

A Shapelet-based Framework for Unsupervised Multivariate Time Series Representation Learning

Zhiyu Liang
Harbin Institute of Technology
Harbin, China
zyliang@hit.edu.cn

Hongzhi Wang*
Harbin Institute of Technology
Harbin, China
wangzh@hit.edu.cn

Jianfeng Zhang
Huawei Noah's Ark Lab
Shenzhen, China
zhangjianfeng3@huawei.com

Zheng Liang
Harbin Institute of Technology
Harbin, China
lz20@hit.edu.cn

Chen Liang
Harbin Institute of Technology
Harbin, China
1190201818@stu.hit.edu.cn

Lujia Pan
Huawei Noah's Ark Lab
Shenzhen, China
panlujia@huawei.com

ABSTRACT

Recent studies have shown great promise in unsupervised representation learning (URL) for multivariate time series, because URL has the capability in learning generalizable representation for many downstream tasks without using inaccessible labels. However, existing approaches usually adopt the models originally designed for other domains (e.g., computer vision) to encode the time series data and rely on strong assumptions to design learning objectives, which limits their ability to perform well. To deal with these problems, we propose a novel URL framework for multivariate time series by learning time-series-specific shapelet-based representation through a popular contrasting learning paradigm. To the best of our knowledge, this is the first work that explores the shapelet-based embedding in the unsupervised general-purpose representation learning. A unified shapelet-based encoder and a novel learning objective with multi-grained contrasting and multi-scale alignment are particularly designed to achieve our goal, and a data augmentation library is employed to improve the generalization. We conduct extensive experiments using tens of real-world datasets to assess the representation quality on many downstream tasks, including classification, clustering, and anomaly detection. The results demonstrate the superiority of our method against not only URL competitors, but also techniques specially designed for downstream tasks. Our code has been made publicly available at <https://github.com/real2fish/CSL>.

PVLDB Reference Format:

Zhiyu Liang, Jianfeng Zhang, Chen Liang, Hongzhi Wang, Zheng Liang, and Lujia Pan. A Shapelet-based Framework for Unsupervised Multivariate Time Series Representation Learning. PVLDB, xx(xx): xxx-xxx, xx.
doi:xx.xxx/xx.xxx

PVLDB Artifact Availability:

The source code, data, and/or other artifacts have been made available at <https://github.com/real2fish/CSL>.

*Corresponding author.

This work is licensed under the Creative Commons BY-NC-ND 4.0 International License. Visit <https://creativecommons.org/licenses/by-nc-nd/4.0/> to view a copy of this license. For any use beyond those covered by this license, obtain permission by emailing info@vldb.org. Copyright is held by the owner/author(s). Publication rights licensed to the VLDB Endowment.

Proceedings of the VLDB Endowment, Vol. xx, No. xx ISSN 2150-8097.
doi:xx.xxx/xx.xxx

1 INTRODUCTION

Multivariate time series (MTS) generally describes a group of dependent variables evolving over time, each of which represents a monitoring metric (e.g., temperature or CPU utilization) of an entity (e.g., system or service). MTS data play a vital role in many practical scenarios, such as manufacturing, medicine, and finance [4, 24, 42].

While MTS are being increasingly collected from various applications, a particular challenge in modeling them is the lack of labels. Unlike images or text that usually contain human-recognizable patterns, label acquisition for time series is much more difficult, because the underlying state of these time-evolving signals can be too complicated even for the domain specialists [46]. For this reason, it has recently become a research focus to explore unsupervised (a.k.a. self-supervised) representation learning (URL) for MTS [10, 56, 59, 60]. *URL aims to train a neural network (called encoder) without accessing the labels to embed the data into feature vectors, by using carefully designed learning objectives to leverage the inherent structure of the raw data.* The learned representations (a.k.a. features or embeddings) can then be used for training models to solve a downstream analysis task using *little annotated data* compared to the traditional supervised methods [60]. And the features are more *general-purpose* since they can facilitate to several tasks.

Unfortunately, unlike in domains such as computer vision (CV)[8, 27, 29] or natural language processing (NLP) [12, 65], URL in the context of time series is still under-explored. MTS are typically continuous-valued data with high noise, diverse temporal patterns and varying semantic meanings, etc [3]. These unique complexities make advanced URL methods in the aforementioned domains difficult to perform well [10, 46]. Although several studies have attempted to fill this gap by considering the characteristics of time series, such as the time-evolving nature [46] and the multi-scale semantics [11, 59], *existing approaches can still be weak in learning well-performed representations* partly due to the following reasons.

First, the existing representation encoder designs are highly inspired by experiences in CV and NLP domains, which may not be well-suited for MTS. Specifically, convolutional neural network (CNN) [16, 49] and Transformer [51] are commonly-used encoders in recent studies [10, 11, 46, 56, 59, 60]. However, the encoders still face many difficulties when applying in MTS due to the lack of capability to deal with the characteristics of time series [32, 45, 66].

Second, some existing approaches rely on domain-specific assumptions, such as the neighbor similarity [11, 46] and the contextual consistency [59], thus are difficult to generalize to various scenarios. For instance, Franceschi et al. [11] and Tonekaboni et al. [46] assume that subsequences distant in time should be dissimilar, which can be easily violated in periodic time series [43].

To tackle the issues mentioned above, we explore the time-series-specific representation encoder without strong assumptions for URL. In particular, we consider the encoder based on a non-parametric time series analysis concept named shapelet [58], i.e. salient subsequence which is tailored to extract time series features from only important time windows to avoid the noises outside. The main reason is that the shapelet-based representation has shown superior performance in specific tasks such as classification [25, 33, 54] and clustering [61]. Besides, compared to the feature extracted from other neural networks such as CNN, the shapelet-based feature can be more intuitive to understand [58]. However, it has never been explored in the recently rising topic of URL for general-purpose representation. To fill this gap, we take the first step and propose to *learn shapelet-based encoder employing contrastive learning*, a popular paradigm that has shown success in URL [8, 10, 59, 64].

We highlight three challenges in learning high-quality and general-purpose shapelet-based representation. The first is how to design a shapelet-based encoder to capture diverse temporal patterns of various time ranges, considering that it is originally proposed to represent only a single shape feature, and exhaustive search or prior knowledge is needed to determine the encoding scale [5, 25, 61]. The second is how to design a URL objective to learn general information for downstream tasks through this shapelet-based encoder, which has never been studied. Last, while contrastive learning leverages the representation similarity of the augmentations of one sample [8] to learn the encoder, it remains an open problem to properly augment the time series to keep the similarity [46, 59].

To cope with these challenges, we propose a novel unsupervised MTS representation learning framework named *Contrastive Shapelet Learning (CSL)*. Specifically, we design a unified architecture that uses multiple shapelets with various (dis)similarity measures and lengths to jointly encode a sample, such that to capture diverse temporal patterns from short to long term. As shapelets of different lengths can separately embed one sample into different representation spaces that are complementary with each other, we propose a multi-grained contrasting objective to simultaneously consider the joint embedding and the representations at each time scale. In parallel, we design a multi-scale alignment loss to encourage the representations of different scales to achieve consensus. The basic idea is to automatically capture the varying semantics by leveraging the intra-scale and inter-scale dependencies of the shapelet-based embedding. Besides, we develop an augmentation library using diverse types of data augmentation methods to further improve the representation quality. To the best of our knowledge, CSL is the first general-purpose URL framework based on shapelets. The main contributions are summarized as follows:

- This paper studies how to improve the URL performance using time-series-specific shapelet-based representation, which has achieved success in specific tasks but has never been explored for the general-purpose URL.

- A novel framework is proposed that adopts contrastive learning to learn shapelet-based representations. A unified shapelet-based encoder architecture and a learning objective with multi-grained contrasting and multi-scale alignment are particularly designed to capture diverse patterns in various time ranges. A library containing various types of data augmentation methods is constructed to improve the representation quality.
- Experiments on tens of real-world datasets from various domains show that i) our learned representations are general to many downstream tasks, such as classification, clustering, and anomaly detection; ii) the proposed method outperforms existing URL competitors and can be comparable to (even better than) tailored techniques for classification and clustering. Additionally, we study the effectiveness of the key components proposed in CSL and the model sensitivity to the key parameters, demonstrate the superiority of CSL against the fully-supervised competitors on partially labeled data, and explain the shapelets learned by CSL. We also study our method in long time series representation and assess its running time.

2 RELATED WORK

There are two lines of research closely related to this paper:

Unsupervised MTS representation learning. Unlike in domains such as CV [8, 27, 29, 55] and NLP [12, 65], the study of URL in time series is still in its infancy.

Inspired by word representation [36], Franceschi et al. [11] adapts the triplet loss to time series to achieve URL. Similarly, Zerveas et al. [60] explores the utility of transformer [51] for URL due to the success of transformer in modeling natural language. Oord et al. [39] proposes to learn the representation by predicting the future in latent space. Eldele et al. [10] extends this idea by conducting both temporal and contextual contrasting to improve the representation quality. Instead of using prediction, Yue et al. [59] combines timestamp-level contrasting with contextual contrasting to achieve hierarchical representation. Tonekaboni et al. [46] assumes consistency between overlapping temporal neighborhoods to model dynamic latent states, while Yang and Hong [56] utilizes the consistency between temporal and spectral domains to enrich the representation. Although these methods have achieved improvements in representation quality, they still have limitations such as the lack of intuitions in encoder design and the dependency on specific assumptions, as discussed in Section 1.

Time series shapelet. The concept of shapelet is first proposed by Ye and Keogh [58] for supervised time series classification tasks. It focuses on extracting features in a notable time range to reduce the interference of noise, which is prevalent in time series.

In the early studies, shapelets are selected by enumerating subsequences of the training time series [5, 17, 38, 58], which suffers from non-optimal representation and high computational overhead [13]. To address these problems, a shapelet learning method is first proposed by Grabocka et al. [13], which directly learns the optimal shapelets through a supervised objective. After this study, many approaches [30, 33, 34, 54] have been proposed to improve the effectiveness and efficiency for classification. Except for supervised classification task, some works [47, 61, 62] employ shapelets for time series clustering and also show competitive performance.

ShapeNet [25] is a special work related to both URL and shapelet. However, it aims to “select” shapelets from existing candidates for MTS classification, while it just adopts a CNN-based URL method extended from [11] to assist the selection. It even contains a supervised feature selection step which uses the true labels. Instead, both our CSL and other URL methods target a different problem that is to automatically “learn” the new features not present in existing feature set without using labels to tackle more than one task.

In summary, although shapelet-based representation has been widely studied for classification and clustering tasks, it has never been explored for the unsupervised learning of general-purpose representations facilitating various tasks as our CSL.

3 PROBLEM STATEMENT

This section defines the key concept used in the paper. At first, we define the data type we are interested in, *multivariate time series*.

Definition 3.1 (Multivariate Time Series). *Multivariate time series (MTS) is a set of variables, each including observations ordered by successive time. Formally, we denote a multivariate time series sample with D variables (a.k.a. dimensions or channels) and T timestamps (a.k.a. length) as $\mathbf{x} \in \mathbb{R}^{D \times T}$, and a dataset containing N samples as $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\} \in \mathbb{R}^{N \times D \times T}$.*

Then, the problem that we are addressing, i.e., *unsupervised representation learning for MTS*, is formulated as follows.

Definition 3.2 (Unsupervised Representation Learning for MTS). *Given an MTS dataset X, the goal of unsupervised representation learning (URL) is to train a neural network model (encoder) $f : \mathbb{R}^{D \times T} \mapsto \mathbb{R}^{D_{repr}}$, such that the representation $\mathbf{z}_i = f(\mathbf{x}_i)$ can be informative for downstream tasks, e.g., classification and anomaly detection. Here unsupervised means that the labels of downstream tasks are unavailable when training f. To simplify the notation, we denote $Z = f(X) = \{\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_N\}$ in following sections.*

It is worthy to note that some works limit “unsupervised (representation) learning” to the unsupervised tasks (e.g., clustering [61]), so the competitors are only the unsupervised methods. Instead, the URL problem mentioned in this paper is to *learn the features that can not only tackle the unsupervised tasks, but also achieve comparable performance to the supervised competitors on the classification task*, which can be more general yet challenging.

