Ensemble Transitive Bidirectional Decoupled Self-Distillation for Time Series Classification

Zhiwen Xiao, *Member, IEEE*, Huanlai Xing, *Member, IEEE*, Rong Qu, *Fellow, IEEE*, Hui Li, Li Feng, Bowen Zhao, and Qian Wan

***Abstract*—Numerous existing deep learning models for time series classification (TSC) tend to overlook the intricate interplay between higher- and lower-level semantic information. While the focus is often on extracting higher-level semantics from lower-level sources, the reciprocal influence of lower-level information on higher levels is undervalued. To address this, we propose an ensemble transitive bidirectional decoupled self-distillation (ETBiDecSD) method for TSC. ETBiDecSD enhances the robustness of higher-level semantic information using an average feature ensemble method to amalgamate the output from each level. Simultaneously, the integrated features are transmitted to each lower level through a directional decoupled distillation structure. Additionally, to promote deep interaction between higher- and lower-level semantic information, ETBiDecSD introduces a transitive bidirectional decoupled distillation structure, facilitating the transfer of target-class and non-target-class knowledge between higher and lower levels. Experimental results demonstrate that whether a fully convolutional network (FCN) with four convolutional blocks or InceptionTime with four Inception blocks is used as the baseline, ETBiDecSD outperforms a quantity of well-established self-distillation algorithms across 85 widely used UCR2018 datasets, as evidenced by the metrics 'win'/'tie'/'lose' and avg. rank, which are derived from accuracy and F_1 scores. Notably, when compared to a non-self-distillation FCN, ETBiDecSD achieves 'win'/'tie'/'lose' results of 64/4/17 in terms of accuracy and 65/4/16 in terms of F_1 score. Similarly, in comparison to a non-self-distillation InceptionTime, ETBiDecSD attains 'win'/'tie'/'lose' results of 60/12/13 for accuracy and 57/12/16 for F_1 score.**

***Index Terms*—Data Mining, Deep Learning, Knowledge Distillation, Representation Learning, Time Series Classification**

I. Introduction

TIME series data represents a sequential arrangement of data points correlated with univariate or multivariate time-dependent variables. This type of data finds extensive applications in diverse domains, including stock prediction [1], signalized traffic corridor [2], emitter identification [3], malware traffic classification [4], anomaly detection [5], electroencephalogram analysis [6], and intelligent diagnosis [7]. Effectively capturing both local and global patterns within data is imperative for the success of any time series classification (TSC) algorithm across diverse feature types [8], [9].

Since Wang *et al.* [10] proposed the multilayer perceptron (MLP), residual network (ResNet), and fully convolutional network (FCN) architectures, the TSC community has witnessed the proliferation of numerous deep learning algorithms. Unlike traditional algorithms, e.g., dynamic time warping (DTW)-based algorithms [11], deep learning algorithms prioritize discerning potential connections among representations through the construction of an internal representation hierarchy within the data [12]. Broadly categorized into two design structures are these algorithms: single-network-based models and dual-network-based models. A single-network-based model typically employs a single (often hybridized) network to capture significant relationships within the internal hierarchy, such as ConvTimeNet [13], adversarial joint-learning algorithm based on recurrent neural network (RNN) [14], and InceptionTime [15]. Conversely, a dual-network-based model integrates two concurrent networks: a local-feature extraction network and a global-relation extraction network. The former, typically grounded in convolutional neural networks (CNNs), concentrates on local feature extraction, while the latter is dedicated to capturing interconnections among the previously extracted features. The long short-term memory (LSTM)-FCN with an LSTM-based network and a FCN [16], the robust neural temporal search network integrating a temporal search network and an attentional LSTM-based network [17], and SelfMatch comprising a ResNet and an attentional LSTM-based network [18] are paradigmatically categorized as dual-network-based models.

Nevertheless, a significant shortcoming observed in numerous deep learning models for TSC is their failure to thoroughly introspect structural complexities. The efficacy of a learning model often hinges upon the quality of both higher- and lower-level semantic information within the representation hierarchy [19]. The derivation of higher-level semantic information from lower levels is a well-established fact. In updating their parameters, learning models typically employ the backpropagation (BP) method, where the updates at lower levels are affected by those at higher levels [20]. In other words, higher-level semantic information can exert a certain degree of influence

This work was partially supported by the Natural Science Foundation of Hebei Province under Grant No. F2022105027, the Ningbo Key Laboratory of Digital Port Technologies (No. NDPT2402), the University of Nottingham Ningbo China, the Doctoral Innovation Fund Program of Southwest Jiaotong University under Grant No. CX-2025ZD09, the Sichuan Science and Technology Program under Grant No. 2024NSFSC1472, and the Fundamental Research Funds for the Central Universities, P. R. China (Corresponding Author: Huanlai Xing).

Z. Xiao, H. Xing, L. Feng, B. Zhao, and Q. Wan are with the School of Computing and Artificial Intelligence, Southwest Jiaotong University, Chengdu 610031, China, with the Tangshan Institute of Southwest Jiaotong University, Tangshan 063000, China, and also with the Engineering Research Center of Sustainable Urban Intelligent Transportation, Ministry of Education, Chengdu 611756, China (Emails: xiao1994zw@163.com; hxx@home.swjtu.edu.cn; fengli@swjtu.edu.cn; cn16bz@icloud.com; qianwan@my.swjtu.edu.cn).

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

H. Li is with the School of Mathematics and Statistics, Xi'an Jiaotong University, Xi'an 710049, China (Email: lihui10@mail.xjtu.edu.cn).

on lower-level semantic information. Consequently, existing frameworks often fail to simultaneously preserve the richness of higher-level semantics and reinforce the representational quality of lower-level features. This limitation results in an incomplete knowledge transfer process, constraining the depth of learned temporal dependencies. In addition, the absence of systematic mechanisms to jointly exploit knowledge from both target and non-target classes creates a critical bottleneck, leaving the interplay of these two complementary information streams largely underexplored. These two issues, i.e., the insufficient bidirectional interaction between hierarchical levels and the underutilization of non-target class knowledge, form the central challenges that this work aims to resolve.

Recently, self-distillation has gained extensive traction within the knowledge distillation (KD) community. Diverging from other KD methods, e.g., dense cross-layer mutual distillation [21] and factor transfer distillation [22], requiring external knowledge introduction, the self-distillation approach uniquely assigns the model a dual role as both student and teacher simultaneously. This method aims to uncover the intricate relationships between higher- and lower-level semantic information within the model, fostering knowledge exchange within the model itself, ultimately leading to regularization [23]. For instance, Zhang *et al.* [24] introduced a be your own teacher distillation method, namely BYOT, aiming to transfer the output level's knowledge to lower-level output. In [25], a feature self-distillation refinement approach was designed to foster the knowledge from the self-teacher network to a refined classifier network. In [26], a progressive self-label approach, namely ProSelfLC, was employed to minimize the regularization entropy during knowledge transfer. In [27], two self-distillation architectures using transitive and densely connected methods were used to facilitate hierarchical knowledge transfer within the model. Notably, most self-distillation algorithms above have several limitations:

- Existing algorithms predominantly focus on a unidirectional flow of knowledge from higher to lower levels of semantic representation. This approach overlooks the critical reciprocal influence that lower-level semantic information can exert on higher-level features. Such an oversight restricts the potential for a more nuanced understanding of data, as demonstrated by methodologies such as BYOT and transitive distillation.
- Furthermore, these frameworks tend to prioritize knowledge associated with target classes, often neglecting the significant "dark knowledge" inherent in non-target classes. The effective harnessing of this latent knowledge is essential for enhancing the overall KD's efficiency, as it offers invaluable insights that enrich model training [28].
- To mitigate the aforementioned limitations, Xiao *et al.* [29] introduced a self-bidirectional decoupled distillation approach, termed Self-BiDecKD, which promotes the exchange of semantic information between non-target and target classes. However, akin to other self-distillation algorithms, including BYOT and ProSelfLC, Self-BiDecKD largely depends on the output layer of the model as the primary source of higher-level semantic information. This dependency inherently restricts the richness of the acquired higher-level features. As indicated by BP principles, the limited richness of higher-level information can adversely impact the quality of lower-level semantic representations. Consequently, both lower- and higher-level semantic information may suffer in quality, hindering the effective learning of rich representations and regularizations from the data.
- Moreover, research into the interplay of knowledge flows between target and non-target classes in self-distillation for TSC remains remarkably sparse, highlighting a critical gap in the literature that warrants further exploration.

To overcome these challenges, we propose an innovative approach: the ensemble transitive bidirectional decoupled self-distillation (ETBiDecSD) method for TSC. ETBiDecSD is specifically designed to (i) enrich higher-level semantic representations by integrating multi-level outputs through an average feature ensemble mechanism, thereby alleviating the limitations of relying solely on final-layer features, and (ii) establish a dual-stream distillation strategy that simultaneously supports directional top-down transfer and transitive bidirectional interaction, ensuring that both target and non-target class knowledge are effectively exchanged across hierarchical levels. Through this design, ETBiDecSD directly addresses the dual challenges of semantic degradation and incomplete knowledge utilization.

The major contributions of this work are summarized below:

1) To address the insufficient interaction between hierarchical levels observed in prior methods, we design a transitive bidirectional decoupled distillation framework. This mechanism enables holistic knowledge circulation between higher- and lower-level representations, ensuring that both target and non-target class information are actively leveraged, thereby enhancing representational richness and learning stability.

2) To mitigate the over-reliance on output-layer features in existing self-distillation methods, we introduce an average feature ensemble strategy. By synthesizing intermediate and higher-level representations, this approach strengthens semantic robustness and preserves diverse temporal cues across multiple layers.

3) To validate the effectiveness and generality of ETBiDecSD, we integrate it with two representative TSC backbones (FCN and InceptionTime) and conduct extensive evaluations on 85 UCR2018 datasets. The results consistently demonstrate that ETBiDecSD achieves superior performance compared to a quantity of state-of-the-art self-distillation baselines, underscoring its broad applicability and competitive advantage in real-world TSC tasks.

This paper proceeds listed below: Section II provides an extensive review of classical TSC and KD algorithms. Subsequently, Section III outlines the ETBiDecSD's fundamental components. Finally, the experimental analysis and conclusions are presented in Sections IV and V, respectively.

II. Related Work

This section provides an overview of several existing algorithms in the fields of TSC and KD.

A. Time Series Classification Algorithms

Below, we provide a review of a diverse array of traditional and deep learning algorithms.

1) Traditional Algorithms: In the realm of TSC, traditional algorithms predominantly fall into two categories: distance-based and feature-based methods. Distance-based algorithms aim to detect notable distinctions between individual samples. Notable examples of such algorithms include nearest neighbor (NN) and dynamic time warping (DTW) [11]. Notable examples include dependent DTW (DTW_D), independent DTW (DTW_I), and adaptive DTW (DTW_A). The integration of ensemble algorithms based on NN and DTW has gained traction in TSC, reflecting emerging trends in the field. For instance, in [30], an elastic ensemble structure integrated 11 classical classifiers, such as time warp with edit and weighted DTW, for feature extraction. In [31], a transformation-based ensemble (COTE) approach incorporated several typical classifiers considering the change and time transformation to address various TSC challenges. A hierarchical method based on probabilistic voting with COTE called HIVE-COTE achieves commendable performance across 85 benchmark TSC datasets [30]. The two representative distance-based algorithms are improved HIVE-COTE [32], and explainable-by-design ensemble methods [33].

