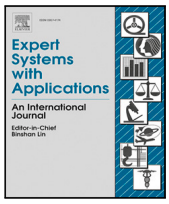
Expert Systems With Applications 283 (2025) 127493

Contents lists available at ScienceDirect

Expert Systems With Applications

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



A contrastive clustering loss function increases class-balanced in time series classification

Wu Chaomin [a], Cheng Xu [b], Wang Hao [a,*]

[a] College of Network and Information Security, Xidian University, Xi'an 710126, China
[b] Department of Technology, Management and Economics, Energy Economics and Modelling, Technical University of Denmark, Lyngby 2800, Denmark

ARTICLE INFO

Keywords:
Imbalanced class
Contrastive learning
Loss function
Time-series classification

ABSTRACT

Addressing class imbalance in temporal data entails distinct complexities arising from substantial disparities in class frequencies within time series, which complicate model training and generalization. When particular classes have substantially fewer instances than others, models are prone to emphasizing the more frequent classes at the expense of adequately recognizing infrequent ones, thereby compromising predictive performance. This imbalance frequently produces biased models with limited capacity to accurately categorize minority classes, undermining the practical utility of temporal sequence classification. Acknowledging the shortcomings of existing techniques, we introduce a novel contrastive clustering (COCL) loss function that promotes discriminative learning of temporal representations under class imbalance. By leveraging contrastive loss principles, the proposed COCL formulation decreases similarity measures both within identical classes and across distinct classes, thereby mitigating class imbalance and strengthening the distinction between minority and majority groups. Furthermore, the COCL loss function is devised to improve clustering by minimizing contrastive losses among observations within the same cluster and between different clusters. Qualitative and quantitative experiments substantiate the superiority of the proposed COCL loss function compared with conventional classification methods and various leading-edge approaches.

1. Introduction

The complex and dynamic nature of time series data poses challenges for classification. Time series data, characterized by temporally dependent and evolving patterns, are prevalent in practical applications such as healthcare for monitoring patient vitals, economic forecasting for predicting market trends, and robotics for motion tracking. These evolving characteristics demand adaptable models, emphasizing the need for flexibility. Conventional static models often fail to adequately capture evolving features due to their limitations in handling temporal dependencies. The inherent non-stationarity in time series data requires advanced modeling techniques that dynamically adjust to changes and trend variations over time. Imbalanced distributions add complexity (Cheng, Shi, Liu, Zhao, & Chen, 2022; He, Siu, & Si, 2023; Yang, Yuan, & Wang, 2023), hindering model performance and causing bias. This challenge is particularly significant in practical applications such as autonomous vessels (Li et al., 2024) and healthcare, where monitoring patient vitals over time is crucial for the early detection of health issues (Ircio, Lojo, Lozano, & Mori, 2022). In economics, predicting market trends depends on understanding evolving economic indicators (Theodossiou, 1993), while in robotics, accurate motion tracking is essential for autonomous operations (Mei, Liu, Wang, & Gao, 2016). Enhancing classification for minority categories remains a central focus in contemporary time series classification research.

Numerous methods have been proposed to address the imbalanced time series classification (ITSC) challenge, which can be broadly categorized into three approaches: resampling methods, cost-sensitive methods, and the application of specific loss functions. Resampling techniques, grounded in a data-centric perspective, mitigate imbalance by either augmenting the minority class through oversampling or reducing the majority class through undersampling. In contrast, cost-sensitive methods operate at the algorithmic level, adjusting misclassification costs to prioritize the minority class and effectively alleviating imbalanced data distribution. The third category centers on devising innovative loss functions to alter classifier learning processes, enabling a more nuanced evaluation of class imbalance.

Existing oversampling methods enhance time series classification by synthesizing new samples rather than duplicating existing ones. Techniques like SMOTE (Chawla, Bowyer, Hall, & Kegelmeyer, 2002), Borderline-SMOTE (Han, Wang, & Mao, 2005), and ADASYN (He, Bai, Garcia, & Li, 2008) are widely used but often struggle to capture the

* Corresponding author.
E-mail addresses: 23151214136@stu.xidian.edu.cn (C. Wu), xu.cheng@ieee.org (X. Cheng), haow@ieee.org (H. Wang).

https://doi.org/10.1016/j.eswa.2025.127493
Received 2 January 2025; Received in revised form 9 March 2025; Accepted 28 March 2025
Available online 28 April 2025

essential temporal correlations in time series data. Therefore, specialized oversampling methods such as INOS (Cao, Ng, Li, & Woon, 2013), MOGT (Pang, Cao, & Tan, 2013), and SPO (Cao, Li, Woon, & Ng, 2011) have been developed to preserve and leverage the temporal information inherent in time series data. Although effective, these approaches can be computationally expensive and may exhibit limitations when dealing with nonlinear or complex data structures. Conversely, undersampling reduces the number of majority class samples to balance the dataset. While this method can mitigate model overfitting by eliminating redundant majority class instances, it risks discarding valuable information and compromising the identification of critical patterns. Additionally, Generative Adversarial Networks (GANs) contribute to oversampling by iteratively refining synthetic data generation through adversarial training. The BFGAN (Lee, Lee, & Kim, 2022) addresses the class imbalance in multimodal time series classification by integrating dataset information and evaluating the proximity of generated samples to the classification boundary. However, BFGAN's neural network heavily relies on a substantial sample corpus, posing challenges when the majority class significantly outweighs the minority class.

Cost-sensitive methodologies (Raj, Magg, & Wermter, 2016; Shen et al., 2018) involve modifying a classifier's loss function by incorporating class-specific weights. However, the applicability of this approach is constrained by the lack of universal support for direct loss function modifications across different classifiers. To overcome this limitation, recent advancements employ adaptive strategies that dynamically adjust misclassification costs throughout the training process, treating these adjustments as learnable parameters. Nevertheless, introducing additional parameters raises concerns regarding increased model complexity and computational overhead. Recent advancements (Gaudreault, Branco, & Gama, 2021) in this field place significant emphasis on performance metrics, particularly in addressing imbalanced scenarios, utilizing metrics such as F1 and AUC. In these frameworks, the primary objective of classifier training is to calibrate model parameters based on these performance metrics to effectively address class imbalances. However, the loss functions (Büttner, Schneider, Krasowski, Pitchika, Krois, Meyer-Lueckel, & Schwendicke, 2024) employed in these methodologies often lack differentiability, posing a conflict with conventional gradient-based learning requirements.

Despite the diversity of existing ITSC methods, each approach has inherent limitations, preventing the establishment of a universal solution for time series classification. Some methods are tailored specifically for one-dimensional time series, rendering them unsuitable for multi-dimensional applications. Furthermore, certain techniques focus exclusively on binary classification, lacking the flexibility required for multi-class scenarios. Traditional clustering methods, such as K-means (Huang, Ye, Xiong, Lau, Jiang, & Wang, 2016) and hierarchical clustering (Ali, Rani, Pravija Raj, & Khedr, 2025), often struggle with time series data due to their inability to capture temporal dependencies and evolving patterns. These approaches typically assume static data distributions, making them ill-suited for handling the sequential nature of time series data. Additionally, traditional clustering techniques may fail to account for data imbalance, leading to biased clustering outcomes that do not accurately reflect the true data structure.

This paper introduces a novel contrastive clustering (COCL) loss function designed to address challenges associated with time-series data. The COCL framework integrates contrastive learning principles, enabling the incorporation of temporal dependencies and capturing the dynamic nature of time series data. This approach is crucial for adapting to variations in data distribution, a common issue in real-world time series applications. We demonstrate how the COCL framework improves the analysis of datasets with ambiguous or undefined class distributions by reducing reliance on predefined categories, which is particularly beneficial for handling imbalanced distributions frequently observed in time series data. By emphasizing the relative interconnections among samples, the COCL framework enhances efficiency in imbalanced classification scenarios. The integration of contrastive learning within the COCL framework effectively reduces its reliance on predefined category distributions, a critical improvement for analyzing datasets with ambiguous or undefined category distributions. This enables a more effective discernment of complex relationships between categories. Furthermore, the framework's emphasis on relative interconnections between samples, as opposed to absolute quantities, strategically confronts the prevalent issue of data imbalance in time-series classification tasks. This approach markedly enhances COCL's efficiency in scenarios characterized by imbalanced data. Moreover, the COCL framework employs a similarity matrix, derived from normalized dot products of feature vectors, to optimize intra-cluster cohesion and inter-cluster separation, facilitating clearer cluster boundaries while preserving internal homogeneity. This method not only improves class distinction but also strengthens the overall clustering integrity. Finally, COCL's robustness to fluctuations in data distributions represents a significant advancement, ensuring adaptability and consistent performance across diverse datasets. This characteristic is particularly important for managing the evolving nature of real-world time series data, thereby maintaining the effectiveness of the framework across various scenarios. The main contributions of this study are summarized as follows:

- We propose a novel COCL loss function to facilitate the learning of distinctive data point representations by the model. By minimizing similarity scores within clusters and maximizing dissimilarity between different clusters, our approach effectively brings closer data points within the same cluster while promoting notable separation between distinct clusters. This advancement improves clustering performance and achieves unsupervised learning without the need for explicit clustering labels.
- To improve clustering performance, we prioritize minimizing contrastive loss between intra-cluster and inter-cluster samples, ensuring a well-structured distribution of similarity scores. The incorporation of a temperature parameter refines the scaling of similarity scores, enhancing adaptability to diverse data distributions.
- We conducted comprehensive experiments across 28 datasets and three classifiers, systematically comparing our method with alternative oversampling techniques. Additionally, we generated three hard disk datasets using publicly available data from the Backblaze company. The results unequivocally demonstrate the superior performance of our approach compared to contemporary state-of-the-art methods.

The remainder of this paper is structured as follows. In Section 2, related work on this topic is reviewed. Section 3 introduces the proposed method in detail. Sections 4 and 5 give the experimental results and discussion. Finally, this study is summarized in Section 6.

2. Related work

2.1. Time series classification

In the domain of time series classification, researchers have introduced various methods aimed at addressing the inherent challenges posed by sequential data classification. These approaches encompass methods based on statistical features, deep learning techniques, and methods centered around capturing sequence similarity. As emphasized by Lei et al. in their work (Lei & Wu, 2020), the initial methods for time series classification relied on statistical features. They utilized statistical properties (such as mean and variance extracted from sequences) to train classifiers. However, these methods often exhibit limited modeling capabilities for temporal changes and long-term dependencies within sequences, possibly overlooking subtle structural differences. Subsequently, the advent of deep learning has brought new developments to the field of time series classification. Wang, Yan, and Oates (2017)

explored the application of deep learning models, which, despite their effectiveness in abstract representation learning, require substantial data and computational resources and are prone to overfitting in low-sample scenarios. More recent architectures have further improved classification performance. Residual networks (Resnet) (Wang et al., 2017) mitigate gradient vanishing through residual connections, enhancing model depth and stability. Similarly, MLSTM-FCN (Karim, Majumdar, Darabi, & Chen, 2017), which integrates multi-layer long short-term memory networks with fully convolutional networks, has demonstrated strong performance in time series tasks. Additionally, some researchers have shifted their focus to methods considering sequence similarity. Keogh et al. (Keogh & Ratanamahatana, 2005) introduced similarity measures such as Dynamic Time Warping to better capture the similarity between sequences. However, the computation of similarity measures can be computationally intensive, especially for large datasets.

Despite significant advancements, existing methods in time series classification remain limited by inadequate modeling of temporal variations, sensitivity to small sample sizes, and high computational complexity. To address these challenges, researchers have increasingly explored advanced loss functions to improve representation learning (Ircio, Lojo, Mori, Malinowski, & Lozano, 2023). This paper builds on this approach by introducing a novel contrastive clustering loss function, detailed in Sections 3 and 4, emphasizing its design and practical applications in time series classification.

2.2. Contrastive learning

Contrastive learning is a widely adopted machine learning technique for acquiring discriminative feature representations by maximizing similarity among related samples and minimizing it among unrelated ones. This approach has driven significant strides in domains like computer vision (Dai & Lin, 2017; Saeed, Grangier, & Zeghidour, 2021) and natural language processing (Le-Khac, Healy, & Smeaton, 2020; Wang, Ding, Li, & Zheng, 2021). For instance, Chen, Kornblith, Norouzi, and Hinton (2020) proposed a framework that enhances image representation by maximizing similarity across different views of the same image. Devlin, Chang, Lee, and Toutanova (2018) applied contrastive learning in self-supervised language modeling, where predicting masked words improved contextual understanding. Furthermore, He, Fan, Wu, Xie, and Girshick (2020) introduced the concept of a "momentum encoder", which maintains feature consistency across training batches, achieving state-of-the-art performance in image representation learning. These contributions underscore the versatility of contrastive learning in refining feature representations across diverse fields.

In time series analysis, contrastive learning plays a crucial role, particularly in clustering tasks, by reinforcing intra-cluster similarity and maximizing inter-cluster differences. Hadsell, Chopra, and LeCun (2006) demonstrated this by mapping samples of the same category into a shared feature space while ensuring inter-class separation to enhance distinctiveness. Additionally, Eldele et al. (2021) integrated contrastive learning with distance metrics, leveraging temporal and contextual information to refine similarity measures, thereby improving clustering accuracy. At the same time, Meng, Qian, Liu, Cui, Xu, and Shen (2023) introduced the MHCCL method for multivariate time series, combining hierarchical clustering and masking mechanisms to strengthen feature representation and classification performance. Additionally, Hao, Wang, Alexander, Yuan, and Zhang (2023) proposed MICOS, which integrates supervised and contrastive learning to further improve multivariate time series classification. In summary, the application of contrastive learning in the field of time series clustering provides robust solutions to pattern recognition and structural discovery issues within sequential data.

3. Contrastive clustering learning

3.1. Problem statement

Given a set of time series $\mathcal{X} = \{x^{(i)}\}_{i=1}^{N}$ where each sequence $x^{(i)} \in \mathbb{R}^{M\times V}$ contains M timestamps and V variables, we explicitly distinguish temporal dimension (M) from variable dimension (V) to prevent notation conflict with sample size N. The encoder $f_q(\cdot) : \mathbb{R}^{M\times V} \to \mathbb{R}^{D\times 1}$ maps inputs to representation vectors:

$$z^{(i)} = f_q(x^{(i)}) \in \mathbb{R}^{D\times 1}.$$

Considering the unique characteristics of time series data, we draw on feature extractors used in Ircio et al. (2023) and Lee et al. (2022). However, as our primary focus lies elsewhere, we do not elaborate on feature extraction in detail. Instead, we emphasize the integration of contrastive clustering as a strategy to address class imbalance effectively.

3.2. Contrastive clustering loss function

Time-series data are particularly prone to class imbalance due to their complex temporal dynamics, which give rise to underrepresented patterns. Traditional clustering methods often fail to address this imbalance, resulting in biased learning that disproportionately favors majority patterns. To tackle this challenge, we introduce a Contrastive Clustering Loss (COCL), a unified approach that integrates contrastive learning with clustering objectives. COCL enhances intra-cluster similarity while simultaneously enforcing inter-cluster separability and balanced cluster assignments. The framework comprises three core components:

1. *Similarity Maximization Loss* ($\mathcal{L}_{\text{SM}}$), which optimizes feature space proximity among temporally related instances to reinforce intra-cluster cohesion.
2. *Dissimilarity Minimization Loss* ($\mathcal{L}_{\text{DM}}$), which penalizes ambiguous inter-cluster boundaries by suppressing overconfident intra-cluster assignments, thus preserving cluster distinctness.
3. *Cluster Loss* ($\mathcal{L}_{\text{CL}}$), which ensures equitable sample distribution across clusters to mitigate class imbalance and prevent the dominance of majority patterns.

