_i)/\tau_{SD}) \qquad i \in \{1, 2, 3\} \quad (3)$$

where, the activation function $f_{softmax}$ is used to compute the probability distribution of a given matrix, and the temperature parameter τ_{SD} is used for self-distillation. As the previous work suggested [34], [71], we set $\tau_{SD} = 1.0$.

The L_2 loss function is utilized to quantify the disparity between q_i and q and is expressed as follows:

$$\mathcal{L}_{KD}^{i} = ||q_i - q||^2 \qquad i \in \{1, 2, 3\} \quad (4)$$

The self-distillation loss for DCRLS, $\mathcal{L}_{KD}$, is the summation of $\mathcal{L}_{KD}^{i}$, shown in Eq. (5)

$$\mathcal{L}_{KD} = \sum_{i=1}^{3} \mathcal{L}_{KD}^{i} \quad (5)$$

Based on $\mathcal{L}_{KD}$ and $\mathcal{L}_{DCL}$, the loss function of DCRLS, $\mathcal{L}$, is written in Eq. (6).

$$\mathcal{L} = \mathcal{L}_{KD} + \mathcal{L}_{DCL} + \epsilon||\theta||^2 \quad (6)$$

where, θ stands for the parameters of DCRLS, and ϵ the coefficient of $||\theta||^2$ (i.e., L_2 regularization). Following [12], [13], we set $\epsilon = 0.0005$.

E. DCRLS-based Classification

To reflect the effectiveness of the instance-level features extracted by DCRLS for classification tasks, this paper adds a classifier behind the first dense (i.e., fully-connected) layer in ResNet. Like most existing time series classification algorithms [29], [30], [31], [72], DCRLS-based classification uses a two-step method. Firstly, unsupervised learning is applied to extract informative features from the data. Subsequently, these features are supplied to the classifier for classification. Following the previous work in [72], we embed a one-nearest-neighbor (1-NN) classifier into the DCRLS-based classification framework.

F. DCRLS-based Clustering

To handle clustering tasks, this paper adds a k-means algorithm behind the first dense layer in ResNet. The DCRLS-based clustering extracts rich representations from data through unsupervised learning and then sends them to the k-means algorithm for clustering. Unlike some previous approaches that include the k-means loss in the overall loss function [66], [68], [73], the DCRLS-based clustering method solely relies on the loss function of DCRLS for parameter updating.

IV. Experiments and Analysis

This section first introduces the experimental setup and performance metrics. Then, it focuses on the hyper-parameter sensitivity and ablation study. Finally, the performance of DCRLS-based classification and clustering is evaluated.

A. Experimental Setup

1) Data Description: In accordance with [66], [68], we evaluate the efficacy of DCRLS-based classification and clustering on 36 commonly used datasets from the UCR2018 archive [74]. The number of categories varies from 2 to 25, and the length of the time series varies from 65 to 637, encompassing various domains such as ECG and motion. For additional information, please refer to Table I.

2) Implementation Details: To prevent overfitting during training, we apply L_2 regularization and dropout techniques. Additionally, we set the decay value of batch normalization to 0.9 to ensure the stability of the training process, following the approach in [11], [13]. Meanwhile, we adopt the Adam optimizer, where the momentum term is fixed to 0.9 and the learning rate is initialized with 0.001 and tuned with a decay value of 0.9. All experiments are run on a computer with Ubuntu 18.04 OS, an Nvidia GTX 1070Ti GPU with 8GB, an Nvidia GTX 1080Ti GPU with 11GB, and an AMD R5 1400 CPU with 16G RAM.

B. Performance Metrics

To compare the performance of various algorithms for time series classification and clustering, we adopt a number of well-known performance metrics.

1) Metrics for Classification: Based on the top-1 accuracy, three metrics, namely, 'win'/'tie'/'lose', mean accuracy (MeanACC), and AVG_rank, are used to rank different supervised algorithms. To be specific, for an arbitrary algorithm, it is associated with 'win', 'tie', and 'lose' values, revealing on how many datasets it is better than, equivalent to, and worse than the other algorithms, respectively. Its 'best' value is the summation of the corresponding 'win' and 'tie' values. As the previous work [11], [12], [13], [44], [52], [53], [56], we use the AVG_rank score based on the Wilcoxon signed-rank test with Holm's alpha (5%) correction to ranking various classification algorithms.

2) Metrics for Clustering: We use a commonly adopted performance indicator, rand index (RI) [66], [68], as defined in Eq. (7).

$$RI = \frac{PTP + NTP}{S(S-1)/2} \tag{7}$$

where, PTP and NTP represent the numbers of positive and negative time series pairs in the clustering, respectively, and S is the dataset size. Note that AVG_RI is the average RI value of a certain algorithm.

C. Hyper-parameter Sensitivity

We use the 36 UCR2018 datasets above to study the impact of hyper-parameter settings on DCRLS.

TABLE II
MeanACC results with different parameter settings on 36 datasets. Abbreviations: L_1–L_1 loss, HL–Huge Loss, CE–Cross Entropy, KL–Kullback Leibler, L_2–L_2 loss, Without–DCRLS without self distillation.

τ_{CL}	DCRLS-based Classification					
	L_1	L_2	KL	CE	HL	Without
0.1	0.7349	0.7392	0.7302	0.7293	0.7345	0.6903
0.2	0.7395	0.7435	0.7349	0.7259	0.7299	0.7045
0.5	0.7418	0.7468	0.7378	0.7349	0.7302	0.7064
0.75	0.7298	0.734	0.7197	0.7201	0.7209	0.6934
1	0.7507	**0.7565**	0.7493	0.7487	0.7435	0.7139

TABLE III
AVG_RI results with different parameter settings on 36 datasets.

τ_{CL}	DCRLS-based Clustering					
	L_1	L_2	KL	CE	HL	Without
0.1	0.6799	0.6794	0.6714	0.6543	0.6602	0.6426
0.2	0.6732	0.6903	0.6769	0.659	0.6698	0.6531
0.5	0.6801	0.6895	0.6769	0.6794	0.6743	0.6539
0.75	0.6689	0.6805	0.6786	0.669	0.6604	0.6492
1	0.6879	**0.6947**	0.6829	0.6809	0.6749	0.6695

1) Effectiveness of τ_{CL}: τ_{CL} is a threshold value for DCRLS to distinguish the similarity of different views from the same sample. Tables II and III show the MeanACC results obtained by DCRLS-based classification and AVG_RI results obtained by DCRLS-based clustering with different τ_{CL} values, respectively. When $\tau_{CL} = 1$, the DCRLS-based classification and clustering result in 0.7565 and 0.6947, the highest MeanACC score and highest AVG_RI value, respectively. It means $\tau_{CL} = 1$ is suitable for DCRLS to capture representations from the input.