Feature-based models are designed to extract representative representations from the input data. Such as, in [34], a model with hidden-unit logistic was introduced to extract temporal dependencies within time series data. In [35], an autoregressive method based on forest ensemble was used for hierarchical relation extraction. Baydogan and Runger [36] designed an auto-pattern approach to extract local dependencies in the data. A symbol method using Fourier approximation and bag of symbolic was presented for spatial feature extraction [37]. To overcome feature selection and weighting challenges, a word extraction method with multivariate unsupervised symbols and derivatives was generated [38]. Wu *et al.* [39] devised a cognitive map approach based on fuzzy technique, sparse auto-encoder, MLP, and high-order cognitive map, for representation learning in time series. In [40], an online method based on rule-based classifier learning was used to mine the connections in unlabeled multivariate time series. In [41], a fuzzy-probabilistic algorithm with representation learning achieved decent performance on 25 TSC benchmark datasets. Zhang and Xiao [42] introduced a time series measurement method with belief Rényi divergence to measure the divergence of time series divergence. In [43], a order-preserving pattern mining algorithm was proposed to discover the top-k contrast patterns as time series shapelets.

2) Deep Learning Algorithms: Designed to construct a hierarchical data representation, deep learning algorithms aim to unearth underlying relationships among representations. Within this framework, two prominent research methodologies in deep learning are single-network-based and dual-network-based models. A single-network-based model typically employs a single (often hybridized) network to capture notable regularizations within the representation hierarchy. Such as, in [44], a multi-relationship extraction approach captured time-series-to-class and inter-time-series relationships, enabling discriminative embeddings. In [45] , a multi-scale capture network was used to capture multi-scale dependencies in time series. In [46], a feature extraction model based on dual attention captured local features and global relationships from the data. A shapelet model with feature embedding searched discriminative shapelets in time series [47]. An approach using random convolutional kernel called ROCKET was introduced for feature extraction [48]. Building upon ROCKET, Dempster *et al.* [49] introduced an enhanced variant termed Improved ROCKET (mini-ROCKET) to expedite feature and relation extraction in time series. Representative single-network-based models encompass FCN [10], MLP [10], InceptionTime [15], ConvTimeNet [13], ResNet [10], learnable dynamic temporal pooling [50], $log\text{-}Sigmoid$ activation-based LSTM [51], similarity-based self-supervised representation learning model [52], clustering time series model based on untrained deep neural networks [53], dynamic graph attention autoencoder [54], ROCKET-based ensemble predictor [55], deep contrastive representation learning model based on self-distillation [56], and adversarial joint-learning RNN [14]. Conversely, in a dual-network-based model, two feature networks are typically integrated in parallel: one dedicated to local-feature extraction and the other to global-relation extraction. The LSTM-FCN [16], RTFN [57], ResNet-Transformer [58], attentional prototypical network [59], SelfMatch [18], densely knowledge-aware network [60], and RNTS [17] are classical dual-network-based methods.

Numerous deep learning models fail to account for the dynamic interaction between higher- and lower-level semantic information within the representation hierarchy. While there is a strong focus on extracting higher-level semantic information from lower-level sources, the reciprocal influence of lower-level information on higher levels is often neglected. In practice, higher and lower-level semantic information mutually influence each other during the learning process. To address this gap, we introduce ETBiDecSD, a model explicitly designed to enhance and facilitate the deep interaction and transfer between higher and lower levels.

B. Knowledge Distillation Algorithms

The learning schemes of KD can be broadly categorized into three main types depending on whether a teacher is updated simultaneously with a student: offline distillation, online distillation, and self-distillation [23].

1) Offline Distillation: The majority of previous KD methods operate in an offline manner. In vanilla KD [61], knowledge transfers from a pre-trained teacher to a student in two stages: (1) the teacher is trained, and (2) it guides the student's training. The offline methods mainly focus on improving various aspects of knowledge transfer, encompassing the design of knowledge [61], [62] and the formulation of loss functions for matching features or distributions [63], [64], [65], [66], [67]. The offline distillation methods typically entail one-way knowledge transfer and a two-phase training process. However, they often encounter challenges related to the employment of a complicated, large-scale teacher model with lengthy training

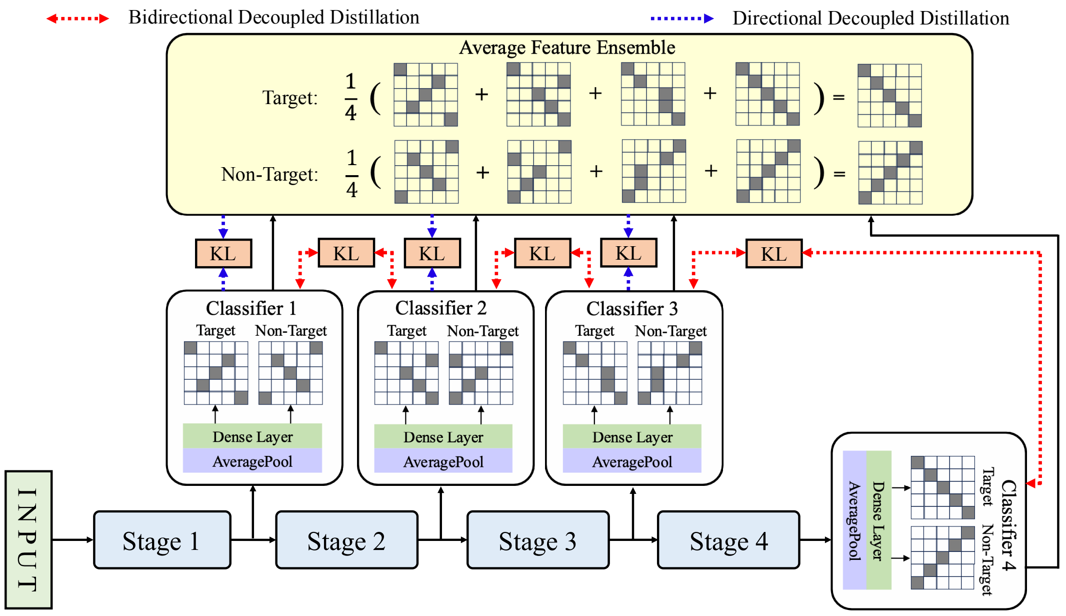


Fig. 1. Schematic diagram of ETBiDecSD. Schematic diagram of ETBiDecSD. The overall framework illustrates how raw time series are progressively processed through multi-level feature extraction, ensemble integration, and decoupled distillation to enable refined semantic transfer across different layers. ETBiDecSD integrates outputs across all levels using the average feature ensemble method to enrich higher-level semantic information, which is then conveyed to lower levels through directional decoupled distillation. It also employs transitive bidirectional decoupled distillation to enhance semantic exchange between higher and lower levels for both target and non-target classes. Both directional and bidirectional decoupled distillation approaches utilize the Kullback-Leibler (KL) divergence function to facilitate interactions between classes. Note: 'target' and 'non-target' represent the classification probabilities for the target and non-target classes of a given classifier, respectively.

durations. While training the student in offline distillation is typically efficient under the teacher's guidance, a capacity gap persists between the complicated teacher and the simple student. Consequently, the student heavily relies on the teacher.

2) *Online Distillation:* To mitigate the drawbacks existed in offline distillation, online distillation is introduced to bolster the student's performance, especially in situations where accessing a high-capacity, high-performance teacher model is challenging [68]. In online distillation, both the teacher and student models are updated simultaneously, rendering the entire KD structure trainable end-to-end. For example, Zhang *et al.* [69] introduced a method based on adversarial co-distillation learning to produce extra diverging image samples. Kshirsagar and Londhe [70] presented an efficient classification method based on convolution and KD for devanagari script-based P300 speller. Su *et al.* [71] devised a deep cross-layer collaborative online distillation learning method for image recognition. Yang *et al.* [72] put forward a mutual contrastive online distillation learning structure to enhance the knowledge flow from the large-scale teacher to the small student. Su *et al.* [73] proposed a synchronous teaching method, seamlessly integrating online teaching and offline teaching to facilitate the transfer of rich and comprehensive knowledge to the student. Online distillation involves a one-phase end-to-end training approach with efficient parallel computing. However, existing online methods often face challenges when dealing with complex teachers in online settings. Therefore, there is a need for further exploration into the dynamic interactions between teacher and student models in online scenarios.

3) *Self-Distillation:* Diverging from the conventional offline and online distillation approaches that necessitate external knowledge introduction, self-distillation innovatively endows the model with a dual role, functioning both as a student and a teacher. This distinctive methodology seeks to unravel the intricate relationships between higher- and lower-level semantic information within the model, promoting internal knowledge exchange and culminating in effective regularization [74]. The typical self-distillation methods include BYOT [24], feature self-distillation refinement [25], self attention distillation [75], densely connected distillation [27], ProSelfLC [26], and transitive distillation [27].

Most existing self-distillation algorithms primarily focus on transferring target-class knowledge between higher and lower levels in the model. However, these approaches often neglect the importance of lower-level semantic information influencing higher-level semantics, as well as the role of non-target-class knowledge in the overall knowledge flow within the model. Additionally, there has been limited investigation into the transfer dynamics of both target-class and non-target-class knowledge in self-distillation. To address these research gaps, we propose ETBiDecSD to facilitate deep interaction between higher- and lower-level target-class and non-target-class semantic information in the model.

III. Method

This section begins by providing an overview of the ETBiDecSD architecture and defining the problem formulation. It concludes with an introduction to two crucial components (i.e., average feature ensemble and transitive bidirectional decoupled distillation) and loss function of ETBiDecSD.

A. Overview

ETBiDecSD establishes a holistic workflow that begins with raw time series input and advances through a four-layer feature extraction backbone, designed to progressively capture increasingly abstract temporal patterns. These extracted representations are then aggregated by the average feature ensemble method, producing enriched higher-level semantics that encapsulate comprehensive contextual information. Once enriched features are obtained, they are transmitted to lower levels via directional decoupled distillation, thereby reinforcing hierarchical information flow and improving the representation capacity of lower-level layers. To further strengthen this interaction, ETBiDecSD incorporates transitive bidirectional decoupled distillation, which enables reciprocal semantic transfer between higher and lower levels. This dual mechanism ensures that both target and non-target class probabilities are simultaneously refined, promoting balanced and nuanced knowledge propagation. As summarized in Fig. 1, the entire process creates a tightly coupled system in which feature extraction, ensemble integration, and decoupled distillation operate in concert. By explicitly modeling inter-level dependencies and facilitating semantic interactions across classes, ETBiDecSD provides a coherent and systematic approach for enhancing TSC performance and generalization.