The complete computation process of COCL is provided in Algorithm 1, followed by a comprehensive discussion of three key loss components in Sections 3.2.1 through 3.2.3.

Algorithm 1: Contrastive Clustering Loss: $\mathcal{L}_{\text{COCL}}$

Input : Time-Series Representations $\{z^{(i)}\}_{i=1}^{N}$, Temperature τ, (Optional) Label Matrix $\mathbf{M}$

Output: Total Contrastive Clustering Loss $\mathcal{L}_{\text{COCL}}$

1 **Compute Cosine Similarities**: $s_{ij} = \frac{z^{(i)}\cdot z^{(j)}}{\|z^{(i)}\|_2 \|z^{(j)}\|_2}$.

2 **Convert to Probabilities**:
$p_{ij} = \frac{\exp(s_{ij}/\tau)}{\sum_{k=1}^{N}\exp(s_{ik}/\tau)}$, $p_i = \frac{1}{N}\sum_j p_{ij}$, $p_j = \frac{1}{N}\sum_i p_{ij}$.

3 **Contrastive Losses**:
$\mathcal{L}_{\text{SM}} = -\frac{1}{N}\sum_i \Big[\log p_i + \log\big(p_j M_{ij} + (1 - M_{ij})\big)\Big]$,
$\mathcal{L}_{\text{DM}} = -\frac{1}{N}\sum_i \log\big(1 - p_i\big)$.

4 **Clustering Loss**: $D_{ij} = \| z^{(i)} - z^{(j)}\|_2$, $C_i = \arg\min_j D_{ij}$, $p(c) = \frac{|\{\, i | C_i = c\}|}{N}$, $\mathcal{L}_{\text{CL}} = -\sum_{c=1}^{K} p(c)\log p(c)$.

5 **Summation**: $\mathcal{L}_{\text{COCL}} = \mathcal{L}_{\text{SM}} + \mathcal{L}_{\text{DM}} + \mathcal{L}_{\text{CL}}$.

6 **return** $\mathcal{L}_{COCL}$

3.2.1. Similarity maximization loss

First, we compute the cosine similarity matrix of the representation vectors, denoted as s. Concretely, for each pair of representation vectors $(z^{(i)}, z^{(j)})$, the similarity is given by

$$s(z^{(i)}, z^{(j)}) = \frac{z^{(i)}\, z^{(j)}}{\|z^{(i)}\|_2 \, \|z^{(j)}\|_2}. \tag{1}$$

Here, $s(\cdot,\cdot)$ denotes the cosine similarity, and $\|z^{(i)}\|_2$ represents the L_2 norm of $z^{(i)}$. Cosine similarity is chosen for its scale invariance, ensuring robust clustering across different data distributions. Using this similarity matrix s, we derive the similarity probability p_{ij}:

$$p_{ij} = \frac{\exp\big(s(z^{(i)}, z^{(j)})/\tau\big)}{\sum_{k=1}^{N} \exp\big(s(z^{(i)}, z^{(k)})/\tau\big)}, \tag{2}$$

where τ is the temperature parameter controlling the scaling of the similarity. By adjusting τ, we can flexibly explore different intensities of similarity, enabling the model to better adapt to various data distributions and tasks.

Subsequently, we calculate the average intra-class probability p_i and the average inter-class probability p_j. Specifically,

$$p_i = \frac{1}{N}\sum_{j=1}^{N} p_{ij}, \quad p_j = \frac{1}{N}\sum_{i=1}^{N} p_{ij}.$$

These probabilities play a pivotal role in constructing the similarity loss term. The similarity maximization loss is defined as:

$$\mathcal{L}_{\mathrm{SM}} = -\frac{1}{N}\sum_{i=1}^{N}\Big[\log p_i + \log\big(p_j \cdot \mathbf{M}_{ij} + (1 - \mathbf{M}_{ij})\big)\Big], \tag{3}$$

where $\mathbf{M}$ is a binary label matrix. If $x^{(i)}$ and $x^{(j)}$ share the same cluster label, then $\mathbf{M}_{ij} = 1$; otherwise $\mathbf{M}_{ij} = 0$. In this manner, $\mathbf{M}$ selectively emphasizes pairs of samples from the same category. Maximizing $\mathcal{L}_{\mathrm{SM}}$ thus encourages clustering of similar time series. Furthermore, the structure of this loss term accounts for data imbalance, ensuring that underrepresented categories are not disregarded during training.

3.2.2. Dissimilarity minimization loss

To promote clear separation between clusters, we introduce the dissimilarity minimization loss, denoted as $\mathcal{L}_{\mathrm{DM}}$. We start by taking the negative logarithm of $(1 - p_i)$, revealing the likelihood that sample $x^{(i)}$ lies outside its designated cluster. Formally,

$$\mathcal{L}_{\mathrm{DM}} = -\frac{1}{N}\sum_{i=1}^{N}\log\big(1 - p_i\big). \tag{4}$$

Intuitively, $\mathcal{L}_{\mathrm{DM}}$ penalizes scenarios in which p_i remains overly high, indicating that $x^{(i)}$ is consistently grouped into a particular cluster with undue confidence. Consequently, $\mathcal{L}_{\mathrm{DM}}$ strategically *discourages* erroneously grouping instances that should be distinct, thus favoring well-separated clusters.

In essence, while $\mathcal{L}_{\mathrm{SM}}$ encourages similarity among instances of the same category, $\mathcal{L}_{\mathrm{DM}}$ dissuades the model from collapsing or conflating different categories. The interplay of these two terms provides a robust way to handle intraclass similarity and interclass dissimilarity concurrently, guiding the model to learn more discriminative representations.

3.2.3. Cluster loss

In traditional clustering tasks, different clusters may contain varying numbers of samples. This imbalance often causes the model to focus excessively on clusters containing more samples, neglecting clusters with fewer samples. To address this, we introduce a *cluster loss* $\mathcal{L}_{\mathrm{CL}}$ designed to specifically handle class imbalance in time-series data.

First, we compute the Euclidean distance matrix $D_{ij} = \|z^{(i)} - z^{(j)}\|_2$, where $\|\cdot\|_2$ denotes the L_2 norm. This matrix captures pairwise distances between normalized vectors, indicating spatial relationships among samples in representation space. Next, for each sample, we select the index corresponding to the minimum distance in D_{ij}, thereby associating each sample with its nearest cluster center. This ensures that each sample is linked to the closest cluster, forming a preliminary clustering scheme.

After establishing these links, we count the number of samples that fall into each cluster and convert these counts into probabilities, yielding a distribution $p(c)$:

$$p(c) = \frac{\big|\{\, i \mid C_i = c \,\}\big|}{N}, \tag{5}$$

where c ranges over all cluster centers. This definition makes explicit that $p(c)$ is the fraction of samples assigned to cluster c.

Finally, the cluster loss is expressed as

$$\mathcal{L}_{\mathrm{CL}} = -\sum_{c=1}^{N} p(c)\log\big(p(c)\big) \quad (K \le N), \tag{6}$$

which is the negative sum of each cluster's probability weighted by its own logarithm. By minimizing this term, we seek to avoid extreme concentration of samples in only a few clusters. This loss term is especially valuable for time-series data, where some patterns may be inherently underrepresented. By highlighting the distribution $p(c)$, $\mathcal{L}_{\mathrm{CL}}$ helps the model to more equitably distribute attention across all clusters.

Combining the three components above yields the final expression for the proposed COCL:

$$\mathcal{L}_{\mathrm{COCL}} = \mathcal{L}_{\mathrm{SM}} + \mathcal{L}_{\mathrm{DM}} + \mathcal{L}_{\mathrm{CL}}, \tag{7}$$

where each loss component is carefully designed to ensure fair cluster assignment and to address intricate patterns and imbalance in time-series data.

3.3. Cross-entropy loss

In addition to the contrastive clustering scheme, we introduce a *cross-entropy loss* $\mathcal{L}_{\mathrm{CE}}$ to incorporate class label supervision:

$$\mathcal{L}_{\mathrm{CE}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_c^{(i)} \log\Big(\hat{y}_c^{(i)}\Big), \tag{8}$$

where N is the total number of samples, C is the number of classes, $y_c^{(i)}$ is the one-hot true label for sample i in class c, and $\hat{y}_c^{(i)}$ is the model-predicted probability that sample i belongs to class c. By minimizing $\mathcal{L}_{\mathrm{CE}}$, the model is encouraged to learn more accurate class predictions, thereby solidifying its performance on downstream classification tasks.

3.4. Total loss

Ultimately, the overall multi-level contrastive loss is obtained by summing the contrastive clustering loss (7) and the cross-entropy loss (8), formally defined as:

$$\mathcal{L} = \alpha\mathcal{L}_{\mathrm{COCL}} + \mathcal{L}_{\mathrm{CE}}, \tag{9}$$

where $\alpha > 0$ is a scaling factor that balances the importance of the unsupervised contrastive component relative to the supervised term. To optimize the model with the constructed loss function, the following procedure is implemented:

1. **Initialization:** Initialize all model parameters and set the scaling factor α.
2. **Forward Pass:** For each batch of time-series data, perform a forward pass to obtain the representation vectors z and the predicted outputs $\hat{y}$.
3. **Loss Calculation:** Compute the similarity maximization loss $\mathcal{L}_{\mathrm{SM}}$, dissimilarity minimization loss $\mathcal{L}_{\mathrm{DM}}$, clustering loss $\mathcal{L}_{\mathrm{CL}}$, and cross-entropy loss $\mathcal{L}_{\mathrm{CE}}$. These loss components are then aggregated as follows:

$$\mathcal{L}_{\mathrm{COCL}} = \mathcal{L}_{\mathrm{SM}} + \mathcal{L}_{\mathrm{DM}} + \mathcal{L}_{\mathrm{CL}}, \quad \mathcal{L} = \alpha\,\mathcal{L}_{\mathrm{COCL}} + \mathcal{L}_{\mathrm{CE}}.$$

Table 1
Description of imbalanced multimodal time-series data sets.

Name	Minority class samples	Majority class samples	Imbalance ratio	Time-series length	Number of classes
FiftyWords_[a]	92–200	705–813	3.52–8.83	270	50
Adiac_[a]	77–88	693–704	7.87–9.14	176	37
SwedishLeaf_[a]	150	975	6.50	128	15
FaceAll_[a]	345–351	1799–1905	3.98–5.52	131	13
OSULeaf_[a]	50–172	270–292	1.56–1.94	427	6
SyntheticControl_[a]	200	400	2.00	60	6
Wafer	97	903	9.30	152	2

[a] Represents the index of the original class converted to the minority class.

4. **Backpropagation:** The selected time-series classification methods (one-nearest neighbor (1NN) algorithm (Keogh & Kasetty, 2002), the 1NN with dynamic time warping distance (1NN-DTW) (Xi, Keogh, Shelton, Wei, & Ratanamahatana, 2006), the support vector machine with radial basis function (SVM) (Yang, Li, & Yang, 2015), MLSTM-FCN and Resnet) are trained using the standard backpropagation algorithm. During the forward pass, PyTorch dynamically constructs the computation graph, whereas during backpropagation, the chain rule is applied the chain rule to compute gradients for all trainable parameters (e.g., convolutional kernels, LSTM gating parameters, batch normalization parameters, and fully connected layer weights).
5. **Parameter Update:** Update the model parameters using an optimizer (e.g., SGD or Adam). Adam adaptively adjusts the learning rate based on estimates of the first and second moments of the gradients (Geng & Luo, 2019), whereas SGD updates parameters using mini-batch stochastic gradient descent (Zheng, Liu, Chen, Ge, & Zhao, 2014). To ensure a fair comparison, all models in this paper are trained using Adam.

Throughout the training process, the COCL loss function balances similarity and dissimilarity constraints, facilitating the model to discriminate between instances that belong to the same cluster or different clusters. Moreover, the clustering loss ensures that underrepresented clusters are not neglected, thereby alleviating data imbalance issues. By incorporating the cross-entropy loss, the model benefits from additional class label supervision, ultimately enhancing both representation quality and classification accuracy. Consequently, the model exhibits robust capability in learning effective representations from complex time-series data while adeptly handling class imbalance and achieving superior classification performance.

4. Experiments

The research methodology and experimental objectives, benchmark datasets, evaluation metrics, and all neural network classifiers employed throughout the experiments are presented in this section. All experiments are implemented in PyTorch and conducted on a system with an Intel Core i9-13900H @2.30 GHz CPU, 16 GB RAM, NVIDIA GeForce RTX 4070 GPU, and Windows 11 operating system.

4.1. Experimental objectives

The primary objective of this study is to evaluate the performance of the proposed model against state-of-the-art methods, particularly in addressing the challenges of imbalanced multimodal binary classification. Additionally, the second objective is to assess the effectiveness of the proposed loss function in comparison to other advanced techniques designed for handling class imbalance. Notably, the proposed approach is independent of the specific neural network classifier employed and is designed to accommodate the dynamic characteristics of time-series data. To ensure fairness and consistency, the experimental setup adheres to the time-series neural network classifiers and feature extractors referenced in Ircio et al. (2023) and Lee et al. (2022).

Table 2
Description of imbalanced multimodal multiclass time-series data sets.

Data sets	Train size	Test size	Ndim	Length	Nclass
ChlorineCon.	467	3840	1	166	3
Dist.Phal.Out.AGr.	400	139	1	80	3
Dist.Phal.TW	400	139	1	80	6
Earthquakes	322	139	1	512	2
ECG200	112	112	1	96	2
ECG5000	500	4500	1	140	5
ElectricDevices	8926	7711	1	96	7
FiftyWords	450	455	1	270	50
Haptics	155	308	1	1092	5
MedicalImages	381	760	1	99	10
Mid.Phal.Out.AGr.	400	154	1	80	3
Mid.Phal.TW	399	154	1	80	6
OSULeaf	200	242	1	427	6
Prox.Phal.Outl.AGr.	400	205	1	80	3
Prox.Phal.TW	400	205	1	80	6
Worms	181	77	1	900	5
Wafer	1008	6176	1	152	2
Phal.Outl.Co.	1800	858	1	80	2
Prox.Phal.Out.Co.	600	291	1	80	2
HD-1%	3715	1162	12	200	2
HD-3%	3792	1185	12	200	2
HD-5%	3891	1217	12	200	2

4.2. Datasets and research methodology

In our preliminary experiments, we implemented the methodology outlined in Lee et al. (2022), focusing on binary classification challenges in imbalanced multimodal situations. We utilized 28 imbalanced datasets from the UCR time series repository (Keogh et al., 2018) (as depicted in Table 1), converting multi-class datasets into binary classifications of minority and majority. While the COCL loss function is primarily optimized for binary classification, it also exhibited robust performance in multi-class scenarios. We evaluated COCL's efficacy against various oversampling techniques, including the recent schemes MHCCL (Meng et al., 2023) and MICOS (Hao et al., 2023), and different classifiers. The GAN-based methods (cGAN, CatGAN, BFGAN) adhered to the DCGAN framework (Radford, Metz, & Chintala, 2015). The assessment employed a five-fold cross-validation process, using the average area under the curve (AUC) as the evaluation metric.