4 METHODOLOGY

In this section, the proposed framework and all components are elaborated.

4.1 Overview

We illustrate the overview framework of the proposed contrastive shapelet learning (CSL) in Fig. 1. Given the input X , two data augmentation methods, denoted as $A'(\mathbf{x})$ and $A''(\mathbf{x})$, are randomly selected from a library (to be discussed later) to produce two correlated views of X as $X' = A'(\mathbf{x})$ and $X'' = A''(\mathbf{x})$, where $A'(\mathbf{x}) = \{A'(\mathbf{x}_1), \dots, A'(\mathbf{x}_N)\}$ and the same to $A''(\mathbf{x})$. Then these two views are fed into a time-series-specific encoder named Shapelet Transformer (ST), which embeds the samples into a latent space (see Section 4.2). CSL explores the representation in this latent space where different shapelets serve as the basis (see Section 4.3

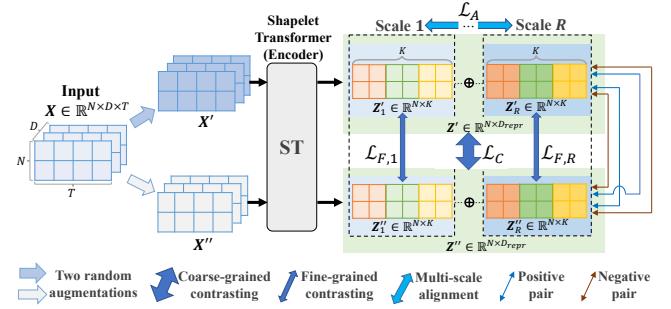


Figure 1: Overview framework of CSL.

and 4.4). We believe that our method is more general as it does not depend on task-specific assumptions like [11, 46, 59].

Formally, given X' and X'' , we have the shapelet-based representations as:

$$\begin{aligned} Z' &= f(X') \in \mathbb{R}^{N \times D_{repr}}, \\ Z'' &= f(X'') \in \mathbb{R}^{N \times D_{repr}}. \end{aligned} \quad (1)$$

Following the paradigm of contrastive learning [8, 10, 64], for each \mathbf{x}_i , the embedding \mathbf{z}'_i should be close to \mathbf{z}''_i whereas far away from \mathbf{z}''_j derived from other samples where $j \neq i$. The encoder is learned through maximizing the similarity of the positive pairs $(\mathbf{z}'_i, \mathbf{z}''_i)$ and minimizing the similarity of the negative pairs $(\mathbf{z}'_i, \mathbf{z}''_j)$. Note that using data augmentation to generate the positive pairs is a common way for contrastive learning which is required by most URL methods, including TS2Vec [59], TS-TCC [10], etc [56, 60]. Alternatively, T-Loss [11] and TNC [46] select subsequences as positive samples. Both augmented and sampled time series serve as the self-supervised signals in URL, which plays the similar role as the labels used by the supervised methods (e.g., OSCNN [45]).

Despite the success of contrastive learning in URL [10, 56, 59, 64], an open question is how to determine proper data augmentation methods to ensure representation similarity of positive samples [8], which could be data- and model-dependent [22]. It is beyond the scope of this paper to develop new augmentation techniques or augmentation selection algorithms. Instead, we construct a data augmentation library which contains diverse types of methods for the random selection at each training step (illustrated in Fig. 1), so that they can be complementary with each other to adapt to various time series data. The library consists of five well-established time series augmentation methods, including *jittering J(x)* that adds random noise to each observation, *cropping C(x)* that crops the time series into a randomly selected subsequence, *time warping TW(x)* that stretches or contracts the randomly selected subsequences, *quantizing Q(x)* that quantizes each observation to the nearest level, and *pooling P(x)* that reduces the temporal resolution using average pooling on each consecutive observations. We illustrate how they are performed using the running examples in Fig. 2, and we refer interested readers to [20, 48] for more details.

The encoder ST is designed to capture the patterns of different time scales using separated shapelets with different lengths. Thus, we propose a multi-grained contrasting objective to simultaneously perform contrastive learning on the shapelet-based embedding of every single scale (fine-grained contrasting) and the representations in the joint space $\mathbb{R}^{D_{repr}}$ regarding all scales (coarse-grained

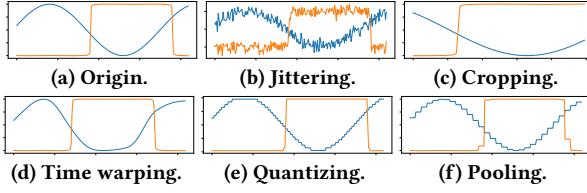


Figure 2: Illustration of the data augmentation methods using a two-dimensional time series. All methods are identically performed on each dimension of the original time series.

contrasting). Additionally, inspired by the consensus principle in multi-view learning [53], we design a multi-scale alignment term to encourage the features at different scales to achieve agreement.

In the rest of this section, we elaborate the key components of the proposed CSL framework, including *Shapelet Transformer*, *multi-grained contrasting*, and *multi-scale alignment*.

4.2 Shapelet Transformer

Shapelet is originally designed to extract representative shape features of univariate time series [58]. In this paper, we simply extend it to a more general multivariate case. Given a sample $\mathbf{x} \in \mathbb{R}^{D \times T}$, a multivariate shapelet $\mathbf{s} \in \mathbb{R}^{D \times L}$ ($L < T$) which has the same dimension D encodes \mathbf{x} using the Euclidean norm between \mathbf{s} and the best-matching subsequence relative to \mathbf{s} within \mathbf{x} , defined as:

$$dist(\mathbf{x}, \mathbf{s}) = \min_{t=1,2,\dots,T-L+1} \|\mathbf{x}[t, L] - \mathbf{s}\|_2 \in \mathbb{R}, \quad (2)$$

where $\mathbf{x}[t, L]$ denotes the subsequence of \mathbf{x} starting at timestamp t and lasting L steps. By taking \mathbf{s} as *trainable parameter*, we can directly learn the optimal shapelet like any neural network using the optimization algorithm such as stochastic gradient descent (SGD) [33]. However, *it is difficult to capture patterns beyond shapes for the original definition in Eq. (2)*, such as the spectral information in the frequency domain, which can limit the capability of the shapelet-based encoder. To address this problem, we extend the representation to a general form as:

$$g(\mathbf{x}, \mathbf{s}, d) = \text{agg}_d \sum_{t=1,2,\dots,T-L+1}^D d(\mathbf{x}^j[t, L], \mathbf{s}^j) \in \mathbb{R}, \quad (3)$$

where \mathbf{x}^j (\mathbf{s}^j) represents the series (shapelet) at j -th dimension, and agg_d is the aggregator that produces the result of d between the most similar pair of $(\mathbf{x}[i, L], \mathbf{s})$. $d(\cdot, \cdot)$ can be *any (dis)similarity measure* for equal-length series. Eq. (2) is obviously a special case of Eq. (3) when d is Euclidean norm, and g corresponds to 1-D convolution when d is the cross-correlation function [9].

Based on Eq. (3), we design a shapelet-based encoder named *Shapelet Transformer* (ST), which is shown in Fig. 3. To extract diverse temporal patterns, ST is a combination of multiple sub-modules with shapelets of R various lengths (scales) and M different (dis)similarity measures. The core idea comes from the observations that i) time series could possess both short-term and long-term patterns in practice [45, 57], and ii) different measures can be complementary with each other to produce more informative features [31]. However, our design is eventually different from existing approaches since we *simultaneously consider these two aspects in a unified shapelet-based architecture*.

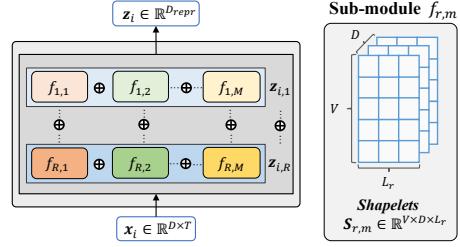


Figure 3: Architecture of Shapelet Transformer (ST)

We denote the sub-module with shapelets of length L_r (scale r) and (dis)similarity measure d_m as $f_{r,m}$. Each $f_{r,m}$ has V shapelets $s_{r,m,1}, \dots, s_{r,m,V}$ that separately embed the input. The outputs of all sub-modules for one sample are concatenated to jointly represent the sample. Formally, the encoder ST is defined as:

$$\begin{aligned} z_i &= f(\mathbf{x}_i) = z_{i,1} \oplus z_{i,2} \oplus \dots \oplus z_{i,R}, \\ z_{i,r} &= f_{r,1}(\mathbf{x}_i) \oplus f_{r,2}(\mathbf{x}_i) \oplus \dots \oplus f_{r,M}(\mathbf{x}_i), \\ f_{r,m}(\mathbf{x}_i) &= [f_{r,m,1}(\mathbf{x}_i), f_{r,m,2}(\mathbf{x}_i), \dots, f_{r,m,V}(\mathbf{x}_i)], \end{aligned} \quad (4)$$

where \oplus is the concatenation operator, $z_{i,r} \in \mathbb{R}^K$ ($K = MV$) denotes the representation of \mathbf{x}_i at scale r , and $f_{r,m,v}(\mathbf{x}_i) = g(\mathbf{x}_i, s_{r,m,v}, d_m)$.

Note that although extracting multi-scale features is a widely adopted idea for time series [45, 57], *the proposed Shapelet Transformer provides a simple yet effective way to achieve contrastive learning for the features of different scales (Section 4.3) and to maximize the agreement of the scales (Section 4.4)*, which we show essential for improving the performance (Section 5.3). The reason is that we simply concatenate the features encoded by each of the shapelets, and thus the representations of different scales can be separated from each other, while the features of existing multi-scale networks [45, 57] are usually fused through complicated layer-by-layer structures. Moreover, integrating multiple measures into the multi-scale shapelet-based architecture is also a simple yet effective idea for improving the URL performance (Section 5.3).

Shapelet Transformer is a unified architecture that can be flexibly changed by varying R , M , V , L_r , d_m if one has prior knowledge, such as the time scale of the MTS patterns. Considering that prior knowledge is not always easy to access, we introduce *a general configuration of the model structure*. Specifically, we fix R to a moderate value of 8 and adaptively set L_r as the evenly spaced numbers over $[0.1T, 0.8T]$, i.e., $L_r = r \times 0.1T$ ($r \in \{1, \dots, R\}$), to approximately match the patterns from short to long term. Three widely adopted (dis)similarity measures are considered, including *Euclidean distance* (d_1), *cosine similarity* (d_2) and *cross correlation* (d_3). As such, given the dimension D_{repr} of the output embedding, the encoder structure can be automatically determined.

4.3 Multi-grained Contrasting

After encoding the training samples, we employ contrastive learning, a popular paradigm that has achieved success in URL, to learn the Shapelet Transformer. The principle of contrastive learning is to pull close the positive pairs and push apart the negative pairs in the embedding space. In this paper, we adopt the InfoNCE loss [39] to separate positive from negative samples because it is one of the most popular loss functions in contrastive learning which has been widely shown effective [15, 39, 41], but the contrastive loss in any

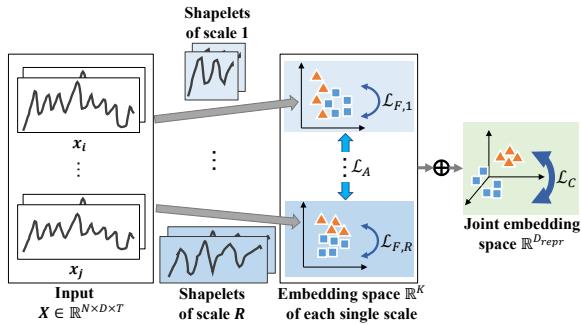


Figure 4: Illustration of multi-grained contrasting and multi-scale alignment. Display one shapelet at each scale for clarity.

other form can also fit in our framework. Given an embedding z_i of a sample x_i and a set Z that contains the embeddings of one positive sample x_i^+ and $N - 1$ negative samples of x_i , the contrastive loss is defined as:

$$\mathcal{L}_{IN}(z_i, Z) = -\log \frac{\exp(\text{sim}(z_i, z_i^+)/\tau)}{\sum_{z' \in Z} \exp(\text{sim}(z_i, z')/\tau)}, \quad (5)$$

where z_i^+ is the embedding of x_i^+ , $\text{sim}(u, v) = u \cdot v / \|u\| \|v\|$ is the cosine similarity, and τ is a temperature parameter that controls the strength of penalties on hard negative samples [39].