TABLE IV
MEANACC AND AVG_RI RESULTS OBTAINED BY VARIOUS DEEPCL-BASED VARIANTS FOR CLASSIFICATION AND CLUSTERING VARIANTS ON 36 DATASETS.

Method	ContrBlock1	ContrBlock2	ContrBlock3	ContrOutput	Classification (MeanACC)	Clustering (AVG_RI)
Baseline				✓	0.6674	0.6327
Baseline-ContrBolck1	✓			✓	0.6699	0.6332
Baseline-ContrBolck2		✓		✓	0.6882	0.6401
Baseline-ContrBolck3			✓	✓	0.6793	0.6379
Baseline-ContrBolck1-ContrBolck2	✓	✓		✓	0.6902	0.6469
Baseline-ContrBolck1-ContrBolck3	✓		✓	✓	0.6935	0.6501
Baseline-ContrBolck2-ContrBolck3		✓	✓	✓	0.7001	0.6602
DeepCL	✓	✓	✓	✓	**0.7139**	**0.6695**

TABLE V
MEANACC AND AVG_RI RESULTS OBTAINED BY DEEPCL WITH VARIOUS SELF DISTILLATION TECHNIQUES FOR CLASSIFICATION AND CLUSTERING ON 36 DATASETS.

Method	BYOT	SAD	TSD		Self Distillation (Ours)	Classification (MeanACC)	Clustering (AVG_RI)
			with KL	with L_2			
DeepCL						0.7139	0.6695
DeepCL-BYOT	✓					0.7493	0.6829
DeepCL-SAD		✓				0.7398	0.6793
DeepCL-TSD with KL			✓			0.7402	0.6756
DeepCL-TSD with L_2				✓		0.7507	0.6889
DCRLS					✓	**0.7565**	**0.6947**

2) Effectiveness of KD loss: A teacher-student model needs to choose an appropriate KD loss to measure the difference between high- and low-level features. Tables II and III also show the impact of KD loss on DCRLS. One can observe that self distillation improves the performance of DCRLS mainly because supervising low-level semantics with high-level ones helps regularize the model itself. Besides, L_2 outperforms the other losses in both classification and clustering. Therefore, we choose L_2 to promote the knowledge flow within the model.

D. Ablation Study

We investigate the effectiveness of two important components of DCRLS on 36 datasets.

1) Effectiveness of Deep Contrastive Learning (DeepCL): As aforementioned, our DeepCL has four parts involving CL, i.e. Residual Blocks 1, 2, and 3, and the output of ResNet, i.e., the whole model, as shown in Fig. 1. Let ContrBlock1, ContrBlock2, ContrBlock3, and ContrOutput denote the contrastive learning at the outputs of Residual Blocks 1, 2, 3, and ResNet, respectively.

To evaluate the impact of low-level semantic information on extraction of high-level semantic information, we compare a number of DeepCL-based variants for classification and clustering variants on 36 datasets.

- Baseline: it uses the proposed ResNet structure (i.e. the whole model) for feature extraction and enables CL at the outputs of ResNet.
- Baseline-ContrBlock1: Baseline with extra CL enabled at the outputs of Residual Block 1.
- Baseline-ContrBlock2: Baseline with extra CL enabled at the outputs of Residual Block 2.
- Baseline-ContrBlock3: Baseline with extra CL enabled at the outputs of Residual Block 3.
- Baseline-ContrBlock1-ContrBlock2: Baseline with extra CL enabled at the outputs of Residual Blocks 1 and 2.
- Baseline-ContrBlock1-ContrBlock3: Baseline with extra CL enabled at the outputs of Residual Blocks 1 and 3.
- Baseline-ContrBlock2-ContrBlock3: Baseline with extra CL enabled at the outputs of Residual Blocks 2 and 3.
- DeepCL: Baseline with extra CL enabled at the outputs of Residual Blocks 1, 2 and 3.

As shown in Table IV, with continuous addition of CL on low-level outputs, the accuracy values of classification and clustering are both increasing. For example, the MeanACC value of Baseline is 0.6674 while that of the Baseline-ContrBolck2 is 0.6882. This, to a certain extend, reflects the importance of low level semantic information to representation learning. DeepCL understands the contrast similarity of high-level semantic information and that of low-level semantic information between peer Residual Blocks, enhancing the ability of representation learning on DCRLS. This is why DeepCL achieves the best performance among all compared variants.

2) Effectiveness of Self Distillation: To evaluate the proposed self distillation method, we select four advanced self-distillation methods for performance comparison, including BYOT, SAD, TSD with KL, and TSD with L_2. These self distillation methods are listed below.

- BYOT: the best your own teacher method with the Kullback Leibler (KL) function used to measure the difference between high- and low-level feature vectors [34].
- SAD: the layer-wise attention self-distillation method transferring knowledge from high levels to low levels [75].
- TSD with KL: the transitive self-distillation method with the Kullback Leibler (KL) function as its loss function [76].

- TSD with L_2: the transitive self-distillation method with the L_2 function as its loss function [76].
- Self Distillation (Ours): using the L_2 function to measure the difference between high- and low-level feature vectors.

As shown in Fig. 1, DCRLS consists of DeepCL and the proposed self distillation. To validate the impact of self distillation on DCRLS, we compare it with a number of DeepCL variants listed below.

- DeepCL-BYOL: DeepCL with the best your own teacher method.
- DeepCL-SAD: DeepCL with the layer-wise attention self-distillation method.
- DeepCL-TSD with KL: DeepCL with the transitive self-distillation method, where the KL function is adopted.
- DeepCL-TSD with L_2: DeepCL with the transitive self-distillation method, where the L_2 function is used.
- DCRLS: DeepCL with the proposed self distillation method.

The MeanACC and AVG_RI results obtained by DeepCL with various self distillation techniques for classification and clustering on 36 datasets are shown in V. Clearly, DeepCL results in the worst performance on both classification and clustering. This is because self distillation can effectively promotes knowledge flow within the model, enhancing its robustness. In addition, DCRLS outperforms DeepCL-BYOT, DeepCL-SAD, DeepCL-TSD with KL, and DeepCL-TSD with L_2 with respect to MeanACC and AVG_RI, which means the proposed self distillation achieves better performance than a number of well-known methods when working with DeepCL on time series classification and clustering tasks.

E. Evaluation of DCRLS-based Classification

To evaluate the performance of DCRLS-based classification, we compare it with a number of existing CL-based classification algorithms against 'win'/'lose'/'tie', MeanACC, and AVG_rank, as listed below.

- T-Loss: a method based on random sub-series technique ResNet as its feature extractor [72].
- SimCLR: a modified version of SimCLR adapted to time series classification with ResNet as its feature extractor [18].
- CRL with KL: a contrastive representation leaning method with the Kullback Leibler (KL) function as its self-distillation loss function and ResNet as its feature extractor [20].
- CRL with L_2: a contrastive representation leaning method with the L_2 function as its self-distillation loss function and ResNet as its feature extractor [20].
- KNCRL with KL: a KD-based contrastive leaning method with the KL function as its KD loss function and ResNet as its feature extractor [21].
- KNCRL with L_2: a KD-based contrastive representation leaning method with the L_2 function as its KD loss function and ResNet as its feature extractor [21].