In a second set of experiments, we employed the UEA & UCR time series classification repository (Bagnall, Lines, Vickers, & Keogh, 2018) and three additional Backblaze datasets (as shown in Table 2). We replicated the method from Ircio et al. (2023) using top-performing classifiers: MLSTM-FCN (MLS) (Karim et al., 2017) and Resnet (RES) (Wang et al., 2017). The classifiers were trained using the Adam optimizer (Zhang, 2018). Considering the stochastic nature of the MLS and RES classifiers, each method was run 10 times for each dataset, and results were measured using the average and standard deviation of both accuracy (acc) and minimum recall ($minRec$). While accuracy is generally not considered ideal for imbalanced scenarios, it was included to control overall classifier performance and ensure majority class classification was not unduly compromised. Conversely, minimum recall directly measures the performance of minority classes, which is crucial

Table 3
The comparison of AUC among various oversampling methods and the proposed COCL on multimodal minority classes using 1NN classifier.

Data sets	Baseline	ROS	SMOTE	BL-SMOTE	ADASYN	INOS	cGAN	CatGAN	BFGAN	COCL	MICOS	MHCCL
FiftyWords_1_2	0.909	0.909	0.923	0.915	0.916	0.925	0.926	0.910	0.933	**0.982**	0.954	0.944
FiftyWords_2_3	0.942	0.942	0.932	0.943	0.940	0.938	0.936	0.936	0.949	0.982	0.925	**0.985**
FiftyWords_3_4	0.833	0.833	0.837	0.864	0.854	0.866	0.864	0.839	0.881	**0.987**	0.943	0.925
FiftyWords_4_5	0.791	0.791	0.789	0.874	0.866	0.868	0.867	0.834	0.880	0.967	0.942	**0.992**
FiftyWords_5_1	0.888	0.888	0.897	0.922	0.914	0.927	0.924	0.884	0.941	0.976	0.949	**0.994**
Adiac_1_2_3_4	0.895	0.895	0.890	0.901	0.909	0.911	0.908	0.909	0.914	**0.976**	0.885	0.843
Adiac_4_5_6_7	0.734	0.734	0.809	0.812	0.836	0.836	0.833	0.820	0.774	**0.939**	0.855	0.855
Adiac_7_8_9_10	0.796	0.796	0.833	0.861	0.865	0.860	0.857	0.832	0.850	**0.949**	0.632	0.826
Adiac_10_11_12_13	0.898	0.898	0.908	0.916	0.935	0.933	0.929	0.931	0.913	**0.977**	0.811	0.836
Adiac_13_14_15_16	0.816	0.816	0.817	0.863	0.868	0.864	0.862	0.857	0.861	**0.969**	0.847	0.926
Adiac_16_17_18_19	0.785	0.785	0.814	0.850	0.846	0.847	0.853	0.823	0.834	**0.949**	0.833	0.929
FaceAll_2_3	0.905	0.906	0.884	0.877	0.880	0.877	0.874	0.869	**0.920**	0.807	0.854	0.867
FaceAll_3_4	0.873	0.873	0.886	0.877	0.874	0.872	0.873	0.878	**0.916**	0.820	0.858	0.731
FaceAll_4_5	0.868	0.868	0.880	0.896	0.897	0.899	0.899	0.889	0.906	**0.914**	0.902	0.893
FaceAll_5_8	0.800	0.800	0.809	0.818	0.839	0.843	0.840	0.815	0.851	**0.961**	0.909	0.857
FaceAll_8_2	0.801	0.801	0.809	0.819	0.828	0.823	0.823	0.827	0.866	0.889	0.910	**0.953**
OSULeaf_1_2	0.720	0.698	0.733	0.747	0.748	0.749	0.747	0.747	0.750	**0.980**	0.943	0.970
OSULeaf_3_4	0.681	0.682	0.678	0.668	0.681	0.673	0.669	0.679	0.692	0.985	0.979	**0.991**
SwedishLeaf_1_2	0.882	0.882	0.892	0.903	0.920	0.921	0.921	0.901	0.903	**0.992**	0.976	0.960
SwedishLeaf_2_3	0.915	0.914	0.920	0.955	0.952	0.949	0.950	0.948	0.942	**0.974**	0.972	0.833
SwedishLeaf_3_4	0.861	0.863	0.903	0.930	0.938	0.932	0.929	0.867	0.877	0.980	**0.986**	0.829
SwedishLeaf_4_5	0.875	0.875	0.933	0.928	0.932	0.932	0.928	0.923	0.940	**0.972**	0.947	0.965
SwedishLeaf_5_1	0.837	0.837	0.860	0.872	0.885	0.891	0.891	0.858	0.886	**0.982**	0.978	0.962
SyntheticControl_1_2	0.880	0.888	0.970	0.976	0.976	**0.981**	0.979	0.940	0.970	0.950	0.959	0.946
SyntheticControl_2_3	0.958	**0.959**	0.882	0.946	0.943	0.942	0.939	0.928	0.948	0.903	0.869	0.849
SyntheticControl_3_4	0.955	**0.959**	0.842	0.936	0.937	0.934	0.928	0.928	0.957	0.830	0.813	0.853
SyntheticControl_4_5	0.945	0.949	0.852	0.928	0.932	0.933	0.930	0.935	**0.950**	0.870	0.845	0.866
Wafer	0.992	0.992	0.992	0.992	0.994	0.991	0.992	0.994	0.994	**0.999**	**0.999**	0.998
Average	0.858	0.858	0.863	0.885	0.889	0.890	0.888	0.875	0.892	**0.945**	0.903	0.906
Average rank	9.04	9.02	8.70	6.45	5.41	5.59	6.36	7.91	4.82	**3.20**	5.86	5.57

in imbalanced contexts. To compare the performance of different methods over multiple datasets, we analyzed significant differences in terms of both metrics. The Wilcoxon signed-rank test with the Shaffer correction (or Shaffer dynamic correction for fewer than 10 methods) was applied at a 0.05 significance level. The final results were graphically presented, emphasizing classifiers without significant differences with bold lines. This dual-metric approach provides a balanced assessment of classifier performance in imbalanced scenarios.

5. Results and analysis

5.1. Comparative experiment for imbalanced multimodal binary classification

5.1.1. Comparative experiment with oversampling method

The comparative evaluation of AUC metrics for various oversampling methodologies, including COCL, MHCCL (Meng et al., 2023), and MICOS (Hao et al., 2023), on minority classes within a multimodal context using a 1NN classifier, is detailed in Table 3. This table encapsulates AUC values for each oversampling technique across diverse datasets, offering a comprehensive assessment of their efficacy. COCL consistently exhibits superior performance, achieving the highest AUC values across multiple datasets, such as FiftyWords_3_4 with an AUC of 0.987, Adiac_10_11_12_13 with an AUC of 0.977, SwedishLeaf_1_2 with an AUC of 0.992, and Wafer with an AUC of 0.999. The mean AUC for COCL is 0.945, surpassing the baseline at 0.858, and other oversampling techniques, including MICOS at 0.903 and MHCCL at 0.906. While MICOS and MHCCL also demonstrate competitive performance, with MHCCL achieving the highest AUC values in datasets such as FiftyWords_2_3 with an AUC of 0.985 and FiftyWords_4_5 with an AUC of 0.992, COCL maintains overall superiority. This is further evidenced by its lowest mean rank of 3.20, compared to MICOS at 5.86 and MHCCL at 5.57. The average rank analysis reinforces COCL's preeminence, establishing it as the top-performing approach among the evaluated methodologies. Additional comparative experimental results for COCL and different oversampling methods using KNN-DTW and SVM classifiers are provided in the A.

5.1.2. Comparative experiment with undersampling method

Table 4 presents the comparison of AUC among various oversampling methods and the proposed COCL on multimodal minority classes using a 1NN classifier. The highest classification AUC accuracy for each dataset is highlighted in bold, and the last two rows of the table provide the average accuracy and average ranking across all datasets. The proposed COCL method achieves an average accuracy ranking of 3.20 across all datasets, with an average count of 14 for obtaining the highest accuracy across the datasets (a total of 28 datasets are used in the experiments). COCL consistently demonstrates higher AUC scores compared to other oversampling methods, including BFGAN, MICOS, and MHCCL. COCL shows notable effectiveness on datasets such as FaceAll_5_8, FaceAll_8_2, SwedishLeaf_1_2, SwedishLeaf_3_4, SwedishLeaf_4_5, SwedishLeaf_5_1, and Wafer, as well as in all binary classification scenarios for the FiftyWords, OSULeaf, and Adiac datasets. While MHCCL (Meng et al., 2023) and MICOS (Hao et al., 2023) exhibit competitive results, COCL consistently ranks higher in terms of average accuracy and the count of the highest AUC scores. The experimental results demonstrate that the proposed COCL method not only surpasses traditional oversampling techniques but also outperforms more recent methods like BFGAN (Lee et al., 2022), MHCCL (Meng et al., 2023), and MICOS (Hao et al., 2023). This highlights the robustness and generalizability of COCL in handling multimodal minority class problems across a wide range of datasets. Similarly, the comparative experimental results among COCL and various undersampling methods in KNN-DTW and SVM classifiers are presented in the A.

5.2. t-SNE visualization results analysis in binary classification tasks

Fig. 1(a)–(h) offer a comprehensive series of t-SNE visualizations, employed as an analytical tool for quantitatively evaluating class distinctions within binary classification paradigms and for assessing the efficacy of the COCL loss function. In the context of the OSULeaf_3_4 dataset, Fig. 1(a) initially exhibits a blend of class distributions, with instances of the minority class interspaced within the domain of the

Table 4
The comparison of AUC among various undersampling methods and the proposed COCL on multimodal minority classes using 1NN classifier.

Data sets	Baseline	RUS	OSS	ENN	T-Link	BFGAN	COCL	MICOS	MHCCL
FiftyWords_1_2	0.909	0.916	0.919	0.917	0.919	0.933	**0.982**	0.954	0.944
FiftyWords_2_3	0.942	0.906	0.942	0.952	0.942	0.949	**0.982**	0.925	**0.985**
FiftyWords_3_4	0.833	0.853	0.840	0.886	0.840	0.881	**0.987**	0.943	0.925
FiftyWords_4_5	0.791	0.787	0.822	0.855	0.822	0.880	**0.967**	0.942	**0.992**
FiftyWords_5_1	0.888	0.869	0.911	0.905	0.911	0.941	**0.976**	0.949	**0.994**
Adiac_1_2_3_4	0.895	0.816	0.895	0.921	0.895	0.914	**0.976**	0.885	0.843
Adiac_4_5_6_7	0.734	0.761	0.747	0.753	0.747	0.774	**0.939**	0.855	0.855
Adiac_7_8_9_10	0.796	0.761	0.806	0.821	0.808	0.850	**0.949**	0.632	0.826
Adiac_10_11_12_13	0.898	0.908	0.897	0.914	0.897	0.913	**0.977**	0.811	0.836
Adiac_13_14_15_16	0.816	0.825	0.890	0.875	0.892	0.861	**0.969**	0.847	0.926
Adiac_16_17_18_19	0.785	0.845	0.836	0.852	0.835	0.834	**0.949**	0.833	0.929
FaceAll_2_3	0.905	0.817	0.903	0.891	0.904	**0.920**	0.807	0.854	0.867
FaceAll_3_4	0.873	0.911	0.977	0.961	**0.978**	0.916	0.820	0.858	0.731
FaceAll_4_5	0.868	**0.917**	0.874	0.904	0.873	0.906	0.914	0.902	0.893
FaceAll_5_8	0.800	0.825	0.806	0.811	0.805	0.851	**0.961**	0.909	0.857
FaceAll_8_2	0.801	0.795	0.802	0.798	0.801	0.866	**0.889**	0.910	**0.953**
OSULeaf_1_2	0.720	0.682	0.698	0.715	0.690	0.750	**0.980**	0.943	0.970
OSULeaf_3_4	0.681	0.662	0.674	0.608	0.680	0.692	**0.985**	0.979	0.991
SwedishLeaf_1_2	0.882	0.922	0.894	0.915	0.887	0.903	**0.992**	0.976	0.960
SwedishLeaf_2_3	0.915	0.870	0.893	0.909	0.892	0.942	**0.974**	0.972	0.833
SwedishLeaf_3_4	0.861	0.922	0.884	0.897	0.867	0.877	**0.980**	**0.986**	0.829
SwedishLeaf_4_5	0.875	0.888	0.870	0.879	0.875	0.940	**0.972**	0.947	0.965
SwedishLeaf_5_1	0.837	0.830	0.848	0.872	0.842	0.886	**0.982**	**0.978**	0.962
SyntheticControl_1_2	0.880	0.920	0.700	0.900	0.900	**0.970**	0.950	0.959	0.946
SyntheticControl_2_3	**0.958**	0.930	0.945	0.948	**0.958**	0.948	0.903	0.869	0.849
SyntheticControl_3_4	0.955	0.895	**0.955**	0.923	**0.955**	0.957	0.830	0.813	0.853
SyntheticControl_4_5	0.945	0.920	**0.948**	0.938	**0.950**	**0.950**	0.870	0.845	0.866
Wafer	**0.992**	0.989	0.991	0.990	0.990	0.994	**0.999**	**0.999**	0.998
Average	0.858	0.855	0.863	0.875	0.870	0.893	**0.945**	0.903	0.906
Average rank	6.59	6.61	5.82	5.05	5.75	3.71	**2.59**	4.61	4.27

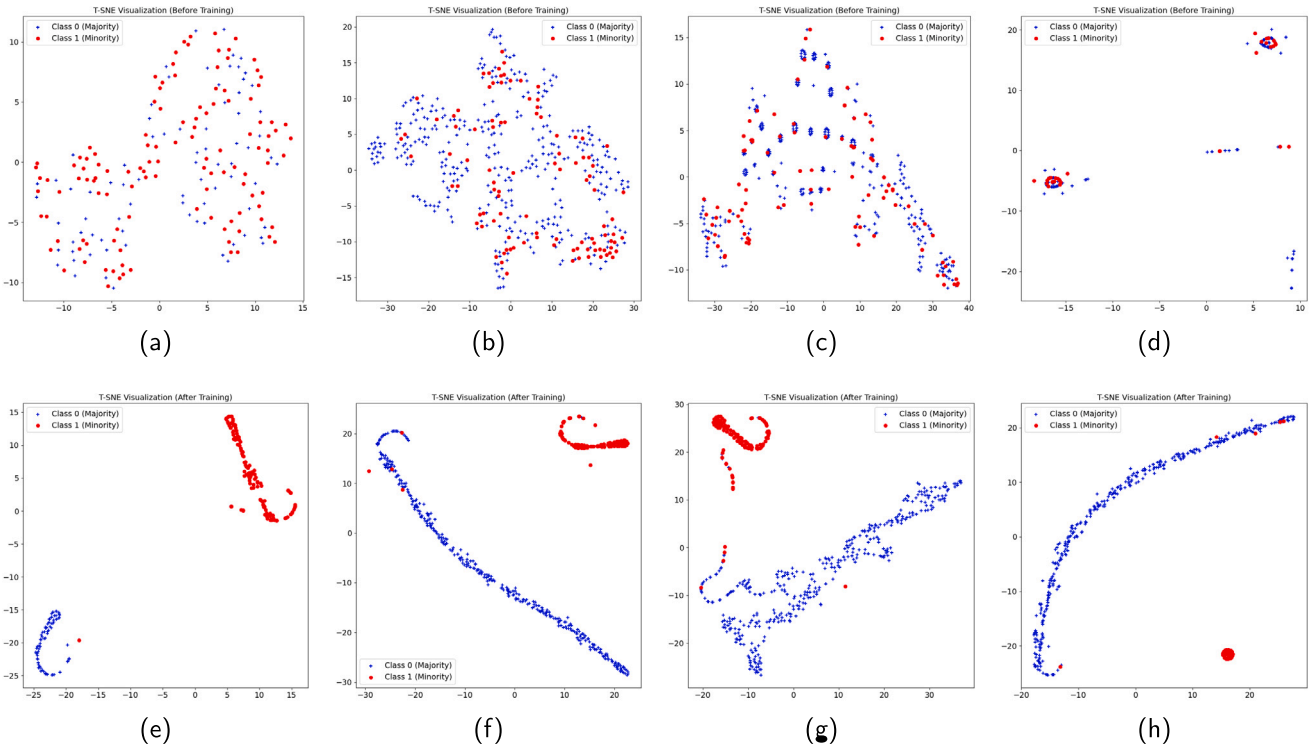


Fig. 1. t-SNE visualizations depicting class distribution and separability in binary classification tasks across four datasets: Adiac_1_2_3_4, FiftyWords_1_2, OSULeaf_3_4, and SwedishLeaf_2_3. (a)–(d) present the data prior to training, while (e)–(h) display the post-training results using the COCL loss function.