B. Problem Formulation

Let $x_i = \{\{x^i_{1,1}, \ldots, x^i_{1,d}\}, \ldots, \{x^i_{l,1}, \ldots, x^i_{l,d}\}\} \in \mathcal{X}$ represent the i-th input instance, where $\mathcal{X} \subseteq \mathbb{R}^{l\times d}$ stands for the input space. The parameters l and d represent the length and dimension of x_i, respectively. The associated categorical label of x_i is denoted by $y_i \in \mathcal{Y}$, where $\mathcal{Y}$ is the target label space. The goal is to derive a predictive model $\mathcal{F} : \mathcal{X} \mapsto \mathcal{Y}$ using the dataset $\mathcal{D} = \{\mathcal{D}_{train}, \mathcal{D}_{val}, \mathcal{D}_{test}\}$. The training set is $\mathcal{D}_{train} = \{x_i, y_i\}_{i=1}^{n_{train}}$, the validation set is $\mathcal{D}_{val} = \{x_i, y_i\}_{i=1}^{n_{val}}$, and the testing set is $\mathcal{D}_{test} = \{x_i, y_i\}_{i=1}^{n_{test}}$. The cardinalities of the training, validation, and testing sets are denoted by n_{train}, n_{val}, and n_{test}, respectively.

Let $O^t_i = [O^t_{i,1}, O^t_{i,2}, \ldots, O^t_{i,C}] \in \mathbb{R}^{1\times C}$, where $i = 1, 2, \ldots, n_{\text{train}}$, denote the i-th output vector of Classifier t, with $t = 1, 2, 3, 4$ and C representing the number of classes. Let $P^t_i = [P^t_{i,1}, P^t_{i,2}, \ldots, P^t_{i,C}] \in \mathbb{R}^{1\times C}$ represent the classification probabilities for the target classes, and $P^t_{i,\backslash} = [P^t_{i,\backslash 1}, P^t_{i,\backslash 2}, \ldots, P^t_{i,\backslash C}] \in \mathbb{R}^{1\times C}$ denote the probabilities for the non-target classes corresponding to O^t_i. Here, $P^t_{i,j} \in P^t_i$ and $P^t_{i,\backslash j} \in P^t_{i,\backslash}$, for $j = 1, 2, \ldots, C$, are as defined in Eq. (1).

$$\begin{aligned} P^t_{i,j} &= \frac{exp(O^t_{i,j}/T)}{\sum_{k=1}^{C} exp(O^t_{i,k}/T)} \\ P^t_{i,\backslash j} &= \frac{\sum_{m=1,m\neq j}^{C} exp(O^t_{i,m}/T)}{\sum_{k=1}^{C} exp(O^t_{i,k}/T)} \end{aligned} \tag{1}$$

where, T is a temperature scaling coefficient, with a specific value of T = 1.0 set for the experiments conducted in this paper. For a more comprehensive discussion, please refer to Subsection IV-C.

C. Average Feature Ensemble

Unlike Self-BiDecKD [29], which connects only the final layer's output to lower-level blocks, the average feature ensemble method adopts a more integrative strategy by consolidating outputs from all levels. This approach enhances the richness of higher-level semantic information and addresses a significant limitation of Self-BiDecKD, which risks overlooking valuable insights from intermediate layers. By transmitting these integrated features downward through directional decoupled distillation, the average feature ensemble method enables a more thorough extraction of semantic knowledge. This strategy not only cultivates a nuanced understanding within the model but also facilitates more effective knowledge transfer, thereby enhancing the overall performance of the proposed ETBiDecSD. Its structure diagram is presented in Fig. 1.

The average feature ensemble method integrates the output from each level to improve the robustness of higher-level semantic information. Let P^{AFE}_i and $P^{AFE}_{i,\backslash}$ present the classification probabilities of target and non-target classes of the integrated features related to x_i, respectively. They are defined as:

$$\begin{aligned} P^{AFE}_i &= \frac{1}{4}\sum_{t=1}^{4} P^t_i \\ P^{AFE}_{i,\backslash} &= \frac{1}{4}\sum_{t=1}^{4} P^t_{i,\backslash} \end{aligned} \tag{2}$$

The directional decoupled distillation transfers the integrated features to each lower level in the model. $\mathcal{L}^i_{AFE}$ is comprised of a target-class loss, $\mathcal{L}^i_{AFET}$, and a non-target-class loss, $\mathcal{L}^i_{AFENT}$, as computed in Eq. (3).

$$\begin{aligned} \mathcal{L}^i_{AFE} &= \alpha\mathcal{L}^i_{AFET} + \beta\mathcal{L}^i_{AFENT} \\ &= \alpha\sum_{t=1}^{3} KL(P^t_i, P^{AFE}_i) + \beta\sum_{t=1}^{3} KL(P^t_{i,\backslash}, P^{AFE}_{i,\backslash}) \end{aligned} \tag{3}$$

where, α and β denote the weights of $\mathcal{L}^i_{AFET}$ and $\mathcal{L}^i_{AFENT}$, respectively. $KL()$ is the Kullback Leibler (KL) loss function. Following the recommendation from prior studies [28], [29], we fix α to 1.0. Additionally, guided by the experiments, this paper sets β to 1.0. For further details, please refer to Subsection IV-C.

D. Transitive Bidirectional Decoupled Distillation

The transitive bidirectional decoupled distillation enables reciprocal knowledge transfer between higher and lower levels, extracting hidden knowledge from both the target and non-target classes, as depicted in Fig. 1.

The transitive bidirectional decoupled distillation loss quantifies the disparity between the classification probabilities of target classes from two specified classifiers, as well as the difference in the classification probabilities of non-target classes from the same classifiers. Let $\mathcal{L}^i_{TBDDT}$ represent the transitive bidirectional decoupled distillation target-class loss of x_i, and $\mathcal{L}^i_{TBDNT}$ denote the transitive bidirectional decoupled distillation non-target-class loss of x_i.

$\mathcal{L}^i_{TBDDT}$ is calculated in Eq. (4).

$$\begin{aligned}\mathcal{L}^i_{TBDDT} = & KL(P_i^4, P_i^3) + KL(P_i^3, P_i^4) + KL(P_i^3, P_i^2) \\ & + KL(P_i^2, P_i^3) + KL(P_i^2, P_i^1) + KL(P_i^1, P_i^2)\end{aligned} \tag{4}$$

$\mathcal{L}^i_{TBDNT}$ is defined as:

$$\begin{aligned}\mathcal{L}^i_{TBDNT} = & KL(P^4_{i,\backslash}, P^3_{i,\backslash}) + KL(P^3_{i,\backslash}, P^4_{i,\backslash}) + KL(P^3_{i,\backslash}, P^2_{i,\backslash}) \\ & + KL(P^2_{i,\backslash}, P^3_{i,\backslash}) + KL(P^2_{i,\backslash}, P^1_{i,\backslash}) + KL(P^1_{i,\backslash}, P^2_{i,\backslash})\end{aligned} \tag{5}$$

$\mathcal{L}^i_{TBDD}$ is a comprehensive integration of $\mathcal{L}^i_{TBDDT}$ and $\mathcal{L}^i_{TBDNT}$, weighted by their respective coefficients, as defined in Eq. (6)

$$\mathcal{L}^i_{TBDD} = \alpha \mathcal{L}^i_{TBDDT} + \beta \mathcal{L}^i_{TBDNT} \tag{6}$$

where, α and β are the weights of $\mathcal{L}^i_{TBDDT}$ and $\mathcal{L}^i_{TBDNT}$, respectively. In alignment with the preceding discussion in Eq. (3), α is fixed at 1.0, while β is similarly determined to be 1.0, based on empirical findings. For additional details, please refer to Subsection IV-C.

E. Loss Function

Like other self-distillation algorithms [24], [25], [26], [27], ETBiDecSD employs a fixed coefficient to integrate multiple loss functions for parameter optimization. The loss function of ETBiDecSD is primarily divided into two components. The first is a supervised loss, $\mathcal{L}_{sup}$, which uses ground truth labels to constrain the predictions of ETBiDecSD. The second is a KD loss, $\mathcal{L}_{KD}$, which combines various self-distillation functions to facilitate deep interaction between higher- and lower-level semantic information, both for target and non-target classes, within the model. The ETBiDecSD's loss function, $\mathcal{L}$, is defined as:

$$\mathcal{L} = (1-\mu)\mathcal{L}_{sup} + \mu\mathcal{L}_{KD} + \lambda||\theta||_2^2 \tag{7}$$

where, μ represents a coefficient of $\mathcal{L}$, λ is the coefficient of $||\theta||_2^2$ (i.e., L_2 regularization), and θ denotes the ETBiDecSD's parameters. Following [24], [25], [26], [29], we set μ = 0.1 in the experiments. To further understand $\mathcal{L}$, we provide a detailed description of its two components: the supervised loss ($\mathcal{L}_{sup}$) and the KD loss ($\mathcal{L}_{KD}$) in the following subsections.

1) Supervised Loss: $\mathcal{L}_{sup}$ represents the aggregate supervised losses across the four classifiers and the average feature ensemble. Each supervised loss leverages the cross-entropy loss function, $CE()$, which measures the divergence between the ground truth labels and the predicted outcomes. The comprehensive definition of $\mathcal{L}_{sup}$ is provided in Eq. (8).

$$\mathcal{L}_{sup} = \frac{1}{n_{train}} \sum_{i=1}^{n_{train}} \left(\sum_{t=1}^{4} CE(P_i^t, y_i) + CE(P_i^{AFE}, y_i)\right) \tag{8}$$

where, y_i is the ground truth label related to x_i.

2) KD Loss: The loss function $\mathcal{L}_{KD}$ facilitates a profound interaction between higher- and lower-level semantic information for both target and non-target classes within the model. It comprises two distinct components: the average feature ensemble loss, $\mathcal{L}_{AFE}$, and the transitive bidirectional decoupled distillation loss, $\mathcal{L}_{TBDD}$, as articulated in Eq. (9).

$$\begin{aligned}\mathcal{L}_{KD} &= \mathcal{L}_{AFE} + \mathcal{L}_{TBDD} \\ &= \frac{1}{n_{train}} \sum_{i=1}^{n_{train}} (\mathcal{L}^i_{AFE} + \mathcal{L}^i_{TBDD})\end{aligned} \tag{9}$$

where, $\mathcal{L}^i_{AFE}$ and $\mathcal{L}^i_{TBDD}$ denote the average feature ensemble and transitive bidirectional decoupled distillation losses associated with each training instance x_i, where $i = 1, 2, \ldots, n_{train}$.

TABLE I
THE PARAMETER SETTINGS OF FCN.

Stage No.	LayerName	KernelSize	ChannelSize	StrideSize
1	Conv1D	11	128	1
	BatchNorm	–	128	–
	ReLU	–	–	–
2	Conv1D	13	128	1
	BatchNorm	–	128	–
	ReLU	–	–	–
3	Conv1D	11	256	1
	BatchNorm	–	256	–
	ReLU	–	–	–
4	Conv1D	11	256	1
	BatchNorm	–	256	–
	ReLU	–	–	–

TABLE II
THE PARAMETER SETTINGS OF INCEPTIONTIME.