Note that the algorithms above all take 1-NN as their classifier for classification. The top-1 accuracy results obtained with different algorithms on 36 selected UCR2018 datasets are shown in Table VI. The DCRLS-based classification is the best among all algorithms as it obtains the highest MeanACC and 'best' values, 28 and 0.7565, and the smallest AVG_rank score, 1.7639. Rather than focusing on the contrastive similarity of high-level semantic information only, DCRLS-based classification considers the contrast similarity of multi-level semantic information, especially that of low-level semantic information between peer Residual Blocks, which is beneficial for mining instance-level features between different perspectives from the same sample. KNCRL with L_2 takes advantage of CL and KD to distinguish the similarity of different views from the same sample and the similarity with the views from different samples. That brings it second in terms of MeanACC and AVG_rank, namely 0.7009 and 3.7361. On the contrary, it is difficult for T-Loss to generate proper random sub-series as positive samples when mining discriminate shapelets from the input, resulting in its performing the worst among all the algorithms.

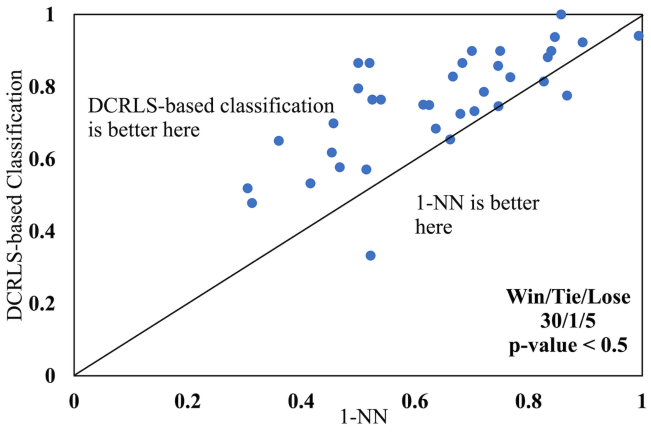


Fig. 3. Accuracy plot showing the performance difference between DCRLS-based classification and 1-NN.

To further study the performance of DCRLS on classification, we compare it with a separate 1-NN algorithm on 36 datasets. Fig. 3 depicts the accuracy plot of DCRLS-based classification against 1-NN for each dataset. The results indicate that DCRLS-based classification outperforms 1-NN in 30 cases, ties in one case, and underperforms in five cases, with a p-value less than 0.5 (about 0.0034). The performance of DCRLS-based classification outstrips significantly that of 1-NN. That is because the instance-level features extracted by DCRLS are more conducive for the 1-NN classifier to distinguish the similarity between different samples.

F. Evaluation of DCRLS-based Clustering

To evaluate the performance of DCRLS-based clustering, we compare it with a number of existing clustering algorithms against three performance metrics: 'win'/'tie'/'lose', AVG_RI, and AVG_rank. These algorithms are listed blow:

- K-means: a simple yet elegant approach that partitions a dataset into K distinct, non-overlapping clusters [68].

TABLE VI
THE TOP-1 ACCURACY RESULTS OF DIFFERENT CLASSIFICATION ALGORITHMS ON 36 DATASETS WHEN USING 1-NN.

Dataset	T-Loss	SimCLR	CRL		KNCRL		DCRLS-based Classification
			with KL	with L_2	with KL	with L_2	
Arrow	0.4457	**0.5714**	0.5485	0.5429	0.5142	0.5086	**0.5714**
Beef	0.4333	0.7	0.7333	0.7	0.7	0.7333	**0.8667**
BeetFly	0.8	0.8	0.85	0.85	**0.9**	**0.9**	**0.9**
BirdChicken	0.8	**0.95**	0.85	**0.95**	0.9	0.85	0.9
Car	0.333	**0.55**	0.517	**0.55**	0.517	0.517	0.5333
ChlorineConcentration	**0.5333**	0.462	0.5183	0.4589	0.451	0.4531	0.4786
Coffee	**1**	**1**	**1**	**1**	**1**	**1**	**1**
DiatomSizeReduction	0.3529	0.4935	0.3987	0.5817	0.585	0.6732	**0.7255**
DistalPhalanxOutlineAgeGroup	0.518	0.6403	**0.7051**	0.7338	0.6547	0.5827	0.6547
DistalPhalanxOutlineCorrect	0.6123	0.6592	0.6232	0.6703	0.6703	0.6667	**0.6992**
ECG200	0.81	0.86	0.89	0.88	0.86	0.89	**0.9**
ECGFIveDays	0.5807	0.6702	0.7405	0.7329	0.698	0.6783	**0.7468**
GunPoint	0.6533	0.8067	0.8133	0.8133	0.8133	0.8333	**0.8667**
Ham	0.6857	0.6667	0.6667	0.6667	0.6762	0.6476	**0.7333**
Herring	0.6875	0.7188	0.7188	0.7031	**0.75**	0.7188	**0.75**
Lighting2	0.6721	**0.8197**	0.7705	0.7705	0.8033	0.7869	0.7869
Meat	0.5	0.5667	0.5833	0.55	0.55	0.7	**0.8667**
MiddlePhalanxOutlineAgeGroup	0.5714	0.4805	0.4674	0.4416	0.4416	0.4751	**0.5195**
MiddlePhalanxOutlineCorrect	**0.6186**	0.6014	**0.6186**	0.6254	0.5842	0.5979	**0.6186**
MiddlePhalanxTW	0.2922	0.5519	0.5029	0.5714	0.5584	0.5519	**0.5779**
MoteStrain	0.5719	0.6414	0.6302	0.5686	0.6022	0.6246	**0.7651**
OSULeaf	0.2355	0.2934	0.5992	0.6033	0.5992	0.6033	**0.7652**
Plane	0.8952	0.9333	**0.9905**	0.8952	0.9619	0.9714	0.9238
ProximalPhalanxOutlineAgeGroup	0.5073	0.5659	0.5415	0.5415	0.5415	0.8341	**0.8585**
ProximalPhalanxTW	0.361	0.8049	0.4592	0.6245	0.7561	0.6732	**0.7512**
SonyAIBORobotSurface1	0.5524	0.7521	0.7121	0.7504	0.777	0.7438	**0.8153**
SonyAIBORobotSurface2	0.6254	0.6737	0.6726	0.7009	0.6789	0.703	**0.7765**
SwedishLeaf	0.5	0.6048	0.5872	0.616	0.5376	0.616	**0.6512**
Symbols	0.2111	0.6462	0.7598	0.6834	0.6915	0.7327	**0.8271**
ToeSegmentation1	0.6095	0.6053	0.6623	0.6667	0.7237	0.6053	**0.8289**
ToeSegmentation2	0.6562	0.6846	0.6923	0.8231	0.6538	0.8538	**0.9385**
TwoPatterns	0.8773	0.8215	0.82	0.82	0.83	0.85	**0.8825**
TwoLeadECG	0.5637	0.6137	**0.7305**	0.6049	0.6383	0.6883	0.6848
Wafer	0.8874	0.932	0.9199	0.9281	0.9156	0.914	**0.9416**
Wine	0.7037	0.7222	0.7407	0.7222	0.7407	0.7593	**0.7963**
WordSyonyms	0.1317	0.163	0.2147	0.1881	0.2947	0.2947	**0.3328**
Win	1	2	3	0	0	0	**23**
Tie	2	3	2	3	2	2	**5**
Lose	33	31	31	33	34	34	**8**
Best	3	5	5	3	2	2	**28**
MeanACC	0.5775	0.6674	0.6735	0.6814	0.6825	0.7009	**0.7565**
AVG_rank	6.0557	4.2222	4.05556	4.1111	4.0556	3.7361	**1.7639**