majority class. This configuration underscores the initial impediment in achieving clear demarcation of class boundaries. Subsequent to the application of the COCL loss function, Fig. 1(e) reveals a notable progression towards enhanced class separation, indicative of an augmentation in the discriminative capacity of the feature space, a development attributable to the COCL loss function. This enhanced delineation of class boundaries post-COCL training is similarly mirrored in the FiftyWords_1_2 dataset. Here, the preliminary intermingling of classes illustrated in Fig. 1(b) transitions to more pronounced segregation in Fig. 1(f), emphasizing the COCL loss function's substantial

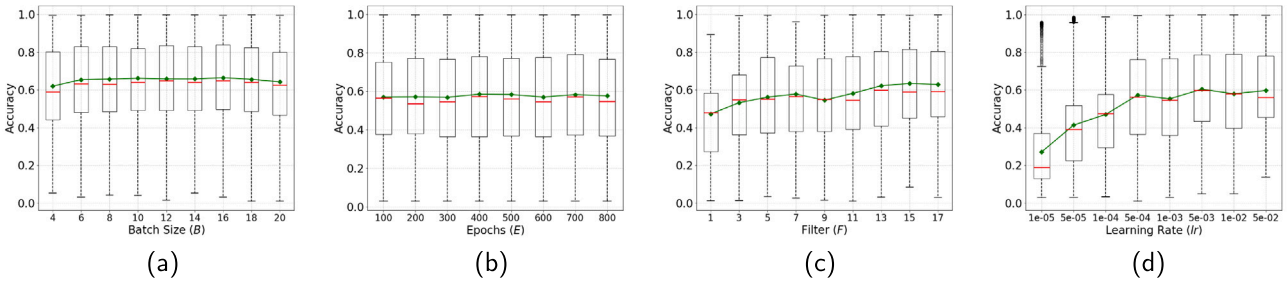


Fig. 2. Hyperparameter stability analysis: (a) Batch size (B), (b) Epochs (E), (c) Filters (F), (d) Learning Rate (lr). The green line represents the average accuracy across the 10 datasets.

influence on the model's class boundary demarcation capabilities. This transformative effect is also observed in the SwedishLeaf_2_3 dataset, where initial substantial class overlap in Fig. 1(c) evolves into a well-defined clustering of classes in Fig. 1(g), thus exemplifying the COCL loss function's proficiency in enhancing the classification landscape. Additionally, the Adiac_1_2_3_4 dataset consolidates this trend, exhibiting a transition from significant class intermixture in Fig. 1(d) to distinct class segregation in Fig. 1(h). This change substantiates the COCL loss function's capacity to foster discriminative clustering. Collectively, these visual representations not only elucidate the pivotal role played by the COCL loss function in augmenting class separability within contexts of imbalanced classification but also underscore its vital contribution to the recalibration of the feature space, consequently amplifying the overall accuracy and efficacy of the model.

5.3. Hyperparameter exploration and impact analysis

To investigate the stability of hyperparameters, we systematically varied the batch size (B), learning rate (lr), number of epochs (E), and number of filters (F). All experiments employed cross-validation on 10 randomly selected datasets: Worms, Mid.Phal.TW, Wafer, ECG200, Prox.Phal.Out.AGr., ChlorineCon., Dist.Phal.TW, OSULeaf, FiftyWords, and Prox.Phal.TW. Fig. 2 presents standard box plots illustrating classification accuracies across these datasets for varying values of B, E, F, and lr. The green curve in each figure represents the mean accuracy. Fig. 2(a) depicts the impact of different batch sizes on model accuracy, showing that as the batch size increased from 4 to 20, the median accuracy remained around 0.6, though individual runs exhibited substantial variability. Fig. 2(b) shows the effect of the number of epochs, indicating that the median accuracy stabilized around 0.6 as epochs increased from 100 to 800, despite significant variability in individual runs. Fig. 2(c) demonstrates the influence of the number of filters, showing that median accuracy improved with an increasing number of filters, peaking between 13 and 15 filters before slightly declining. Smaller filter numbers were associated with greater variability, which decreased as the number of filters increased, indicating more consistent performance with a higher number of filters. Fig. 2(d) illustrates the effect of different learning rates on accuracy, showing that median accuracy improved as the learning rate increased from 1×10^{-5} to 1×10^{-3} before stabilizing. Lower learning rates exhibited higher variability, with some runs achieving very high accuracy but less consistency. To achieve optimal model performance, the selected hyperparameter values are $B = 8$, $lr = 0.0005$, $E = 500$, and $F = 7$.

5.4. Computational efficiency analysis

Table 5 presents the average total training time and per-epoch training time for each method in the binary classification task under the imbalanced multimodal setting, as outlined in the preliminary experiments of Section 4.2. The results are ordered in ascending training time. To ensure a fair comparison, all methods were initialized with identical parameter settings. The results indicate that MICOS (130.32 s) and RUS (136.30 s) exhibit the shortest total training times, with per-epoch durations of 0.29 s and 0.36 s, respectively. The proposed COCL completes training in 138.03 s (0.38 s per epoch), surpassing CatGAN (150.37 s), SMOTE, ROS, and the computationally intensive MHCCL (713.28 s).

Table 5
The comparison of average total training time and per-epoch average time among various methods and the proposed COCL.

Model	Average total training time (s)	Average per-epoch time (s)
MICOS	130.32	0.29 ± 0.12
RUS	136.30	0.36 ± 0.32
COCL(Ours)	138.03	0.38 ± 0.10
CatGAN	150.37	0.50 ± 0.31
SMOTE	157.24	0.57 ± 0.08
ROS	158.56	0.58 ± 0.20
Baseline	158.97	0.59 ± 0.32
ADASYN	159.45	0.59 ± 0.25
BL-SMOTE	163.88	0.64 ± 0.31
BFGAN	165.03	0.65 ± 0.32
T-Link	166.64	0.66 ± 0.51
cGAN	175.22	0.77 ± 0.34
OSS	181.87	0.82 ± 0.24
ENN	207.78	1.08 ± 0.15
INOS	221.06	1.21 ± 0.48
MHCCL	713.28	5.50 ± 2.72

Further analysis based on AUC and the average ranking results presented in Tables 3 and 4 reveals that COCL incurs only a 5.92% increase in computational cost relative to MICOS while achieving a 4.65% improvement in AUC. Compared to RUS, COCL improves AUC by 6.92% with only a 1.27% increase in computational cost, highlighting its effectiveness in balancing computational efficiency and classification performance.

5.5. Ablation study

Furthermore, to further validate the effectiveness of the contrastive clustering loss function, we conduct ablation experiments on the 28 imbalanced datasets from the UCR (University of California at Riverside) repository (Keogh et al., 2018), and the experimental results are presented in Table 6.

As shown in Table 6, the classification accuracy with COCL achieved the highest across all 28 datasets. Across the spectrum of considered datasets, the incorporation of the COCL loss function consistently yields heightened AUC values. Particularly noteworthy instances of improvement are observed in datasets such as FiftyWords_1_2, FiftyWords_2_3, and FiftyWords_5_1, where the introduction of COCL results in substantial enhancements, manifesting as AUC values of 0.989, 0.989, and 0.988, respectively, in contrast to their counterparts lacking the COCL loss function. Analogous trends are discerned in datasets including FaceAll_5_8, FaceAll_8_2, OSULeaf_1_2, and OSULeaf_3_4, where the

Table 6
The ablation experiment of COCL in multi-modal minority classes.

Datasets	w/ $\mathcal{L}_{\text{COCL}}$	w/o $\mathcal{L}_{\text{COCL}}$	Datasets	w/ $\mathcal{L}_{\text{COCL}}$	w/o $\mathcal{L}_{\text{COCL}}$
FitfyWords_1_2	**0.989**	0.987	FaceAll_5_8	**0.988**	0.911
FitfyWords_2_3	**0.989**	0.963	FaceAll_8_2	**0.966**	0.891
FitfyWords_3_4	**0.967**	0.961	OSULeaf_1_2	**0.968**	0.916
FitfyWords_4_5	**0.973**	0.925	OSULeaf_3_4	**0.992**	0.988
FitfyWords_5_1	**0.988**	0.987	SwedishLeaf_1_2	**0.974**	0.950
Adiac_1_2_3_4	**0.953**	0.934	SwedishLeaf_2_3	**0.980**	0.978
Adiac_4_5_6_7	**0.901**	0.873	SwedishLeaf_3_4	**0.981**	0.976
Adiac_7_8_9_10	**0.899**	0.881	SwedishLeaf_4_5	**0.976**	0.974
Adiac_10_11_12_13	**0.965**	0.926	SwedishLeaf_5_1	**0.985**	0.969
Adiac_13_14_15_16	**0.938**	0.924	SyntheticControl_1_2	**0.945**	0.720
Adiac_16_17_18_19	**0.914**	0.893	SyntheticControl_2_3	**0.933**	0.863
FaceAll_2_3	**0.942**	0.788	SyntheticControl_3_4	**0.853**	0.823
FaceAll_3_4	**0.938**	0.915	SyntheticControl_4_5	**0.870**	0.843
FaceAll_4_5	**0.936**	0.915	Wafer	**0.999**	0.998

Table 7
Ablation results of cocl loss function components.

Ablation modules	Average accuracy	Average rank
$\mathcal{L}_{\text{COCL}}$	**0.945**	**1.00**
$\mathcal{L}_{\text{SM}} + \mathcal{L}_{\text{DM}}$	0.857	2.61
$\mathcal{L}_{\text{CL}}$	0.868	2.36

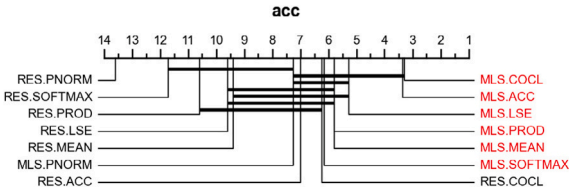


Fig. 3. Pairwise difference plot of *acc* with two classifiers (MLSTM-FCN (MLS) and Resnet (RES)) using the proposed COCL loss function versus other loss functions with a significance level of 0.05.

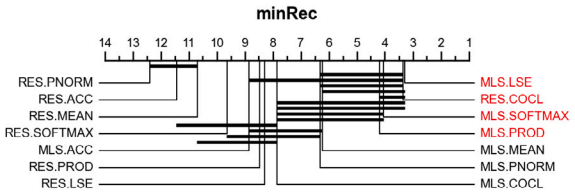


Fig. 4. Pairwise difference plot of *minRec* with two classifiers (MLSTM-FCN (MLS) and Resnet (RES)) using the proposed COCL loss function versus other loss functions with a significance level of 0.05.

inclusion of COCL consistently augments AUC values, thereby underscoring the efficacy of COCL in mitigating imbalances within multi-modal minority classes. The discernible impact of COCL is notably accentuated in datasets characterized by inherent complexity, such as SyntheticControl_1_2 and SyntheticControl_3_4, where the introduction of COCL significantly elevates AUC values.

To clarify the contributions of contrastive learning and clustering losses to COCL performance, we conducted ablation experiments with these configurations: $\mathcal{L}_{\text{COCL}}$ (combined contrastive and clustering loss), $\mathcal{L}_{\text{SM}} + \mathcal{L}_{\text{DM}}$ (separate contrastive and clustering losses), and $\mathcal{L}_{\text{CL}}$ (clustering loss only). The results in Table 7 indicate that $\mathcal{L}_{\text{COCL}}$ achieved the highest average accuracy of 0.945 and secured the first rank, demonstrating its superior performance by synergistically leveraging both losses. The $\mathcal{L}_{\text{SM}} + \mathcal{L}_{\text{DM}}$ module, which treats the losses separately, achieved an average accuracy of 0.857 and ranked 2.61, showing a decline in performance compared to $\mathcal{L}$COCL. Similarly, the $\mathcal{L}_{\text{CL}}$ module, which focuses solely on clustering loss, attained an average accuracy of 0.868 and ranked 2.36, further underscoring the benefits of the combined approach.

5.6. Comparative experiment on the performance of different minimum recall-based loss functions

The average values and standard deviations of *acc* and *minRec* for all datasets using the MLS classifier are listed in Tables 8 and 9. The tables highlight the loss functions that achieved optimal performance for both *acc* and *minRec* metrics for each dataset using distinctive colors. Additionally, paired statistical difference plots for *acc* and *minRec* measurements are presented in Figs. 3 and 4.

To validate our approach to ITSC issues, we employed the COCL loss function in training classifiers, aiming to achieve higher minimum recall values compared to accuracy maximization training. This was done without significantly compromising overall accuracy. We conducted training using the proposed loss function as well as traditional accuracy loss functions on MLS and RES classifiers across all benchmark datasets. To mitigate potential learning issues arising from nearly non-differentiable functions, we set the parameter for parameter approximation to $\alpha = 10$. Tables 8 and 9 present the mean and standard deviation of *acc* and *minRec* for all runs and datasets. The highlighted loss function yielding the best performance in both *acc* and *minRec* for each dataset is specified. Additionally, Figs. 3 and 4 illustrate charts of paired statistical differences obtained for *acc* and *minRec* metrics.

Firstly, focusing on *acc*, as shown in Tables 8 and 9, the COCL loss function achieved optimal results on 11/21 RES datasets and 16/21 MLS datasets, with particularly significant improvements in class-imbalanced scenarios. For RES, COCL increased *acc* by 21.4% in ChlorineCon. (from 0.56 to 0.68). and 60.0% in FiftyWords (from 0.25 to 0.40). In the MLS classifier, COCL attained an accuracy of 1.00 ± 0.00 on Wafer while maintaining $minRec = 0.92 \pm 0.02$, indicating near-perfect predictive performance. Notably, the differences in performance between MLS trained with LSE, MEAN, product, and softmax loss functions, as depicted in Fig. 3, were not statistically significant. Subsequently, regarding *minRec*, the COCL loss function stood out, as shown in Fig. 4, with 18 out of 21 optimal results for RES and 5 out of 21 for MLS. On average, COCL improved *minRec* for RES by 43.7% (from 0.12 to 0.55) and for MLS by 22.4% (from 0.31 to 0.53). For instance, COCL transformed a zero-recall scenario in Dist.Phal.TW (MLS) from 0.00 ± 0.00 to 0.30 ± 0.00.