Recall that our Shapelet Transformer embeds one MTS sample into R separate embedding spaces which capture temporal features at different time scales. Thus, we propose a multi-grained contrasting objective that *explicitly considers not only the joint embedding space of the R scales, but also the latent space for each single scale*, as illustrated in Fig. 4. Specifically, we consider contrastive learning in the joint embedding space of all scales as the coarse-grained contrasting. The loss is defined as:

$$\mathcal{L}_C = \sum_{i=1}^N \mathcal{L}_{IN}(z'_i, Z''). \quad (6)$$

In parallel, the fine-grained contrasting is performed for embedding at each time scale $r \in \{1, \dots, R\}$, defined as:

$$\mathcal{L}_{F,r} = \sum_{i=1}^N \mathcal{L}_{IN}(z'_{i,r}, Z''_r). \quad (7)$$

The multi-grained contrastive loss is the sum of the coarse-fined and the fine-grained loss functions, i.e.,

$$\mathcal{L}_M = \mathcal{L}_C + \sum_{r=1}^R \mathcal{L}_{F,r}. \quad (8)$$

One may wonder *why the coarse-grained contrasting is required given that the optimal representations at each single scale seem to be learned using the fine-grained losses*. The intuition is that with only the fine-grained contrasting, the learning process could be dominated by some scales, i.e., the embeddings of some scales are well learned but the others are not, such that the joint embedding is not optimal. Thus, we design the coarse-grained loss to explicitly encourage feature similarity in the joint embedding space, so that to “balance” the representation quality of each scale to improve the final performance. We further clarify this issue in Section 5.3.

4.4 Multi-scale Alignment

As illustrated in Fig. 4, the joint embedding space $\mathbb{R}^{D_{repr}}$ ($D_{repr} = RK$) is composed by the space \mathbb{R}^K of each single scale. For one time series, the features of different scales are extracted using shapelets of different lengths, and thus can be seen as different views of the sample (similar to images of a 3-D object taken from different viewpoints [53]). From the perspective of multi-view learning, the representations of different lengths have not only complementary information but also consensus because the shapelets of different lengths can match some correlated time regions. Since we have leveraged the complementary information by using the joint embeddings, we propose to *enhance the consensus over different scales*, which can help to reduce the error rate of each view (scale) [53].

Inspired by Canonical Correlation Analysis (CCA) [1], we design a multi-scale alignment strategy to promote the consensus. The basic idea is to *encourage the embeddings of one sample for different scales to be maximally correlated*. Formally, given representations $Z_1, Z_2, \dots, Z_R \in \mathbb{R}^{N \times K}$ that have been *column-wise normalized*, the objective is minimizing the L_2 distance between each orthogonal features and their mean centers:

$$\begin{aligned} & \arg \min \sum_{r=1}^R \|Z_r - \bar{Z}\|_F^2, \\ & \text{s.t. } Z_r^T Z_r = I, \forall r \in \{1, 2, \dots, R\}, \end{aligned} \quad (9)$$

where $\bar{Z} = \frac{1}{R} \sum_{r=1}^R Z_r$ are the mean centers of the representations.

Eq. (9) has orthogonality constraints thus cannot be optimized end-to-end with other objective functions, which could limit its effectiveness [6]. Inspired by the soft decorrelation method proposed in [6], we *formulate the orthogonality constraints as a loss function to achieve end-to-end learning*. The core idea is to approximate the full-batch covariance matrix at each training step by stochastic incremental learning and encourage sparsity in the off-diagonal elements of the approximated covariance matrix using L_1 regularization. Consider the mini-batch representation $Z_B \in \mathbb{R}^{B \times K}$ of size B which has been batch normalized [21]. At t -th training step, we compute the mini-batch covariance matrix $C_B^t = \frac{1}{B-1} Z_B^T Z_B$ and an accumulative covariance matrix over each mini batch as:

$$C_{accu}^t = \alpha C_{accu}^{t-1} + C_B^t, \quad (10)$$

where $\alpha \in [0, 1]$ is a forgetting/decay rate, and C_{accu}^0 is initialized as all-zero matrix. As such, the full-batch covariance matrix $C^t = Z^T Z$ can be approximated by \hat{C}^t as:

$$\hat{C}^t = \frac{C_{accu}^t}{c^t}, \quad (11)$$

where $c^t = \alpha c^{t-1} + 1$ is a normalizing factor with $c^0 = 0$.

Given the approximate full-batch covariance matrix \hat{C}^t in Eq. (11), the orthogonality constraints $Z^T Z = I$ can be achieved in a soft procedure by minimizing an L_1 loss in the off-diagonal elements to penalize the correlation. Denote the element of \hat{C}^t at entry (i, j) as $\phi_{i,j}^t$. The soft orthogonality loss is defined as:

$$\mathcal{L}_S(Z) = \sum_{i=1}^K \sum_{j=i+1}^K |\phi_{i,j}^t|. \quad (12)$$

Table 1: Time and space complexity for CSL and the URL baselines.

Complexity	TS2Vec	T-Loss	TNC	TS-TCC	TST	CSL
Time	$O(NT(T+D^2))$	$O(NTD)$	$O(NTD)$	$O(NT(T+D))$	$O(NT(T+D^2))$	$O(NT^2D)$
Space	$O(TD+T^2)$	$O(D)$	$O(D)$	$O(TD+T^2)$	$O(TD+T^2)$	$O(TD)$

Based on Eq. (9) and (12), we define our multi-scale alignment loss on top of both Z' and Z'' as:

$$\mathcal{L}_A = \sum_{Z \in \{Z', Z''\}} \left(\sum_{r=1}^R \|Z_r - \bar{Z}\|_F^2 + \lambda_S \sum_{r=1}^R \mathcal{L}_S(Z_r) \right), \quad (13)$$

where λ_S controls the importance of the soft orthogonality loss.

It is noteworthy that the idea behind \mathcal{L}_A of aligning multi-scale information can be a general scheme for modeling MTS, which is worth further exploration in the future.

4.5 Summary and Complexity Analysis

Summary. Based on the discussion in Section 4.3 and Section 4.4, the total loss of the proposed CSL is defined as:

$$\mathcal{L} = \mathcal{L}_M + \lambda \mathcal{L}_A = \left(\mathcal{L}_C + \sum_{r=1}^R \mathcal{L}_{F,r} \right) + \lambda \mathcal{L}_A, \quad (14)$$

where λ controls the importance of multi-scale alignment.

The encoder (i.e., ST) $f(\mathbf{x}; \theta)$ is *unsupervisedly* trained by minimizing the loss \mathcal{L} using the popular back-propagation algorithm [7], where θ denotes trainable parameters which are updated within each mini batch. The learned encoder maps MTS into latent representation as $\mathbf{z}_i = f(\mathbf{x}_i; \theta)$, and \mathbf{z}_i is used for downstream tasks.

Complexity analysis. All data augmentation methods used in CSL take $O(BTD)$ time for MTS samples in a mini batch of size B [20]. The encoder ST takes $O(BL_s(T-L_s+1)DD_{repr})$ time for embedding the time series into representations, where L_s is the shapelet length. Recall that $RK = D_{repr}$. Therefore, both \mathcal{L}_C and $\sum_{r=1}^R \mathcal{L}_{F,r}$ can be computed in $O(B^2D_{repr})$ time and computing $\sum_{r=1}^R \|Z_r - \bar{Z}\|_F^2$ takes $O(BD_{repr})$ time. The computation of $\mathcal{L}_S(Z_r)$ takes $O(B^2K)$ time dominated by computing $C_B^t = \frac{1}{B-1} Z_B^T Z_B$. Since each training epoch has $\lfloor \frac{N}{B} \rfloor$ batches, the total time complexity for training CSL is $O(NTD + NL_s(T-L_s+1)DD_{repr} + NBD_{repr})$. Considering that B and D_{repr} are constants and we set L_s to be proportional to T , the time complexity can be simplified as $O(NT^2D)$. Similarly, the space complexity for the CSL training algorithm is $O(TD)$.

We compare the complexity of CSL to that of the advanced URL baselines in Table 1. CSL is theoretically more scalable than TS2Vec and TST when $D \gg T$, otherwise they have the same time complexity. CSL also has less space complexity than TS2Vec and TST. Compared to TS-TCC, the time complexity of CSL is the same or somewhat greater according to the relation between T and D , but CSL has less space complexity. T-Loss and TNC are more scalable in both time and space. However, the two methods rely on many sequential operations which can not be accelerated by GPUs. The experimental results in Section 5.8 show that they run much slower than CSL, TS2Vec, TST and TS-TCC with a considerably large input scale. Moreover, we show that our CSL, though primarily designed for improving the representation quality, also has faster training speed for real-world tasks, saying that it can achieve better performance with equal or less training time.

Table 2: Statistics of the 30 UEA datasets. All datasets are used for classification evaluation and the 12 subsets marked by * are used for clustering evaluation following [61].

Dataset	# Train	# Test	# Dim	Length	# Class
ArticularyWordRecognition (AW)*	275	300	9	144	25
AtrialFibrillation (AF)*	15	15	2	640	3
BasicMotions (BM)*	40	40	6	100	4
CharacterTrajectories (CT)	1422	1436	3	182	20
Cricket (Cr)	108	72	6	1197	12
DuckDuckGeese (DD)	50	50	1345	270	5
EigenWorms (EW)	128	131	6	17984	5
Epilepsy (Ep)*	137	138	3	206	4
EthanolConcentration (EC)	261	263	3	1751	4
ERing (ER)*	30	270	4	65	6
FaceDetection (FD)	5890	3524	144	62	2
FingerMovements (FM)	316	100	28	50	2
HandMovementDirection (HM)*	160	74	10	400	4
Handwriting (Ha)	150	850	3	152	26
Heartbeat (He)	204	205	61	405	2
InsectWingbeat (IW)	30000	20000	200	30	10
JapaneseVowels (JV)	270	370	12	29	9
Libras (Li)*	180	180	2	45	15
LSST (LS)	2459	2466	6	36	14
MotorImagery (MI)	278	100	64	3000	2
NATOPS (NA)*	180	180	24	51	6
PenDigits (PD)*	7494	3498	2	8	10
PEMS-SF (PE)*	267	173	963	144	7
PhonemeSpectra (PS)	3315	3353	11	217	39
RacketSports (RS)	151	152	6	30	4
SelfRegulationSCP1 (SR1)	268	293	6	896	2
SelfRegulationSCP2 (SR2)	200	180	7	1152	2
SpokenArabicDigits (SA)	6599	2199	13	93	10
StandWalkJump (SW)*	12	15	4	2500	3
UWaveGestureLibrary (UW)*	120	320	3	315	8

Table 3: Statistics of used anomaly detection datasets.