- UDFS: a combination of discriminative analysis and $l_{2,1}$-norm minimization for unsupervised feature selection [58].
- NDFS: an unsupervised feature selection algorithm using non-negative spectral analysis [59].
- RUFS: a robust unsupervised feature selection algorithm using local learning regularized robust nonnegative matrix factorization [60].
- RSFS: a robust local learning method for graph Laplacian and spectral regression construction [61].
- KSC: a K-spectral centroid clustering algorithm to mine patterns of temporal variation [62].
- KBDB: a dynamic time warping-based barycenter averaging method [63].
- K-shape: a domain-independent, highly accurate, and efficient clustering approach for partitional, hierarchical, and spectral clustering [64].
- U-shapelet: a shapelet method for time series clustering [64].
- DTC: a deep temporal clustering algorithm that integrates dimensionality reduction and temporal clustering into end-to-end learning [66].
- DEC: an unsupervised deep embedding algorithm for clustering analysis that learns feature representations and clustering assignments using deep neural network [67].
- IDEC: an improved version of DEC [67].
- DTCR: a deep temporal clustering representation algorithm integrating the temporal reconstruction and K-means objective into the seq2seq model [68].

Table VII shows the RI results of different clustering algorithms on 36 datasets. DTCR and DCRLS-based clustering are the best and second-best among all compared algorithms. DTCR integrates the temporal reconstruction and K-means into the seq2seq model, jointly optimizing its parameters for cluster structure improvement and temporal representation mining. However, DTCR is mainly used for addressing cluster-

TABLE VII
THE RI RESULTS OF DIFFERENT CLUSTERING ALGORITHMS ON 36 DATASETS.

Dataset	K-means	UDFS	NDFS	RUFS	RSFS	KSC	KDBA	K-shape	U-shapelet	DTC	DEC	IDEC	DTCR	DCRLS-based Clustering
Arrow	0.6095	0.7254	0.7381	**0.7476**	0.7108	0.7254	0.7222	0.7254	0.646	0.6692	0.5817	0.621	0.6868	0.6229
Beef	0.6713	0.6759	0.7034	0.7149	0.6975	0.7057	0.6713	0.5402	0.6966	0.6345	0.5954	0.6276	0.8046	**0.8115**
BeetFly	0.4789	0.4949	0.5579	0.6053	0.6516	0.6053	0.6052	0.6053	0.7314	0.5211	0.4947	0.6053	**0.9**	0.6632
BirdChicken	0.4947	0.4947	0.7361	0.5579	0.6632	0.7316	0.6053	0.6632	0.5579	0.4947	0.4737	0.4789	**0.8105**	0.7316
Car	0.6345	0.6757	0.626	0.6667	0.6708	0.6898	0.6254	0.7028	0.6418	0.6695	0.6859	0.687	**0.7501**	0.7011
ChlorineConcentration	0.5241	0.5282	0.5225	0.533	0.5316	0.5256	0.53	0.4111	0.5318	0.5353	0.5348	0.535	0.5357	**0.5386**
Coffee	0.746	0.8624	**1**	0.5467	**1**	**1**	0.4851	**1**	**1**	0.4841	0.4921	0.5767	0.9286	0.746
DiatomSizeReduction	0.9583	0.9583	0.9583	0.9333	0.9137	**1**	0.9583	**1**	0.7083	0.8792	0.9294	0.7347	0.9682	0.7042
DistalPhalanxOutlineAgeGroup	0.6171	0.6531	0.6239	0.6252	0.6539	0.6535	0.675	0.602	0.6273	0.7812	0.7785	0.7786	**0.7825**	0.6197
DistalPhalanxOutlineCorrect	0.5252	0.5362	0.5362	0.5252	0.5327	0.5235	0.5203	0.5252	0.5098	0.501	0.5029	0.533	**0.6075**	0.5694
ECG200	0.6315	0.6533	0.6315	**0.7018**	0.6916	0.6315	0.6018	**0.7018**	0.5758	0.6018	0.6422	0.6233	0.6648	**0.7018**
ECGFIveDays	0.4783	0.502	0.5573	0.502	0.5953	0.5257	0.5573	0.502	0.5968	0.5016	0.5103	0.5114	**0.9638**	0.6467
GunPoint	0.4971	0.5029	0.5102	0.6498	0.4994	0.4971	0.542	0.6278	0.6278	0.54	0.4981	0.4974	0.6398	**0.6779**
Ham	0.5025	0.5219	0.5362	0.5107	0.5127	0.5362	0.5141	0.5311	0.5362	0.5648	**0.5963**	0.4956	0.5362	0.5451
Herring	0.4965	0.5099	0.5164	0.5238	0.5151	0.494	0.5164	0.4965	0.5417	0.5045	0.5099	0.5099	**0.579**	0.5322
Lighting2	0.4966	0.5119	0.5373	0.5729	0.5269	0.6263	0.5119	**0.6548**	0.5192	0.577	0.5311	0.5519	0.5913	0.623
Meat	0.6595	0.6483	0.6635	0.6578	0.6657	0.6723	0.6816	0.6575	0.6742	0.322	0.6475	0.622	**0.9763**	0.7633
MiddlePhalanxOutlineAgeGroup	0.5351	0.5269	0.535	0.5315	0.5473	0.5364	0.5513	0.5105	0.5396	0.5757	0.7059	0.68	**0.7982**	0.5751
MiddlePhalanxOutlineCorrect	0.5	0.5431	0.5047	0.5114	0.5149	0.5014	0.5563	0.5114	0.5218	0.5272	0.5423	0.5423	**0.5617**	0.5248
MiddlePhalanxTW	0.0983	0.1225	0.1919	0.792	0.8062	0.8187	0.8046	0.6213	0.792	0.7115	0.859	0.8626	**0.8638**	0.815
MoteStrain	0.4947	0.5579	0.6053	0.5579	0.6168	0.6632	0.4789	0.6053	0.4789	0.5062	0.7435	0.7342	**0.7686**	0.6337
OSULeaf	0.5615	0.5372	0.5622	0.5497	0.5665	0.5714	0.5541	0.5538	0.5525	0.7329	0.7484	0.7607	**0.7739**	0.7652
Plane	0.9081	0.8949	0.8954	0.922	0.9314	0.9603	0.9225	0.9901	**1**	0.904	0.9447	0.9447	0.9549	0.9509
ProximalPhalanxOutlineAgeGroup	0.5288	0.4997	0.5463	0.578	0.5384	0.5305	0.5192	0.5617	0.5206	0.743	0.4263	**0.8091**	**0.8091**	0.5603
ProximalPhalanxTW	0.4789	0.4947	0.6053	0.5579	0.5211	0.6053	0.5211	0.5211	0.4789	0.838	0.8189	**0.903**	0.9023	0.8493
SonyAIBORobotSurface1	0.7721	0.7695	0.7721	0.7787	0.7928	0.7726	0.7988	0.8084	0.7639	0.5563	0.5732	0.69	**0.8769**	0.7311
SonyAIBORobotSurface2	0.8697	0.8745	0.8865	0.8756	0.8948	**0.9039**	0.8684	0.5617	0.877	0.7012	0.6514	0.6572	0.8354	0.683
SwedishLeaf	0.4987	0.4923	0.55	0.5192	0.5038	0.4923	0.55	0.5533	0.6154	0.8871	0.8837	0.8893	0.9223	**0.9243**
Symbols	0.881	0.8548	0.8562	0.8525	0.906	0.8982	**0.9774**	0.8373	0.9603	0.9053	0.8841	0.8857	0.9168	0.8912
ToeSegmentation1	0.4873	0.4921	0.5873	0.5429	0.4968	0.5	0.6143	0.6143	0.5873	0.5077	0.4984	0.5017	0.5659	**0.6926**
ToeSegmentation2	0.5257	0.5257	0.5968	0.5968	0.5826	0.5257	0.5573	0.5257	0.502	0.5348	0.4991	0.4991	0.8286	**0.8569**
TwoPatterns	0.8529	0.8259	0.853	0.8385	**0.8588**	0.8585	0.8446	0.8046	0.7757	0.6251	0.6293	0.6338	0.6984	0.693
TwoLeadECG	0.5476	0.5495	0.6328	0.8246	0.5635	0.5464	0.5476	0.8246	0.5404	0.5116	0.5007	0.5016	**0.7114**	0.5279
Wafer	0.4925	0.4925	0.5263	0.5263	0.4925	0.4925	0.4925	0.4925	0.4925	0.5324	0.5679	0.5597	0.7338	**0.8076**
Wine	0.4984	0.4987	0.5123	0.5021	0.5033	0.5006	0.5064	0.5001	0.5033	0.4906	0.4913	0.5157	**0.6271**	0.5353
WordSyonyms	0.8775	0.8697	0.876	0.8861	0.8817	0.8727	0.8159	0.7844	0.823	0.8855	0.8893	0.8947	**0.8984**	0.8902
Win	0	0	0	1	1	1	1	1	0	0	1	1	**16**	7
Lose	0	0	1	1	1	2	0	**3**	2	0	0	1	1	1
Tie	36	36	35	34	34	33	35	32	34	36	35	34	**19**	28
Best	0	0	1	2	2	3	1	4	2	0	1	2	**17**	8
AVG_RI	0.5957	0.6077	0.6403	0.6477	0.6542	0.6582	0.6335	0.6419	0.6402	0.6238	0.6351	0.6515	**0.7714**	0.6974
AVG_rank	10.7361	9.5556	7.1667	7.3333	6.8056	7	7.8889	7.9444	8.0278	8.7083	8.7083	7.5694	**2.75**	4.806