These findings indicate that the proposed method aligns with our expectations. Specifically, classifiers trained with COCL loss functions for RES and MLS outperform accuracy maximization training on *minRec* for most datasets. Moreover, MLS classifiers trained with accuracy-centric losses yielded $minRec = 0$ in multiple datasets, implying an inability to correctly predict any test instances, at least for one category. In contrast, classifiers trained with the COCL loss function achieved higher *minRec* values without a significant decrease in accuracy, making more

Table 8
Comparison results of average and standard deviation of *acc* and *minRec* obtained by the proposed COCL loss function and other loss functions learning for resnet classifier.

Dataset	RESNET													
	ACC		MEAN		PROD		SOFTMAX		LSE		PNORM		COCL	
	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec
ChlorineCon.	0.56 ± 0.00	0.00 ± 0.00	0.25 ± 0.01	0.00 ± 0.00	0.35 ± 0.02	0.30 ± 0.03	0.33 ± 0.06	0.18 ± 0.11	0.34 ± 0.06	0.19 ± 0.07	0.23 ± 0.00	0.00 ± 0.00	**0.68 ± 0.00**	**0.93 ± 0.04**
Dist.Phal.Out.AGr.	0.68 ± 0.01	0.00 ± 0.00	**0.69 ± 0.01**	0.11 ± 0.24	0.67 ± 0.05	0.55 ± 0.06	0.67 ± 0.05	0.47 ± 0.13	0.68 ± 0.08	0.54 ± 0.14	0.27 ± 0.17	0.00 ± 0.00	0.64 ± 0.00	**0.98 ± 0.01**
Dist.Phal.TW	0.58 ± 0.00	0.00 ± 0.00	0.58 ± 0.00	0.00 ± 0.00	0.52 ± 0.09	0.00 ± 0.00	0.22 ± 0.13	0.00 ± 0.00	0.48 ± 0.10	0.00 ± 0.00	0.17 ± 0.09	0.00 ± 0.00	**0.63 ± 0.00**	**0.72 ± 0.11**
Earthquakes	**0.75 ± 0.00**	0.00 ± 0.00	0.60 ± 0.04	0.53 ± 0.07	0.61 ± 0.04	0.57 ± 0.08	0.57 ± 0.01	0.51 ± 0.01	0.57 ± 0.02	0.52 ± 0.05	0.00 ± 0.00	0.00 ± 0.00	0.71 ± 0.00	**0.98 ± 0.01**
ECG200	0.69 ± 0.01	0.37 ± 0.09	0.72 ± 0.01	0.59 ± 0.02	0.72 ± 0.01	0.60 ± 0.02	0.68 ± 0.01	0.57 ± 0.01	0.68 ± 0.01	0.57 ± 0.04	0.33 ± 0.35	0.27 ± 0.28	**0.88 ± 0.00**	**0.98 ± 0.01**
ECG5000	**0.90 ± 0.01**	0.00 ± 0.00	0.90 ± 0.00	0.00 ± 0.00	0.38 ± 0.25	0.00 ± 0.01	0.01 ± 0.00	0.00 ± 0.00	0.73 ± 0.08	0.00 ± 0.00	0.20 ± 0.25	0.00 ± 0.00	0.86 ± 0.02	**0.26 ± 0.06**
ElectricDevices	0.60 ± 0.01	0.00 ± 0.00	**0.62 ± 0.03**	0.00 ± 0.00	0.57 ± 0.01	0.09 ± 0.04	0.38 ± 0.08	0.10 ± 0.09	0.57 ± 0.04	**0.17 ± 0.09**	0.15 ± 0.07	0.00 ± 0.00	0.34 ± 0.00	0.00 ± 0.00
FiftyWords	0.25 ± 0.03	0.00 ± 0.00	0.25 ± 0.02	0.00 ± 0.00	0.03 ± 0.04	0.00 ± 0.00	0.28 ± 0.01	0.00 ± 0.00	0.34 ± 0.02	0.00 ± 0.00	0.02 ± 0.04	0.00 ± 0.00	**0.40 ± 0.00**	**0.01 ± 0.00**
Haptics	0.27 ± 0.01	0.00 ± 0.00	0.26 ± 0.02	0.00 ± 0.00	0.31 ± 0.01	0.14 ± 0.03	0.26 ± 0.03	0.02 ± 0.04	0.29 ± 0.04	0.08 ± 0.09	0.19 ± 0.00	0.00 ± 0.00	**0.36 ± 0.00**	**0.89 ± 0.04**
MedicalImages	**0.55 ± 0.01**	0.00 ± 0.00	0.41 ± 0.08	0.00 ± 0.00	0.11 ± 0.11	0.00 ± 0.00	0.32 ± 0.08	0.02 ± 0.05	0.45 ± 0.02	0.12 ± 0.08	0.06 ± 0.02	0.00 ± 0.00	0.54 ± 0.00	**0.40 ± 0.13**
Mid.Phal.Out.AGr.	**0.60 ± 0.01**	0.12 ± 0.16	0.55 ± 0.12	0.12 ± 0.13	0.52 ± 0.11	0.20 ± 0.10	0.54 ± 0.08	0.21 ± 0.08	0.54 ± 0.07	0.22 ± 0.08	0.35 ± 0.18	0.00 ± 0.00	0.53 ± 0.00	**0.82 ± 0.06**
Mid.Phal.TW	0.56 ± 0.00	0.00 ± 0.00	0.53 ± 0.04	0.00 ± 0.00	0.31 ± 0.04	0.00 ± 0.00	0.24 ± 0.06	0.00 ± 0.00	0.35 ± 0.03	0.00 ± 0.00	0.24 ± 0.09	0.00 ± 0.00	**0.57 ± 0.00**	**0.47 ± 0.15**
OSULeaf	0.38 ± 0.03	0.00 ± 0.00	**0.42 ± 0.03**	0.00 ± 0.00	0.36 ± 0.04	0.21 ± 0.08	0.30 ± 0.04	0.02 ± 0.04	0.41 ± 0.12	0.20 ± 0.11	0.17 ± 0.06	0.00 ± 0.00	0.41 ± 0.00	**0.89 ± 0.05**
Prox.Phal.Out.AGr.	**0.82 ± 0.12**	0.01 ± 0.04	0.72 ± 0.07	0.48 ± 0.18	0.76 ± 0.02	0.56 ± 0.07	0.73 ± 0.07	0.53 ± 0.15	0.78 ± 0.03	0.56 ± 0.15	0.57 ± 0.19	0.33 ± 0.18	0.42 ± 0.00	**0.82 ± 0.10**
Prox.Phal.TW	0.70 ± 0.03	0.00 ± 0.00	0.73 ± 0.03	0.00 ± 0.00	0.56 ± 0.14	0.02 ± 0.05	0.25 ± 0.14	0.00 ± 0.00	0.45 ± 0.10	0.00 ± 0.00	0.08 ± 0.11	0.00 ± 0.00	**0.74 ± 0.00**	**0.73 ± 0.09**
Worms	**0.46 ± 0.03**	0.00 ± 0.00	0.31 ± 0.03	0.00 ± 0.00	0.21 ± 0.04	0.11 ± 0.07	0.23 ± 0.08	0.03 ± 0.04	0.30 ± 0.13	0.08 ± 0.14	0.40 ± 0.09	0.00 ± 0.00	0.38 ± 0.00	**0.59 ± 0.10**
Wafer	0.95 ± 0.00	0.64 ± 0.04	0.94 ± 0.01	0.75 ± 0.01	0.92 ± 0.01	0.78 ± 0.01	0.82 ± 0.02	0.80 ± 0.01	0.86 ± 0.01	0.80 ± 0.00	0.19 ± 0.25	0.00 ± 0.00	**0.99 ± 0.00**	**0.99 ± 0.00**
Phal.Out.Co.	0.70 ± 0.01	0.31 ± 0.02	0.69 ± 0.02	0.53 ± 0.04	**0.71 ± 0.02**	0.69 ± 0.03	0.69 ± 0.03	0.63 ± 0.05	**0.71 ± 0.01**	0.69 ± 0.01	0.47 ± 0.18	0.43 ± 0.16	0.62 ± 0.00	**0.89 ± 0.01**
Prox.Phal.Out.Co.	**0.83 ± 0.01**	0.47 ± 0.05	0.65 ± 0.01	0.50 ± 0.01	0.77 ± 0.07	0.71 ± 0.11	0.68 ± 0.04	0.57 ± 0.06	0.77 ± 0.05	0.71 ± 0.09	0.51 ± 0.27	0.43 ± 0.22	0.74 ± 0.00	**0.91 ± 0.02**
HD-1%	0.80 ± 0.41	**0.34 ± 0.26**	0.79 ± 0.41	0.29 ± 0.25	0.70 ± 0.47	0.27 ± 0.27	0.60 ± 0.51	0.15 ± 0.18	0.89 ± 0.31	0.32 ± 0.26	0.40 ± 0.51	0.00 ± 0.00	**0.99 ± 0.00**	0.03 ± 0.00
HD-3%	0.98 ± 0.00	0.25 ± 0.20	0.88 ± 0.30	0.25 ± 0.18	0.88 ± 0.30	0.11 ± 0.13	0.88 ± 0.30	0.17 ± 0.17	0.88 ± 0.30	**0.26 ± 0.18**	0.59 ± 0.49	0.00 ± 0.00	**0.99 ± 0.00**	0.06 ± 0.00
HD-5%	0.96 ± 0.01	0.30 ± 0.12	0.42 ± 0.47	0.15 ± 0.21	0.87 ± 0.29	**0.34 ± 0.19**	0.78 ± 0.38	0.32 ± 0.20	0.87 ± 0.29	0.40 ± 0.15	0.50 ± 0.47	0.00 ± 0.00	**0.99 ± 0.00**	0.06 ± 0.00
Time the best	6	1	2	5	1	1	0	0	1	2	0	0	**11**	**18**

Table 9
Comparison results of average and standard deviation of *acc* and *minRec* obtained by the proposed COCL loss function and other loss functions learning for MLSTM-FCN classifier.

Dataset	RESNET													
	ACC		MEAN		PROD		SOFTMAX		LSE		PNORM		COCL	
	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec
ChlorineCon.	0.58 ± 0.00	0.01 ± 0.02	0.43 ± 0.01	0.37 ± 0.01	0.46 ± 0.01	**0.38 ± 0.01**	0.41 ± 0.02	0.37 ± 0.01	0.45 ± 0.01	0.37 ± 0.00	0.42 ± 0.01	0.37 ± 0.01	**0.66 ± 0.02**	0.35 ± 0.04
Dist.Phal.Out.AGr.	**0.73 ± 0.02**	0.58 ± 0.02	0.68 ± 0.04	0.57 ± 0.03	0.71 ± 0.02	0.63 ± 0.04	0.72 ± 0.01	0.61 ± 0.02	**0.73 ± 0.01**	**0.67 ± 0.02**	0.39 ± 0.17	0.05 ± 0.17	**0.73 ± 0.07**	0.11 ± 0.00
Dist.Phal.TW	0.69 ± 0.01	0.00 ± 0.00	0.59 ± 0.02	0.01 ± 0.03	0.58 ± 0.01	0.14 ± 0.03	0.58 ± 0.04	0.20 ± 0.06	0.59 ± 0.02	0.22 ± 0.04	0.56 ± 0.01	0.17 ± 0.04	**0.70 ± 0.04**	**0.30 ± 0.00**
Earthquakes	**0.78 ± 0.01**	0.15 ± 0.03	0.66 ± 0.01	0.57 ± 0.03	0.66 ± 0.03	0.51 ± 0.03	0.67 ± 0.01	**0.64 ± 0.01**	0.66 ± 0.02	0.61 ± 0.02	0.66 ± 0.02	0.62 ± 0.01	0.75 ± 0.00	0.58 ± 0.08
ECG200	0.85 ± 0.01	0.69 ± 0.00	0.82 ± 0.01	**0.80 ± 0.01**	0.81 ± 0.01	**0.80 ± 0.01**	0.81 ± 0.01	0.76 ± 0.01	0.80 ± 0.01	0.77 ± 0.02	0.80 ± 0.01	0.77 ± 0.01	**0.92 ± 0.04**	0.36 ± 0.00
ECG5000	**0.93 ± 0.00**	0.00 ± 0.00	0.91 ± 0.03	0.20 ± 0.04	0.85 ± 0.04	0.32 ± 0.05	0.45 ± 0.21	0.11 ± 0.08	0.75 ± 0.11	0.28 ± 0.06	0.48 ± 0.10	0.00 ± 0.00	**0.93 ± 0.02**	**0.34 ± 0.05**
ElectricDevices	0.63 ± 0.17	0.19 ± 0.08	**0.74 ± 0.01**	0.32 ± 0.01	0.72 ± 0.01	0.34 ± 0.02	0.62 ± 0.01	0.33 ± 0.03	0.66 ± 0.06	0.32 ± 0.03	0.66 ± 0.04	**0.35 ± 0.01**	0.43 ± 0.00	0.15 ± 0.05
FiftyWords	0.28 ± 0.00	0.00 ± 0.00	0.34 ± 0.02	0.00 ± 0.00	0.04 ± 0.03	0.00 ± 0.00	0.42 ± 0.01	0.00 ± 0.00	0.47 ± 0.01	0.00 ± 0.00	0.05 ± 0.02	0.00 ± 0.00	**0.61 ± 0.04**	**0.16 ± 0.00**
Haptics	0.40 ± 0.02	0.04 ± 0.04	0.38 ± 0.01	0.00 ± 0.00	0.39 ± 0.02	0.23 ± 0.02	0.40 ± 0.03	0.24 ± 0.02	0.41 ± 0.03	**0.27 ± 0.01**	0.37 ± 0.02	0.18 ± 0.02	**0.46 ± 0.02**	0.22 ± 0.00
MedicalImages	0.57 ± 0.01	0.00 ± 0.00	0.41 ± 0.01	0.00 ± 0.00	0.26 ± 0.17	0.00 ± 0.00	0.50 ± 0.01	0.42 ± 0.01	0.56 ± 0.01	0.48 ± 0.01	0.40 ± 0.07	0.14 ± 0.12	**0.68 ± 0.02**	**0.51 ± 0.00**
Mid.Phal.Out.AGr.	0.58 ± 0.10	0.18 ± 0.07	0.49 ± 0.08	0.31 ± 0.07	0.52 ± 0.08	0.33 ± 0.08	0.51 ± 0.03	0.36 ± 0.03	0.51 ± 0.02	**0.39 ± 0.04**	0.56 ± 0.05	0.28 ± 0.03	**0.71 ± 0.05**	0.02 ± 0.00
Mid.Phal.TW	0.55 ± 0.01	0.00 ± 0.00	0.54 ± 0.01	0.00 ± 0.00	0.40 ± 0.02	0.22 ± 0.00	0.27 ± 0.02	0.00 ± 0.00	0.26 ± 0.04	0.09 ± 0.08	0.24 ± 0.02	0.00 ± 0.00	**0.62 ± 0.01**	**0.42 ± 0.00**
OSULeaf	0.73 ± 0.02	0.00 ± 0.00	**0.79 ± 0.03**	0.56 ± 0.05	0.76 ± 0.02	0.61 ± 0.01	0.74 ± 0.01	0.59 ± 0.01	0.76 ± 0.02	**0.63 ± 0.02**	0.72 ± 0.02	0.57 ± 0.01	0.59 ± 0.03	0.24 ± 0.00
Prox.Phal.Out.AGr.	0.75 ± 0.13	0.50 ± 0.03	0.82 ± 0.01	**0.70 ± 0.01**	0.82 ± 0.00	0.71 ± 0.00	0.83 ± 0.02	0.60 ± 0.11	0.82 ± 0.03	0.62 ± 0.07	0.84 ± 0.01	0.63 ± 0.05	**0.86 ± 0.07**	0.04 ± 0.02
Prox.Phal.TW	**0.82 ± 0.00**	0.00 ± 0.00	0.72 ± 0.01	0.00 ± 0.00	0.69 ± 0.03	0.16 ± 0.03	0.58 ± 0.01	**0.49 ± 0.01**	0.63 ± 0.02	0.40 ± 0.05	0.77 ± 0.01	0.02 ± 0.03	**0.82 ± 0.04**	0.33 ± 0.02
Worms	0.61 ± 0.01	0.00 ± 0.00	0.54 ± 0.06	0.27 ± 0.08	0.66 ± 0.02	0.36 ± 0.05	0.65 ± 0.02	**0.42 ± 0.06**	0.66 ± 0.02	0.42 ± 0.07	**0.67 ± 0.03**	0.39 ± 0.03	0.57 ± 0.02	0.32 ± 0.05
Wafer	0.99 ± 0.00	0.96 ± 0.01	0.99 ± 0.00	**0.97 ± 0.00**	0.99 ± 0.00	**0.97 ± 0.00**	0.99 ± 0.00	0.96 ± 0.01	0.99 ± 0.01	**0.97 ± 0.00**	0.99 ± 0.01	0.96 ± 0.01	**1.00 ± 0.00**	0.92 ± 0.02
Phal.Out.Co.	**0.79 ± 0.01**	0.64 ± 0.05	0.75 ± 0.02	**0.72 ± 0.01**	0.76 ± 0.02	**0.72 ± 0.02**	0.75 ± 0.02	**0.72 ± 0.01**	0.74 ± 0.01	**0.72 ± 0.01**	0.75 ± 0.02	0.62 ± 0.04	0.67 ± 0.01	0.35 ± 0.06
Prox.Phal.Out.Co.	0.86 ± 0.02	0.65 ± 0.05	0.66 ± 0.00	0.53 ± 0.00	0.86 ± 0.02	0.74 ± 0.05	0.83 ± 0.02	**0.75 ± 0.05**	0.84 ± 0.02	**0.75 ± 0.03**	0.82 ± 0.02	0.71 ± 0.07	**0.87 ± 0.01**	0.38 ± 0.08
HD-1%	**0.99 ± 0.00**	0.48 ± 0.05	**0.99 ± 0.00**	0.66 ± 0.03	**0.99 ± 0.00**	**0.67 ± 0.00**	**0.99 ± 0.00**	**0.67 ± 0.00**	**0.99 ± 0.00**	**0.67 ± 0.00**	0.93 ± 0.13	0.24 ± 0.21	**0.99 ± 0.00**	0.00 ± 0.00
HD-3%	**0.99 ± 0.00**	0.57 ± 0.01	0.98 ± 0.00	0.61 ± 0.02	0.98 ± 0.00	0.63 ± 0.02	0.98 ± 0.00	**0.66 ± 0.00**	0.98 ± 0.00	0.65 ± 0.01	0.80 ± 0.39	0.24 ± 0.20	0.98 ± 0.00	0.00 ± 0.00
HD-5%	**0.98 ± 0.00**	0.61 ± 0.01	0.97 ± 0.00	**0.63 ± 0.00**	0.97 ± 0.00	**0.63 ± 0.00**	0.96 ± 0.04	**0.63 ± 0.01**	0.97 ± 0.00	**0.63 ± 0.01**	0.97 ± 0.00	**0.63 ± 0.00**	**0.98 ± 0.00**	0.00 ± 0.00
Time the best	8	0	3	5	1	6	1	8	2	**9**	1	2	**16**	5