Stage No.	LayerName	KernelSize	ChannelSize	StrideSize
1	Conv1D	2	32	1
	Conv1D	3	32	1
	Conv1D	5	32	1
	Conv1D	9	32	1
	Conv1D	11	32	1
	Conv1D	17	32	1
	Maxpooling1D	2	–	1
	BatchNorm	–	192	–
	ReLU	–	–	–
2	Conv1D	2	32	1
	Conv1D	3	32	1
	Conv1D	5	32	1
	Conv1D	9	32	1
	Conv1D	11	32	1
	Conv1D	17	32	1
	Maxpooling1D	2	–	1
	BatchNorm	–	192	–
	ReLU	–	–	–
3	Conv1D	2	64	1
	Conv1D	3	64	1
	Conv1D	5	64	1
	Conv1D	9	64	1
	Conv1D	11	64	1
	Conv1D	17	64	1
	Maxpooling1D	2	–	1
	BatchNorm	–	384	–
	ReLU	–	–	–
4	Conv1D	2	64	1
	Conv1D	3	64	1
	Conv1D	5	64	1
	Conv1D	9	64	1
	Conv1D	11	64	1
	Conv1D	17	64	1
	Maxpooling1D	2	–	1
	BatchNorm	–	384	–
	ReLU	–	–	–

IV. PERFORMANCE EVALUATION AND ANALYSIS

This section initially elucidates the experimental setup and outlines the performance metrics employed. Subsequently, it delves into the exploration of hyper-parameter sensitivity and presents the findings of an ablation study. Finally, the section conducts a comprehensive analysis of the experimental results.

A. Experimental Setup

1) Data Description: Consistent with prior studies [15], [47], [48], [49], we opt for 85 extensively employed datasets sourced from the UCR 2018 archive [76]. These datasets analyzed display a considerable range of characteristics. The lengths of the time series span from a minimum of 24 to a maximum of 2709. The number of classes varies significantly as well, ranging from as few as 2 to as many as 60. Furthermore, they encompass a diverse array of domains, including motion, electroencephalogram (EEG), and electrocardiogram (ECG), underscoring their broad applicability and utility in various research contexts.

2) Baseline: In this paper, we adopt two well-established baselines: FCN [10] and InceptionTime [15]. Specifically, FCN is composed of four convolutional blocks for feature extraction. Each convolutional block consists of a 1-dimensional convolutional neural network (Conv1D) layer, a batch normalization (BatchNorm) module, and a rectified linear unit (ReLU) function. The parameter settings of FCN are detailed in Table I. On the other hand, InceptionTime comprises four Inception blocks designed to capture multi-scale dependencies in time series data. Each Inception block includes five Conv1D layers and one Maxpooling block in parallel. The parameter settings of InceptionTime are detailed in Table II.

3) Implementation Details: In this paper, the BatchNorm decay value is set to 0.9. To mitigate overfitting during training, L_2 regularization is employed. Additionally, the Adam optimizer is utilized with an initial learning rate of 0.001, and both the decay and momentum term values are set to 0.9.

All experiments are run with a computer with Python 3.6, an AMD central processing unit (CPU) R1400, Pytorch 1.3.1, 32GB memory, an Nvidia graphics processing unit (GPU) 2080Ti, Thop 0.0.31, Scikit-learn 0.24.2, and Numpy 1.16.4. During the data preprocessing stage, no data augmentation techniques or specialized preprocessing strategies are applied; the time series data are directly fed into the model following standard normalization procedures. In addition, our code is accessible at https://github.com/xiaozw1994/ETBiDecSD.

B. Evaluation Metrics

To assess the comprehensive efficacy of ETBiDecSD, we introduce two distinct metrics: performance metric and efficiency metric. The performance metric is primarily utilized to gauge the algorithm's effectiveness in achieving its intended objectives. Conversely, the efficiency metric is designed to reflect the operational efficiency of the algorithm, quantifying the resources expended relative to its performance outcomes.

1) Performance Metrics: To assess the ETBiDecSD's effectiveness, this study employs two well-established performance metrics: $Accuracy$ and F_1. Their mathematical definitions are written below.

$$\begin{aligned} Accuracy &= \frac{N_{TP} + N_{TN}}{N_{TP} + N_{FP} + N_{TN} + N_{FN}} \\ F_1 &= \frac{2 * N_{TP}}{2 * N_{TP} + N_{FP} + N_{FN}} \end{aligned} \tag{10}$$

where, N_{TP} and N_{TN} represent the number of true positive and true negative instances, and N_{FP} and N_{FN} stand for the number of false positive and false negative instances.

Building upon the foundational performance metrics previously discussed, we introduce two additional refined statistical metrics: 'win', 'tie', 'lose', and avg. rank. Following established methodologies in the literature [10], [15], [16], [17], [18], [46], [47], [48], [49], the metrics 'win', 'tie', and 'lose' quantify the comparative performance of each algorithm by recording the number of datasets on which it outperforms, ties with, or underperforms against other algorithms, respectively. The 'best' metric is derived as the sum of the 'win' and 'tie' values. Further, in alignment with previous studies [10], [15], [16], [17], [18], [48], [49], we employ the avg. rank metric to facilitate comparative analysis of the algorithms based on their accuracy performance. The avg. rank score is computed using the Wilcoxon signed-rank test [77], with Holm's alpha (5%) correction applied to adjust for multiple comparisons.

2) Efficiency Metrics: When different self-distillation algorithms utilize the same baseline model, the model parameters and floating point operations (FLOPs) remain unchanged. This constancy arises because the model parameters and FLOPs are intrinsically tied to the architecture of the baseline model itself.

Nevertheless, distinct self-distillation algorithms can influence the training process and resource utilization in various ways. To effectively compare the efficiency of these self-distillation algorithms, we introduced four evaluation metrics: training time, CPU usage, GPU usage, and memory usage. These metrics are consistently measured under identical conditions, including the same training duration, batch size, and initial values, ensuring the experimental fairness and reliability, as explained below.

- Training time: The time required to optimize the parameters of a self-distillation model until satisfactory performance metrics are achieved.
- CPU usage: This parameter reflects the proportion of CPU resources consumed by a self-distillation model during training. It is determined by dividing the CPU usage of the model by the total CPU capacity.
- GPU usage: This metric indicates the extent of GPU resources utilized by a self-distillation model during the training process. It is calculated by the ratio of the GPU resources occupied by the model to the total available GPU resources.
- Memory usage: This metric quantifies the memory consumption of a self-distillation model during the training phase. It is computed as the ratio of the memory usage of the model to the total available memory.

3) Visual Comparison via Accuracy/F_1 Plot: As demonstrated in previous studies [10], [15], [16], [17], [18], [46], [47], [48], [49], we utilize an accuracy/F_1 plot to illustrate the performance differences between two algorithms. Each point in the plot represents the comparative accuracy of algorithms A and B across various datasets, with the diagonal reference line $y = x$ indicating equal performance. Points above this line show that algorithm A outperforms algorithm B, while those below indicate the opposite. Points on the line reflect equivalent accuracy.

TABLE III
STATISTICAL OUTCOMES OBTAINED FROM DIFFERENT β VALUES ACROSS 85 WELL-KNOWN UCR2018 DATASETS.

Baseline	Evaluation Metric		β				
	Fundamental Metric	Statistical Metric	0.1	0.5	1.0	2.0	5.0
FCN	*Accuracy*	Best	20	28	**39**	26	22
		Win	10	14	**21**	9	13
		Tie	10	14	**18**	17	9
		Lose	65	57	**46**	59	63
		avg. rank	3.7294	2.8706	**2.6118**	2.8824	2.9059
	F_1	Best	19	27	**38**	27	22
		Win	9	13	**20**	10	13
		Tie	10	14	**18**	17	9
		Lose	66	58	**47**	58	63
		avg. rank	3.8463	2.9043	**2.6425**	2.9123	2.9315
Inception Time	*Accuracy*	Best	19	25	**36**	25	21
		Win	9	12	**20**	11	10
		Tie	10	13	**16**	14	11
		Lose	66	60	**49**	60	64
		avg. rank	3.9234	2.8126	**2.5917**	2.9068	2.9118
	F_1	Best	18	26	**36**	25	21
		Win	8	12	**18**	9	12
		Tie	10	14	**18**	16	9
		Lose	67	59	**49**	60	64
		avg. rank	4.0464	2.8457	**2.6222**	2.9370	2.9375

TABLE IV
STATISTICAL OUTCOMES OBTAINED FROM DIFFERENT T VALUES ACROSS 85 WELL-KNOWN UCR2018 DATASETS.

Baseline	Evaluation Metric		T				
	Fundamental Metric	Statistical Metric	0.1	0.5	1.0	2.0	5.0
FCN	*Accuracy*	Best	14	24	**38**	25	15
		Win	7	11	**21**	13	8
		Tie	7	13	**17**	12	7
		Lose	71	61	**47**	60	70
		avg. rank	4.3682	3.0251	**2.4313**	2.8183	4.1025
	F_1	Best	14	27	**38**	27	14
		Win	7	13	**20**	10	5
		Tie	7	14	**18**	17	9
		Lose	71	58	**47**	58	71
		avg. rank	4.5052	3.0607	**2.4599**	2.8476	4.1387
Inception Time	*Accuracy*	Best	13	25	**37**	24	16
		Win	5	12	**20**	12	8
		Tie	8	13	**17**	12	8
		Lose	72	60	**48**	61	69
		avg. rank	4.5293	3.1165	**2.4625**	2.8834	4.2791
	F_1	Best	13	26	**36**	25	15
		Win	5	12	**18**	9	7
		Tie	8	14	**18**	16	8
		Lose	72	59	**49**	60	70
		avg. rank	4.6713	3.1531	**2.4914**	2.9133	4.3168

To quantify the performance of algorithm A relative to B, we calculate the 'win' value by counting the points above the $y = x$ line, the 'tie' value from points on the line, and the 'lose' value from points below it. This visual representation effectively assesses the relative efficacy of algorithms across diverse datasets.

C. Hyper-parameter Sensitivity

To analyze the impact of various hyperparameter configurations on ETBiDecSD's performance, we leverage a comprehensive set of 85 UCR2018 datasets.

1) ETBiDecSD with various β values: As previously mentioned, β represents the weight assigned to the non-target class loss function. Table III presents the statistical outcomes from experiments conducted across 85 widely recognized UCR2018 datasets, evaluating the impact of various β values.

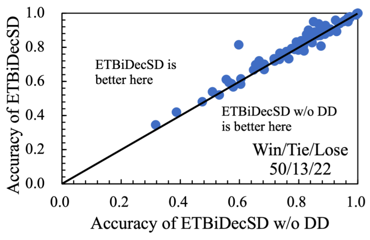
The results reveal that a β value of 1.0 consistently enhances the performance of ETBiDecSD in terms of both accuracy and F_1 score, regardless of whether FCN or InceptionTime is used as the baseline. Specifically, when InceptionTime is employed as the baseline, β = 1.0 achieves a 'win'/'tie'/'lose' ratio of 20/16/49 and demonstrates the lowest avg. rank (2.5917) in terms of accuracy. These findings strongly indicate that β = 1.0 optimizes performance, suggesting its significant value for a wide range of applications.