specific problems and it has complicated model structure. On the other hand, DCRLS-based clustering updates the model's parameters via self-supervised learning and it adopts a K-means algorithm for feature classification. Despite its simple structure, DCRLS-based clustering achieves decent performance on the 36 datasets, thanks to the DCRLS's strong ability for feature extraction.

To further study the effectiveness of DCRLS in time series clustering, we compare DCRLS-based clustering with a separate K-means algorithm on 36 datasets. Fig. 4 depicts the RI plot of DCRLS-based clustering against K-means for each dataset. The results show that DCRLS-based clustering achieves 'win'/'tie'/ 'loss' in 30/1/5 cases, respectively, with a p-value less than 0.5 (about 0.0048). The performance of DCRLS-based clustering is significantly better than that of K-means. That is because the DCRLS-based clustering is provided with sufficient features captured by DCRLS, especially those hiding deeply in the input data.

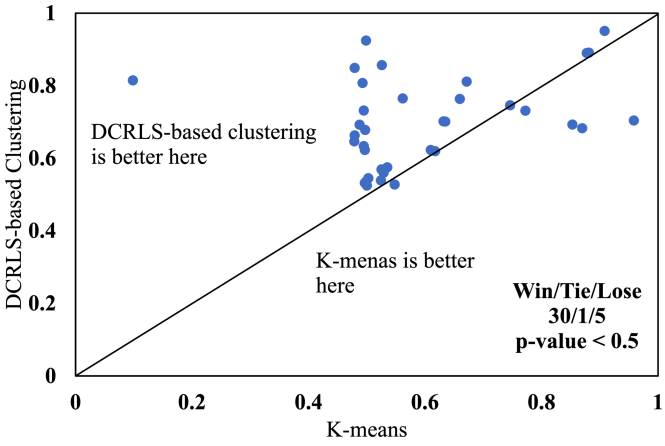


Fig. 4. RI plot showing the performance difference between DCRLS-based clustering and K-means.

V. CONCLUSION

In the proposed DCRLS, deep contrastive learning pays attention to the contrast similarity of multi-level semantic information to mine instance-level features between different perspectives from the same sample, while self distillation transfers knowledge from high-level to low-level blocks to

regularize DCRLS during knowledge transfer. Experimental results demonstrate that DCRLS obtains promising results in both classification and clustering tasks. Specifically, DCRLS-based classification wins on 28 datasets in terms of mean accuracy, 'win'/'tie'/'lose' measure, and AVG_rank, compared with six classification algorithms. DCRLS-based clustering wins on 8 datasets and takes the second best position among 14 clustering algorithms regarding AVG_RI, 'win'/'tie'/'lose' measure, and AVG_rank. on With 36 UCR2018 datasets considered, DCRLS-based classification and clustering result in significant performance improvement when compared with separate 1-NN and K-mean algorithms, respectively. Meanwhile, DCRLS is generic and has potential to be applied to other domains.

Due to the limited GPU resources for training, this work considered 36 out of 128 UCR2018 datasets for model evaluation and comparison. In the future, we will validate DCRLS on all UCR2018 datasets in the time series domain and on larger datasets in other domains, such as ImageNet.