predictions across various datasets. Additionally, COCL, LSE, softmax, and product loss functions emerged as the most effective for addressing ITSC issues, displaying no significant differences in performance metrics *acc* and *minRec* compared to the best-performing loss functions. Moreover, MLS classifiers trained with the COCL loss function obtained the highest results in *acc* and *minRec* for a greater number of datasets. Therefore, we consider it the optimal proposed approach.

5.7. Comparative experiments with state-of-the-art methods

The objective of this investigation is to evaluate the efficacy of the MLS classifier trained with the COCL loss function as a prospective solution to the ITSC problem, in comparison to other contemporary methodologies. Notably, a comprehensive review of pertinent literature was conducted to identify methodologies that have sufficiently addressed the ITSC problem and possess the requisite details for reproducibility. The selected methodologies can be categorized as follows:

The objective of this investigation is to evaluate the efficacy of the MLS classifier trained with the COCL loss function as a prospective solution to the ITSC problem, in comparison to other contemporary methodologies. Notably, a comprehensive review of pertinent literature was conducted to identify methodologies that have sufficiently addressed the ITSC problem and possess the requisite details for reproducibility. The selected methodologies can be categorized as follows:

- Oversampling methods not explicitly tailored for time series: This category includes SMOTE (Chawla et al., 2002) and ADASYN (He et al., 2008), implemented through the Imbalanced-learn Python package.
- Oversampling methods specifically designed for time series: Encompassing INOS (Cao et al., 2013), MOGT (Pang et al., 2013), and SPO (Cao et al., 2011), these methodologies were originally formulated without consideration for multidimensional time series or multiclass issues.
- Cost-sensitive methodologies: Specifically, CS-CNN (Shen et al., 2018) and K3-CNN (Raj et al., 2016), originally conceived for binary classification scenarios. Adapting these methodologies for multiclass problems involves calculating misclassification cost metrics, such as accuracy and Gmeans, for multiclass settings.
- State-Of-The-Art Methods: Selected the recently proposed MHCCL method (Meng et al., 2023) and MICOS method (Hao et al., 2023) for handling multivariate time series data.

The meticulously organized comparative experimental results, presented in Tables 10 and 11, are indicative of the performance across datasets. Supplementary visual representations elucidating pairwise differentials in *acc* and *minRec* measures are depicted in Figs. 5 and 6. Both Tables 10 and 11 underscore the significance of the proposed COCL loss function, particularly evident in the majority of datasets, with a pronounced impact in the *acc* column. Fig. 5 substantiates this observation, revealing the superiority of the proposed methodology in *acc* when compared with other state-of-the-art methods. Simultaneously, while the proposed COCL loss function does not exhibit optimal performance in the *minRec* column, it consistently maintains

Table 10
Comparison results of average and standard deviation of *acc* and *minRec* obtained by the proposed loss function and state-of-the-art methods learning for resnet classifier.

Dataset	RESNET																					
	COCL		LSE		SMOTE		ADASYN		SPO		INOS		MOGT		CS-CNN		K3-CNN		MICOS		MHCCL	
	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec
ChlorineConc.	0.68 ± 0.00	**0.93 ± 0.04**	0.34 ± 0.06	0.19 ± 0.07	0.35 ± 0.05	0.19 ± 0.09	0.30 ± 0.03	0.17 ± 0.07	0.43 ± 0.02	0.26 ± 0.04	0.34 ± 0.01	0.27 ± 0.04	0.38 ± 0.03	0.30 ± 0.02	0.24 ± 0.03	0.03 ± 0.09	0.43 ± 0.06	0.00 ± 0.00	0.63 ± 0.00	0.19 ± 0.01	0.57 ± 0.00	0.12 ± 0.01
Dist.Phal.Out.AGr.	0.64 ± 0.00	**0.98 ± 0.01**	0.68 ± 0.08	0.54 ± 0.14	0.71 ± 0.01	0.58 ± 0.00	0.69 ± 0.01	0.58 ± 0.00	**0.72 ± 0.01**	0.58 ± 0.00	0.70 ± 0.01	0.58 ± 0.00	0.71 ± 0.01	0.58 ± 0.00	0.48 ± 0.08	0.11 ± 0.16	0.60 ± 0.11	0.00 ± 0.00	0.68 ± 0.10	0.11 ± 0.08	0.69 ± 0.00	0.57 ± 0.01
Dist.Phal.TW	0.63 ± 0.00	**0.72 ± 0.11**	0.48 ± 0.10	0.00 ± 0.00	0.61 ± 0.02	0.00 ± 0.00	0.60 ± 0.01	0.00 ± 0.00	0.57 ± 0.01	0.05 ± 0.08	0.56 ± 0.02	0.06 ± 0.07	0.60 ± 0.01	0.00 ± 0.00	0.28 ± 0.06	0.00 ± 0.00	0.28 ± 0.00	0.00 ± 0.00	**0.69 ± 0.01**	0.40 ± 0.00	0.61 ± 0.00	0.02 ± 0.00
Earthquakes	0.71 ± 0.00	**0.98 ± 0.01**	0.57 ± 0.02	0.52 ± 0.05	**0.75 ± 0.01**	0.08 ± 0.05	**0.75 ± 0.00**	0.00 ± 0.00	0.74 ± 0.01	0.00 ± 0.02	0.74 ± 0.01	0.02 ± 0.08	0.74 ± 0.00	0.00 ± 0.00	0.68 ± 0.14	0.12 ± 0.12	0.44 ± 0.03	0.27 ± 0.05	0.57 ± 0.00	0.23 ± 0.01	0.74 ± 0.00	0.02 ± 0.00
ECG200	**0.88 ± 0.00**	**0.98 ± 0.01**	0.68 ± 0.01	0.57 ± 0.04	0.75 ± 0.01	0.68 ± 0.02	0.72 ± 0.04	0.47 ± 0.07	0.74 ± 0.04	0.63 ± 0.16	0.76 ± 0.03	0.67 ± 0.14	0.72 ± 0.04	0.59 ± 0.15	0.37 ± 0.04	0.04 ± 0.12	0.65 ± 0.03	0.61 ± 0.03	**0.86 ± 0.00**	0.67 ± 0.02	0.87 ± 0.00	0.75 ± 0.00
ECG5000	0.86 ± 0.02	**0.26 ± 0.06**	0.73 ± 0.08	0.00 ± 0.00	–	–	0.78 ± 0.01	0.06 ± 0.05	0.89 ± 0.01	0.15 ± 0.10	0.89 ± 0.02	0.12 ± 0.09	0.86 ± 0.01	0.10 ± 0.02	0.66 ± 0.24	0.05 ± 0.08	0.35 ± 0.00	0.00 ± 0.00	**0.92 ± 0.00**	0.03 ± 0.00	0.91 ± 0.00	0.01 ± 0.00
ElectricDevices	0.34 ± 0.00	0.00 ± 0.00	0.57 ± 0.04	0.17 ± 0.09	0.66 ± 0.01	0.00 ± 0.00	0.62 ± 0.04	0.01 ± 0.01	0.65 ± 0.03	**0.24 ± 0.05**	0.67 ± 0.02	0.18 ± 0.04	**0.70 ± 0.01**	0.18 ± 0.05	0.69 ± 0.02	0.20 ± 0.05	0.50 ± 0.02	0.00 ± 0.00	0.53 ± 0.00	0.21 ± 0.00	0.57 ± 0.00	0.18 ± 0.00
FiftyWords	0.40 ± 0.00	**0.01 ± 0.00**	0.34 ± 0.02	0.00 ± 0.00	–	–	–	–	–	–	–	–	–	–	0.17 ± 0.09	0.00 ± 0.00	0.11 ± 0.04	0.00 ± 0.00	**0.46 ± 0.00**	0.00 ± 0.00	0.40 ± 0.04	0.00 ± 0.00
Haptics	0.36 ± 0.00	**0.89 ± 0.04**	0.29 ± 0.04	0.08 ± 0.09	0.28 ± 0.01	0.00 ± 0.00	0.25 ± 0.02	0.00 ± 0.01	0.29 ± 0.02	0.00 ± 0.00	0.28 ± 0.02	0.00 ± 0.00	0.28 ± 0.02	0.00 ± 0.01	0.20 ± 0.01	0.00 ± 0.00	0.22 ± 0.02	0.00 ± 0.00	0.33 ± 0.00	0.00 ± 0.00	**0.38 ± 0.00**	0.13 ± 0.00
MedicalImages	0.54 ± 0.00	**0.40 ± 0.13**	0.45 ± 0.02	0.12 ± 0.08	0.34 ± 0.01	0.00 ± 0.00	0.31 ± 0.02	0.00 ± 0.00	0.31 ± 0.01	0.00 ± 0.00	0.32 ± 0.01	0.00 ± 0.00	0.33 ± 0.01	0.00 ± 0.00	0.21 ± 0.03	0.00 ± 0.00	0.15 ± 0.01	0.00 ± 0.00	**0.56 ± 0.00**	0.11 ± 0.00	0.55 ± 0.00	0.13 ± 0.01
Mid.Phal.Out.AGr.	0.53 ± 0.00	**0.82 ± 0.06**	0.54 ± 0.07	0.22 ± 0.08	0.60 ± 0.01	0.24 ± 0.00	0.56 ± 0.02	0.23 ± 0.05	0.60 ± 0.01	0.24 ± 0.00	0.60 ± 0.01	0.24 ± 0.00	0.59 ± 0.03	0.24 ± 0.00	0.39 ± 0.13	0.04 ± 0.14	**0.61 ± 0.02**	0.00 ± 0.00	0.57 ± 0.10	0.22 ± 0.01	0.53 ± 0.00	0.28 ± 0.00
Mid.Phal.TW	0.57 ± 0.00	**0.47 ± 0.15**	0.35 ± 0.03	0.00 ± 0.00	0.49 ± 0.01	0.00 ± 0.00	0.43 ± 0.01	0.00 ± 0.00	0.48 ± 0.01	0.00 ± 0.00	0.48 ± 0.01	0.00 ± 0.00	0.44 ± 0.02	0.02 ± 0.03	0.31 ± 0.12	0.00 ± 0.00	0.48 ± 0.08	0.00 ± 0.00	**0.61 ± 0.12**	0.08 ± 0.00	0.55 ± 0.00	0.00 ± 0.00
OSULeaf	0.41 ± 0.00	**0.89 ± 0.05**	0.41 ± 0.12	0.20 ± 0.11	0.45 ± 0.03	0.10 ± 0.08	0.38 ± 0.04	0.00 ± 0.00	0.34 ± 0.02	0.00 ± 0.01	0.41 ± 0.03	0.00 ± 0.00	0.33 ± 0.02	0.00 ± 0.00	0.20 ± 0.03	0.00 ± 0.00	0.21 ± 0.03	0.00 ± 0.00	**0.46 ± 0.00**	0.17 ± 0.00	0.50 ± 0.00	0.23 ± 0.01
Prox.Phal.Out.AGr.	0.42 ± 0.00	**0.82 ± 0.10**	0.78 ± 0.03	0.56 ± 0.15	0.78 ± 0.02	0.60 ± 0.03	0.73 ± 0.09	0.53 ± 0.17	0.77 ± 0.01	0.59 ± 0.03	0.75 ± 0.01	0.55 ± 0.03	0.76 ± 0.02	0.57 ± 0.07	0.38 ± 0.20	0.03 ± 0.07	0.73 ± 0.17	0.00 ± 0.00	**0.85 ± 0.00**	0.08 ± 0.00	0.84 ± 0.00	0.49 ± 0.02
Prox.Phal.TW	0.74 ± 0.00	**0.73 ± 0.09**	0.45 ± 0.10	0.00 ± 0.00	0.68 ± 0.02	0.04 ± 0.08	0.68 ± 0.02	0.07 ± 0.00	0.72 ± 0.02	0.08 ± 0.02	0.68 ± 0.01	0.12 ± 0.06	0.66 ± 0.03	0.00 ± 0.00	0.37 ± 0.08	0.00 ± 0.00	0.35 ± 0.05	0.00 ± 0.00	**0.78 ± 0.01**	0.00 ± 0.00	0.76 ± 0.00	0.00 ± 0.00
Worms	0.38 ± 0.00	**0.59 ± 0.10**	0.30 ± 0.13	0.08 ± 0.14	**0.58 ± 0.01**	0.18 ± 0.04	0.46 ± 0.02	0.00 ± 0.00	0.42 ± 0.07	0.08 ± 0.08	0.42 ± 0.07	0.06 ± 0.07	0.46 ± 0.08	0.01 ± 0.03	0.17 ± 0.02	0.00 ± 0.00	0.19 ± 0.02	0.00 ± 0.00	0.45 ± 0.00	0.12 ± 0.01	0.46 ± 0.00	0.15 ± 0.00
Wafer	**0.99 ± 0.00**	**0.99 ± 0.00**	0.86 ± 0.01	0.80 ± 0.00	0.97 ± 0.00	0.93 ± 0.01	0.94 ± 0.01	0.58 ± 0.08	0.96 ± 0.01	0.69 ± 0.06	0.96 ± 0.00	0.82 ± 0.06	0.96 ± 0.00	0.72 ± 0.02	0.68 ± 0.29	0.43 ± 0.20	0.38 ± 0.17	0.31 ± 0.20	0.98 ± 0.00	0.92 ± 0.00	0.98 ± 0.00	0.90 ± 0.02
Phal.Out.Co.	0.62 ± 0.00	**0.89 ± 0.01**	0.71 ± 0.01	0.69 ± 0.01	0.72 ± 0.02	0.69 ± 0.05	0.68 ± 0.01	0.65 ± 0.02	0.74 ± 0.01	0.49 ± 0.04	0.74 ± 0.01	0.60 ± 0.05	**0.76 ± 0.01**	0.53 ± 0.04	0.55 ± 0.13	0.13 ± 0.13	0.59 ± 0.08	0.19 ± 0.11	0.71 ± 0.00	0.41 ± 0.03	**0.76 ± 0.00**	0.49 ± 0.01
Prox.Phal.Out.Co.	0.74 ± 0.00	**0.91 ± 0.02**	0.77 ± 0.05	0.71 ± 0.09	0.81 ± 0.02	0.75 ± 0.02	0.74 ± 0.05	0.68 ± 0.09	0.85 ± 0.02	0.62 ± 0.05	0.76 ± 0.06	0.69 ± 0.08	0.85 ± 0.01	0.64 ± 0.04	0.61 ± 0.16	0.06 ± 0.15	0.56 ± 0.17	0.31 ± 0.27	0.82 ± 0.00	0.39 ± 0.03	**0.86 ± 0.00**	0.66 ± 0.02
HD-1%	**0.99 ± 0.00**	0.03 ± 0.00	0.89 ± 0.31	**0.32 ± 0.26**	0.01 ± 0.00	0.00 ± 0.00	0.94 ± 0.22	0.21 ± 0.27	0.89 ± 0.30	0.23 ± 0.23	0.94 ± 0.22	0.25 ± 0.28	–	–	0.70 ± 0.47	0.18 ± 0.13	0.11 ± 0.31	0.06 ± 0.18	**0.99 ± 0.00**	0.01 ± 0.00	0.98 ± 0.00	0.32 ± 0.02
HD-3%	**0.99 ± 0.00**	0.06 ± 0.00	0.88 ± 0.30	0.26 ± 0.18	0.03 ± 0.00	0.00 ± 0.00	0.93 ± 0.21	0.16 ± 0.18	0.97 ± 0.00	0.16 ± 0.11	0.97 ± 0.00	0.18 ± 0.14	–	–	0.69 ± 0.46	0.10 ± 0.10	0.03 ± 0.00	0.00 ± 0.00	0.98 ± 0.00	0.08 ± 0.00	0.98 ± 0.00	**0.33 ± 0.01**
HD-5%	**0.99 ± 0.00**	0.06 ± 0.00	0.87 ± 0.29	**0.40 ± 0.15**	0.33 ± 0.44	0.19 ± 0.30	0.91 ± 0.20	0.26 ± 0.13	0.91 ± 0.20	0.26 ± 0.15	0.87 ± 0.28	0.26 ± 0.15	–	–	0.15 ± 0.29	0.06 ± 0.20	0.06 ± 0.00	0.00 ± 0.00	0.98 ± 0.01	0.08 ± 0.00	0.98 ± 0.00	0.33 ± 0.01
Time the best	6	**18**	0	2	2	0	1	0	1	1	1	0	2	0	0	0	1	0	**9**	0	3	1