Dataset	# Entity	# Dim	Train length	Test length	Anomaly ratio (%)
SMAP	55	25	135183	427617	13.13
MSL	27	55	58317	73729	10.72
SMD	12	38	304168	304174	5.84
ASD	12	19	102331	51840	4.61

5 EXPERIMENTS

5.1 Experimental Setup

We conduct extensive experiments using total 34 real-world datasets to assess the representation quality of CSL. Three main tasks are investigated including the supervised classification task and the unsupervised clustering and anomaly detection tasks. Note that URL considers the MTS representation at the segment level, thus we work on segment-level anomaly detection (rather than observation-level [28, 44]). In specific, we consider series at each sliding window $\mathbf{x}_i[t, w]$, $t = 1, 2, \dots, N-w+1$ as an anomaly if it contains at least one anomalous observation. We train the popular SVM, K-means, and Isolation Forest on top of the learned representations to solve the three tasks respectively. We describe the datasets, baselines, implementations and evaluation metrics as follows.

Datasets. We use 34 MTS datasets with various sample size, dimension, series length, number of classes and application scenario to evaluate the representation quality on the three downstream tasks. We use the default train/test split for all datasets where only the training data are used for learning the encoder and task-specific models. The datasets used for each task are present below.

(1) *Classification*. To benchmark the result, we evaluate the performance of MTS classification on all 30 datasets of the popular UEA archive [3]. These data are collected from various domains, e.g., human action recognition, Electrocardiography monitoring and audio classification. The dataset statistics is present in Table 2.

(2) *Clustering*. Following a recent work of multivariate time series clustering [61], we evaluate the clustering performance using 12 UEA subsets which are highly heterogeneous in train/test size, length, and the number of dimensions and classes. The statistics of these 12 datasets are shown in Table 2 (marked by *).

(3) *Anomaly Detection*. Four recently published datasets collected from several challenging real-world applications are used for anomaly detection. Soil Moisture Active Passive satellite (SMAP) and Mars Science Laboratory rover (MSL) are two spacecraft anomaly detection datasets from NASA [18]. Server Machine Dataset (SMD) is a 5-week-long dataset collected by [44] from a large Internet company. Application Server Dataset (ASD) is a 45-day-long MTS characterizing the status of the servers recently collected by [28]. Following [28], for SMD, we use the 12 entities that do not suffer concept drift for evaluation. Table 3 shows the dataset statistics.

Baselines. We use 21 baselines for comparison, which are categorized into two groups:

(1) *URL methods*. We compare our CSL with 5 URL baselines specially designed for time series, including TS2Vec [59], T-Loss [11], TNC [46], TS-TCC [10], and TST [60]. All URL competitors are evaluated in the same way as CSL for a fair comparison. More details of these methods are discussed in Section 2.

(2) *Task-specific methods*. We also include baselines tailored for downstream tasks. We select outstanding approaches for classification, containing the most popular baseline DTWD which adopts the one-nearest-neighbor classifier with dynamic time warping as the distance metric [3] and five supervised techniques, including the RNN-based MLSTM-FCNs [23], the attentional prototype-based TapNet [63], the shapelet-based ShapeNet [25], and the CNN-based OSCNN [45] and DSN [52]. To avoid an unfair comparison, we let outside the ensemble methods like [31]. Recall that the supervised classification methods use the true labels to learn the features, which is benchmarked against the data augmentation or sampling in URL. Thus, the comparison is fair without further applying the data augmentation methods of CSL on the baseline approaches.

We consider six advanced clustering baselines including the dimension reduction-based MC2PCA [26] and TCK [35], the distance-based m-kAVG+ED and m-kDBA [40], the deep learning-based DeTSEC [19], and the shapelet-based MUSLA [61]. In addition, we design ShapeNet-Clustering (SN-C), an adaption of the classification baseline ShapeNet which is also based on the shapelets. In SN-C, we dismiss the supervised feature selection of ShapeNet and use K-means rather than SVM upon the features for clustering.

Since no evaluation is reported on the anomaly detection datasets under the segment-level setting, we develop 2 baselines on top of the raw MTS using also Isolation Forest for a fair comparison. The models take the observations either at each timestamp (denoted as IF-p) or within each sliding window (denoted as IF-s) as the input. Similarly, we also adapt ShapeNet for anomaly detection (SN-AD) by dismissing the supervised feature selection of ShapeNet and using Isolation Forest upon the shapelet-transformed features.

Implementations. We implement the CSL model using PyTorch 1.10.2 and run all experiments on a Ubuntu machine with Tesla V100 GPU. The SVM, K-means, and Isolation Forest are implemented using Scikit-learn 1.1.1 and the data augmentation methods are implemented using tsaug [20] with default parameters. Most of the hyper-parameters of CSL are set to fixed values for all experiments without tuning. We adopt SGD optimizer to learn the ST encoder. The learning rate is set to 0.01. Batchnorm is applied after the encoding. We set $\alpha = 0.5$ in soft orthogonality and $\lambda = 0.01$, $\lambda_S = 1$ in loss functions. The batch size is set to 8 for all UEA datasets and 256 for the anomaly detection datasets. The temperature τ is selected from $\{0.1, 0.01, 0.001\}$ by cross validation for the UEA datasets and is fixed to 0.1 for the anomaly detection datasets. Following previous works [59, 61], the embedding dimension is fixed to $D_{repr} = 320$ for classification and is chosen from $\{80, 240, 320\}$ for clustering for a fair comparison. On anomaly detection tasks, we set D_{repr} to 240, 320, 48, 32 for SMAP, MSL, SMD, and ASD respectively.

We reproduce the URL baselines using the open source code from the authors' implementations with the recommended configurations. The results of the classification baselines and the task-specific clustering baselines are taken from the published papers [3, 25, 45, 52, 59, 61]. Other results are based on our reproduction.

Metrics. Standard metrics are employed to evaluate the performance of the downstream tasks. We utilize Accuracy (Acc) [3] in classification tasks. Clustering results are evaluated using Rand Index (RI) and Normalized Mutual Information (NMI) [61, 62]. And F1-score is adopted for anomaly detection [14, 28].

5.2 Main Results

Table 4, 5 and 6 summarize the results on classification, clustering, and anomaly detection tasks. We report the average ranking (AR) of algorithms on each dataset, and count the number of datasets in which the CSL wins/ties/loses (W/T/L) the counterparts in the one-versus-one comparisons. The Wilcoxon rank test's p-values (p-val) are employed to quantitatively evaluate the significance.

In summary, the proposed CSL outperforms the URL competitors on most of the tasks and datasets, achieving the best overall performance. Moreover, CSL can achieve performance comparable to the approaches customized for classification and clustering. The results show the excellent ability of CSL in unsupervised learning of high-quality and general-purpose MTS representation. Below we discuss the results in detail for each task.

Classification. As shown in Table 4, CSL achieves competitive performance on most of the datasets. It has the highest average accuracy and accuracy ranking. Specifically, among the 30 datasets, CSL achieves the best accuracy in 21 of them if compared to URL methods only, and the highest accuracy in 12 of them (best in all algorithms) if all methods are considered. In the one-versus-one comparison, CSL outperforms all URL competitors in terms of the number of wins on the datasets. These results are in line with our expectations, as shapelets are originally designed to extract time series patterns that can effectively distinguish different classes. CSL further enhances the advantages of shapelets by jointly using the shapelets of different lengths and multiple (dis)similarity measures, and by using a novel objective for model training. To our surprise, CSL achieves better overall accuracy than the fully supervised counterparts. Compared to the supervised learning methods and based

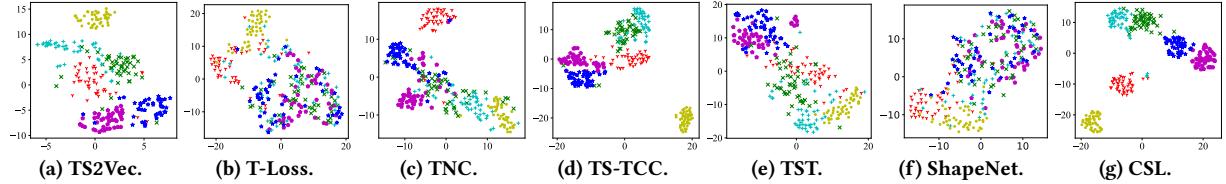


Figure 5: Two-dimensional t-SNE [50] visualization of the unsupervised learned representation for ERing test set. Classes are distinguishable using their respective marker shapes and colors.

Table 4: Performance comparison on MTS classification. The best results among URL methods are highlighted in bold and \dagger indicates the best among all competitors. The underlined value indicates significant difference under a statistical level of 0.05.

Dataset	Tailored Classification Approaches						Unsupervised Representation Learning + Classifier					
	DTWD	MLSTM-FCNs	TapNet	ShapeNet	OSCNN	DSN	TS2Vec	T-Loss	TNC	TS-TCC	TST	CSL
AW	0.987	0.973	0.987	0.987	0.988	0.984	0.987	0.943	0.973	0.953	0.977	0.990 \dagger
AF	0.200	0.267	0.333	0.400	0.233	0.067	0.200	0.133	0.133	0.267	0.067	0.533 \dagger
BM	0.975	0.950	1.000 \dagger	1.000 \dagger	1.000 \dagger	1.000 \dagger	0.975	1.000 \dagger	0.975	1.000 \dagger	0.975	1.000 \dagger
CT	0.989	0.985	0.997	0.980	0.998 \dagger	0.994	0.995	0.993	0.967	0.985	0.975	0.991
Cr	1.000 \dagger	0.917	0.958	0.986	0.993	0.989	0.972	0.972	0.958	0.917	1.000 \dagger	0.994
DD	0.600	0.675	0.575	0.725 \dagger	0.540	0.568	0.680	0.650	0.460	0.380	0.620	0.380
EW	0.618	0.504	0.489	0.878 \dagger	0.414	0.391	0.847	0.840	0.840	0.779	0.748	0.779
Ep	0.964	0.761	0.971	0.987	0.980	0.999 \dagger	0.964	0.971	0.957	0.957	0.949	0.986
ER	0.133	0.133	0.133	0.133	0.882	0.922	0.874	0.133	0.852	0.904	0.874	0.967 \dagger
EC	0.323	0.373	0.323	0.312	0.241	0.245	0.308	0.205	0.297	0.285	0.262	0.498 \dagger
FD	0.529	0.545	0.556	0.602	0.575	0.635 \dagger	0.501	0.513	0.536	0.544	0.534	0.593
FM	0.530	0.580	0.530	0.580	0.568	0.492	0.480	0.580	0.470	0.460	0.560	0.59 \dagger
HM	0.231	0.365	0.378	0.338	0.443 \dagger	0.373	0.338	0.351	0.324	0.243	0.243	0.432
Ha	0.286	0.286	0.357	0.451	0.668 \dagger	0.337	0.515	0.451	0.249	0.498	0.225	0.533
He	0.717	0.663	0.751	0.756	0.489	0.783 \dagger	0.683	0.741	0.746	0.751	0.746	0.722
IW	N/A	0.167	0.208	0.250	0.667 \dagger	0.386	0.466	0.156	0.469	0.264	0.105	0.256
JV	0.949	0.976	0.965	0.984	0.991 \dagger	0.987	0.984	0.989	0.978	0.930	0.978	0.919
Li	0.870	0.856	0.850	0.856	0.950	0.964 \dagger	0.867	0.883	0.817	0.822	0.656	0.906
LS	0.551	0.373	0.568	0.590	0.413	0.603	0.537	0.509	0.595	0.474	0.408	0.617 \dagger
MI	0.500	0.510	0.590	0.610 \dagger	0.535	0.574	0.510	0.580	0.500	0.610 \dagger	0.500	0.610 \dagger
NA	0.883	0.889	0.939	0.883	0.968	0.978 \dagger	0.928	0.917	0.911	0.822	0.850	0.878
PE	0.711	0.699	0.751	0.751	0.760	0.801	0.682	0.676	0.699	0.734	0.740	0.827 \dagger
PD	0.977	0.978	0.980	0.977	0.986	0.987	0.989	0.981	0.979	0.974	0.56	0.990 \dagger
PS	0.151	0.110	0.175	0.298	0.299 \dagger	0.320	0.233	0.222	0.207	0.252	0.085	0.255
RS	0.803	0.803	0.868	0.882 \dagger	0.877	0.862	0.855	0.855	0.776	0.816	0.809	0.882 \dagger
SR1	0.775	0.874 \dagger	0.652	0.782	0.835	0.717	0.812	0.843	0.799	0.823	0.754	0.846
SR2	0.539	0.472	0.550	0.578 \dagger	0.532	0.464	0.578 \dagger	0.539	0.550	0.533	0.550	0.494
SA	0.963	0.990	0.983	0.975	0.997 \dagger	0.991	0.988	0.905	0.934	0.970	0.923	0.990
SW	0.200	0.067	0.400	0.533	0.383	0.387	0.467	0.333	0.400	0.333	0.267	0.667 \dagger
UW	0.903	0.891	0.894	0.906	0.927 \dagger	0.916	0.906	0.875	0.759	0.753	0.575	0.922
Avg (excl. IW)	0.650	0.637	0.673	0.714	0.706	0.701	0.712	0.675	0.677	0.682	0.635	0.751 \dagger
Avg (incl. IW)	N/A	0.621	0.657	0.699	0.704	0.691	0.704	0.658	0.670	0.668	0.617	0.735 \dagger
AR (URL)	/	/	/	/	/	/	2.82	3.52	4.02	3.87	4.70	2.08
AR (All)	8.17	8.10	6.00	4.68	4.53	4.98	5.87	6.97	8.10	7.82	9.02	3.77 \dagger
W/T/L	25/0/5	25/1/4	23/1/6	18/3/9	17/1/12	18/1/11	23/0/7	22/1/7	23/0/7	22/4/4	25/0/5	/
p-val	0.0003	0.0003	0.0038	0.1075	0.4174	0.2177	0.0316	0.0058	0.0066	0.0002	0.0002	/