ACKNOWLEDGMENT

The authors thank the editors and reviewers for their valuable suggestions to improve the quality of this work.

REFERENCES

[1] S. Supriya, S. Siuly, H. Wang, and Y. Zhang, "Eeg sleep stages analysis and classification based on weighed complex network features," *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 5, no. 2, pp. 236–246, 2021.

[2] P. Fergus, C. Chalmers, C. C. Montanez, D. Reilly, P. Lisboa, and B. Pineles, "Modelling segmented cardiotocography time-series signals using one-dimensional convolutional neural networks for the early detection of abnormal birth outcomes," *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 5, no. 6, pp. 882–892, 2021.

[3] Y. Yao, H. Chen, and X. Yao, "Discriminative learning in the model space for symbolic sequence classification," *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 5, no. 2, pp. 154–167, 2021.

[4] A. M. Roy and J. Bhaduri, "Densesph-yolov5: An automated damage detection model based on densenet and swin-transformer prediction head-enabled yolov5 with attention mechanism," *Adv. Eng. Informatics*, vol. 56, pp. 1–16, 2023.

[5] P. Jain, S. Jain, O. R. Zaïane, and A. Srivastava, "Anomaly detection in resource constrained environments with streaming data," *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 6, no. 3, pp. 7103–7125, 2022.

[6] E. Tsang and J. Chen, "Regime change detection using directional change indicators in the foreign exchange market to chart brexit," *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 2, no. 3, pp. 185–193, 2018.

[7] S. Jamil and A. M. Roy, "An efficient and robust phonocardiography (pcg)-based valvular heart diseases (vhd) detection framework using vision transformer (vit)," *Comp. Bio. Med.*, vol. 158, pp. 1–15, 2023.

[8] S. Wang, Y. Yin, D. Wang, Z. Lv, Y. Wang, and Y. Jin, "An interpretable deep neural network for colorectal polyp diagnosis under colonoscopy," *Knowl. Based Syst.*, vol. 234, pp. 1–14, 2022.

[9] A. R. Shirazi and Y. Jin, "A strategy for self-organized coordinated motion of a swarm of minimalist robots," *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 1, no. 5, pp. 326–338, 2017.

[10] H. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: a review," *Data Min. Knowl. Disc.*, vol. 33, pp. 917–963, 2019.

[11] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," *In Proc. IEEE IJCNN 2017*, pp. 1578–1585, 2017.

[12] H. Fawaz, B. Lucas, G. Forestier, C. Pelletier, D. Schmidt, J. Weber, G. Webb, L. Idoumghar, P.-A. Muller, and F. Petitjean, "Inceptiontime: finding alexnet for time series classification," *Data Min. Knowl. Disc.*, vol. 34, pp. 1936–1962, 2020.

[13] Z. Xiao, X. Xu, H. Xing, S. Luo, P. Dai, and D. Zhan, "Rtfn: A robust temporal feature network for time series classification," *Inform. Sciences*, vol. 571, pp. 65–86, 2021.

[14] A. Jaiswal, A. Babu, M. Zadeh, D. Banerjee, and F. Makedon, "A survey on contrastive self-supervised learning," *arXiv preprint arXiv: 2011.00362v3*, 2021.

[15] Y. Lin, Y. Gou, X. Liu, J. Bai, J. Lv, and X. Peng, "Dual contrastive prediction for incomplete multi-view representation learning," *IEEE Trans. Pattern Anal.*, pp. 1–14, 2022.

[16] Y. Li, M. Yang, D. Peng, T. Li, J. Huang, and X. Peng, "Twin contrastive learning for online clustering," *Int. J. Comput. Vision*, vol. 130, no. 9, p. 2205–2221, sep 2022.

[17] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," *In Proc. CVPR*, pp. 9726–9735, 2020.

[18] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," *In Proc. ICML*, 2020.

[19] X. Chen and K. He, "Exploring simple siamese representation learning," *In Proc. CVPR 2021*, pp. 15 745–15 753, 2021.

[20] M. Jin, Y. Zheng, Y. Li, C. Gong, C. Zhou, and S. Pan, "Multi-scale contrastive siamese networks for self-supervised graph representation learning," *In Proc. IJCAI 2021*, 2021.

[21] P. Bhat, E. Arani, and B. Zonooz, "Distill on the go: Online knowledge distillation in self-supervised learning," *In Proc. IEEE/CVF CVPRW*, pp. 2672–2681, 2021.

[22] P. Hu, H. Zhu, J. Lin, D. Peng, Y.-P. Zhao, and X. Peng, "Unsupervised contrastive cross-modal hashing," *IEEE Trans. Pattern Anal.*, vol. 45, no. 3, pp. 3877–3889, 2023.

[23] Y. Li, P. Hu, Z. Liu, D. Peng, J. Zhou, and X. Peng, "Contrastive clustering," *In Proc. AAAI*, vol. 35, no. 10, pp. 8547–8555, 2021.

[24] X. Shu, B. Xu, L. Zhang, and J. Tang, "Multi-granularity anchor-contrastive representation learning for semi-supervised skeleton-based action recognition," *IEEE Trans. Pattern Anal.*, pp. 1–18, 2022.

[25] B. Xu, X. Shu, and Y. Song, "X-invariant contrastive augmentation and representation learning for semi-supervised skeleton-based action recognition," *IEEE Trans. Image Process.*, vol. 31, pp. 3852–3867, 2022.

[26] Y. Lin, M. Yang, J. Yu, P. Hu, C. Zhang, and X. Peng, "Graph matching with bi-level noisy correspondence," *arXiv preprint arXiv: 2212.04085*, 2022.

[27] J. Yu, J. Liu, Y. Cheng, R. Feng, and Y. Zhang, "Modality-aware contrastive instance learning with self-distillation for weakly-supervised audio-visual violence detection," *In Proc. ACM MM*, p. 6278–6287, 2022.

[28] X. Wang, J. Zhu, Z. Yan, Z. Zhang, Y. Zhang, Y. Chen, and H. Li, "Last: Label-free self-distillation contrastive learning with transformer architecture for remote sensing image scene classification," *IEEE Geosci. Remote S.*, vol. 19, pp. 1–5, 2022.

[29] M. Mohsenvand, M. R. Izadi, and P. Maes, "Contrastive representation learning for electroencephalogram classification," *In Proc. Machine Learning for Health NeurIPS Workshop*, 2020.

[30] E. Eldele, M. Ragab, Z. Chen, M. Wu, C. Kwoh, X. Li, and C. Guan, "Time-series representation learning via temporal and contextual contrasting," *In Proc. IJCAI 2021*, 2021.

[31] Z. Yue, Y. Wang, J. Duan, T. Yang, C. Huang, Y. Tong, and B. Xu, "Ts2vec: Towards universal representation of time series," *In Proc. AAAI*, vol. 36, pp. 8980–8987, 2022.