2) ETBiDecSD with various T values: T represents the temperature scaling coefficient used to adjust the prediction distribution of both the non-target and target classes in each classifier of ETBiDecSD. Table IV presents the statistical outcomes derived from experiments with varying T values across 85 well-established UCR2018 datasets.

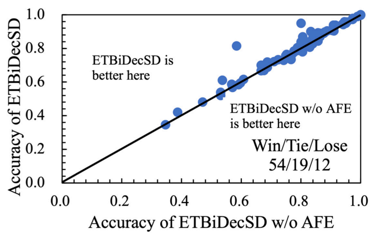
Upon evaluating the performance of ETBiDecSD across different T values, it becomes clear that T = 1.0 consistently outperforms the other configurations. For instance, when FCN is used as the baseline and accuracy serves as the reference metric, T = 1.0 achieves the most favorable 'win'/'tie'/'lose'/'best' outcomes and also records the lowest avg. rank. These results underscore the rationale for selecting T = 1.0, as it delivers the optimal performance across the

TABLE V
STATISTICAL OUTCOMES OBTAINED FROM DIFFERENT μ VALUES ACROSS 85 WELL-KNOWN UCR2018 DATASETS.

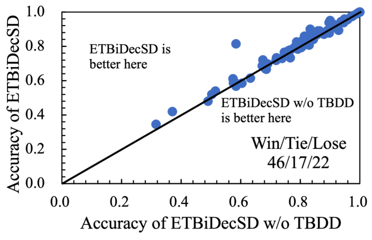
Baseline	Evaluation Metric		μ								
	Fundamental Metric	Statistical Metric	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
FCN	*Accuracy*	Best	**36**	27	18	16	16	14	12	5	4
		Win	**20**	12	8	8	7	7	6	0	1
		Tie	**16**	15	10	8	9	7	6	5	3
		Lose	**49**	58	67	69	69	71	73	80	81
		avg. rank	**3.4106**	3.9433	4.6332	4.5102	5.3293	5.1023	5.9932	6.2345	6.8324
	F_1	Best	**37**	26	18	16	16	13	11	5	3
		Win	**21**	11	8	8	7	6	5	0	0
		Tie	**16**	15	10	8	9	7	6	5	3
		Lose	**48**	59	67	69	69	72	74	80	82
		avg. rank	**3.5247**	4.0752	4.7881	4.6610	5.5075	5.2729	6.1936	6.4430	7.0609
Inception Time	*Accuracy*	Best	**37**	28	18	15	14	13	12	5	3
		Win	**21**	13	8	7	6	7	6	0	1
		Tie	**16**	15	10	8	8	6	6	5	2
		Lose	**58**	57	67	70	71	72	73	80	82
		avg. rank	**3.5002**	4.0234	4.6908	4.5663	5.3956	5.1658	6.0678	6.3121	6.9174
	F_1	Best	**36**	27	18	15	14	13	10	5	3
		Win	**20**	13	8	7	6	7	5	0	1
		Tie	**16**	14	10	8	8	6	5	5	2
		Lose	**49**	58	67	70	71	72	75	80	82
		avg. rank	**3.6172**	4.1579	4.8477	4.7190	5.5760	5.3385	6.2707	6.5231	7.1487



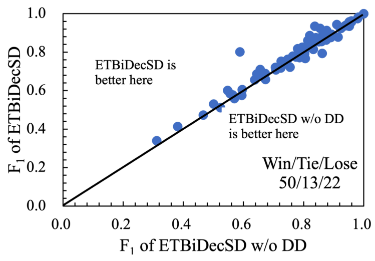
(a) ETBiDecSD vs. ETBiDecSD w/o DD



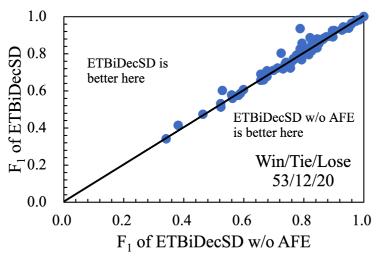
(b) ETBiDecSD vs. ETBiDecSD w/o AFE



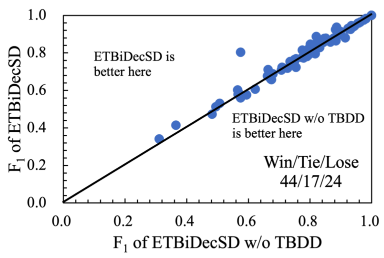
(c) ETBiDecSD vs. ETBiDecSD w/o TBDD



(d) ETBiDecSD vs. ETBiDecSD w/o DD



(e) ETBiDecSD vs. ETBiDecSD w/o AFE



(f) ETBiDecSD vs. ETBiDecSD w/o TBDD

Fig. 2. Accuracy and F_1 score plots highlighting the discernible performance disparity between two given ETBiDecSD variants across 85 well-known UCR2018 datasets, with the baseline being FCN.

tested scenarios.

3) ETBiDecSD with various μ values: μ is a key coefficient employed to balance the various loss functions in ETBiDecSD. Table V presents the statistical results from experiments using different μ values across 85 well-known UCR2018 datasets.

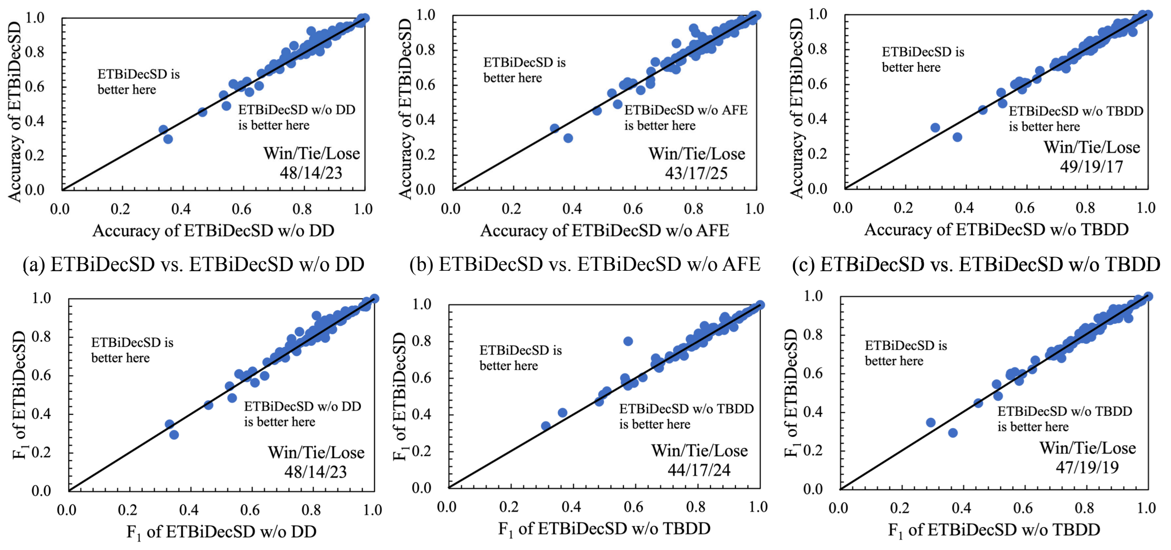
Upon evaluating the performance of ETBiDecSD with varying μ values, it is evident that $\mu = 0.1$ yields the highest 'win'/'tie'/'lose'/'best' outcomes and the lowest avg. rank, outperforming all other configurations. This confirms that μ = 0.1 strikes the optimal balance, enhancing ETBiDecSD's performance and making it the most effective setting for the model.

D. Ablation Study

To assess the crucial components' efficacy on ETBiDecSD, we contrast it with three ETBiDecSD variants, detailed below.

- ETBiDecSD w/o DD: ETBiDecSD without the decoupled distillation.
- ETBiDecSD w/o AFE: ETBiDecSD without the average feature ensemble.
- ETBiDecSD w/o TBDD: ETBiDecSD without the transitive bidirectional decoupled distillation.

1) Efficacy of Decoupled Distillation: To assess the efficacy of decoupled distillation (DD), we compare ETBiDecSD with ETBiDecSD w/o DD across 85 renowned UCR2018 datasets. Incorporating DD, while beneficial, introduces additional computational costs. As presented in Table VI, ETBiDecSD necessitates approximately 21.3471 hours to train on these datasets, in contrast to 20.2228 hours required by the version without DD, when the baseline is FCN. Despite this, the integration of DD significantly enhances model performance, as demonstrated in Figs. 2 (a)(d) and 3 (a)(d). Specifically, Fig. 3 (a) shows that ETBiDecSD, with DD, achieves a 'win'/'tie'/'lose'



(a) ETBiDecSD vs. ETBiDecSD w/o DD (b) ETBiDecSD vs. ETBiDecSD w/o AFE (c) ETBiDecSD vs. ETBiDecSD w/o TBDD

(d) ETBiDecSD vs. ETBiDecSD w/o DD (e) ETBiDecSD vs. ETBiDecSD w/o AFE (f) ETBiDecSD vs. ETBiDecSD w/o TBDD

Fig. 3. Accuracy and F_1 score plots highlighting the discernible performance disparity between two given ETBiDecSD variants across 85 well-known UCR2018 datasets, with the baseline being InceptionTime.

TABLE VI
STATISTICAL EFFICIENCY OUTCOMES OBTAINED FROM FOUR ETBIDECSD VARIANTS ACROSS 85 WELL-KNOWN UCR2018 DATASETS.

Baseline	Efficiency Metric	ETBiDecSD w/o DD	ETBiDecSD w/o AFE	ETBiDecSD w/o TBDD	ETBiDecSD
FCN	Total Training Time (h)	20.2228	19.8356	**19.5222**	21.3471
	Average CPU Usage (%)	21.1249	21.0339	**20.8109**	22.2994
	Average GPU Usage (%)	58.7963	59.3012	**58.5698**	60.6784
	Average Memory Usage (%)	19.5297	19.4473	**19.2876**	20.1892
Inception Time	Total Training Time (h)	24.6728	23.3664	**22.9360**	26.0445
	Average CPU Usage (%)	21.4622	21.3697	**21.1431**	22.6554
	Average GPU Usage (%)	60.8149	60.5582	**60.3839**	62.8686
	Average Memory Usage (%)	20.9606	20.8721	**20.6154**	21.6684

ratio of 48/14/23 in accuracy compared to its non-DD variant. This improvement underscores the model's capability to utilize both target-class and non-target-class knowledge flows, thereby capturing complex relationships and latent regularities within the data. Such findings affirm the substantial performance gains attributed to DD, notwithstanding the associated computational overhead.