on the Wilcoxon rank test, our CSL has surpassed MSLTM-FCNs and TapNet and is on par with ShapeNet, OSCNN, and DSN, showing that CSL has reached a comparable level to supervised learning. This implies that class-specific features can be learned from the inherent structure of the data without supervised information, thus labels are only needed for classifier training. Furthermore, we observe that CSL performs poorly on DuckDuckGeese (DD), which has a very high dimension of 1345 (see Table 2). This may indicate a relative weakness of CSL in dealing with high-dimensional MTS, which is a possible direction to further improve our method. **Clustering.** The results of the clustering tasks are shown in Table 5. CSL outperforms all the other competitors except MUSLA. We note that the best performance for most of the datasets is achieved by either CSL or MUSLA, which are both based on time series shapelet methods. This result shows the superiority of shapelet

features for the MTS clustering tasks. CSL outperforms MUSLA in terms of average ranking, the number of best performances, and the number of wins in one-versus-one comparisons, while slightly underperforming MUSLA in terms of average RI and NMI. Overall, there is no statistically significant difference between these two methods. We would like to emphasize that MUSLA is a specialized clustering method, while our CSL is a generic URL algorithm that can be used for a variety of downstream tasks. MUSLA also relies on exhaustive search or prior knowledge to determine the length of shapelets, while CSL can achieve comparable performance without any effort in this regard. Besides, we notice that SN-C has almost the worst overall performance, which indicates that the URL-based shapelet selection method of ShapeNet which is customized for classification cannot be well generalized to the clustering problem.

Table 5: Performance comparison on MTS clustering. The best results among URL methods are highlighted in bold, and \dagger indicates the best among all competitors. The underlined value indicates significant difference under a statistical level of 0.05.

Dataset	Metric	Tailored Clustering Approaches						Unsupervised Representation Learning + Clustering						
		MC2PCA	TCK	m-kAVG+ED	m-kDBA	DeTSEC	MUSLA	SN-C	TS2Vec	T-Loss	TNC	TS-TCC	TST	CSL
AW	RI	0.989	0.973	0.952	0.934	0.972	0.977	0.936	0.980	0.975	0.938	0.946	0.978	0.990 \dagger
	NMI	0.934	0.873	0.834	0.741	0.792	0.838	0.492	0.880	0.842	0.565	0.621	0.866	0.942 \dagger
AF	RI	0.514	0.552	0.705	0.686	0.629	0.724	0.410	0.465	0.469	0.518	0.469	0.444	0.743 \dagger
	NMI	0.514	0.191	0.516	0.317	0.293	0.538	0.213	0.080	0.149	0.147	0.164	0.171	0.587 \dagger
BM	RI	0.791	0.868	0.772	0.749	0.717	1.000 \dagger	0.836	0.854	0.936	0.719	0.856	0.844	1.000 \dagger
	NMI	0.674	0.776	0.543	0.639	0.800	1.000 \dagger	0.736	0.820	0.871	0.394	0.823	0.790	1.000 \dagger
Ep	RI	0.613	0.786	0.768	0.777	0.840	0.816	0.695	0.706	0.705	0.650	0.736	0.718	0.873 \dagger
	NMI	0.173	0.534	0.409	0.471	0.346	0.601	0.250	0.312	0.306	0.156	0.451	0.357	0.705 \dagger
ER	RI	0.756	0.772	0.805	0.775	0.770	0.841	0.734	0.925	0.885	0.764	0.821	0.867	0.968 \dagger
	NMI	0.336	0.399	0.400	0.406	0.392	0.722	0.349	0.775	0.672	0.346	0.478	0.594	0.906 \dagger
HM	RI	0.627	0.635	0.697	0.685	0.628	0.719 \dagger	0.636	0.609	0.599	0.613	0.608	0.607	0.651
	NMI	0.067	0.103	0.168	0.265	0.112	0.398 \dagger	0.125	0.044	0.034	0.051	0.053	0.039	0.175
Li	RI	0.892	0.917	0.911	0.913	0.907	0.941 \dagger	0.855	0.904	0.922	0.896	0.881	0.886	0.941 \dagger
	NMI	0.577	0.620	0.622	0.602	0.724	0.353	0.542	0.654	0.464	0.373	0.400	0.761 \dagger	
NA	RI	0.882	0.833	0.853	0.876	0.714	0.976 \dagger	0.754	0.817	0.836	0.700	0.792	0.809	0.876
	NMI	0.698	0.679	0.643	0.643	0.043	0.855 \dagger	0.313	0.523	0.558	0.222	0.513	0.565	0.657
PE	RI	0.424	0.191	0.817	0.755	0.806	0.892 \dagger	0.730	0.765	0.746	0.763	0.789	0.726	0.858
	NMI	0.011	0.066	0.491	0.402	0.425	0.614 \dagger	0.217	0.290	0.102	0.278	0.331	0.026	0.537
PD	RI	0.929	0.922	0.935	0.881	0.885	0.946	0.845	0.941	0.936	0.873	0.857	0.818	0.950 \dagger
	NMI	0.713	0.693	0.738	0.605	0.563	0.826 \dagger	0.273	0.776	0.749	0.537	0.339	0.090	0.822
SW	RI	0.591	0.762	0.733	0.695	0.733	0.771 \dagger	0.467	0.410	0.410	0.457	0.589	0.467	0.724
	NMI	0.350	0.536	0.559	0.466	0.556	0.609 \dagger	0.188	0.213	0.213	0.193	0.187	0.248	0.554
UW	RI	0.883	0.913	0.920	0.893	0.879	0.913	0.795	0.865	0.893	0.817	0.796	0.779	0.927 \dagger
	NMI	0.570	0.710	0.713	0.582	0.558	0.728	0.233	0.511	0.614	0.322	0.215	0.244	0.731 \dagger
Avg	RI	0.741	0.760	0.822	0.801	0.790	0.876 \dagger	0.724	0.770	0.776	0.726	0.762	0.745	0.875
	NMI	0.468	0.515	0.553	0.513	0.457	0.704 \dagger	0.312	0.480	0.480	0.306	0.379	0.366	0.698
AR (URL)		/	/	/	/	/	/	/	3.33	3.56	4.71	4.06	4.33	1.00 \dagger
AR (All)		7.83	6.15	5.23	6.46	7.35	2.17	10.60	7.38	7.12	10.62	8.98	9.19	1.92 \dagger
W/T/L		22/0/2	22/0/2	21/0/3	21/1/2	22/0/2	12/3/9	24/0/0	24/0/0	24/0/0	24/0/0	24/0/0	24/0/0	/
p-val		0.0000	0.0000	0.0000	0.0002	0.0000	0.9169	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	/

Table 6: Performance comparison on MTS anomaly detection. w represents the length of the sliding window and the best results are highlighted in bold. The underlined value indicates significant difference under a statistical level of 0.05.

Dataset	w	IF-s	IF-p	SN-AD	TS2Vec	T-Loss	TNC	TS-TCC	TST	CSL
SMAP	25	0.1040	0.2146	0.2408	0.2680	0.3479	0.3111	0.3383	0.2279	0.4088
	50	0.0982	0.2090	0.2253	0.3096	0.3834	0.3089	0.3662	0.2399	0.3989
	75	0.0377	0.2149	0.2195	0.2365	0.3806	0.3133	0.3578	0.2383	0.3964
	100	0.0890	0.2166	0.2744	0.2834	0.3862	0.3218	0.3501	0.2511	0.4049
MSL	25	0.0212	0.2235	0.2408	0.1557	0.2300	0.2097	0.2420	0.2104	0.3312
	50	0.0160	0.2375	0.2571	0.1422	0.2563	0.2485	0.2744	0.2292	0.3725
	75	0.0092	0.2522	0.2796	0.1588	0.2661	0.2513	0.2898	0.2474	0.3813
	100	0.0077	0.2653	0.2907	0.1753	0.2771	0.2629	0.3074	0.2645	0.4033
SMD	25	0.2453	0.1664	0.1807	0.2035	0.1685	0.1754	0.1870	0.1394	0.2723
	50	0.2666	0.1799	0.1981	0.2345	0.1955	0.1867	0.2140	0.1416	0.2777
	75	0.2715	0.1963	0.1953	0.2659	0.2211	0.2070	0.2327	0.1704	0.2782
	100	0.2912	0.2152	0.2101	0.2846	0.2472	0.2291	0.2454	0.1859	0.2784
ASD	25	0.1916	0.1878	0.1403	0.2978	0.3223	0.1591	0.1938	0.1351	0.3272
	50	0.2149	0.2368	0.2084	0.3412	0.3504	0.1812	0.2324	0.1933	0.3799
	75	0.2391	0.2782	0.2488	0.3655	0.3544	0.2367	0.2844	0.2398	0.4094
	100	0.2585	0.3193	0.3062	0.3735	0.3449	0.2945	0.2679	0.2926	0.4349
Avg F1		0.1476	0.2558	0.2323	0.2560	0.2957	0.2436	0.2740	0.2129	0.3597
AR		6.69	6.38	5.63	4.56	3.50	6.13	3.63	7.38	1.13
W/T/L		15/0/1	16/0/0	16/0/1	16/0/0	16/0/0	16/0/0	16/0/0	16/0/0	/
p-val		0.0002	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	/

Anomaly Detection. In Table 6, we can see that CSL outperforms the baselines in every setting, except for SMD with a window length of 100, where CSL is slightly inferior to IF-s and TS2Vec. This may indicate that these two algorithms are more effective in detecting outliers with long SMD windows. For each dataset, the performance of all methods tend to improve as the sliding window size increases, because larger windows allow more normal observations to be seen to better detect the outliers. CSL achieves superior performance on the MSL dataset, outperforming the second-best TS-TCC by more

than 30% on each window size. Although the difference in performance is not as large on the other three datasets, CSL is almost always the best and significantly outperformed each competitor. This indicates that CSL has an outstanding ability to identify anomalies. We also observe that the second-best method is different for each dataset, with T-Loss on SMAP, TS-TCC on MSL, and IF-s on SMD, while TS2Vec and T-Loss are the second-best methods with about the same performance on ASD. The reason may be that these URL algorithms are developed based on specific assumptions that may fail in other domains. In contrast, CSL exhibits more general capabilities. The variant of ShapeNet, i.e., SN-AD, does not achieve competitive performance like in classification, showing again the limitation of ShapeNet in terms of task generality.