[32] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal.*, vol. 35, no. 8, p. 1798–1828, 2013.

[33] P. Munro, "Backpropagation," *Sammut, C., Webb, G.I. (eds) Encyclopedia of Machine Learning*, 2011.

[34] L. Zhang, J. Song, A. Gao, J. Chen, C. Bao, and K. Ma, "Be your own teacher: Improve the performance of convolutional neural networks via self distillation," *In Prco. ICCV 2019*, pp. 3712–3721, 2019.

[35] J. Lines and A. Bagnall, "Time series classification with ensembles of elastic distance measures," *Data Min. Knowl. Disc.*, vol. 29, pp. 565–592, 2015.

[36] J. Lines, S. Taylor, and A. Bagnall, "Time-series classification with cote: The collective of transformation-based ensembles," *IEEE Trans. Knowl. Data En.*, vol. 27, no. 9, pp. 2522–2535, 2015.

[37] J. Lines *et al.*, "Time series classification with hive-cote: The hierarchical vote collective of transformation-based ensembles," *ACM Trans. Knowl. Discov. D.*, vol. 21, no. 52, pp. 1–35, 2018.

[38] M. Middlehurst, J. Large, M. Flynn, J. Lines, A. Bostrom, and A. Bagnall, "Hive cote 2.0: a new meta ensemble for time series classification," *Machine Learning*, vol. 110, p. 3211–3243, 2021.

[39] K. Fauvel, É. Fromont, V. Masson, P. Faverdin, and A. Termier, "Local cascade ensemble for multivariate data classification," *arXiv preprint arXiv:2005.03645*, 2020.

[40] I. Karlsson, P. Papapetrou, and H. Boström, "Generalized random shapelet forests," *Data Min. Knowl. Disc.*, vol. 30, no. 5, pp. 1053–1083, 2016.

[41] M. G. Baydogan and G. Runger, "Time series representation and similarity based on local autopatterns," *Data Min. Knowl. Disc.*, vol. 30, no. 2, pp. 476–509, 2016.

[42] A. Quattoni, S. Wang, L. Morency, M. Collins, and T. Darrell, "Hidden conditional random fields," *IEEE Trans. Pattern Anal.*, vol. 29, no. 10, pp. 1848–1852, 2007.

[43] F. Baldán and J. Benítez, "Multivariate times series classification through an interpretable representation," *Inform. Sciences*, vol. 569, pp. 596–614, 2021.

[44] Z. Xiao, X. Xu, H. Zhang, and E. Szczerbicki, "A new multi-process collaborative architecture for time series classification," *Knowl. Based Syst.*, vol. 220, pp. 1–11, 2021.

[45] W. Cheng, P. Sunganthan, and R. Katuwal, "Time series classification using diversified ensemble deep random vector functional link and resnet features," *Appl. Soft Comput.*, vol. 112, pp. 1–12, 2021.

[46] H. Zhu, J. Zhang, H. Cui, K. Wang, and Q. Tang, "Tcran: Multivariate time series classification using residual channel attention networks with time correction," *Appl. Soft Comput.*, vol. 114, pp. 1–10, 2022.

[47] H. Xing, Z. X. R. Qu, Z. Zhu, and B. Zhao, "An efficient federated distillation learning system for multi-task time series classification," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–12, 2022.

[48] Z. Huang, C. Yang, X. Chen, X. Zhou, G. Chen, T. Huang, and W. Gui, "Functional deep echo state network improved by a bi-level optimization approach for multivariate time series classification," *Appl. Soft Comput.*, vol. 106, pp. 1–12, 2021.

[49] A. Dempster, D. Petitjean, and G. Webb, "Rocket: exceptionally fast and accurate time series classification using random convolutional kernels," *Data Min. Knowl. Disc.*, vol. 34, p. 1454–1495, 2020.

[50] A. Dempster *et al.*, "Minirocket: A very fast (almost) deterministic transform for time series classification," *In Proc. KDD'21*, p. 248–257, 2021.

[51] K. Kashiparekh, J. Narwariya, P. Malhotra, L. Vig, and G. Shroff, "Convtimenet: A pre-trained deep convolutional neural network for time series classification," *In Proc. IJCNN*, pp. 1–8, 2019.

[52] Z. Xiao, X. Xu, H. Xing, R. Qu, F. Song, and B. Zhao, "Rnts: Robust neural temporal search for time series classification," *In Proc. IJCNN*, pp. 1–8, 2021.

[53] F. Karim, S. Majumdar, H. Darabi, and S. Harford, "Multivariate lstm-fcns for time series classification," *Neural Networks*, vol. 116, pp. 237–245, 2019.

[54] S. H. Huang, L. Xu, and C. Jiang, "Residual attention net for superior cross-domain time sequence modeling," *Fintech with Artificial Intelligence, Big Data, and Blockchain. Springer*, 2021.

[55] H. Xing, Z. Xiao, D. Zhan, S. Luo, P. Dai, and K. Li, "Selfmatch: Robust semisupervised time-series classification with self-distillation," *Int. J. Intell. Syst.*, vol. 37, pp. 8583–8610, 2022.

[56] X. Zhang, Y. Gao, J. Lin, and C.-T. Lu, "Tapnet: Multivariate time series classification with attentional prototypical network," *in Proc. AAAI*, pp. 1–8, 2020.

[57] B. Lafabregue, J. Weber, P. Gançarski, and G. Forestier, "End-to-end deep representation learning for time series clustering: a comparative study," *Data Min. Knowl. Disc.*, vol. 36, pp. 29–81, 2022.

[58] Y. Yang, H. Shen, Z. Ma, Z. Huang, and X. Zhou, "L2, 1-norm regularized discriminative feature selection for unsupervised," *in Proc. IJCAI*, pp. 1589–1594, 2011.

[59] Z. Li, Y. Yang, J. Liu, X. Zhou, and H. Lu, "Unsupervised feature selection using nonnegative spectral analysis," *in Proc. AAAI*, vol. 26, pp. 1026–1032, 2012.

[60] M. Qian and C. Zhai, "Robust unsupervised feature selection," *in Proc. IJCAI 2013*, pp. 1621–1627, 2013.

[61] L. Shi, L. Du, and Y. Shen, "Robust spectral learning for unsupervised feature selection," *in Proc. IEEE ICDM 2014*, pp. 977–982, 2014.

[62] J. Yang and J. Leskovec, "Patterns of temporal variation in online media," *in Proc. ACM WSDM 2011*, pp. 177–186, 2011.

[63] F. Petitjean, A. Ketterlin, and P. Gancarski, "A global averaging method for dynamic time warping, with the applications to clustering," *Pattern Recogn.*, vol. 44, no. 3, pp. 678–693, 2011.

[64] J. Paparrizos and L. Gravano, "K-shape: efficient and accurate clustering of time series," *in Proc. ACM SIGMOD 2015*, pp. 1855–1870, 2015.