2) Efficacy of Average Feature Ensemble: To evaluate the impact of the average feature ensemble (AFE), we compare ETBiDecSD with ETBiDecSD w/o AFE across 85 widely recognized UCR2018 datasets. While integrating AFE introduces additional computational overhead, as evidenced by increased CPU and GPU resource utilization in Table VI, it significantly enhances model performance. Figs. 2 (b)(e) and 3 (b)(e) demonstrate that AFE substantially enriches the higher-level semantic information within ETBiDecSD. This enhancement results in notable performance improvements compared to the model without AFE. Specifically, Figs. 2 (b) and 3 (b) show that ETBiDecSD with AFE achieves 'win'/'tie'/'lose' results of 54/19/12 and 43/17/25 in accuracy, respectively, when evaluated against FCN and InceptionTime baselines. These results underscore the effectiveness of AFE in capturing complex data patterns and improving accuracy, despite the associated increase in computational demands.

3) Efficacy of Transitive Bidirectional Decoupled Distillation: To validate the efficacy of transitive bidirectional decoupled distillation (TBDD), we compare ETBiDecSD with ETBiDecSD w/o TBDD across 85 well-known UCR2018 datasets. As detailed in Table VI, the inclusion of TBDD increases computational resource overhead for ETBiDecSD, leading to longer training times and higher CPU and memory usage. However, the integration of TBDD significantly enhances the interaction between higher and lower levels within the model, resulting in improved overall performance. For instance, Figs. 2 (c) and 3 (c) depict the accuracy of ETBiDecSD versus ETBiDecSD w/o TBDD, using FCN and InceptionTime as baselines. ETBiDecSD with TBDD demonstrates superior performance, achieving 'win'/'tie'/'lose' results of 46/17/22 and 49/19/17, respectively, compared to its non-TBDD counterpart. These results underscore the effectiveness of TBDD in fostering deeper interactions within the model, thereby capturing complex data patterns and enhancing accuracy, despite the associated increase in computational demands.

In summary, while DD, AFE, and TBD introduce significant computational overhead to ETBiDecSD, they are crucial components of the model. Each plays a vital role in addressing various TSC challenges, enhancing the model's performance by capturing complex patterns and improving accuracy. Despite the increased resource demands, these components are

TABLE VII
STATISTICAL OUTCOMES OBTAINED FROM SEVERAL WELL-ESTABLISHED SELF-DISTILLATION ALGORITHMS ACROSS 85 WELL-KNOWN UCR2018 DATASETS, WHEN THE BASELINE IS FCN.

Baseline	Evaluation Metric		Baseline	BYOT	TSD	SAD	ProSelfLC	SelfRef	ESD	Self-BiDecKD	ETBiDecSD
	Fundamental Metric	Statistical Metric	(FCN)								
FCN	*Accuracy*	Best	12	14	12	14	15	12	17	18	**33**
		Win	6	7	4	5	7	6	7	4	**18**
		Tie	6	7	8	9	8	6	10	14	**15**
		Lose	73	71	73	71	70	73	68	67	**52**
		avg. rank	6.1588	5.2059	5.4882	5.2529	4.6834	5.4529	5.2882	3.8471	**3.6235**
	F_1	Best	12	14	12	15	14	12	17	19	**33**
		Win	6	7	4	4	7	6	7	5	**18**
		Tie	6	7	8	11	7	6	10	14	**15**
		Lose	73	71	73	70	71	73	68	66	**52**
		avg. rank	6.3648	5.3800	5.6717	5.4286	4.8400	5.6352	5.4650	3.9757	**3.7447**
	Total Training Time (h)		**15.0206**	16.5227	16.8531	18.5384	19.6507	20.0438	20.4446	20.8535	21.3471
	Average CPU Usage (%)		**20.1151**	20.3162	20.5194	20.7246	20.9318	21.1412	21.3526	21.5661	22.2994
	Average GPU Usage (%)		**58.3588**	58.5339	58.7095	58.8856	59.0623	59.2395	59.4172	59.5954	60.6784
	Average Memory Usage (%)		**18.7931**	18.9810	19.1708	19.3625	19.4206	19.4789	19.5373	19.5959	20.1892

TABLE VIII
STATISTICAL OUTCOMES OBTAINED FROM SEVERAL WELL-ESTABLISHED SELF-DISTILLATION ALGORITHMS ACROSS 85 WELL-KNOWN UCR2018 DATASETS, WHEN THE BASELINE IS INCEPTIONTIME.

Baseline	Evaluation Metric		Baseline	BYOT	TSD	SAD	ProSelfLC	SelfRef	ESD	Self-BiDecKD	ETBiDecSD
	Fundamental Metric	Statistical Metric	(InceptionTime)								
Inception Time	*Accuracy*	Best	13	14	14	16	18	13	20	22	**35**
		Win	5	7	5	7	9	6	8	10	**20**
		Tie	8	7	9	9	9	7	12	12	**15**
		Lose	72	71	71	69	67	72	65	63	**50**
		avg. rank	6.3217	5.0710	5.5019	5.1282	4.5634	5.3882	5.2559	3.8158	**3.4528**
	F_1	Best	13	14	15	16	17	13	20	23	**34**
		Win	5	7	6	7	8	6	8	11	**19**
		Tie	8	7	9	9	9	7	12	12	**15**
		Lose	72	71	70	69	68	72	65	62	**51**
		avg. rank	6.5331	5.2406	5.6859	5.2997	4.7160	5.5684	5.4317	3.9434	**3.5683**
	Total Training Time (h)		**17.6119**	19.3731	19.7606	21.7366	23.0408	23.7320	24.4440	25.1773	26.0445
	Average CPU Usage (%)		**18.9896**	19.1795	19.3713	19.9524	20.9501	21.1596	21.5827	21.7986	22.6554
	Average GPU Usage (%)		**59.5388**	59.7174	59.8965	60.0762	60.3766	60.5577	60.9211	61.1038	62.8686
	Average Memory Usage (%)		**15.9963**	18.2358	19.3299	19.7165	20.3080	20.8360	20.9402	21.0030	21.6684

indispensable for effectively solving diverse TSC problems within ETBiDecSD.

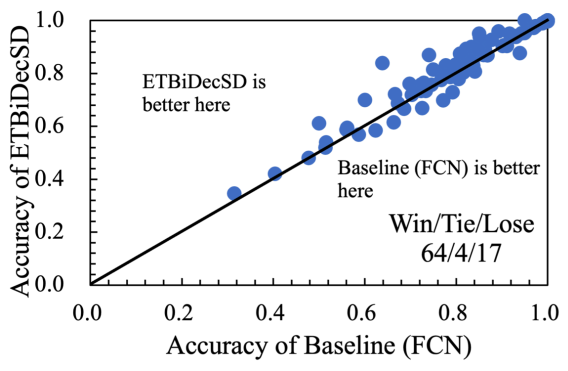
E. Experimental Outcome Analysis and Discussion

To assess the efficacy of ETBiDecSD, we benchmark its performance against several well-established self-distillation algorithms across 85 widely recognized UCR2018 datasets.

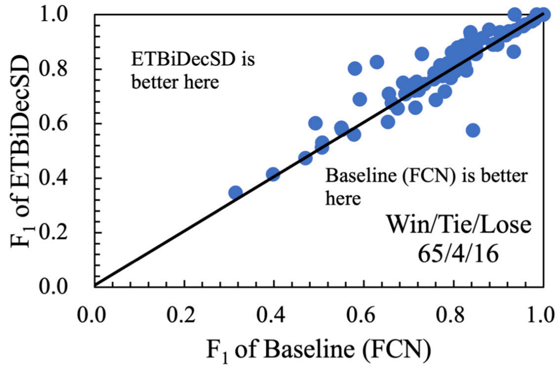
- Baseline (FCN): the pure FCN without any self-distillation.
- Baseline (InceptionTime): the pure InceptionTime without any self-distillation.
- TSD: the self-distillation approach based on transitive technique for time series knowledge transfer [27].
- BYOT: the be your own teacher distillation approach, adapted for time series representation learning [24].
- ProSelfLC: the modified time series end-to-end approach based on progressive self-label correction [26].
- SAD: the modified time series self-distillation approach embedding layer-wise attention [75] .
- ESD: the modified time series self-distillation model integrating ensemble technique [27].
- SelfRef: the effective self-distillation model with feature refinement, adapted for TSC [25].
- Self-BiDecKD: the self-bidirectional decoupled distillation approach for TSC [29].

Tables VII and VIII present the statistical outcomes obtained from several well-established self-distillation algorithms across 85 well-known UCR2018 datasets, with FCN and InceptionTime as the baselines, respectively. ETBiDecSD emerges as the best-performing self-distillation algorithm among all comparison methods, excelling in both reference accuracy and F_1 score. It achieves the highest 'win'/'tie'/'lose'/'best' values and the lowest avg. rank score. For instance, with InceptionTime as the baseline, ETBiDecSD attains a 'win'/'tie'/'lose'/'best' ratio of 20/15/50/35 and an avg. rank score of 3.4528 in accuracy. This exceptional performance is attributed to ETBiDecSD's effective utilization of the average feature ensemble, which integrates the output from each level, thereby enhancing the robustness of higher-level semantic information. Furthermore, the transitive bidirectional decoupled distillation facilitates deep interaction between higher- and lower-level target-class and non-target-class knowledge within the model. Self-BiDecKD follows closely, employing bidirectional decoupled knowledge distillation to enhance knowledge transfer within the model and achieving competitive results in terms of 'best' and avg. rank, regardless of whether the baseline is FCN or InceptionTime. In contrast, TSD struggles to extract sufficient features through transitive self-distillation alone, resulting in its comparatively lower performance among the evaluated self-distillation algorithms.

Then, as demonstrated in Tables VII and VIII, when examining the resource expenditure of various algorithms, it becomes evident that ETBiDecSD, which incorporates both an average feature ensemble and a transitive bidirectional decoupled distillation module, inevitably incurs higher computational costs. Specifically, when InceptionTime serves as the baseline, ET-

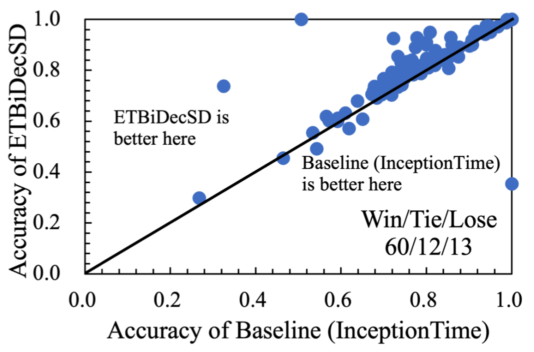


(a) ETBiDecSD vs. Baseline (FCN) in accuracy

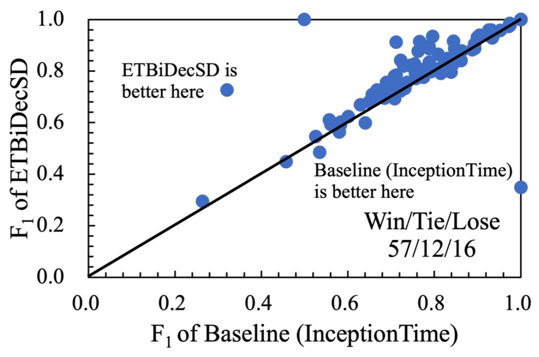


(b) ETBiDecSD vs. Baseline (FCN) in F₁

Fig. 4. Accuracy and F_1 score plots highlighting the discernible performance disparity between ETBiDecSD and Baseline (FCN) across 85 well-known UCR2018 datasets, with ETBiDecSD utilizing FCN as the baseline.