Table 11
Comparison results of average and standard deviation of *acc* and *minRec* obtained by the proposed loss function and state-of-the-art methods learning for MLSTM-FCN classifier.

Dataset	RESNET																					
	COCL		LSE		SMOTE		ADASYN		SPO		INOS		MOGT		CS-CNN		K3-CNN		MICOS		MHCCL	
	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec
ChlorineConc.	0.66 ± 0.02	0.35 ± 0.04	0.45 ± 0.01	0.37 ± 0.00	0.48 ± 0.04	0.38 ± 0.02	0.45 ± 0.02	0.31 ± 0.05	0.53 ± 0.04	0.31 ± 0.07	0.52 ± 0.03	0.37 ± 0.06	0.56 ± 0.02	0.18 ± 0.09	0.47 ± 0.09	0.14 ± 0.07	0.42 ± 0.04	0.22 ± 0.03	0.61 ± 0.08	0.11 ± 0.01	**0.76 ± 0.01**	**0.55 ± 0.04**
Dist.Phal.Out.AGr.	**0.73 ± 0.07**	0.11 ± 0.00	**0.73 ± 0.01**	**0.67 ± 0.02**	0.60 ± 0.05	0.48 ± 0.08	0.66 ± 0.08	0.44 ± 0.18	0.64 ± 0.07	0.33 ± 0.21	0.65 ± 0.09	0.44 ± 0.21	0.68 ± 0.07	0.38 ± 0.20	0.69 ± 0.03	0.58 ± 0.02	0.67 ± 0.06	0.47 ± 0.08	0.73 ± 0.02	0.25 ± 0.08	0.68 ± 0.00	0.54 ± 0.01
Dist.Phal.TW	**0.70 ± 0.04**	**0.30 ± 0.00**	0.59 ± 0.02	0.22 ± 0.04	0.64 ± 0.02	0.17 ± 0.06	0.60 ± 0.02	0.12 ± 0.03	0.67 ± 0.02	0.19 ± 0.07	0.65 ± 0.02	0.20 ± 0.08	0.64 ± 0.03	0.09 ± 0.07	0.68 ± 0.01	0.00 ± 0.00	0.68 ± 0.00	0.00 ± 0.00	0.62 ± 0.02	0.00 ± 0.00	0.60 ± 0.00	0.02 ± 0.00
Earthquakes	0.75 ± 0.00	0.58 ± 0.08	0.66 ± 0.02	**0.61 ± 0.02**	0.57 ± 0.11	0.36 ± 0.04	0.75 ± 0.00	0.00 ± 0.00	**0.76 ± 0.03**	0.23 ± 0.18	0.74 ± 0.06	0.22 ± 0.18	0.65 ± 0.20	0.12 ± 0.18	0.72 ± 0.03	0.46 ± 0.04	0.58 ± 0.08	0.43 ± 0.06	0.57 ± 0.00	0.43 ± 0.01	0.65 ± 0.00	0.27 ± 0.00
ECG200	**0.92 ± 0.04**	0.36 ± 0.00	0.80 ± 0.01	0.77 ± 0.02	0.83 ± 0.03	0.68 ± 0.12	0.80 ± 0.04	0.55 ± 0.17	0.84 ± 0.02	0.72 ± 0.08	0.84 ± 0.02	0.75 ± 0.07	0.68 ± 0.19	0.40 ± 0.37	0.74 ± 0.07	0.53 ± 0.10	0.83 ± 0.01	**0.81 ± 0.01**	0.86 ± 0.00	0.77 ± 0.02	0.89 ± 0.00	0.83 ± 0.01
ECG5000	**0.93 ± 0.02**	**0.34 ± 0.05**	0.75 ± 0.11	0.28 ± 0.06	–	–	0.93 ± 0.00	0.00 ± 0.00	0.93 ± 0.01	0.16 ± 0.03	0.92 ± 0.01	0.16 ± 0.05	0.93 ± 0.01	0.13 ± 0.05	0.43 ± 0.12	0.03 ± 0.03	0.63 ± 0.03	0.00 ± 0.00	0.91 ± 0.01	0.05 ± 0.00	0.92 ± 0.00	0.12 ± 0.01
ElectricDevices	0.43 ± 0.00	0.15 ± 0.05	0.66 ± 0.06	**0.32 ± 0.03**	0.52 ± 0.21	0.20 ± 0.03	0.64 ± 0.03	0.04 ± 0.02	**0.74 ± 0.02**	0.17 ± 0.04	0.72 ± 0.01	0.16 ± 0.06	0.73 ± 0.01	0.22 ± 0.07	0.59 ± 0.20	0.23 ± 0.07	0.56 ± 0.14	0.25 ± 0.09	0.56 ± 0.00	0.20 ± 0.00	0.58 ± 0.00	0.18 ± 0.00
FiftyWords	0.61 ± 0.04	**0.16 ± 0.00**	0.47 ± 0.01	0.00 ± 0.00	–	–	–	–	–	–	–	–	–	–	0.22 ± 0.01	0.00 ± 0.00	0.16 ± 0.00	0.00 ± 0.00	0.59 ± 0.05	0.00 ± 0.00	**0.62 ± 0.00**	0.00 ± 0.00
Haptics	**0.46 ± 0.02**	0.22 ± 0.00	0.41 ± 0.03	**0.27 ± 0.01**	0.42 ± 0.01	0.11 ± 0.02	0.21 ± 0.00	0.00 ± 0.00	0.38 ± 0.02	0.02 ± 0.03	0.37 ± 0.03	0.01 ± 0.02	0.40 ± 0.04	0.10 ± 0.08	0.36 ± 0.05	0.02 ± 0.04	0.28 ± 0.02	0.00 ± 0.00	0.39 ± 0.00	0.04 ± 0.00	0.43 ± 0.00	0.13 ± 0.00
MedicalImages	0.68 ± 0.02	**0.51 ± 0.00**	0.56 ± 0.01	0.48 ± 0.01	0.61 ± 0.01	0.45 ± 0.01	0.66 ± 0.03	0.04 ± 0.03	0.72 ± 0.01	0.28 ± 0.06	**0.73 ± 0.02**	0.39 ± 0.07	0.72 ± 0.01	0.28 ± 0.04	0.32 ± 0.04	0.01 ± 0.03	0.16 ± 0.00	0.00 ± 0.00	0.67 ± 0.00	0.01 ± 0.00	0.69 ± 0.00	0.34 ± 0.01
Mid.Phal.Out.AGr.	**0.71 ± 0.05**	0.02 ± 0.00	0.51 ± 0.02	**0.39 ± 0.04**	0.55 ± 0.03	0.29 ± 0.02	0.58 ± 0.02	0.24 ± 0.00	0.61 ± 0.02	0.23 ± 0.01	0.60 ± 0.02	0.24 ± 0.01	0.59 ± 0.02	0.24 ± 0.01	0.54 ± 0.09	0.25 ± 0.02	0.54 ± 0.07	0.09 ± 0.07	0.52 ± 0.00	0.25 ± 0.01	0.51 ± 0.00	0.27 ± 0.00
Mid.Phal.TW	**0.62 ± 0.01**	**0.42 ± 0.00**	0.26 ± 0.04	0.09 ± 0.08	0.49 ± 0.01	0.00 ± 0.00	0.51 ± 0.03	0.00 ± 0.00	0.57 ± 0.02	0.00 ± 0.00	0.53 ± 0.03	0.03 ± 0.06	0.53 ± 0.02	0.02 ± 0.04	0.46 ± 0.04	0.00 ± 0.00	0.56 ± 0.03	0.00 ± 0.00	0.59 ± 0.15	0.00 ± 0.00	0.54 ± 0.00	0.08 ± 0.00
OSULeaf	0.59 ± 0.03	0.24 ± 0.00	0.76 ± 0.02	**0.63 ± 0.02**	0.78 ± 0.03	0.55 ± 0.04	0.36 ± 0.04	0.00 ± 0.00	**0.79 ± 0.03**	0.52 ± 0.07	0.67 ± 0.02	0.22 ± 0.12	0.68 ± 0.04	0.25 ± 0.18	0.69 ± 0.08	0.19 ± 0.11	0.56 ± 0.02	0.00 ± 0.00	0.26 ± 0.00	0.07 ± 0.00	0.54 ± 0.00	0.32 ± 0.00
Prox.Phal.Out.AGr.	**0.86 ± 0.07**	0.04 ± 0.02	0.82 ± 0.03	0.62 ± 0.07	0.83 ± 0.01	0.58 ± 0.06	0.82 ± 0.02	0.64 ± 0.07	0.85 ± 0.01	0.55 ± 0.09	0.82 ± 0.02	**0.67 ± 0.04**	0.84 ± 0.02	0.63 ± 0.08	0.69 ± 0.18	0.27 ± 0.26	0.67 ± 0.18	0.41 ± 0.02	0.89 ± 0.00	0.00 ± 0.00	0.83 ± 0.00	0.54 ± 0.02
Prox.Phal.TW	**0.82 ± 0.04**	0.33 ± 0.02	0.63 ± 0.02	**0.40 ± 0.05**	0.73 ± 0.03	0.00 ± 0.00	0.72 ± 0.02	0.00 ± 0.00	0.78 ± 0.02	0.05 ± 0.06	0.76 ± 0.04	0.10 ± 0.06	0.73 ± 0.02	0.00 ± 0.02	0.80 ± 0.01	0.00 ± 0.00	0.44 ± 0.02	0.00 ± 0.00	0.73 ± 0.01	0.00 ± 0.00	0.72 ± 0.00	0.01 ± 0.00
Worms	0.57 ± 0.02	0.32 ± 0.05	0.66 ± 0.02	**0.42 ± 0.07**	0.66 ± 0.06	0.36 ± 0.08	0.49 ± 0.02	0.00 ± 0.00	0.71 ± 0.01	0.22 ± 0.03	**0.72 ± 0.02**	0.34 ± 0.09	0.66 ± 0.02	0.05 ± 0.06	0.61 ± 0.10	0.19 ± 0.14	0.47 ± 0.05	0.00 ± 0.00	0.47 ± 0.00	0.17 ± 0.01	0.46 ± 0.00	0.17 ± 0.00
Wafer	**1.00 ± 0.00**	0.92 ± 0.02	0.99 ± 0.01	0.97 ± 0.00	0.99 ± 0.01	0.96 ± 0.02	0.97 ± 0.02	0.81 ± 0.15	0.99 ± 0.02	0.92 ± 0.11	0.99 ± 0.01	0.93 ± 0.07	0.97 ± 0.07	0.90 ± 0.09	0.98 ± 0.01	0.88 ± 0.07	0.69 ± 0.01	0.65 ± 0.01	0.99 ± 0.00	0.97 ± 0.00	0.99 ± 0.00	**0.98 ± 0.00**
Phal.Out.Co.	0.67 ± 0.01	0.35 ± 0.06	0.74 ± 0.01	**0.72 ± 0.01**	0.72 ± 0.01	0.70 ± 0.01	0.72 ± 0.02	0.43 ± 0.09	0.77 ± 0.03	0.61 ± 0.16	**0.78 ± 0.02**	0.68 ± 0.08	0.77 ± 0.02	0.58 ± 0.14	0.73 ± 0.02	0.65 ± 0.03	**0.78 ± 0.02**	0.68 ± 0.01	0.73 ± 0.00	0.45 ± 0.03	0.78 ± 0.00	0.62 ± 0.02
Prox.Phal.Out.Co.	**0.87 ± 0.01**	0.38 ± 0.08	0.84 ± 0.02	**0.75 ± 0.03**	0.83 ± 0.05	0.74 ± 0.02	0.84 ± 0.02	0.66 ± 0.12	**0.87 ± 0.03**	0.67 ± 0.12	0.86 ± 0.02	0.73 ± 0.09	0.86 ± 0.02	0.67 ± 0.08	0.77 ± 0.05	0.64 ± 0.09	0.77 ± 0.08	0.61 ± 0.09	0.79 ± 0.00	0.41 ± 0.03	0.86 ± 0.00	0.71 ± 0.00
HD-1%	**0.99 ± 0.00**	0.00 ± 0.00	0.99 ± 0.00	**0.67 ± 0.00**	0.90 ± 0.14	0.58 ± 0.01	0.55 ± 0.50	0.15 ± 0.18	0.55 ± 0.50	0.14 ± 0.16	0.46 ± 0.50	0.11 ± 0.15	–	–	0.99 ± 0.00	0.57 ± 0.05	0.22 ± 0.39	0.09 ± 0.16	0.98 ± 0.00	0.13 ± 0.00	0.98 ± 0.00	0.18 ± 0.02
HD-3%	**0.98 ± 0.00**	0.00 ± 0.00	**0.98 ± 0.00**	**0.65 ± 0.01**	**0.98 ± 0.00**	0.61 ± 0.01	0.68 ± 0.45	0.33 ± 0.25	0.77 ± 0.41	0.38 ± 0.25	0.72 ± 0.43	0.36 ± 0.25	–	–	0.96 ± 0.06	0.60 ± 0.01	0.12 ± 0.29	0.04 ± 0.12	**0.98 ± 0.00**	0.28 ± 0.00	**0.98 ± 0.00**	0.30 ± 0.03
HD-5%	**0.98 ± 0.00**	0.00 ± 0.00	0.97 ± 0.00	**0.63 ± 0.01**	0.97 ± 0.00	**0.63 ± 0.01**	0.80 ± 0.36	0.43 ± 0.23	0.55 ± 0.47	0.29 ± 0.28	0.80 ± 0.36	0.39 ± 0.22	–	–	0.97 ± 0.00	**0.63 ± 0.00**	0.06 ± 0.00	0.00 ± 0.00	**0.98 ± 0.00**	0.16 ± 0.00	**0.98 ± 0.00**	0.28 ± 0.02
Time the best	**14**	**5**	2	13	1	1	0	0	4	0	3	1	0	0	0	1	1	1	2	0	4	2