Finally, we visualize the unsupervised learned representation of ERing test data using t-SNE [50]. We compare CSL with the five URL baselines and the variant of ShapeNet which excludes the supervised feature selection. As Fig. 5 shows, the representation learned by the proposed CSL forms more separated clusters, which also suggests representation of lower entropy. This explains why CSL can outperform the competitors on downstream analysis tasks.

5.3 Ablation Study

To validate the effectiveness of the key components in CSL, we conduct ablation studies using classification tasks on all 30 UEA datasets. Due to space limitations, only the statistical results are reported here. The best value in a comparison is highlighted in bold and underlining indicates a significant difference under a statistical level of 0.05. The results are discussed as follows.

Table 7: Effectiveness of multi-scale shapelets.

Statistic	Short scale only	Long scale only	Better scale	Both (CSL)
Avg Acc/AR	0.710/2.88	0.687/2.98	0.723/2.12	0.735/2.02
W/T/L	19/4/7	21/3/6	13/6/11	/
p-val	<u>0.0054</u>	<u>0.0001</u>	0.1791	/

Table 8: Effectiveness of diverse (dis)similarity measures.

Statistic	Euclidean only	Cosine only	Cross only	All (CSL)
Avg Acc/AR	0.652/3.33	0.698/2.93	0.709/2.05	0.735/1.68
W/T/L	25/4/1	24/1/5	17/2/11	/
p-val	<u>0.0000</u>	<u>0.0001</u>	<u>0.0426</u>	/

Table 9: Effectiveness of multi-grained contrasting and multi-scale alignment.

Statistic	w/o \mathcal{L}_C	w/o \mathcal{L}_F	w/o \mathcal{L}_A	All (CSL)
Avg Acc/AR	0.720/2.38	0.708/3.33	0.709/2.92	0.735/1.37
W/T/L	20/5/5	27/3/0	26/4/0	/
p-val	<u>0.0006</u>	<u>0.0000</u>	<u>0.0000</u>	/

Effectiveness of components in Shapelet Transformer. There are two major designs within the Shapelet Transformer, including using the shapelets of different scales (lengths) and the diverse dis(similarity) measures. As they improve the representation in orthogonal directions, we assess their effectiveness individually.

(1) *Multi-scale shapelets.* ST contains shapelets ranging from short to long. Here we compare CSL with its three variants: *short scale only* (where the shapelet length ranges from $0.1T$ to $0.4T$), *long scale only* (from $0.5T$ to $0.8T$) and the *better of the two*. To make a fair comparison, we fix the embedding dimension D_{repr} and the number of scales R for all experiments. The results are shown in Table 7. Both the short- and long-scale variants perform much worse than CSL. Even the best of the two variants still performs slightly worse than CSL. These results demonstrate the necessity of using shapelets with a wider range of time scales.

(2) *Diverse (dis)similarity measures.* To investigate the role of the dis(similarity) measures in the Shapelet Transformer, we compare our CSL with its three variants, i.e., separately using one measure of the *Euclidean norm*, *cosine similarity*, and *cross correlation*. The results are summarized in Table 8. We can see that the cross correlation is the best performer among the three single measures, while the Euclidean norm variant using the original definition of shapelet is the worst. This validates our hypothesis that the Euclidean norm-based shapelet has limitations in representing time series. All three variants perform much worse than CSL with the p-values less than 0.05. This shows the need to combine the different types of measures in the shapelet-based MTS representation.

Effectiveness of components in loss function. There are three terms in our loss function, i.e., *coarse-grained contrastive loss* \mathcal{L}_C , *fine-grained contrastive loss* $\mathcal{L}_F = \sum_{r=1}^R \mathcal{L}_{F,r}$, and *multi-scale alignment loss* \mathcal{L}_A . We investigate the effect size of each term by removing them one by one. As we can see in Table 9, CSL is significantly better than the variant without the term \mathcal{L}_F or \mathcal{L}_A , which proves the importance of these two components. In contrast, removing the coarse-grained loss \mathcal{L}_C has the least impact. This may imply that, when the representations on each time scale have been sufficiently trained and aligned, the joint version is already near-optimal, thus the coarse-grained contrasting can no longer lead to a huge (but still statistically significant) improvement like the other two terms.

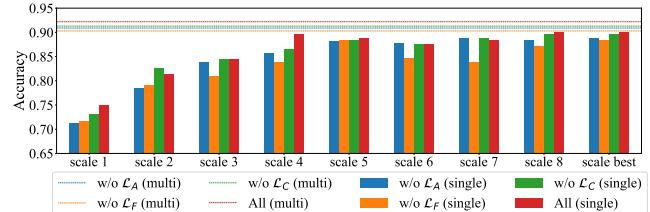


Figure 6: Study of the multi-grained contrasting and the multi-scale alignment on UWaveGestureLibrary. Dashed line corresponds to the joint embedding in $\mathbb{R}^{D_{repr}}$ with multiple scales and bar corresponds to embedding at each single scale.

Table 10: Effectiveness of the data augmentation library.

Statistic	w/o $J(x)$	w/o $C(x)$	w/o $TW(x)$	w/o $Q(x)$	w/o $P(x)$	All (CSL)
Avg Acc/AR	0.715/3.63	0.716/3.70	0.707/4.52	0.712/3.88	0.718/3.28	0.735/1.98
W/T/L	19/7/4	21/7/2	24/3/3	23/5/2	19/7/4	/
p-val	<u>0.0004</u>	<u>0.0003</u>	<u>0.0001</u>	<u>0.0001</u>	<u>0.0006</u>	/

We further explore how multi-grained contrasting and multi-scale alignment work using a case study in Fig. 6. We find that removing \mathcal{L}_F decreases the representation quality of every single scale (the orange bar), and thus has a great negative impact on the joint embedding (the orange line). Similar phenomena can be observed for \mathcal{L}_A . It indicates that \mathcal{L}_F and \mathcal{L}_A improve the final performance through improving the quality of each scale. Compared to the variants without \mathcal{L}_C (the green bar), the representation quality of each single scale is *balanced* with the loss (the red bars), saying that the quality of scale 1, 4, 5 and 8 is improved, while the quality of scale 2 and 7 is a little decreased. As a result, the joint embedding learned using \mathcal{L}_C (the red line) is better than that without the loss (the green line). It validates the hypothesis in Section 4.3 that \mathcal{L}_C can coordinate the multiple scales to improve the joint embedding.

From the above exhaustive analysis, we can conclude that all the components included in the loss function of CSL are necessary.

Effectiveness of the data augmentation library. We remove the methods in the data augmentation library one by one to evaluate their effectiveness. As can be seen in Table 10, the variant without time warping get the lowest average ranking (4.52), implying that removing time warping has a broader negative effect among the 30 datasets than the other data augmentation methods. The data augmentation libraries without each of the other four methods have a close average ranking. In contrast, the complete version has the highest average ranking (1.98), suggesting that the performance of CSL can probably be further improved when more types of data augmentation approaches are included in the library. This is an interesting finding and may imply a general data-independent data augmentation scheme for unsupervised representation learning of MTS. We leave the further exploration in our future work.

5.4 Sensitivity Analysis

We perform sensitivity analysis to study the key parameters, including the number of shapelet scales R (default 8), the minimum and maximum lengths of the shapelets L_{min} (default $0.1T$) and L_{max} (default $0.8T$), the decay rate α (default 0.5), and the regularization coefficients λ (default 0.01) and λ_S (default 1).

Similar to the setting in Section 3.2, given L_{min} , L_{max} and R , the shapelet lengths are simply set to the evenly spaced numbers over

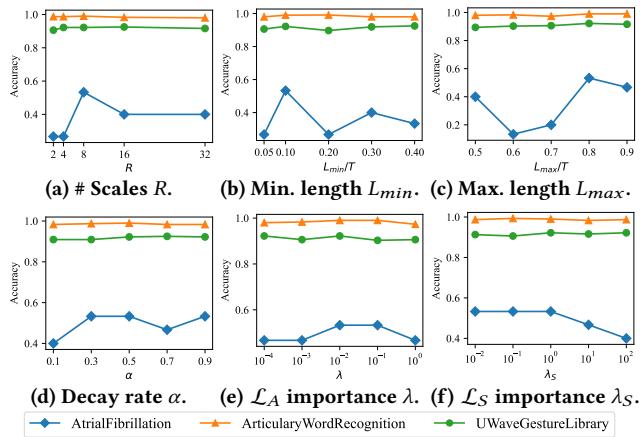


Figure 7: Sensitivity analysis of the key parameters.

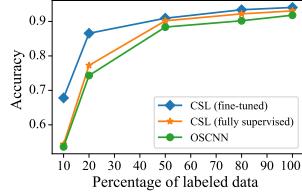


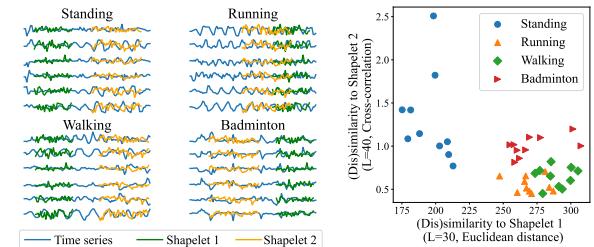
Figure 8: Accuracy of OSCNN, fully supervised and fine-tuned CSL w.r.t. the ratio of labeled data on UWaveGestureLibrary.

$[L_{min}, L_{max}]$, i.e., $L_r = L_{min} + (r - 1) \frac{L_{max} - L_{min}}{R-1}$ ($r \in \{1, \dots, R\}$). The performance is evaluated using classification accuracy. The results on three diverse UEA datasets are shown in Fig. 7 and the similar trends can be observed on the other datasets. Please note that the performance on AtrialFibrillation seems sensitive just because the dataset has only 15 testing samples, the minimum number among the 30 UEA datasets, which is a corner case of our evaluation. We discuss the results in detail as follows.

The sensitivity analysis of R . To capture multi-scale information, the number of scales R cannot be too small. But too large R will cause a small number of shapelets K under a fixed representation dimension D_{repr} , which can also decrease the representation quality. As the result in Fig. 7a shows, the model is relatively more sensitive to small values of R than larger values, while a moderate value of 8 can lead to good overall performance among the datasets.

The sensitivity analysis of L_{min} and L_{max} . From Fig. 7b-7c, we can see that for UWaveGestureLibrary, the best choice of L_{min} is about $0.4T$ and the values of $0.8T$ - $0.9T$ are the best for L_{max} , which indicates that the long-term features can be more effective than the short-term ones. For ArticularyWordRecognition, the relatively small values of L_{min} ($0.1T$ - $0.2T$) and large values of L_{max} ($0.8T$ - $0.9T$) are better, showing the importance of both short- and long-term features. While on the AtrialFibrillation dataset, $L_{min} = 0.1T$ and $0.8T$ - $0.9T$ for L_{max} are empirically the best choice. Overall, the default settings of $L_{min} = 0.1T$ and $L_{max} = 0.8T$ can be decent for different datasets without any tuning (also validated in Section 5.3), while one can manually optimize them for further improvement.

The sensitivity analysis of α . As the result shown in Fig. 7d, our model is more sensitive to the small values of the decay rate α than the large values for the UWaveGestureLibrary and AtrialFibrillation datasets, and the opposite for ArticularyWordRecognition. Overall,



(a) Time series of the four classes and two shapelets with different lengths and (dis)similarity measures learned by CSL.
(b) The two-dimensional representations of all time series encoded using the two learned shapelets.