(a) ETBiDecSD vs. Baseline (InceptionTime) in accuracy



(b) ETBiDecSD vs. Baseline (InceptionTime) in F₁

Fig. 5. Accuracy and F_1 score plots highlighting the discernible performance disparity between ETBiDecSD and Baseline (InceptionTime) across 85 well-known UCR2018 datasets, with ETBiDecSD utilizing InceptionTime as the baseline.

BiDecSD requires the longest training duration for 85 datasets and exhibits the highest CPU, GPU, and memory usage. Conversely, algorithms that employ a simple self-distillation module, such as BOYT and TSD, demonstrate significantly lower computational overhead but at the expense of suboptimal performance. This clearly illustrates that while ETBiDecSD delivers notable performance enhancements, it does so with a corresponding increase in resource consumption.

To comprehensively evaluate the efficacy of ETBiDecSD, we conduct a comparative analysis against a non-self-distillation baseline model across 85 well-known UCR2018 datasets. As shown in Tables VII and VIII, ETBiDecSD does indeed consume more computing resources than the non-self-distillation baseline model, with increased training time and higher CPU and GPU usage. However, this increase in resource expenditure is accompanied by a significant improvement in performance. Fig. 4 presents accuracy and F_1 score plots, highlighting the performance of ETBiDecSD relative to the baseline (FCN). When using FCN as the baseline, ETBiDecSD achieves 'win'/'tie'/'lose' results of 64/4/17 in accuracy and 65/4/16 in F_1 score. Similarly, as illustrated in Fig. 5, when compared with the InceptionTime baseline, ETBiDecSD attains 'win'/'tie'/'lose' results of 60/12/13 in accuracy and 57/12/16 in F_1 score. These results demonstrate that the integration of the average feature ensemble and the transitive bidirectional decoupled distillation effectively facilitates deep interaction between higher- and lower-level semantic information within the model. Such mechanisms enable ETBiDecSD to extract rich and diverse connections and regularizations from the data, underscoring its effectiveness across a wide range of TSC problems.

V. Conclusion

This work revisits self-distillation for TSC from the perspective of hierarchical semantic interaction rather than architectural complexity alone. Unlike conventional approaches that primarily rely on the final output layer as the sole source of supervisory signals, ETBiDecSD explicitly targets two structural limitations in existing methods: the progressive degradation of semantic richness across layers and the systematic neglect of non-target class knowledge. By introducing an average feature ensemble mechanism, ETBiDecSD reconstructs a more informative and stable source of higher-level semantics that integrates multi-level temporal representations. This design aligns with the intrinsic characteristics of time series data, where meaningful patterns are distributed across different temporal resolutions and abstraction depths, rather than being concentrated at a single terminal layer. Building upon this enriched semantic foundation, the proposed transitive bidirectional decoupled distillation framework enables structured knowledge circulation between higher and lower layers while simultaneously disentangling target and non-target class information. This dual-stream interaction reflects a key property of time series learning: robust temporal understanding emerges from both discriminative cues and contextual regularities embedded in non-target classes. Empirical results across diverse UCR2018 datasets consistently validate that reinforcing lower-level representations with semantically enriched, bidirectionally exchanged knowledge leads to more stable optimization and improved generalization. Collectively, these findings suggest that performance gains stem not merely from additional supervision, but from a principled reorganization of semantic flow tailored to the hierarchical and multi-scale nature of time series data.

While ETBiDecSD achieves competitive performance, its reliance on multiple distillation objectives with fixed coefficients incurs notable computational overhead. This static configuration may yield suboptimal synergy among loss terms and introduces redundant computations. To address this, future work will explore automated weighting and loss selection strategies, such as reinforcement learning, to improve efficiency. In addition, we will investigate optimization techniques including dynamic sampling and knowledge compression to reduce training costs while preserving semantic fidelity. These directions aim to enhance the scalability of ETBiDecSD in resource-constrained scenarios.

Acknowledgement

The authors wish to convey their heartfelt gratitude to the editors and reviewers for their invaluable insights, which have substantially enriched the quality of this research.

References

[1] H. Tian, X. Zhang, X. Zheng, and D. D. Zeng, "Learning dynamic dependencies with graph evolution recurrent unit for stock predictions," *IEEE Trans. Syst. Man. Cybern. Syst.*, vol. 53, no. 11, pp. 6705–6717, 2023.

[2] S. Das, S. Mustavee, S. Agarwal, and S. Hasan, "Koopman-theoretic modeling of quasiperiodically driven systems: Example of signalized traffic corridor," *IEEE Trans. Syst. Man. Cybern. Syst.*, vol. 53, no. 7, pp. 4466–4476, 2023.

[3] L. Xu, X. Fu, Y. Wang, Q. Zhang, H. Zhao, Y. Lin, and G. Gui, "Enhanced few-shot specific emitter identification via phase shift prediction and decoupling," *IEEE Trans. Cogn. Commun.*, pp. 1–11, 2024.

[4] X. Zhang, Q. Wang, M. Qin, Y. Wang, T. Ohtsuki, B. Adebisi, H. Sari, and G. Gui, "Enhanced few-shot malware traffic classification via integrating knowledge transfer with neural architecture search," *IEEE Trans. Inf. Foren. Sec.*, vol. 19, pp. 5245–5256, 2024.

[5] A. Marchioni, A. Enttsel, M. Mangia, R. Rovatti, and G. Setti, "Anomaly detection based on compressed data: An information theoretic characterization," *IEEE Trans. Syst. Man. Cybern. Syst.*, vol. 54, no. 1, pp. 23–38, 2024.

[6] Z. Liu, F. Xiao, C.-T. Lin, and Z. Cao, "A robust evidential multisource data fusion approach based on cooperative game theory and its application in eeg," *IEEE Trans. Syst. Man. Cybern. Syst.*, vol. 54, no. 2, pp. 729–740, 2024.

[7] J. Jiao, H. Li, J. Lin, and H. Zhang, "Entropy-oriented domain adaptation for intelligent diagnosis of rotating machinery," *IEEE Trans. Syst. Man. Cybern. Syst.*, vol. 54, no. 2, pp. 1239–1249, 2024.

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

[9] H. Xing, Z. Xiao, 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.

[10] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," in *Proc. Int. Jt. Conf. Neural Networks (IJCNN)*, 2017, pp. 1578–1585.

[11] A. Bagnall., J. Lines, A. Bostrom, J. Large, and E. Keogh, "The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances," *Data Min. Knowl. Disc.*, vol. 31, pp. 1–55, 2017.

[12] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, pp. 436–444, 2015.

[13] 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. Int. Jt. Conf. Neural Networks (IJCNN)*, 2019, pp. 1–8.

[14] Q. Ma, S. Li, and G. W. Cottrell, "Adversarial joint-learning recurrent neural network for incomplete time series classification," *IEEE Trans. Pattern Anal.*, vol. 44, no. 4, pp. 1765–1776, 2022.

[15] H. I. Fawaz, B. Lucas, G. Forestier, C. Pelletier, D. F. Schmidt, J. Weber, G. I. 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.

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

[17] Z. Xiao, X. Xu, H. Xing, R. Qu, F. Song, and B. Zhao, "Rnts: Robust neural temporal search for time series classification," in *Proc. Int. Jt. Conf. Neural Networks (IJCNN)*, 2021, pp. 1–8.

[18] 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.

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

[20] L. Jin, J. Cheng, J. Shi, and F. Huang, "Brief introduction of back propagation (bp) neural network algorithm and its improvement," *In: Jin, D., Lin, S. (eds) Advances in Computer Science and Information Engineering. Advances in Intelligent and Soft Computing*, 2012.

[21] A. Yao and D. Sun, "Knowledge transfer via dense cross-layer mutual-distillation," in *Proc. Lect. Notes Comput. Sci., ECCV*, 2020, pp. 294–311.

[22] J. Kim, S. Park, and N. Kwak, "Paraphrasing complex network: network compression via factor transfer," in *Proc. Conf. Neural Inf. Process. Syst.*, 2018, pp. 2765–2774.

[23] J. Guo, B. Yu, S. Maybank, and D. Tao, "Knowledge distillation: A survey," *Int. J. Comput. Vision*, vol. 129, p. 1789–1819, 2021.

[24] 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 *Proc. IEEE Int. Conf. Comput. Vision (ICCV)*, 2019, pp. 3712–3721.

[25] M. Ji, S. Shin, S. Hwang, G. Park, and I.-C. Moon, "Refine myself by teaching myself: Feature refinement via self-knowledge distillation," in *Proc IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit. (CVPR)*, 2021, pp. 10 659–10 668.

[26] X. Wang, Y. Hua, E. Kodirov, D. A. Clifton, and N. M. Robertson, "Proselflc: Progressive self label correction for training robust deep neural networks," in *Proc IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit. (CVPR)*, 2021, pp. 752–761.

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

[28] B. Zhao, Q. Cui, R. Song, Y. Qiu, and J. Liang, "Decoupled knowledge distillation," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit. (CVPR)*, 2022, pp. 11 943–11 952.

[29] Z. Xiao, H. Xing, R. Qu, H. Li, L. Feng, B. Zhao, and J. Yang, "Self-bidirectional decoupled distillation for time series classification," *IEEE Trans. Artif. Intell.*, vol. 5, no. 8, pp. 4101–4110, 2024.

[30] J. Lines, S. Taylor, and A. Bagnall, "Time series classification with hive-cote: the hierarchical of transformation-based ensembles," *ACM Trans. Knowl. Discov. D.*, vol. 21, no. 52, pp. 1–35, 2018.

[31] A. Bagnall, J. Lines, J. Hills, and A. Bostrom, "Time series classification with cote: the collective of transformation-based ensembles," in *Proc. IEEE Int. Conf. Data Eng. (ICDE)*, 2016, pp. 1548–1549.

[32] 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," *Mach. Learn.*, vol. 110, pp. 3211–3243, 2021.

[33] K. Fauvel, É. Fromont, V. Masson, P. Faverdin, and A. Termier, "Xem: An explainable-by-design ensemble method for multivariate time series classification," *Data Min. Knowl. Disc.*, vol. 36, pp. 917–957, 2022.

[34] W. Pei, H. Dibeklioğlu, D. M. J. Tax, and L. van der Maaten, "Multivariate time-series classification using the hidden-unit logistic model," *IEEE Trans. Neur. Net. Lear.*, vol. 29, no. 4, pp. 920–931, 2018.

[35] K. S. Tuncel and M. G. Baydogan, "Autoregressive forests for multivariate time series modeling," *Pattern Recogn.*, vol. 73, pp. 202–215, 2018.

[36] M. G. Baydogan and G. Runger, "Time series representation and similarity based on local auto patterns," *Data Min. Knowl. Disc.*, vol. 30, pp. 476–509, 2016.

[37] J. Large, A. Bagnall, S. Malinowski, and R. Tavenard, "From bop to boss and beyond: time series classification with dictionary based classifier," *arXiv preprint arXiv:1809.06751*, 2018.