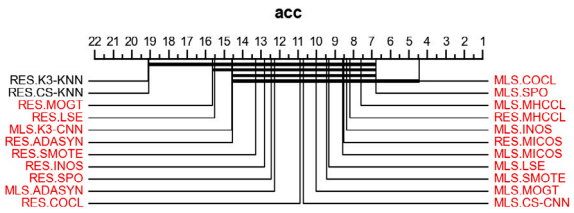


Fig. 5. Pairwise difference plot of *acc* with two classifiers (MLSTM-FCN (MLS) and Resnet (RES)) using the proposed COCL loss function versus other state-of-the-art methods with a significance level of 0.05.

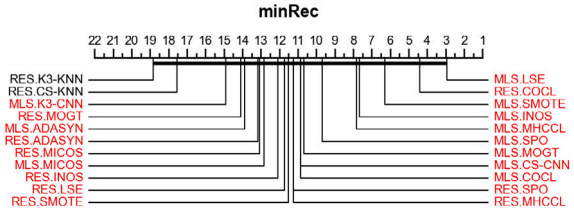


Fig. 6. Pairwise difference plot of *minRec* with two classifiers (MLSTM-FCN (MLS) and Resnet (RES)) using the proposed COCL loss function versus other state-of-the-art methods with a significance level of 0.05.

a commendable second-place position in performance outcomes (refer to Fig. 6).

In summary, despite class imbalance, the RES classifier trained with the COCL loss function demonstrates the capacity to optimize recall values across all classes, concurrently achieving noteworthy levels of overall *acc* and *minRec* across diverse datasets.

6. Conclusions

In succinct terms, this paper introduces and examines a novel contrastive clustering (COCL) loss function designed to augment the acquisition of distinctive representations for data points, thereby offering a fresh perspective for mitigating the challenges associated with the imbalanced time series classification problem. The advanced loss function posited herein adeptly minimizes similarity scores within intra-cluster and inter-cluster contexts, fostering increased proximity among data points within the same cluster in the feature space while simultaneously ensuring substantial dissimilarity between points originating from disparate clusters. This enhancement not only serves to elevate clustering efficacy but also facilitates unsupervised clustering devoid of explicit labels. The modus operandi of our approach underscores the optimization of clustering performance through judicious minimization of contrastive loss between samples within identical clusters and those emanating from distinct clusters. This strategic objective engenders a more rationalized distribution of similarity scores, thereby facilitating the model's assimilation of generalized representations. Moreover, the introduction of a temperature parameter augments adaptability to the characteristics of the data distribution, thereby bolstering the overall robustness of the model. Rigorous experimental assessments, spanning 28 datasets, three classifiers, and three hard drive datasets, substantiate the superior performance of our proposed methodology relative to contemporary state-of-the-art approaches. These findings corroborate the efficacy and applicability of our COCL loss function in confronting clustering intricacies arising from imbalances in data distribution.

CRediT authorship contribution statement

Wu Chaomin: Methodology, Conceptualization, Investigation, Writing – original draft, Writing – review & editing. **Cheng Xu:** Methodology, Conceptualization, Software, Writing – review & editing. **Wang Hao:** Visualization, Writing – review & editing.

Declaration of competing interest

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

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.eswa.2025.127493.

Data availability

Data will be made available on request.

References

Ali, A. I. A., Rani, S. Sheeja., Pravija Raj, P. V., & Khedr, A. M. (2025). Hierarchical cluster-based IELM for financial distress prediction with imbalanced data. *Neural Computing and Applications*, *37*(5), 2925–2943.

Bagnall, A. J., Lines, J., Vickers, W., & Keogh, E. J. (2018). The UEA & UCR time series classification repository. [Online]. Available: www.timeseriesclassification.com.

Büttner, M., Schneider, L., Krasowski, A., Pitchika, V., Krois, J., Meyer-Lueckel, H., et al. (2024). Conquering class imbalances in deep learning-based segmentation of dental radiographs with different loss functions. *Journal of Dentistry*, *148*, Article 105063.

Cao, H., Li, X.-L., Woon, Y.-K., & Ng, S.-K. (2011). SPO: Structure preserving oversampling for imbalanced time series classification. In *2011 IEEE 11th international conference on data mining* (pp. 1008–1013).

Cao, H., Ng, K., Li, X. L., & Woon, Y.-K. (2013). Integrated oversampling for imbalanced time series classification. *IEEE Transactions on Knowledge and Data Engineering*, *25*(12), 2809–2822.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, *16*, 321–357.

Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020). A simple framework for contrastive learning of visual representations. In *International conference on machine learning* (pp. 1597–1607). PMLR.

Cheng, X., Shi, F., Liu, X., Zhao, M., & Chen, S. (2022). A novel deep class-imbalanced semi-supervised model for wind turbine blade icing detection. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(6), 2558–2570.

Dai, B., & Lin, D. (2017). Contrastive learning for image captioning. *Advances in Neural Information Processing Systems*, 30.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

Eldele, E., Ragab, M., Chen, Z., Wu, M., Kwoh, C. K., Li, X., et al. (2021). Time-series representation learning via temporal and contextual contrasting. arXiv preprint arXiv:2106.14112.

Gaudreault, J. G., Branco, P., & Gama, J. (2021). An analysis of performance metrics for imbalanced classification. In *International conference on discovery science* (pp. 67–77). Cham: Springer International Publishing.

Geng, Y., & Luo, X. (2019). Cost-sensitive convolutional neural networks for imbalanced time series classification. *Intelligent Data Analysis*, *23*(2), 357–370.

Hadsell, R., Chopra, S., & LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. In *2006 IEEE computer society conference on computer vision and pattern recognition* (pp. 1735–1742).

Han, H., Wang, W. Y., & Mao, B. H. (2005). Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. In *International conference on intelligent computing* (pp. 878–887).

Hao, S., Wang, Z., Alexander, A. D., Yuan, J., & Zhang, W. (2023). MICOS: Mixed supervised contrastive learning for multivariate time series classification. *Knowledge-Based Systems*, 260.

He, H., Bai, Y., Garcia, E. A., & Li, S. (2008). ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)* (pp. 1322–1328).

He, K., Fan, H., Wu, Y., Xie, S., & Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 9729–9738).

He, Q. Q., Siu, S. W. I., & Si, Y. W. (2023). Attentive recurrent adversarial domain adaptation with top-k pseudo-labeling for time series classification. *Applied Intelligence: The International Journal of Artificial Intelligence, Neural Networks, and Complex Problem-Solving Technologies*, *53*, 13110–13129.

Huang, X., Ye, Y., Xiong, L., Lau, R. Y., Jiang, N., & Wang, S. (2016). Time series k-means: A new k-means type smooth subspace clustering for time series data. *Information Sciences*, *367*, 1–13.

Ircio, J., Lojo, A., Lozano, J. A., & Mori, U. (2022). A multivariate time series streaming classifier for predicting hard drive failures [application notes]. *IEEE Computational Intelligence Magazine*, *17*(1), 102–114.

Ircio, J., Lojo, A., Mori, U., Malinowski, S., & Lozano, J. A. (2023). Minimum recall-based loss function for imbalanced time series classification. *IEEE Transactions on Knowledge and Data Engineering*, *35*(10), 10024–10034.

Karim, F., Majumdar, S., Darabi, H., & Chen, S. (2017). LSTM fully convolutional networks for time series classification. *IEEE Access*, *6*, 1662–1669.

Keogh, E., & Kasetty, S. (2002). On the need for time series data mining benchmarks: A survey and empirical demonstration. In *Proceedings of the eighth ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 102–111).

Keogh, E., & Ratanamahatana, C. (2005). Exact indexing of dynamic time warping. *Knowledge and Information Systems*, *7*, 358–386.

Keogh, E., et al. (2018). The UCR time series classification archive. [Online]. Available: www.cs.ucr.edu/eamonn/time_series_data/.

Le-Khac, P. H., Healy, G., & Smeaton, A. F. (2020). Contrastive representation learning: A framework and review. *IEEE Access*, *8*, 193907–193934.

Lee, H. K., Lee, J., & Kim, S. B. (2022). Boundary-focused generative adversarial networks for imbalanced and multimodal time series. *IEEE Transactions on Knowledge and Data Engineering*, *34*(9), 4102–4118.

Lei, Y., & Wu, Z. (2020). Time series classification based on statistical features. *EURASIP Journal on Wireless Communications and Networking*, 1–13.

Li, S., Cheng, X., Shi, F., Zhang, H., Dai, H., Zhang, H., et al. (2024). A novel robustness-enhancing adversarial defense approach to AI-powered sea state estimation for autonomous marine vessels. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*.

Mei, J., Liu, M., Wang, Y.-F., & Gao, H. (2016). Learning a mahalanobis distance-based dynamic time warping measure for multivariate time series classification. *IEEE Transactions on Cybernetics*, *46*(6), 1363–1374.

Meng, Q., Qian, H., Liu, Y., Cui, L., Xu, Y., & Shen, Z. (2023). MHCCL: Masked hierarchical cluster-wise contrastive learning for multivariate time series. *vol. 37*, In *Proceedings of the AAAI conference on artificial intelligence* (pp. 9153–9161). 8.

Pang, J. Z. F., Cao, H., & Tan, V. Y. F. (2013). MOGT: Oversampling with a parsimonious mixture of Gaussian trees model for imbalanced time-series classification. In *2013 IEEE international workshop on machine learning for signal processing* (pp. 1–6).

Radford, A., Metz, L., & Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434.

Raj, V., Magg, S., & Wermter, S. (2016). Towards effective classification of imbalanced data with convolutional neural networks. In *IAPR workshop on artificial neural networks in pattern recognition* (pp. 150–162).

Saeed, A., Grangier, D., & Zeghidour, N. (2021). Contrastive learning of general-purpose audio representations. In *ICASSP 2021-2021 IEEE international conference on acoustics, speech and signal processing* (pp. 3875–3879).

Shen, Y., Han, T., Yang, Q., Yang, X., Wang, Y., Li, F., et al. (2018). CS-CNN: Enabling robust and efficient convolutional neural networks inference for internet-of-things applications. *IEEE Access*, *6*, 13439–13448.

Theodossiou, P. T. (1993). Predicting shifts in the mean of a multivariate time series process: an application in predicting business failures. *Journal of the American Statistical Association*, *88*(422), 441–449.

Wang, D., Ding, N., Li, P., & Zheng, H. T. (2021). Cline: Contrastive learning with semantic negative examples for natural language understanding. arXiv preprint arXiv:2107.00440.

Wang, Z., Yan, W., & Oates, T. (2017). Time series classification from scratch with deep neural networks: A strong baseline. In *2017 international joint conference on neural networks*.

Xi, X., Keogh, E., Shelton, C., Wei, L., & Ratanamahatana, C. A. (2006). Fast time series classification using numerosity reduction. In *Proceedings of the 23rd international conference on machine learning* (pp. 1033–1040).

Yang, Y., Li, J., & Yang, Y. (2015). The research of the fast SVM classifier method. In *2015 12th international computer conference on wavelet active media technology and information processing* (pp. 121–124).

Yang, W., Yuan, J., & Wang, X. (2023). SFCC: Data augmentation with stratified Fourier coefficients combination for time series classification. *Neural Processing Letters*, *55*, 1833–1846.

Zhang, Z. (2018). Improved adam optimizer for deep neural networks. In *2018 IEEE/ACM 26th international symposium on quality of service*.

Zheng, Y., Liu, Q., Chen, E., Ge, Y., & Zhao, J. L. (2014). Time series classification using multi-channels deep convolutional neural networks. In *International conference on web-age information management* (pp. 298–310). Cham: Springer International Publishing.