Figure 9: Explanation of the shapelets learned by CSL.

our model is less sensitive to α than the other parameters, and a moderate value around 0.5 is better for different datasets.

The sensitivity analysis of λ and λ_S . As shown in Fig. 7e, by varying λ from 1 to 0.0001, we observe that our model achieves good performance among different datasets when λ is around 0.01. Similarly, we vary λ_S from 100 to 0.01. The result in Fig. 7f indicates that our model is more robust to the larger values of λ_S (1 to 100) for UWaveGestureLibrary and the opposite for AtrialFibrillation (where the model is more sensitive when $\lambda_S > 1$). Overall, a moderate value around 1 can be a good choice for different datasets.

5.5 Study of Partially Labeled Classification

To further demonstrate the superiority of our CSL, we perform a case study on UWaveGestureLibrary under a practical setting of partially labeled MTS classification. Specifically, we compare CSL with the best-performing supervised OSCNN on the dataset where only a portion of the randomly selected data is labeled. For CSL, we first use all the data to train the Shapelet Transformer without using labels. Then, we append a linear classifier on top of the representations, and fine-tune the encoder and linear layer using the available labeled data by minimizing the standard cross-entropy loss as used in OSCNN. In contrast, OSCNN is supervisedly trained using the same labeled data (fully supervised). For comparison, we also train a CSL model in the same fully supervised way as OSCNN.

As Fig. 8 shows, the fully supervised CSL performs very closely to OSCNN. The fine-tuned CSL consistently outperforms the two competitors, especially when the proportion of labeled data is small. Taking advantage of URL which can “pre-train” the encoder using all available data regardless of annotations, the fine-tuned CSL uses only 20% labeled data to achieve accuracy comparable to the fully supervised OSCNN and CSL trained with 50% labeled data. The results show the superiority of our URL method in partially-label settings, compared to the traditional fully supervised techniques.

5.6 Study of the Learned Shapelets

To provide an intuitive understanding of the features CSL extracts, we study the learned shapelets using an easy-to-understand BasicMotions problem from UEA archive. The time series are sensor records of four human motions, i.e., *Standing*, *Walking*, *Running* and *Badminton*. Each sample has six dimensions and of length $T = 100$.

We plot four time series of the four classes and two shapelets with different lengths and measures learned by our CSL (see Fig. 9a). Shapelet 1 is of length 30 and encodes the samples using Euclidean

Table 11: Accuracy and SVM training time on long time series.

Dataset	Raw values		Unsupervised learned representation						
	Acc	Time	TS2Vec	T-Loss	TNC	TS-TCC	TST	CSL	CSL (Speedup)
BH	0.732	<i>0.6204s</i>	0.712	0.732	0.732	0.658	0.720	0.732	0.0038s (163x)
CD	0.482	<i>0.4369s</i>	0.520	0.695	0.549	0.629	0.625	0.712	0.0031s (141x)
DA	0.240	<i>0.1968s</i>	0.193	0.400	0.220	0.286	0.307	0.373	0.0012s (164x)
US	0.210	<i>205.7195s</i>	0.155	0.600	0.112	0.379	0.183	0.692	1.2834s (160x)

distance (Green). Shapelet 2 has a length of 40 and is used in conjunction with the cross-correlation function for encoding (Orange). The shapelets are matched to the most similar subsequences for each of the time series samples. Note that CSL uses multivariate shapelets to jointly capture the information among different variables, where each shapelet has the same dimensions as the time series. The two shapelets encode each time series sample into a two-dimensional representation (Fig. 9b), where each axis is the (dis)similarity between the shapelet and the matching subsequence according to Eq. (3).

From Fig. 9, we observe that the representation based on Shapelet 1 (X-axis) can distinguish the *Standing* motion while the other three motions can be effectively classified by the features extracted using both shapelets. Thus, the shapelets can be seen as the prototypes of some classes and the representations are explained as the degree the shapelets exist in the time series, which is intuitive to understand. Our proposal not only extends the original shapelet which is designed only for supervised classification to general-purpose URL, but also retains its benefit in terms of explainability or interpretability [58]. Although the interpretation method is ad-hoc, it remains a nice property of the shapelet compared to the complex neural networks which are harder to explain [37].

5.7 Study of Long Time Series Representation

We assess the ability of the URL methods on long time series representation. Four datasets from the Time Series Machine Learning Website [2] are used including BinaryHeartbeat (BH), Cats-Dogs (CD), DucksAndGeese (DA) and UrbanSound (US). The series lengths of the four datasets are 18530, 14773, 236784 and 44100 respectively and the other statistics can be found on the website. Following Section 5.1, we train an SVM using either the unsupervised learned representation or the raw values of the training data, and report the test accuracy and the SVM training time (marked in italics) in Table 11. In terms of accuracy, CSL performs the best on three data sets and the second-best on DA, showing its higher ability in long series representation. T-Loss is also well-performed, but is still inferior to CSL on CD and US. TS2Vec and TST cannot handle long series due to high space complexity, so they have to shorten the raw data by truncation or subsampling following [59], which may cause information loss and result in their low performance. Compared to analysis on the raw values, using the representation learned by CSL can not only improve the accuracy, but also achieve more than 140x of speedups for the SVM training. This indicates the superiority of the proposed CSL in long time series analysis.

5.8 Running Time Analysis

Although our main goal is to improve the representation quality of URL, we show that the running time of the proposed CSL is also less than or comparable to the URL baselines. We first assess the accuracy with respect to the training time using two medium-sized datasets. As shown in Fig. 10, CSL and TS2Vec achieve the

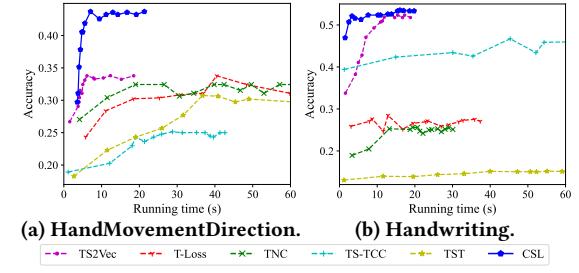


Figure 10: Accuracy w.r.t. total training time.

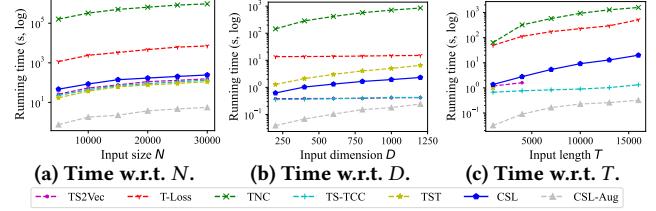


Figure 11: Training time per epoch of varying input size (N), dimension (D) and series length (T) on InsectWingbeat, DuckDuckGeese and EigenWorms respectively.

same or higher accuracy using much less time, showing that they are faster to train than the other URL methods, while CSL is also faster than TS2Vec. Next, we evaluate the training time per epoch on InsectWingbeat, DuckDuckGeese and EigenWorms, the UEA datasets with the largest input size, dimension and series length. The results are shown in Fig. 11a-11c respectively. TS-TCC runs fast in most cases. TS2Vec is also time-efficient, but it cannot scale to large length T due to high memory consumption (Fig. 11c). TST is slower than CSL for high-dimensional time series (Fig. 11b) and runs out of memory for large T (Fig. 11c). T-Loss and TNC, though have smaller time complexity, are much slower than the others with considerably large N , D and T . The reason is that they consist of many sequential operations which cannot be sped up with GPUs. CSL is fairly efficient among the URL methods in terms of running time per epoch. More importantly, CSL can be faster to train as we have illustrated above as it converges using less number of epochs. Besides, we observe that the time spent on data augmentation (CSL-Aug) is very little during the CSL training.

6 CONCLUSION

This paper presents a novel URL framework named CSL, which leverages contrastive learning for MTS-specific representation. Particularly, we design a unified shapelet-based encoder and an objective with multi-grained contrasting and multi-scale alignment to capture information in various time ranges. We also build a data augmentation library including diverse types of methods to improve the generality. Extensive experiments on tens of real-world datasets demonstrate the superiority of CSL over the baselines on downstream classification, clustering, and anomaly detection tasks.

ACKNOWLEDGMENTS

This paper was supported by NSFC grant (62232005, 62202126) and The National Key Research and Development Program of China (2020YFB1006104).