[38] P. Schäfer and U. Leser, "Multivariate time series classification with weasel+muse," *arXiv preprint arXiv:1711.11343*, 2017.

[39] K. Wu, K. Yuan, Y. Teng, J. Liu, and L. Jiao, "Broad fuzzy cognitive map systems for time series classification," *App. Soft Comput.*, vol. 128, pp. 1–13, 2022.

[40] G. He, X. Xin, R. Peng, M. Han, J. Wang, and X. Wu, "Online rule-based classifier learning on dynamic unlabeled multivariate time series data," *IEEE Trans. Syst. Man. Cybern. Syst.*, vol. 52, no. 2, pp. 1121–1134, 2022.

[41] F. J. Erazo-Costa, P. C. L. Silva, and F. G. Guimarães, "A fuzzy-probabilistic representation learning method for time series classification," *IEEE Trans. Fuzzy Syst.*, vol. 32, no. 5, pp. 2940–2952, 2024.

[42] L. Zhang and F. Xiao, "Belief rényi divergence of divergence and its application in time series classification," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 8, pp. 3670–3681, 2024.

[43] Y. Wu, Y. Meng, Y. Li, L. Guo, X. Zhu, P. Fournier-Viger, and X. Wu, "Copp-miner: Top-k contrast order-preserving pattern mining for time series classification," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 6, pp. 2372–2387, 2024.

[44] P. Shi, X. Dang, W. Ye, Z. Li, and Z. Qin, "Mrm2: Multi-relationship modeling module for multivariate time series classification," in *Proc. IEEE Int. Conf. Data Min. (ICDM)*, 2022, pp. 1185–1190.

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

[46] R. Chen, X. Yan, S. Wang, and G. Xiao, "Da-net: Dual-attention network for multivariate time series classification," *Inf. Sci.*, vol. 610, pp. 472–487, 2022.

[47] G. Li, B. Choi, J. Xu *et al.*, "Shapenet: A shapelet-neural network approach for multivariate time series classification," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 8375–8383.

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

[49] A. Dempster, D. F. Schmidt, and G. I. Webb, "Minirocket: A very fast (almost) deterministic transform for time series classification," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2021, pp. 248–257.

[50] D. Lee, S. Lee, and H. Yu, "Learnable dynamic temporal pooling for time series classification," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 8288–8296.

[51] P. Ranjan, P. Khan, S. Kumar, and S. K. Das, "log-sigmoid activation-based long short-term memory for time-series data classification," *IEEE Trans. Artif. Intell.*, vol. 5, no. 2, pp. 672–683, 2024.

[52] N. M. Foumani, C. W. Tan, G. I. Webb, H. Rezatofighi, and M. Salehi, "Series2vec: similarity-based self-supervised representation learning for time series classification," *Data Min. Knowl. Disc.*, pp. 1–25, 2024.

[53] X. Li, W. Xi, and J. Lin, "Randomnet: clustering time series using untrained deep neural networks," *Data Min. Knowl. Disc.*, pp. 1–30, 2024.

[54] L. Sun, C. Li, Y. Ren, and Y. Zhang, "A multitask dynamic graph attention autoencoder for imbalanced multilabel time series classification," *IEEE Trans. Neur. Net. Lear.*, pp. 1–14, 2024.

[55] A. Campagner, M. Barandas, D. Folgado, H. Gamboa, and F. Cabitza, "Ensemble predictors: Possibilistic combination of conformal predictors for multivariate time series classification," *IEEE Trans. Pattern Anal.*, pp. 1–12, 2024.

[56] Z. Xiao, H. Xing, B. Zhao, R. Qu, S. Luo, P. Dai, K. Li, and Z. Zhu, "Deep contrastive representation learning with self-distillation," *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 8, no. 1, pp. 3–15, 2024.

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

[58] 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.

[59] X. Zhang, Y. Gao, J. Lin, and C.-T. Lu, "Tapnet: Multivariate time series classification with attentional prototypical network," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 6845–6852.

[60] Z. Xiao, H. Xing, R. Qu, L. Feng, S. Luo, P. Dai, B. Zhao, and Y. Dai, "Densely knowledge-aware network for multivariate time series classification," *IEEE Trans. Syst. Man. Cybern. Syst.*, vol. 54, no. 4, pp. 2192–2204, 2024.

[61] G. Hinton, O. Vinyals, and J. Dean, "Distillation the knowledge in a neural network," *arXiv preprint arXiv: 1503.02531*, 2015.

[62] A. Romero, N. Ballas, S. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Fitnet: hints for thin deep nets," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–13.

[63] N. Passalis and A. Tefas, "Learning deep representations with probabilistic knowledge transfer," in *Proc. Lect. Notes Comput. Sci., ECCV*, 2018, p. 283–299.

[64] S. I. Mirzadeh, M. Farajtabar, A. Li, and H. Ghasemzadeh, "Improved knowledge distillation via teacher assistant," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 5191–5198.

[65] T. Li, J. Li, Z. Liu, and C. Zhang, "Few sample knowledge distillation for efficient network compression," in *Proc IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit. (CVPR)*, 2020, pp. 14 627–14 635.

[66] T. Su, J. Zhang, Z. Yu, G. Wang, and X. Liu, "Stkd: Distilling knowledge from synchronous teaching for efficient model compression," *IEEE Trans. Neur. Net. Lear.*, vol. 34, no. 12, pp. 10 051–10 064, 2023.

[67] Q. Zhao, S. Lyu, L. Chen, B. Liu, T.-B. Xu, G. Cheng, and W. Feng, "Learn by oneself: Exploiting weight-sharing potential in knowledge distillation guided ensemble network," *IEEE Trans. Circ. Syst. Vid.*, vol. 33, no. 11, pp. 6661–6678, 2023.

[68] D. Chen, J. P. Mei, C. Wang, Y. Feng, and C. Chen, "Online knowledge distillation with diverse peers," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 3430–3437.

[69] H. Zhang, Z. Hu, W. Qin, M. Xu, and M. Wang, "Adversarial co-distillation learning for image recognition," *Pattern Recogn.*, vol. 111, pp. 1–10, 2021.

[70] G. B. Kshirsagar and N. D. Londhe, "Ds-p3snet: An efficient classification approach for devanagari script-based p300 speller using compact channelwise convolution and knowledge distillation," *IEEE Trans. Syst. Man. Cybern. Syst.*, vol. 52, no. 12, pp. 7431–7443, 2022.

[71] T. Su, Q. Liang, J. Zhang, Z. Yu, Z. Xu, G. Wang, and X. Liu, "Deep cross-layer collaborative learning network for online knowledge distillation," *IEEE Trans. Circ. Syst. Vid.*, vol. 33, no. 5, pp. 2075–2087, 2023.

[72] C. Yang, Z. An, H. Zhou, F. Zhuang, Y. Xu, and Q. Zhang, "Online knowledge distillation via mutual contrastive learning for visual recognition," *IEEE Trans. Pattern Anal.*, vol. 45, no. 8, pp. 10 212–10 227, 2023.

[73] T. Su, J. Zhang, Z. Yu, G. Wang, and X. Liu, "Stkd: Distilling knowledge from synchronous teaching for efficient model compression," *IEEE Trans. Neur. Net. Lear.*, vol. 34, no. 12, pp. 10 051–10 064, 2023.

[74] H. Mobahi, M. Farajtabar, and P. L. Bartlett, "Self-distillation amplifies regularization in hilbert space," in *Proc. Adv. neural inf. proces. syst.*, 2020, pp. 1–11.

[75] Y. Hou, Z. Ma, C. Liu, and C. C. Loy, "Learning lightweight lane detection cnns by self attention distillation," in *Proc. IEEE Int. Conf. Comput. Vision (ICCV)*, 2019, pp. 1013–1021.

[76] 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 J. Autom. Sin.*, vol. 6, no. 6, pp. 1293–1305, 2019.

[77] D. Rey and M. Neuhäuser, "Wilcoxon-signed-rank test," *In: Lovric, M. (eds) International Encyclopedia of Statistical Science*, 2021.

Zhiwen Xiao (Member, IEEE) received the B.Eng. degree in network engineering from the Chengdu University of Information Technology, Chengdu, China, in 2019, and the M.Eng. degree in computer science from the Northwest A&F University, Yangling, China, in 2023. He is currently pursuing the Ph.D. degree in computer science with Southwest Jiaotong University, Chengdu. He was honored as a Distinguished Program Committee Member for ECML-PKDD'25. His research interests include data mining, time series analysis, federated learning, representation learning, semantic communication, and computer vision.

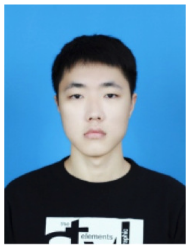
Huanlai Xing (Member, IEEE) received Ph.D. degree in computer science from University of Nottingham (Supervisor: Dr Rong Qu), Nottingham, U.K., in 2013. He was a Visiting Scholar in Computer Science, The University of Rhode Island (Supervisor: Dr. Haibo He), USA, in 2020-2021. Huanlai Xing is with the School of Computing and Artificial Intelligence, Southwest Jiaotong University (SWJTU), and Tangshan Institute of SWJTU. He is an associate editor of IEEE Signal Processing Letters. 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 IJCAI, 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.

Rong Qu (Fellow, IEEE) is a full 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/area editor at IEEE Computational Intelligence Magazine, IEEE Transactions on Evolutionary Computation, Swarm and Evolutionary Computation, Journal of Operational Research Society, and Journal of Information and Intelligence. 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.

Hui Li received the B.Sc. and M.Sc. degrees in applied mathematics from the School of Mathematics and Statistics, Xi'an Jiaotong University, Xi'an, China, in 1999 and 2002, respectively, and the Ph.D. degree in computer science from the University of Essex, Colchester, U.K., in 2008. He is currently a Professor with the School of Mathematics and Statistics, Xi'an Jiaotong University. His current research interests include evolutionary computation, multiobjective optimization, and machine learning. Dr. Li was a recipient of the 2010 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Award as one of the inventors for MOEA/D.

Li Feng received his PhD degree in control science and engineering from the Xi'an Jiaotong University, Xi'an, China, in 2005, under the supervision of Prof. Xiaohong Guan (Academian of CAS). He is a Research Professor and PhD supervisor with the School of Computing and Artificial Intelligence, Southwest Jiaotong University, Chengdu. His research interests include artificial intelligence, cyber security and its applications.



Bowen Zhao received his B. Eng. degree in Computer Science and Technology in 2020, from Southwest Jiaotong University, Sichuan, China. He recieved the Ph.D. degree in computer science in 2025, from Southwest Jiaotong University, Chengdu, China. His research interests include deep reinforcement learning, semantic communication, cloud computing, and deep learning.

Qian Wan received the B.Eng. degree in computer science from the Wuhan University, Wuhan, China, in 2016, and the M.Des. degree in interaction design from the China University of Geosciences, Wuhan, China, in 2020. He is pursuing the Ph.D. degree in computer science at Southwest Jiaotong University, Chengdu, China. His research interests include data model and mining, virtual reality, object detection, computer vision applications for art design, and augmented reality.