[65] H. Kamalzadeh, A. Ahmadi, and S. Mansour, "Clustering time-series by a novel slope-based similarity measure considering particle swarm optimization," *Appl. Soft Comput.*, vol. 96, pp. 1–16, 2020.

[66] N. Madiraju, S. Sadat, D. Fisher, and H. Karimabadi, "Deep temporal clustering: Fully unsupervised learning of time-domain features," *arXiv preprint arXiv:1802.01059*, 2018.

[67] X. Guo, L. Gao, X. Liu, and J. Yin, "Improved deep embedded clustering with local structure preservation," *in Proc. IJCAI 2017*, pp. 1753–1759, 2017.

[68] Q. Ma, J. Zheng, S. Li, and G. Cottrell, "Learning representations for time series clustering," *In Proc. NeurIPS 2019*, 2019.

[69] K. Ghasedi, X. Wang, C. Deng, and H. Huang, "Balanced self-paced learning for generative adversarial clustering network," *In Proc. CVPR*, p. 4391–4400, 2019.

[70] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," *arXiv preprint arXiv:1708.04552*, 2017.

[71] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *In Proc. NeurIPS 2015*, 2015.

[72] J.-Y. Franceschi, A. Dieuleveut, and M. Jaggi, "Unsupervised scalable representation learning for multivariate time series," *In Proc. NeurIPS 2019*, 2019.

[73] X. Guo, L. Gao, X. Liu, and J. Yin, "Improved deep embedded clustering with local structure preservation," *In Proc. IJCAI*, p. 1753–1759, 2017.

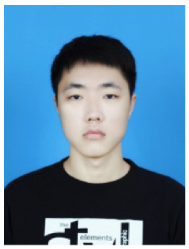
[74] H. A. Dau, A. Bagnall, K. Kamgar, C.-C. M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana, and E. Keogh, "The ucr time series archive," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 6, pp. 1293–1305, 2019.

[75] Y. Hou, Z. Ma, C. Liu, and C. C. Loy, "Learning lightweight lane detection cnns by self attention distillation," *In Proc. ICCV*, pp. 1013–1021, 2019.

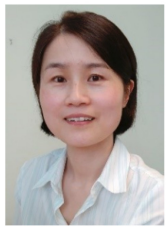
[76] L. Zhang, C. Bao, and K. Ma, "Self-distillation: Towards efficient and compact neural networks," *IEEE Trans. Pattern Anal.*, vol. 44, no. 8, pp. 4388–4403, 2022.

Zhiwen Xiao (M), received the B.Eng. degree in network engineering from the Chengdu University of Information Technology, Chengdu, China, and the M.Eng. degree in computer science from the Northwest A & F University, Yangling, China. He is pursuing the Ph.D. degree in computer science at Southwest Jiaotong University, Chengdu, China. His research interests include semantic communication, federated learning (FL), representation learning, data mining, and computer vision.

Huanlai Xing (M), received Ph.D. degree in computer science from University of Nottingham (Supervisor: Dr Rong Qu), Nottingham, U.K., in 2013. He was a Visiting Scholar in Computer Science, The University of Rhode Island (Supervisor: Dr. Haibo He), USA, in 2020-2021. Huanlai Xing is with the School of Computing and Artificial Intelligence, Southwest Jiaotong University (SWJTU), and Tangshan Institute of SWJTU. He was on Editorial Board of SCIENCE CHINA INFORMATION SCIENCES. He was a member of several international conference program and senior program committees, such as ECML-PKDD, MobiMedia, ISCIT, ICCC, TrustCom, IJCNN, and ICSINC. His research interests include semantic communication, representation learning, data mining, reinforcement learning, machine learning, network function virtualization, and software defined networking.



Bowen Zhao, received his B. Eng. degree in Computer Science and Technology in 2020, from Southwest Jiaotong University, Sichuan, China. He is currently pursuing the master's degree in the School of Information Science and Technology, Southwest Jiaotong University, Chengdu, China. His research interests include deep reinforcement learning, cloud computing, and deep learning.



Rong Qu (SM'12) is an Associate Professor at the School of Computer Science, University of Nottingham. She received her B.Sc. in Computer Science and Its Applications from Xidian University, China in 1996 and Ph.D. in Computer Science from The University of Nottingham, U.K. in 2003. Her research interests include the modelling and optimisation for logistics transport scheduling, personnel scheduling, network routing, portfolio optimization and timetabling problems by using evolutionary algorithms, mathematical programming, constraint programming in operational research and artificial intelligence. These computational techniques are integrated with knowledge discovery, machine learning and data mining to provide intelligent decision support on logistic fleet operations at SMEs, workforce scheduling at hospitals, policy making in education, and cyber security for connected and autonomous vehicles.

Dr. Qu is an associated editor at Engineering Applications of Artificial Intelligence, IEEE Computational Intelligence Magazine, IEEE Transactions on Evolutionary Computation, Journal of Operational Research Society and PeerJ Computer Science. She is a Senior IEEE Member since 2012 and the Vice-Chair of Evolutionary Computation Task Committee since 2019 and Technical Committee on Intelligent Systems Applications (2015-2018) at IEEE Computational Intelligence Society. She has guest edited special issues on the automated design of search algorithms and machine learning at the IEEE Transactions on Pattern Analysis and Machine Intelligence and IEEE Computational Intelligence Magazine.

Shouxi Luo (M), received the bachelor's degree in communication engineering and the Ph.D. degree in communication and information systems from the University of Electronic Science and Technology of China, Chengdu, China, in 2011 and 2016, respectively. He is an Associate Professor with the School of Computing and Artificial Intelligence, Southwest Jiaotong University, Chengdu. His research interests include data center networks, software-defined networking, and networked systems.

Penglin Dai (M), received the B.S. degree in mathematics and applied mathematics and the Ph.D. degree in computer science from Chongqing University, Chongqing, China, in 2012 and 2017, respectively. He is currently an Assistant Professor with the School of Information Science and Technology, Southwest Jiaotong University, Chengdu, China. His current research interests include intelligent transportation systems and vehicular cyber–physical systems.

Ke Li, received the master's degree from Sichuan University in 2008 and the Ph.D. degree in communication and information systems from the University of Electronic Science and Technology of China in 2012. She is a Lecturer with the School of Information Science and Technology, Southwest Jiaotong University. Her research interests include network optimization, network design, and future Internet.

Zonghai Zhu received his B.Sc. degree in the Department of Information, Mechanical and Electrical Engineer, Shanghai Normal University, China, in 2010, and received his Ph.D. degree from the Department of Computer Science and Engineering, East China University of Science and Technology, China, in 2021. He is now an Assistant Professor in the School of Computing and Artificial Intelligence, Southwest Jiaotong University, China. His research interests include imbalanced problems, kernel-based methods, and graph-structured data.