REFERENCES

- [1] Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. 2013. Deep canonical correlation analysis. In *International conference on machine learning*. PMLR, 1247–1255.
- [2] Anthony Bagnall, Eamonn Keogh, Jason Lines, Aaron Bostrom, James Large, and Matthew Middlehurst. [n.d.]. Time Series Machine Learning Website. www.timeseriesclassification.com.
- [3] Anthony J. Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn J. Keogh. 2018. The UEA multivariate time series classification archive, 2018. *CoRR* abs/1811.00075 (2018). arXiv:1811.00075 <http://arxiv.org/abs/1811.00075>
- [4] Stefanos Bennett, Mihai Cucuringu, and Gesine Reinert. 2022. Detection and clustering of lead-lag networks for multivariate time series with an application to financial markets. (2022).
- [5] Aaron Bostrom and Anthony Bagnall. 2017. Binary shapelet transform for multiclass time series classification. In *Transactions on Large-Scale Data-and Knowledge-Centered Systems XXXII*. Springer, 24–46.
- [6] Xiaobin Chang, Tao Xiang, and Timothy M Hospedales. 2018. Scalable and effective deep CCA via soft decorrelation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1488–1497.
- [7] Yves Chauvin and David E Rumelhart. 2013. *Backpropagation: theory, architectures, and applications*. Psychology press.
- [8] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A Simple Framework for Contrastive Learning of Visual Representations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event (Proceedings of Machine Learning Research)*, Vol. 119. PMLR, 1597–1607. <http://proceedings.mlr.press/v119/chen20j.html>
- [9] Timothy Derrick and Joshua Thomas. 2004. Time series analysis: the cross-correlation function. (2004).
- [10] Emadeldene Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, Chee Keong Kwoh, Xiaoli Li, and Cuntai Guan. 2021. Time-Series Representation Learning via Temporal and Contextual Contrasting. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21, Zhi-Hua Zhou (Ed.)*. International Joint Conferences on Artificial Intelligence Organization, 2352–2359. <https://doi.org/10.24963/ijcai.2021/324> Main Track.
- [11] Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. 2019. Unsupervised scalable representation learning for multivariate time series. *Advances in neural information processing systems* 32 (2019).
- [12] Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple Contrastive Learning of Sentence Embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (Eds.). Association for Computational Linguistics, 6894–6910. <https://doi.org/10.18653/v1/2021.emnlp-main.552>
- [13] Josif Grabocka, Nicolas Schilling, Martin Wistuba, and Lars Schmidt-Thieme. 2014. Learning time-series shapelets. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 392–401.
- [14] Sihoo Han and Simon S Woo. 2022. Learning Sparse Latent Graph Representations for Anomaly Detection in Multivariate Time Series. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2977–2986.
- [15] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. 2020. Momentum Contrast for Unsupervised Visual Representation Learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*. Computer Vision Foundation / IEEE, 9726–9735. <https://doi.org/10.1109/CVPR42600.2020.00975>
- [16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.
- [17] Jon Hills, Jason Lines, Edgaras Baranauskas, James Mapp, and Anthony Bagnall. 2014. Classification of time series by shapelet transformation. *Data Mining and Knowledge Discovery* 28, 4 (2014), 851–881.
- [18] Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Söderström. 2018. Detecting Spacecraft Anomalies Using LSTMs and Non-parametric Dynamic Thresholding. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, Yike Guo and Faisal Farooq (Eds.). ACM, 387–395. <https://doi.org/10.1145/3219819.3219845>
- [19] Dino Ienco and Roberto Interdonato. 2020. Deep Multivariate Time Series Embedding Clustering via Attentive-Gated Autoencoder. In *Advances in Knowledge Discovery and Data Mining - 24th Pacific-Asia Conference, PAKDD 2020, Singapore, May 11-14, 2020, Proceedings, Part I (Lecture Notes in Computer Science)*, Hady W. Lauw, Raymond Chi-Wing Wong, Alexandros Ntoulas, Ee-Peng Lim, See-Kiong Ng, and Simo Jialin Pan (Eds.), Vol. 12084. Springer, 318–329. https://doi.org/10.1007/978-3-030-47426-3_25
- [20] Arundo Analytics Inc. 2019. *tsaug: An open-source package for time series data augmentation*. Retrieved January 1, 2023 from <https://tsaug.readthedocs.io/en/stable/references.html>
- [21] Sergey Ioffe and Christian Szegedy. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015 (JMLR Workshop and Conference Proceedings)*, Francis R. Bach and David M. Blei (Eds.), Vol. 37. JMLR.org, 448–456. <http://proceedings.mlr.press/v37/ioffe15.html>
- [22] Brian Kenji Iwana and Seiichi Uchida. 2021. An empirical survey of data augmentation for time series classification with neural networks. *Plos one* 16, 7 (2021), e0254841.
- [23] Fazole Karim, Somshubra Majumdar, Houshang Darabi, and Samuel Harford. 2019. Multivariate LSTM-FCNs for time series classification. *Neural Networks* 116 (2019), 237–245. <https://doi.org/10.1016/j.neunet.2019.04.014>
- [24] Eunji Kim, Sungzoon Cho, Byeongeon Lee, and Myoungsu Cho. 2019. Fault detection and diagnosis using self-attentive convolutional neural networks for variable-length sensor data in semiconductor manufacturing. *IEEE Transactions on Semiconductor Manufacturing* 32, 3 (2019), 302–309.
- [25] Guozhong Li, Byron Choi, Jianliang Xu, Sourav S. Bhowmick, Kwok-Pan Chun, and Grace Lai-Hung Wong. 2021. ShapeNet: A Shapelet-Neural Network Approach for Multivariate Time Series Classification. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. AAAI Press, 8375–8383. <https://ojs.aaai.org/index.php/AAAI/article/view/17018>
- [26] Hailin Li. 2019. Multivariate time series clustering based on common principal component analysis. *Neurocomputing* 349 (2019), 239–247. <https://doi.org/10.1016/j.neucom.2019.03.060>
- [27] Junnan Li, Pan Zhou, Caiming Xiong, and Steven C. H. Hoi. 2021. Prototypical Contrastive Learning of Unsupervised Representations. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. <https://openreview.net/forum?id=KmnykpSuqeq>
- [28] Zhihan Li, Youjian Zhao, Jiaqi Han, Ya Su, Rui Jiao, Xidao Wen, and Dan Pei. 2021. Multivariate Time Series Anomaly Detection and Interpretation using Hierarchical Inter-Metric and Temporal Embedding. In *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, Feida Zhu, Beng Chin Ooi, and Chunyan Miao (Eds.). ACM, 3220–3230. <https://doi.org/10.1145/3447548.3467075>
- [29] Zhaowen Li, Yousong Zhu, Fan Yang, Wei Li, Chaoyang Zhao, Yingying Chen, Zhiyang Chen, Jiahao Xie, Liwei Wu, Rui Zhao, et al. 2022. Univip: A unified framework for self-supervised visual pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14627–14636.
- [30] Zhiyi Liang and Hongzhi Wang. 2021. Efficient class-specific shapelets learning for interpretable time series classification. *Information Sciences* 570 (2021), 428–450.
- [31] Jason Lines, Sarah Taylor, and Anthony Bagnall. 2018. Time series classification with HIVE-COTE: The hierarchical vote collective of transformation-based ensembles. *ACM transactions on knowledge discovery from data* 12, 5 (2018).
- [32] Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X. Liu, and Schahram Dastidar. 2022. Pyraformer: Low-Complexity Pyramidal Attention for Long-Range Time Series Modeling and Forecasting. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net. <https://openreview.net/forum?id=0EXmFzUn5l>
- [33] Qianli Ma, Wanqing Zhuang, and Garrison Cottrell. 2019. Triple-shapelet networks for time series classification. In *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 1246–1251.
- [34] Qianli Ma, Wanqing Zhuang, Sen Li, Desen Huang, and Garrison Cottrell. 2020. Adversarial dynamic shapelet networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 5069–5076.
- [35] Karl Øyvind Mikalsen, Filippo Maria Bianchi, Cristina Soguero-Ruiz, and Robert Jenssen. 2018. Time series cluster kernel for learning similarities between multivariate time series with missing data. *Pattern Recognit.* 76 (2018), 569–581. <https://doi.org/10.1016/j.patcog.2017.11.030>
- [36] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). <http://arxiv.org/abs/1301.3781>
- [37] Christoph Molnar. 2022. *Interpretable Machine Learning* (2 ed.). <https://christophm.github.io/interpretable-ml-book>
- [38] Abdullah Mueen, Eamonn Keogh, and Neal Young. 2011. Logical-shapelets: an expressive primitive for time series classification. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1154–1162.
- [39] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748* (2018).
- [40] Mert Ozer, Anna Sapienza, Andrés Abeluk, Goran Muric, and Emilio Ferrara. 2020. Discovering patterns of online popularity from time series. *Expert Syst. Appl.* 151 (2020), 113337. <https://doi.org/10.1016/j.eswa.2020.113337>

- [41] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. 2020. GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash (Eds.). ACM, 1150–1160. <https://doi.org/10.1145/3394486.3403168>
- [42] Ann Riley and Elvira Nica. 2021. Internet of things-based smart healthcare systems and wireless biomedical sensing devices in monitoring, detection, and prevention of COVID-19. *American Journal of Medical Research* 8, 2 (2021), 51–64.
- [43] Saeid Sanei and Jonathon A Chambers. 2013. *EEG signal processing*. John Wiley & Sons.
- [44] Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. 2019. Robust Anomaly Detection for Multivariate Time Series through Stochastic Recurrent Neural Network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evinaria Terzi, and George Karypis (Eds.). ACM, 2828–2837. <https://doi.org/10.1145/3292500.3330672>
- [45] Wensi Tang, Guodong Long, Lu Liu, Tianyi Zhou, Michael Blumenstein, and Jing Jiang. 2022. Omni-Scale CNNs: a simple and effective kernel size configuration for time series classification. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net. <https://openreview.net/forum?id=PDYs7Z2XFGv>
- [46] Sana Tonekaboni, Danny Eytan, and Anna Goldenberg. 2021. Unsupervised Representation Learning for Time Series with Temporal Neighborhood Coding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. <https://openreview.net/forum?id=8qDwejCuCN>
- [47] Liudmila Ulanova, Nurjahan Begum, and Eamonn J. Keogh. 2015. Scalable Clustering of Time Series with U-Shapelets. In *Proceedings of the 2015 SIAM International Conference on Data Mining, Vancouver, BC, Canada, April 30 - May 2, 2015*, Suresh Venkatasubramanian and Jieping Ye (Eds.). SIAM, 900–908. <https://doi.org/10.1137/1.978161974010.101>
- [48] Terry Taewoong Um, Franz Michael Josef Pfister, Daniel Pichler, Satoshi Endo, Muriel Lang, Sandra Hirche, Urban Fietzek, and Dana Kulic. 2017. Data augmentation of wearable sensor data for parkinson's disease monitoring using convolutional neural networks. In *Proceedings of the 19th ACM International Conference on Multimodal Interaction, ICMI 2017, Glasgow, United Kingdom, November 13 - 17, 2017*, Edward Lank, Alessandro Vinciarelli, Eve E. Hoggan, Sriram Subramanian, and Stephen A. Brewster (Eds.). ACM, 216–220. <https://doi.org/10.1145/3136755.3136817>
- [49] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. 2016. WaveNet: A Generative Model for Raw Audio. *CoRR* abs/1609.03499 (2016). arXiv:1609.03499 <http://arxiv.org/abs/1609.03499>
- [50] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research* 9, 86 (2008), 2579–2605. <http://jmlr.org/papers/v9/vandermaaten08a.html>
- [51] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.), 5998–6008. <https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fb0d053c1c4a845aa-Abstract.html>
- [52] Qiao Xiao, Boqian Wu, Yu Zhang, Shiwei Liu, Mykola Pechenizkiy, Elena Mocanu, and Decebal Constantin Mocanu. 2022. Dynamic Sparse Network for Time Series Classification: Learning What to "see". *CoRR* abs/2212.09840 (2022). <https://doi.org/10.48550/arXiv.2212.09840> arXiv:2212.09840
- [53] Chang Xu, Dacheng Tao, and Chao Xu. 2013. A Survey on Multi-view Learning. *CoRR* abs/1304.5634 (2013). arXiv:1304.5634 <http://arxiv.org/abs/1304.5634>
- [54] Akihiro Yamaguchi, Ken Ueo, and Hisashi Kashima. 2022. Learning Evolvable Time-series Shapelets. In *38th IEEE International Conference on Data Engineering, ICDE 2022, Kuala Lumpur, Malaysia, May 9-12, 2022*. IEEE, 793–805. <https://doi.org/10.1109/ICDE53745.2022.000064>
- [55] Jinyu Yang, Jiali Duan, Son Tran, Yi Xu, Sampath Chanda, Liqun Chen, Belinda Zeng, Trishul Chilimbi, and Junzhou Huang. 2022. Vision-language pre-training with triple contrastive learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 15671–15680.
- [56] Ling Yang and Shenda Hong. 2022. Unsupervised time-series representation learning with iterative bilinear temporal-spectral fusion. In *International Conference on Machine Learning*. PMLR, 25038–25054.
- [57] Junchen Ye, Zihhan Liu, Bowen Du, Leilei Sun, Weimiao Li, Yanjie Fu, and Hui Xiong. 2022. Learning the evolutionary and multi-scale graph structure for multivariate time series forecasting. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2296–2306.
- [58] Lexiang Ye and Eamonn Keogh. 2011. Time series shapelets: a novel technique that allows accurate, interpretable and fast classification. *Data mining and knowledge discovery* 22, 1-2 (2011), 149–182.
- [59] Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. 2022. Ts2vec: Towards universal representation of time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 8980–8987.
- [60] George Zerveas, Srideepika Jayaraman, Dhaval Patel, Anuradha Bhamidipaty, and Carsten Eickhoff. 2021. A transformer-based framework for multivariate time series representation learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2114–2124.
- [61] Nan Zhang and Shiliang Sun. 2022. Multiview Unsupervised Shapelet Learning for Multivariate Time Series Clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022), 1–16. <https://doi.org/10.1109/TPAMI.2022.3198411>
- [62] Qin Zhang, Jia Wu, Peng Zhang, Guodong Long, and Chengqi Zhang. 2019. Salient Subsequence Learning for Time Series Clustering. *IEEE Trans. Pattern Anal. Mach. Intell.* 41, 9 (2019), 2193–2207. <https://doi.org/10.1109/TPAMI.2018.2847699>
- [63] Xuchao Zhang, Yifeng Gao, Jessica Lin, and Chang-Tien Lu. 2020. TapNet: Multivariate Time Series Classification with Attentional Prototypical Network. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAII 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 6845–6852. <https://ojs.aaai.org/index.php/AAAI/article/view/6165>
- [64] Xiang Zhang, Ziyuan Zhao, Theodoros Tsiligkaridis, and Marinka Zitnik. 2022. Self-Supervised Contrastive Pre-Training For Time Series via Time-Frequency Consistency. *CoRR* abs/2206.08496 (2022). <https://doi.org/10.48550/arXiv.2206.08496>
- [65] Yanzhao Zhang, Richong Zhang, Samuel Mensah, Xudong Liu, and Yongyi Mao. 2022. Unsupervised sentence representation via contrastive learning with mixing negatives. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 11730–11738.
- [66] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. 2022. FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (Eds.), Vol. 162. PMLR, 27268–27286.